**Business Intelligence Lab**

**Experiment 01**

**Aim:**

To design Data Warehouse for given case study and perform ETL and OLAP operations on it

**Theory:**

**Data Warehouse**

Data Warehouse is a relational database management system (RDBMS) construct to meet the requirements of transaction processing systems. It can be loosely described as any centralized data repository which can be queried for business benefits. It is a relational database that is designed for query and analysis rather than transaction processing. It includes historical data derived from transaction data from single and multiple sources.

Data Warehousing (DW) is a process for collecting and managing data from varied sources to provide meaningful business insights. Data Warehousing supports architectures and tools for business executives to systematically organize, understand and use their information to make strategic decisions.

**Characteristics of Data Warehouse**

1. Subject-Oriented

A data warehouse target on the modeling and analysis of data for decision-makers. Therefore, data warehouses typically provide a concise and straightforward view around a particular subject, such as customer, product, or sales, instead of the global organization's ongoing operations. This is done by excluding data that are not useful concerning the subject and including all data needed by the users to understand the subject.

2. Integrated

A data warehouse integrates various heterogeneous data sources like RDBMS, flat files, and online transaction records. It requires performing data cleaning and integration during data warehousing to ensure consistency in naming conventions, attributes types, etc., among different data sources.

3. Time-Variant

Historical information is kept in a data warehouse. For example, one can retrieve files from 3 months, 6 months, 12 months, or even previous data from a data warehouse. These variations with a transactions system, where often only the most current file is kept.

4. Non-Volatile

The data warehouse is a physically separate data storage, which is transformed from the source operational RDBMS. The operational updates of data do not occur in the data warehouse, i.e., update, insert, and delete operations are not performed. It usually requires only two procedures in data accessing: Initial loading of data and access to data. Therefore, the DW does not require transaction processing, recovery, and concurrency capabilities, which allows for substantial speedup of data retrieval. Non-Volatile defines that once entered into the warehouse, and data should not change.

**Types of Data Warehouse**

1. Enterprise Data Warehouse (EDW)

Enterprise Data Warehouse (EDW) is a centralized warehouse. It provides decision support service across the enterprise. It offers a unified approach for organizing and representing data. It also provides the ability to classify data according to the subject and give access according to those divisions.

2. Operational Data Store

Operational Data Store, which is also called ODS, is nothing but a data store required when neither Data warehouse nor OLTP systems support organizations reporting needs. In ODS, the Data warehouse is refreshed in real time. Hence, it is widely preferred for routine activities like storing records of the Employees.

3. Data Mart

A data mart is a subset of the data warehouse. It is specially designed for a particular line of business, such as sales, finance, sales or finance. In an independent data mart, data can be collected directly from sources.

**Benefits of Data Warehouse**

1. Understand business trends and make better forecasting decisions.
2. Data Warehouses are designed to perform enormous amounts of data.
3. The structure of data warehouses is more accessible for end-users to navigate, understand, and query.
4. Queries that would be complex in many normalized databases could be easier to build and maintain in data warehouses.
5. Data warehousing is an efficient method to manage demand for lots of information from lots of users.
6. Data warehousing provides the capabilities to analyze a large amount of historical data.

**Data warehouse designing phases**

There are 8 core steps for that go into building a data warehouse

**1. Requirements Gathering**

Data warehouses touch all areas of business, so every department needs to be on board with the design. Since your warehouse is only as powerful as the data contained within it, aligning department needs and goals with the overall project is critical to success.

This Requirements Gathering stage should focus on the following objectives.

1. Aligning department goals with the overall project
2. Determining the scope of the project in relation to business objectives
3. Discovering future needs and current needs by diving deep into your data
4. Creating a disaster recovery plan in the case of system failure
5. Thinking about each layer of security
6. Anticipating compliance needs and mitigating regulatory risks

Data warehouse is like a blueprint.

**2.** **Setting up Physical Environments**

Data warehouses typically have three primary physical environments

1. Development
2. Testing
3. Production.

This mimics standard software development best practices, and your three environments will exist on completely separate physical servers.

**3. Introducing Data Modeling**

Data modeling is the process of visualizing data distribution in your warehouse. Data modeling helps you visualize the relationships between data, and it's useful for setting standardized naming conventions, creating relationships between data sets, and establishing compliance and security processes that align with your overarching IT goals.

Data modeling typically takes place at the data mart level and branches out into your data warehouse. It's the logic of how you're storing data in relation to other data.

The three most popular data models for warehouses are:

1. Snowflake Schema
2. Star Schema
3. Galaxy Schema

**4. Choosing Your Extract, Transfer, Load (ETL) Solution**

ETL or Extract, Transfer, Load is the process you'll use to pull data out of your current tech stack or existing storage solutions and put it into your warehouse. You should pay careful attention to the ETL solution that you use.

Since, ETL is responsible for the bulk of the in-between work, choosing a subpar or developing a poor ETL process can break your entire warehouse. You want optimal speeds, good visualization, and the ability to build easy, replicable, and consistent data pipelines between all of your existing architecture and your new warehouse.

**5. Online Analytic Processing (OLAP) Cube**

You will likely need to address OLAP cubes if you're designing your entire database from scratch, or if you have to maintain your own OLAP cube — which typically requires specialized personnel. So, if you plan on using a vendor warehouse solution you probably won't need to utilize an OLAP cube. If you have a set of BI tools that require you to utilize an OLAP cube for ad-hoc reporting, you may need to develop one or use a vendor solution.

**6. Creating the Front End**

So far, we've only covered backend processes. There needs to be front-end visualization, so users can immediately understand and apply the results of data queries.

That's the job of your front-end. There are plenty of tools on the market that helps with visualization. BI tools like Tableau or PowerBI for those using BigQuery are great for visualization. You can also develop a custom solution — though that's a significant undertaking.

Most small-to-medium-sized businesses lean on established BI kits like those mentioned above. But, some businesses may need to develop their own BI tools to meet ad-hoc analytic needs. For example, a Sales Ops manager at a large company may need a specific BI tool for territory strategies. This tool may need to be custom developed given the scope of their sales objectives.

**7. Optimizing Queries**

Optimizing your queries is a complex process that's unique to your specific needs.

**8. Establishing a Rollout**

Once you're ready to launch your warehouse, it's time to start thinking about education, training, and use cases. Most of the time, it will be a week or two before your end-users start seeing any functionality from that warehouse (at least at scale). But, they should be adequately trained before the rollout is completed.

**Dimensional Schema**

Multidimensional Schema is especially designed to model data warehouse systems. The schemas are designed to address the unique needs of very large databases designed for the analytical purpose (OLAP).

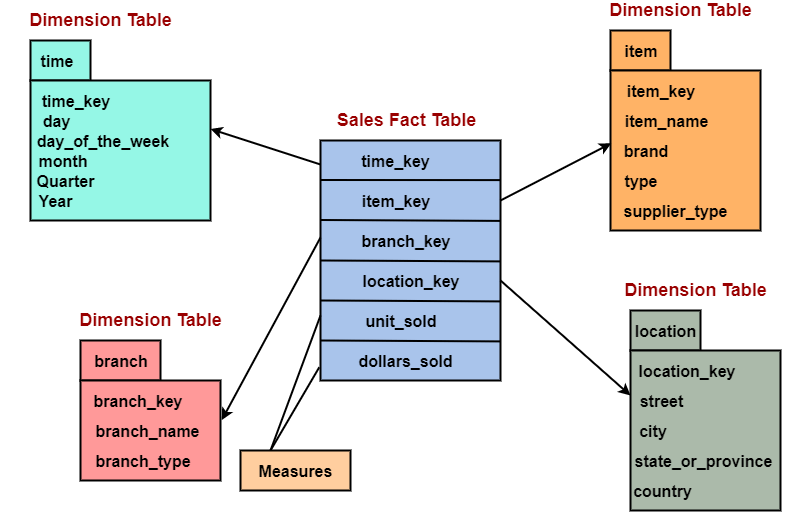
Types of Data Warehouse Schema:

Following are 3 chief types of multidimensional schemas each having its unique advantages:

1. Star Schema
2. Snowflake Schema
3. Galaxy Schema

**Star Schema**

Star Schema in a data warehouse, in which the center of the star can have one fact table and a number of associated dimension tables. It is known as star schema as its structure resembles a star. The Star Schema data model is the simplest type of Data Warehouse schema. It is also known as Star Join Schema and is optimized for querying large data sets.



Characteristics of Star Schema:

* Every dimension in a star schema is represented with the only one-dimension table.
* The dimension table should contain the set of attributes.
* The dimension table is joined to the fact table using a foreign key
* The dimension table are not joined to each other
* Fact table would contain key and measure
* The Star schema is easy to understand and provides optimal disk usage.
* The dimension tables are not normalized.
* The schema is widely supported by BI Tools

**Snowflake Schema**

Snowflake Schema in a data warehouse is a logical arrangement of tables in a multidimensional database such that the ER diagram resembles a snowflake shape. A Snowflake Schema is an extension of a Star Schema, and it adds additional dimensions. The dimension tables are normalized which splits data into additional tables.

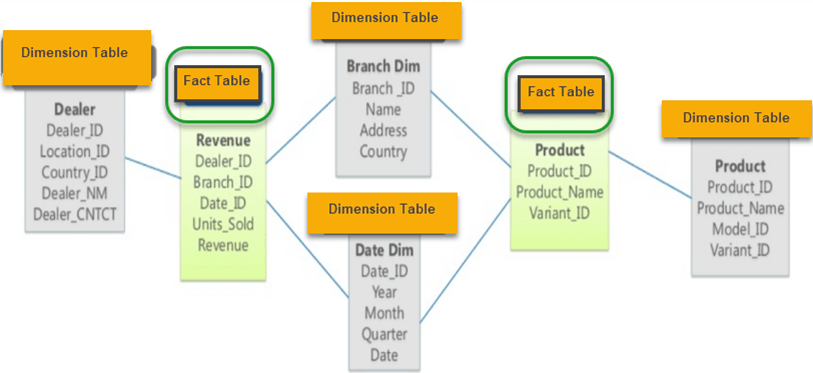


Characteristics of Snowflake Schema:

* The main benefit of the snowflake schema is that it uses smaller disk space.
* Easier to implement a dimension is added to the Schema
* Due to multiple tables query performance is reduced
* The primary challenge that you will face while using the snowflake Schema is that you need to perform more maintenance efforts because of the more lookup tables.

**Galaxy Schema**

A Galaxy Schema contains two fact tables that share dimension tables between them. It is also called Fact Constellation Schema. The schema is viewed as a collection of stars hence the name Galaxy Schema.



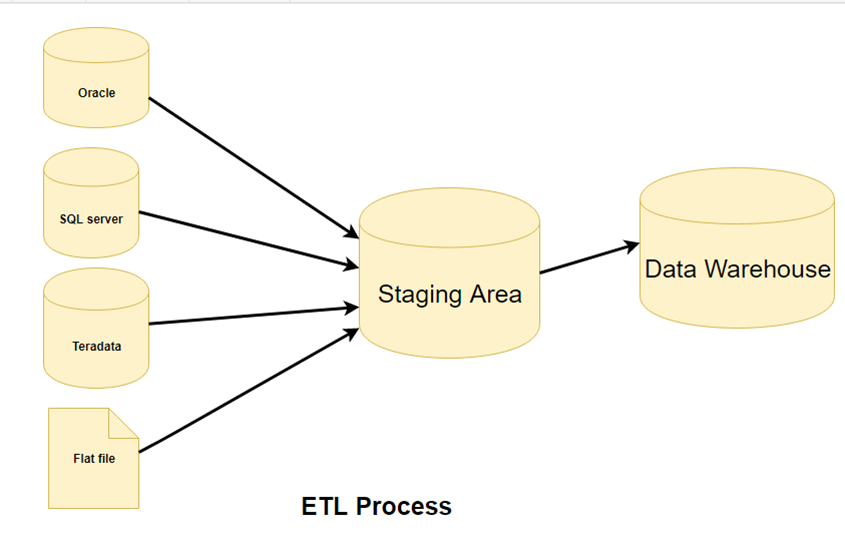
Characteristics of Galaxy Schema

* The dimensions in this schema are separated into separate dimensions based on the various levels of hierarchy.
* For example, if geography has four levels of hierarchy like region, country, state, and city then the Galaxy schema should have four dimensions.
* Moreover, it is possible to build this type of schema by splitting the one-star schema into more Star schemes.
* The dimensions are large in this schema which is needed to build based on the levels of hierarchy.
* This schema is helpful for aggregating fact tables for better understanding.

**ETL**

ETL is a process that extracts the data from different source systems, then transforms the data (like applying calculations, concatenations, etc.) and finally loads the data into the Data Warehouse system. Full form of ETL is Extract, Transform and Load.

The ETL process requires active inputs from various stakeholders including developers, analysts, testers, top executives and is technically challenging.



**Why we need ETL -**

1. It helps companies to analyze their business data for taking critical business decisions.
2. Transactional databases cannot answer complex business questions that can be answered by ETL examples.
3. ETL provides a method of moving the data from various sources into a data warehouse.
4. ETL processes can perform complex transformations and require the extra area to store the data.
5. ETL is a predefined process for accessing and manipulating source data into the target database.

**Step 1: Extraction**

Data is extracted from the source system into the staging area. Transformations if any are done in the staging area so that performance of the source system is not degraded. Also, if corrupted data is copied directly from the source into a Data warehouse database, rollback will be a challenge.

**Step 2: Transformation**

Data extracted from the source server is raw and not usable in its original form. Therefore it needs to be cleansed, mapped and transformed. In fact, this is the key step where the ETL process adds value and changes data such that insightful BI reports can be generated.

It is one of the important ETL concepts where you apply a set of functions on extracted data. Data that does not require any transformation is called as direct move or pass through data.

**Step 3: Loading**Loading data into the target data warehouse database is the last step of the ETL process. In a typical Data warehouse, a huge volume of data needs to be loaded in a relatively short period (nights). Hence, the load process should be optimized for performance.

**OLAP**

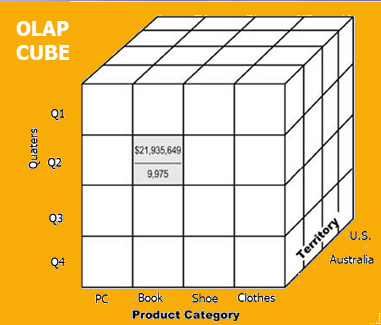
Online Analytical Processing (OLAP) is a category of software that allows users to analyze information from multiple database systems at the same time. It is a technology that enables analysts to extract and view business data from different points of view.

Analysts frequently need to group, aggregate and join data. These OLAP operations in data mining are resource intensive. With OLAP data can be pre-calculated and pre-aggregated, making analysis faster.

OLAP databases are divided into one or more cubes. The cubes are designed in such a way that creating and viewing reports become easy. OLAP stands for Online Analytical Processing.

**OLAP cube**:

At the core of the OLAP concept, is an OLAP Cube. The OLAP cube is a data structure optimized for very quick data analysis. The OLAP Cube consists of numerical facts called measures which are categorized by dimensions. OLAP Cube is also called the hypercube.



Usually, data operations and analysis are performed using the simple spreadsheet, where data values are arranged in row and column format. This is ideal for two-dimensional data. However, OLAP contains multidimensional data, with data usually obtained from a different and unrelated source. Using a spreadsheet is not an optimal option. The cube can store and analyze multidimensional data in a logical and orderly manner.

**Basic analytical operations of OLAP**

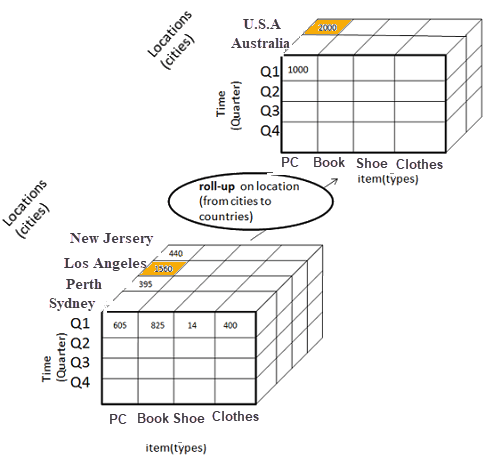
Four types of analytical OLAP operations are:

1. Roll-up:

Roll-up is also known as “consolidation” or “aggregation.” The Roll-up operation can be performed in 2 ways:

1. Reducing dimensions
2. Climbing up concept hierarchy. Concept hierarchy is a system of grouping things based on their order or level.

Consider this diagram:



In this example, cities New jersey and Lost Angles and rolled up into country USA

The sales figure of New Jersey and Los Angeles are 440 and 1560 respectively. They become 2000 after roll-up

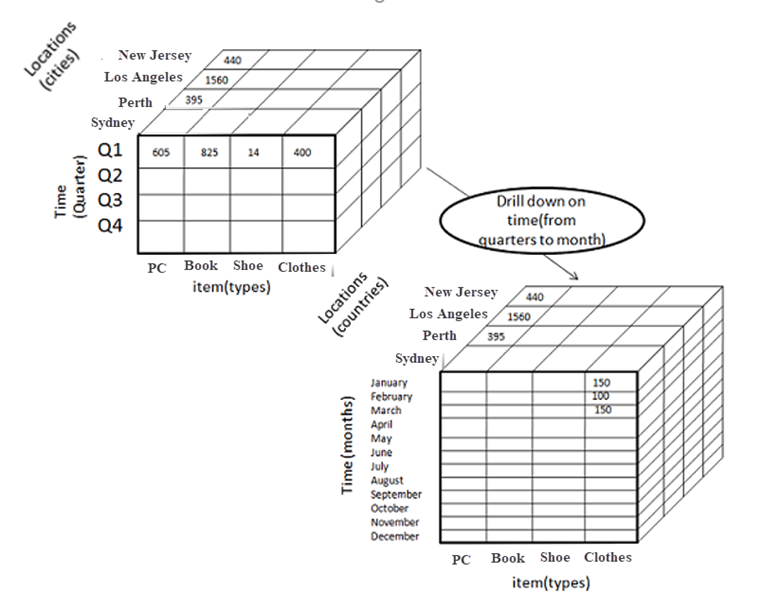
In this aggregation process, data location hierarchy moves up from city to country.

In the roll-up process at least one or more dimensions need to be removed. In this example, the Cities dimension is removed.

2. Drill-down

In drill-down data is fragmented into smaller parts. It is the opposite of the rollup process. It can be done via

1. Moving down the concept hierarchy
2. Increasing a dimension



Consider the diagram above

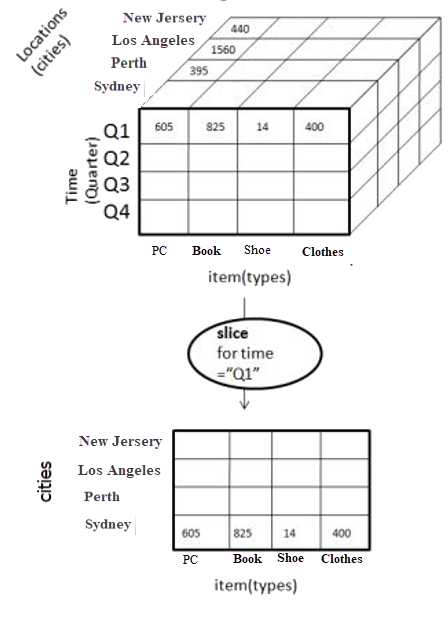
Quater Q1 is drilled down to months January, February, and March. Corresponding sales are also registers.

In this example, dimension months are added.

3. Slice:

Here, one dimension is selected, and a new sub-cube is created.

Following diagram explain how slice operation performed:

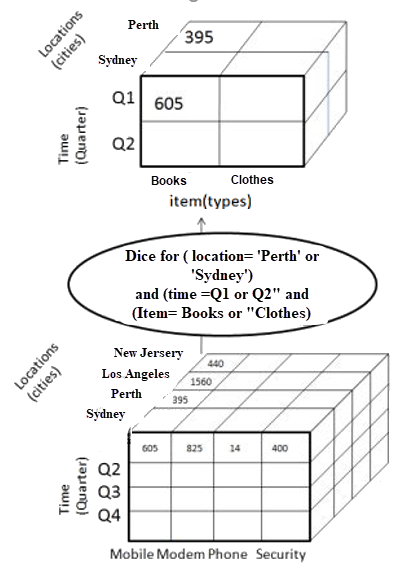


Dimension Time is Sliced with Q1 as the filter.

A new cube is created altogether.

Dice:

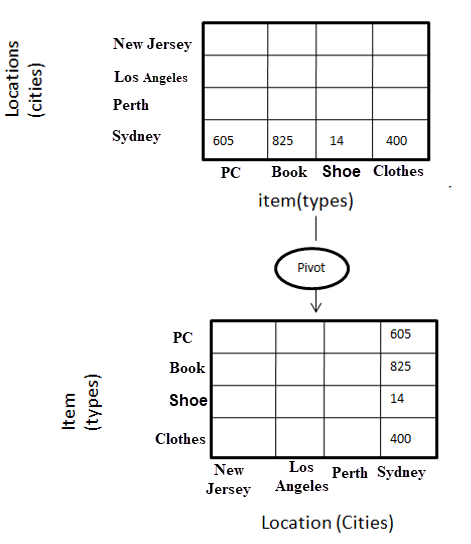
This operation is similar to a slice. The difference in dice is you select 2 or more dimensions that result in the creation of a sub-cube.



4. Pivot:

In Pivot, you rotate the data axes to provide a substitute presentation of data.

In the following example, the pivot is based on item types.



Other OLAP operations may contain ranking the top-N or bottom-N elements in lists, as well as calculate moving average, growth rates, and interests, internal rates of returns, depreciation, currency conversions, and statistical tasks.

OLAP offers analytical modeling capabilities, containing a calculation engine for determining ratios, variance, etc. and for computing measures across various dimensions. It can generate summarization, aggregation, and hierarchies at each granularity level and at every dimensions intersection. OLAP also provide functional models for forecasting, trend analysis, and statistical analysis. In this context, the OLAP engine is a powerful data analysis tool.

**Output:**

**Case Study on eWallet**

**Business Requirements**

One of the online retail company’s features is an e-wallet service that holds credit and debit information of all the transactions of a customer that can be used to pay for products purchased on the platform.

The Finance department of the company would like to build reporting and analytics on the e-wallet service so they can understand the extent of the wallet usage and also provide the details of all the transactions to the consumer.

The aim is the retrieval of credit amount, debit amount for a given year, for a given month, for a given date for every customer who can have any type of wallet.

**Design Data Warehouse**

The four key decisions made during the design of a dimensional model include:

1. Select the business process.
2. Declare the grain.
3. Identify the dimensions.
4. Identify the facts.

**The decision steps for our e-Wallet case:**

**1. Select the business process:** Design is developed based on the requirements and background given but also keeping flexibility in mind. All the required fields are assumed to be available from the company’s transactional database.

**2. Grain definition:** Atomic grain refers to the lowest level at which data is captured by a given business process.The lowest level of data that can be captured in this context is wallet transactions i.e., all the credit and debit transactions on e-wallet.

**3. Dimensions:** Dimensions provide all the possible context surrounding the event of transaction.

**Implementation of Data Warehouse**

Creating database named eWalletDB with the following dimension tables:

DimWallet

CREATE TABLE [DimWallet] (

[Wallet\_id] INT PRIMARY KEY,

[Type] VARCHAR(20),

[Start\_Date] DATE,

[Expiry\_Date] DATE,

[Wallet\_Price] INT

)

DimCustomer

CREATE TABLE [DimCustomer] (

[Customer\_ID] INT PRIMARY KEY,

[dim\_Customer\_ID] INT,

[First\_Name] VARCHAR(30),

[Last\_Name] VARCHAR(30),

[Gender] VARCHAR(30),

[Email] VARCHAR(100),

[Address] VARCHAR(200),

[Start\_Date] DATE,

[End\_Date] DATE

)

DimDate

CREATE TABLE [DimDate] (

[DateKey] INT primary key,

[Date] DATE,

[FullDateUK] VARCHAR(30),

[DayOfMonth] VARCHAR(30),

[DayName] VARCHAR(30),

[DayOfWeekUK] VARCHAR(30),

[DayOfWeekInMonth] VARCHAR(30),

[DayOfWeekInYear] VARCHAR(30),

[DayOfYear] VARCHAR(30),

[WeekOfMonth] VARCHAR(30),

[WeekOfYear] VARCHAR(30),

[Month] VARCHAR(30),

[MonthName] VARCHAR(30),

[Year] VARCHAR(30),

[MonthYear] VARCHAR(300),

[MMYYYY] VARCHAR(30),

)

Following is the fact table:

FactWallet

CREATE TABLE [FactWallet] (

Transaction\_Id INT PRIMARY KEY,

Customer\_Id INT FOREIGN KEY REFERENCES DimCustomer(Customer\_ID),

Transaction\_Date INT FOREIGN KEY REFERENCES DimDate(DateKey),

Wallet\_Id INT FOREIGN KEY REFERENCES DimWallet(Wallet\_Id),

Type VARCHAR(50),

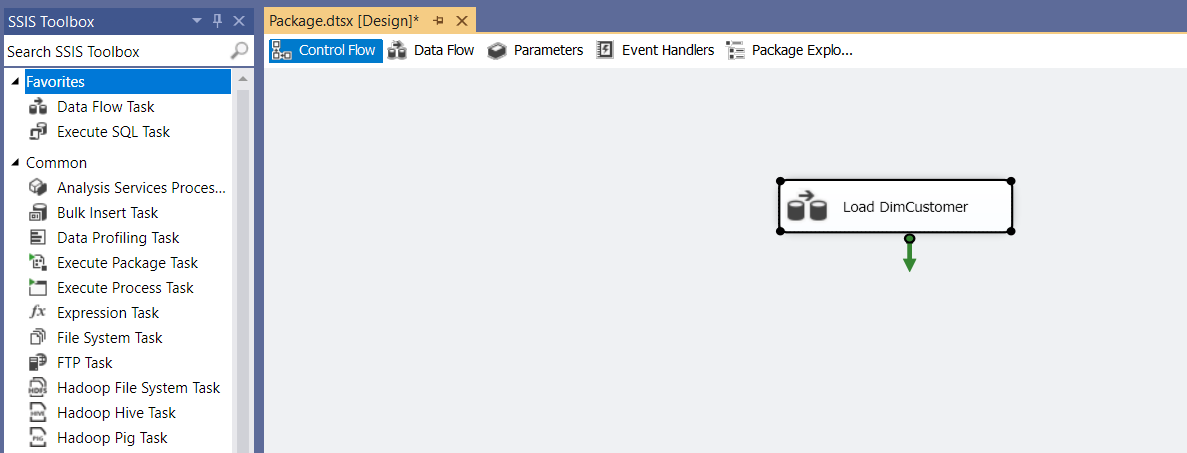
Credit INT,

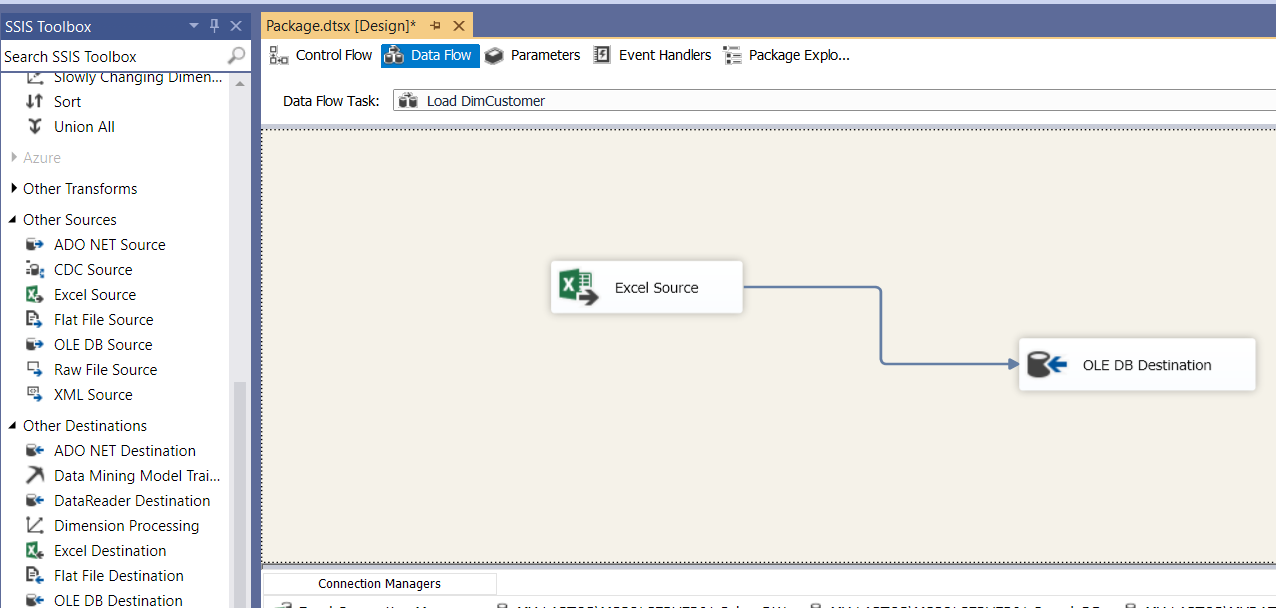
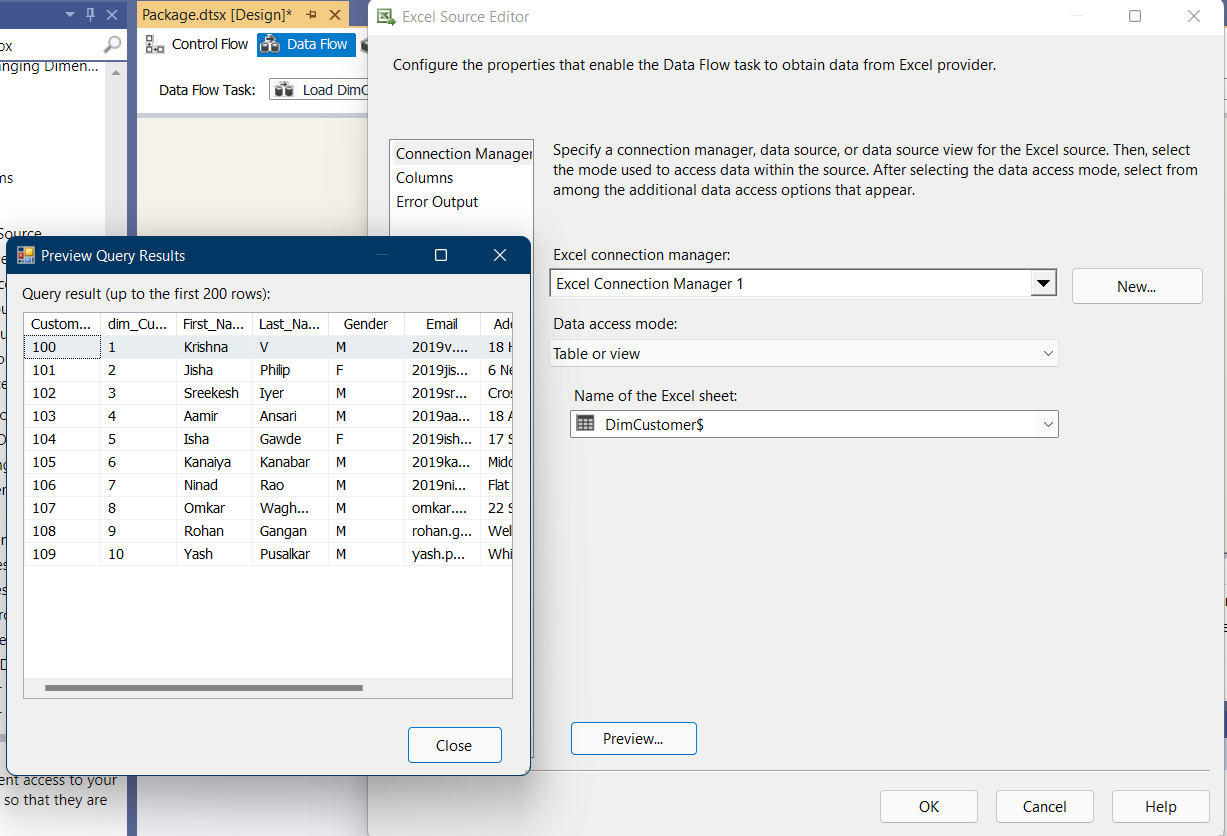
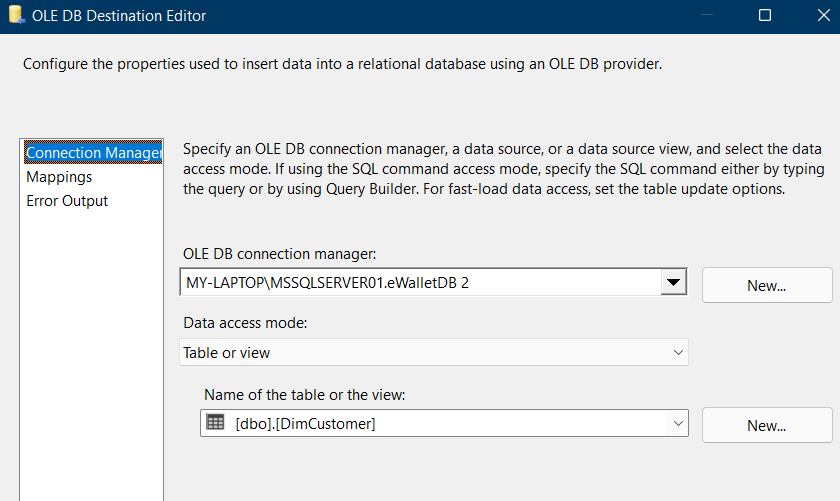
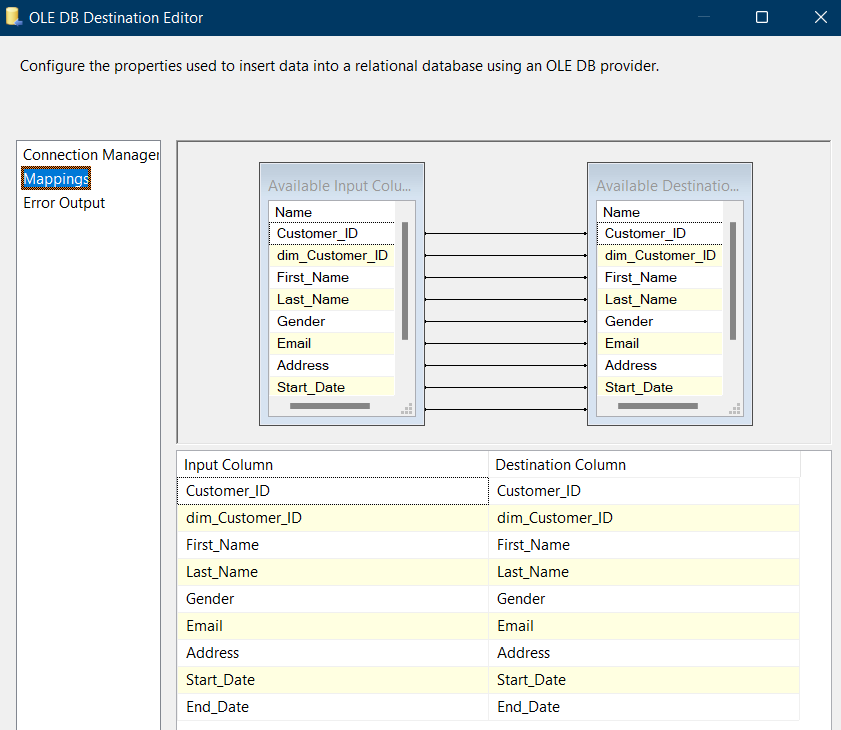
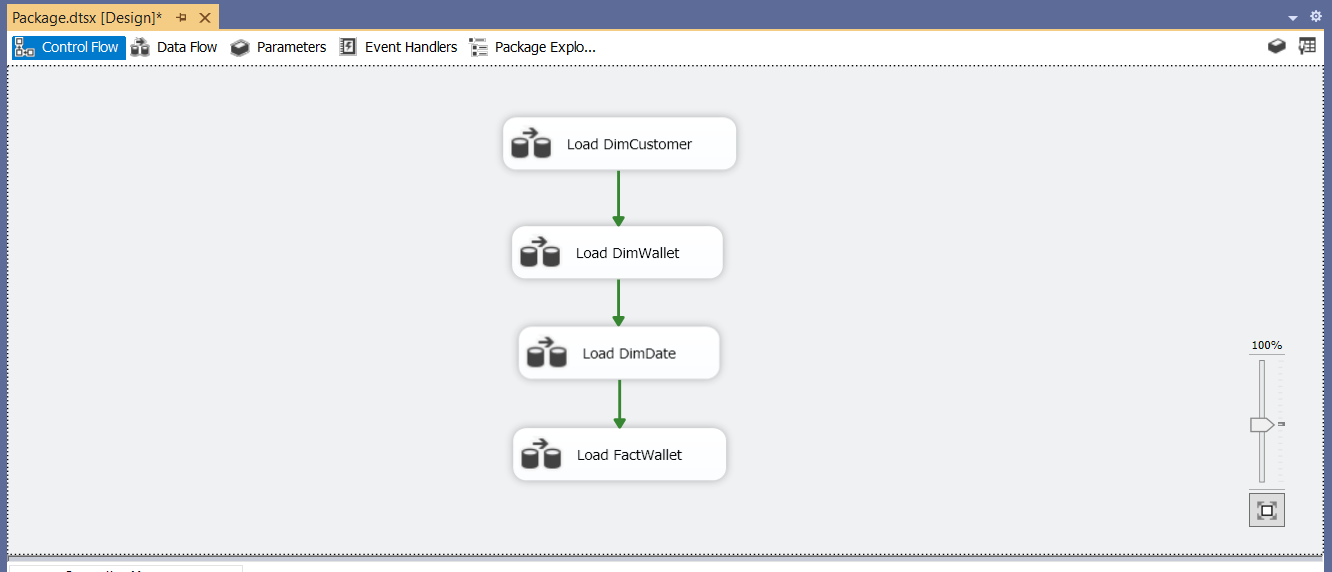
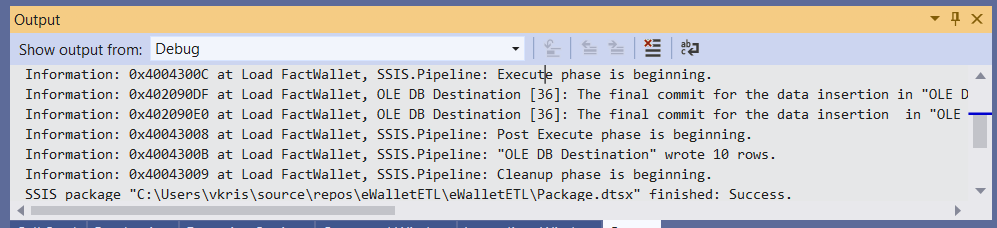
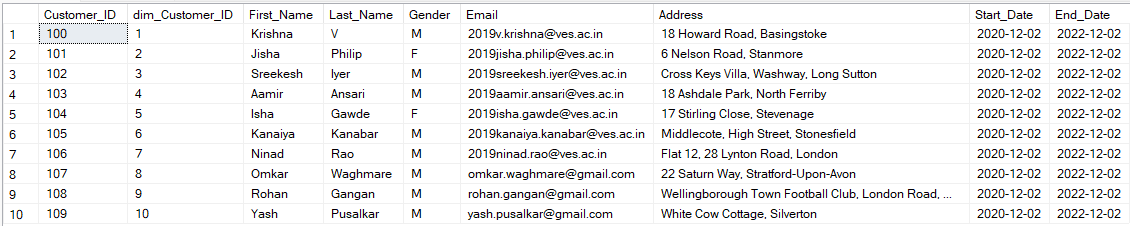
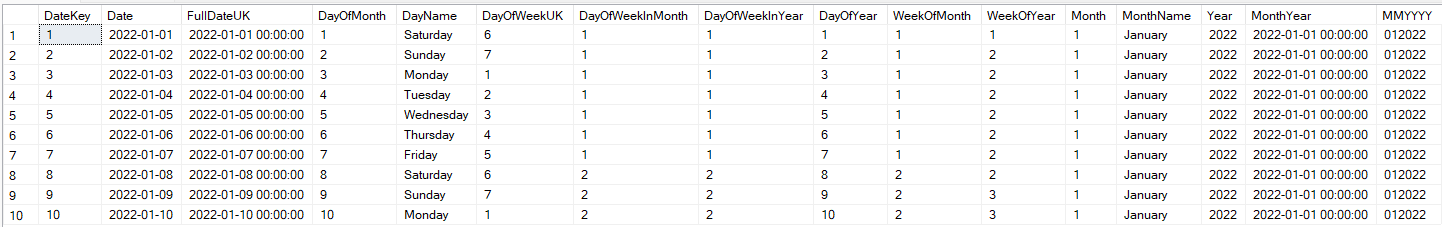
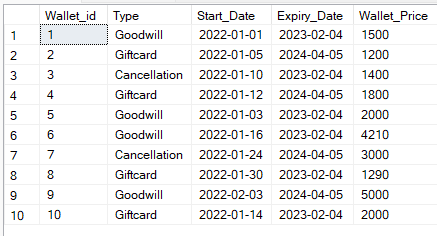
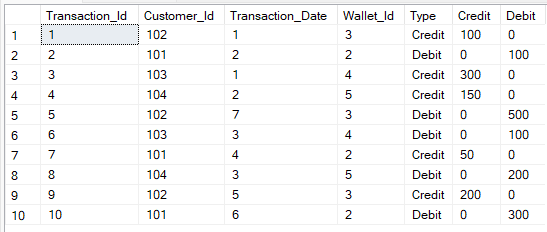
Debit INT

)

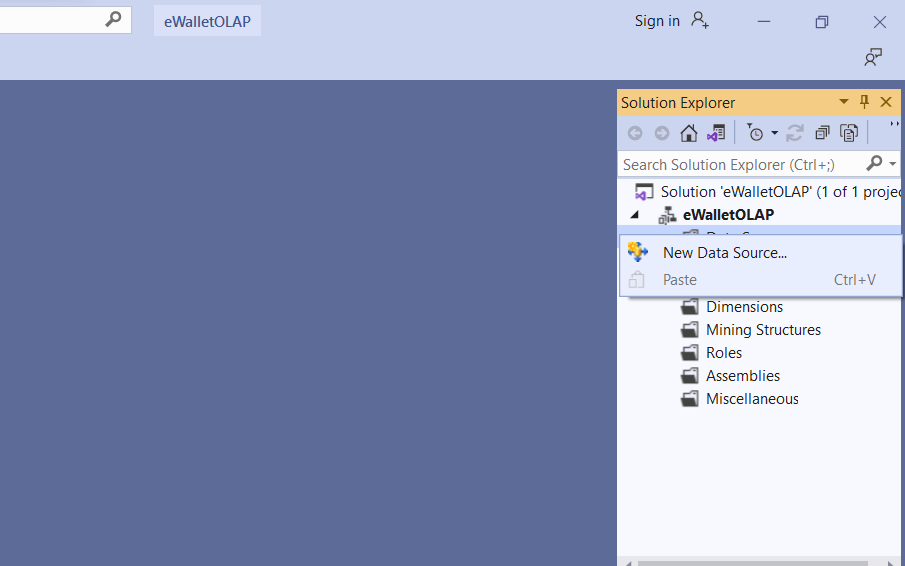
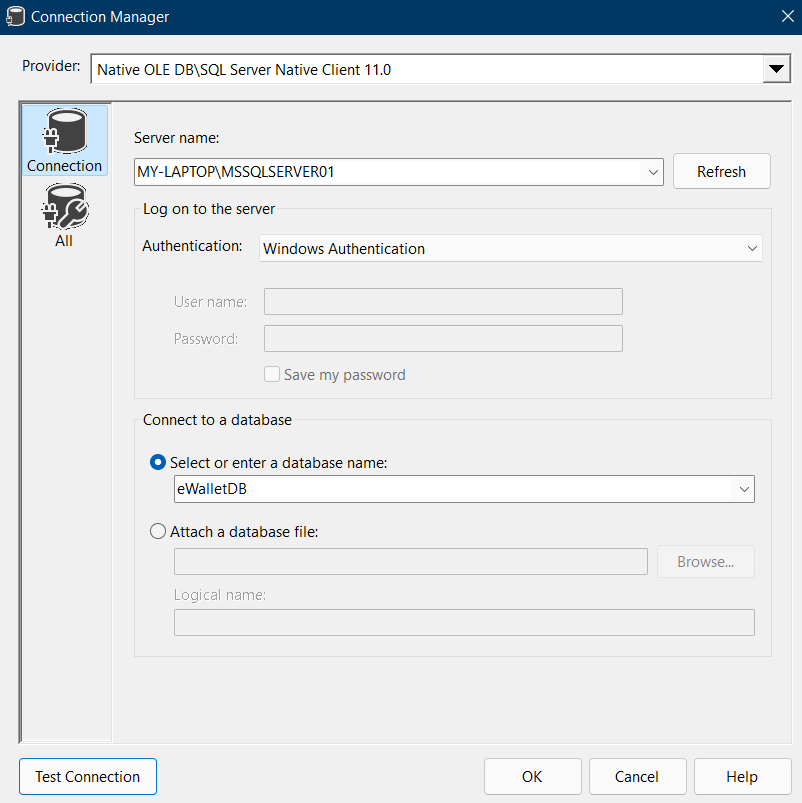
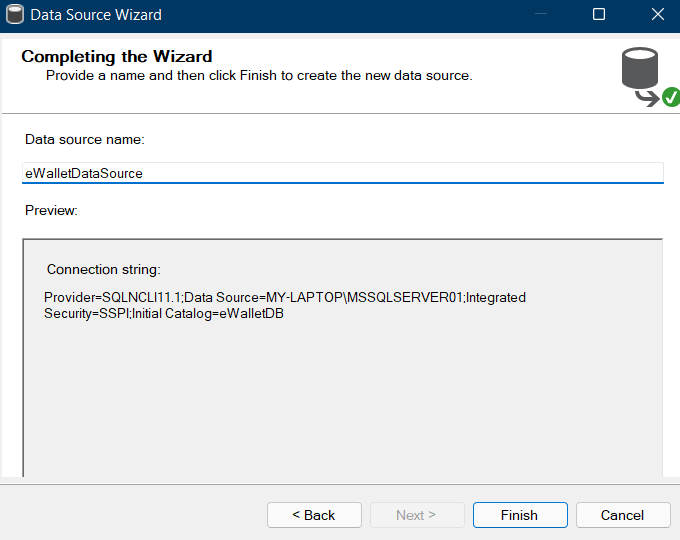
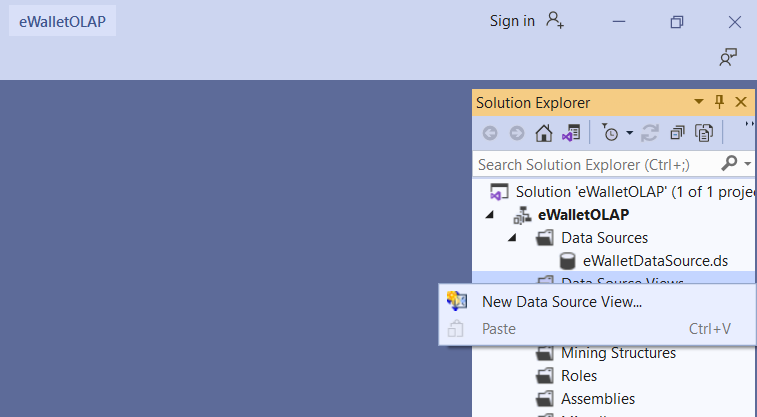
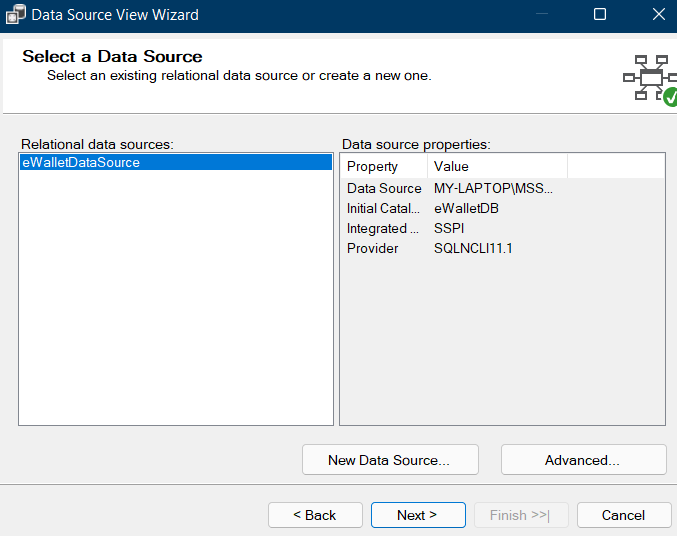
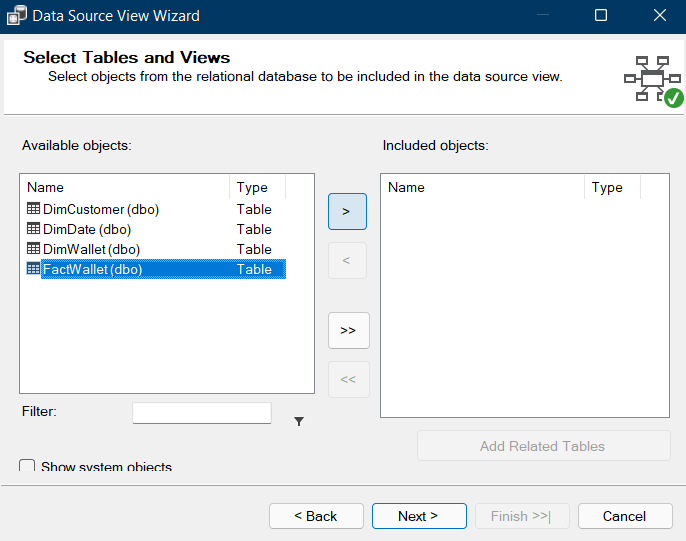
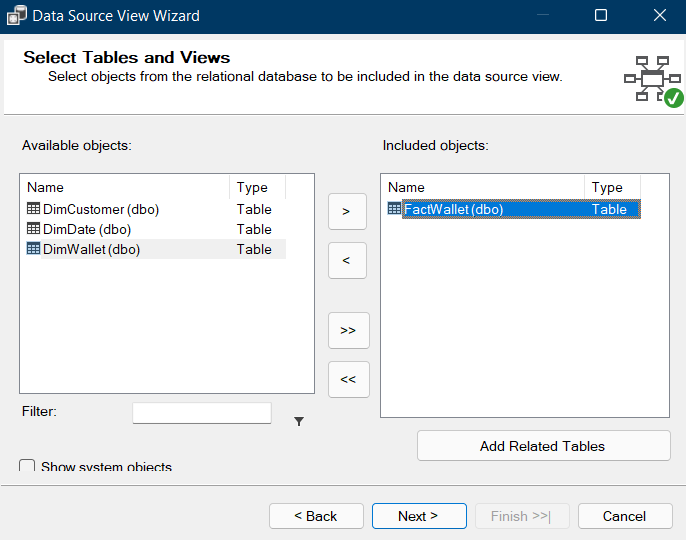
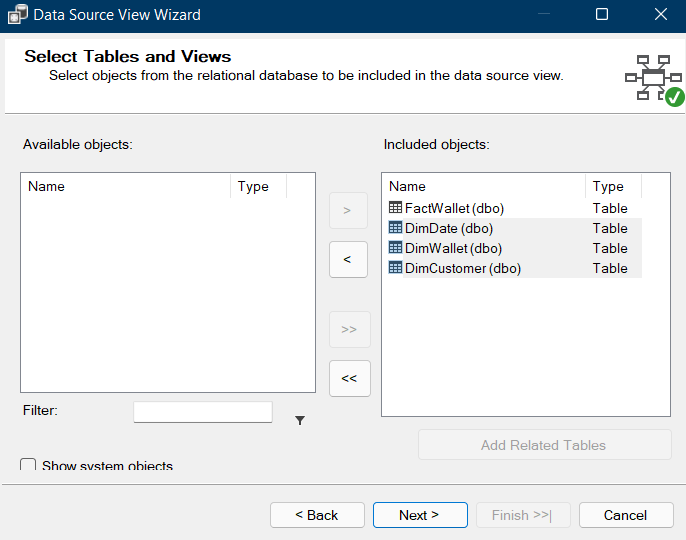
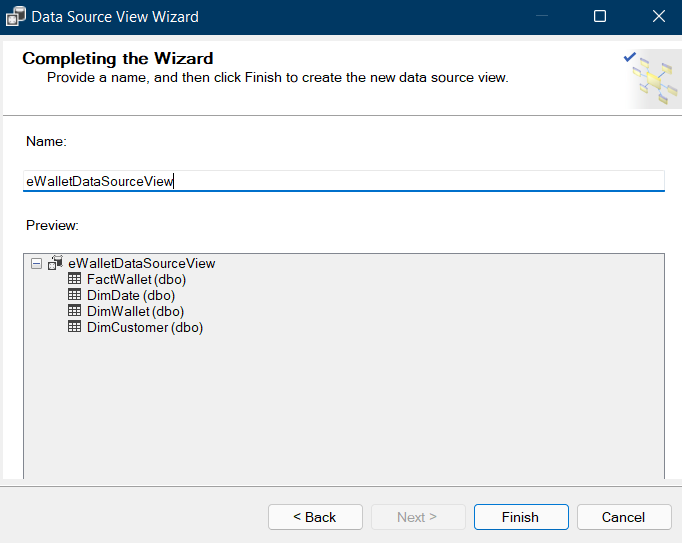
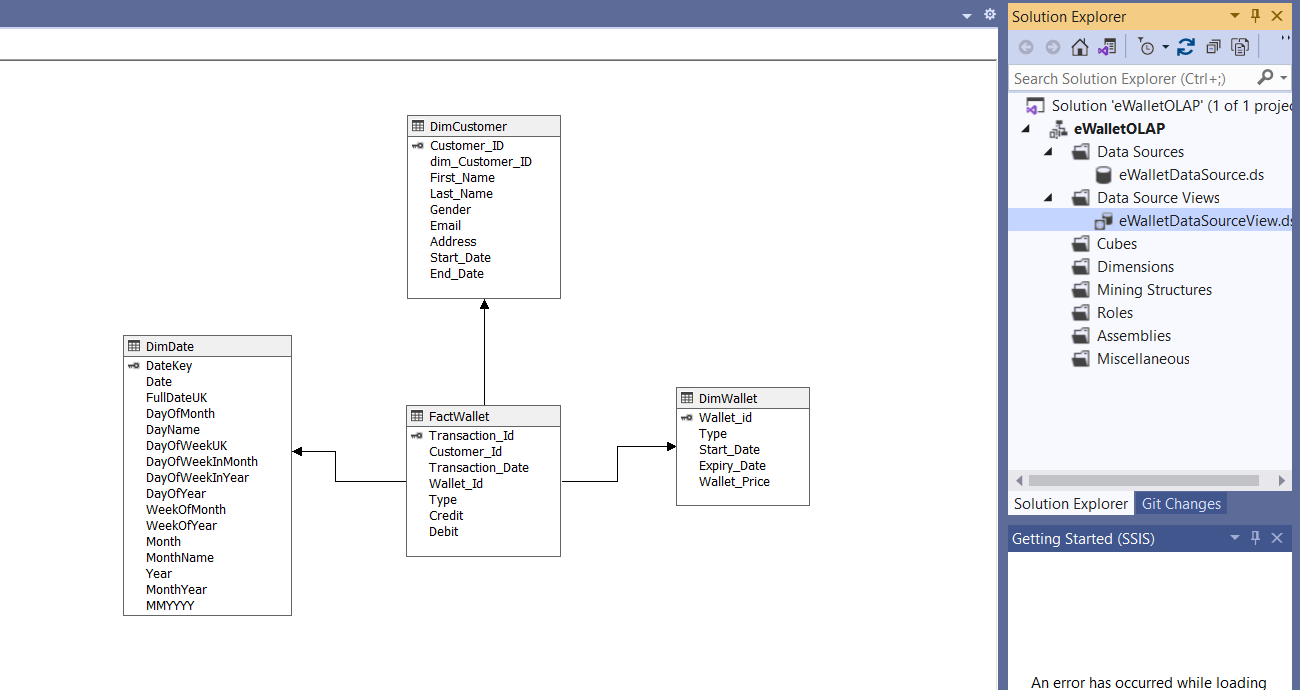
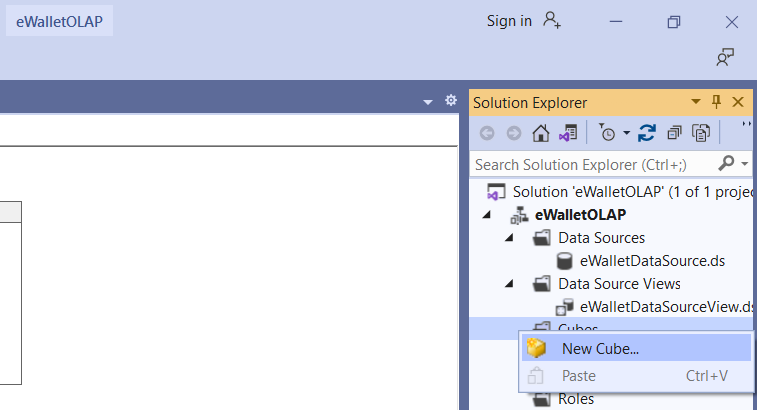
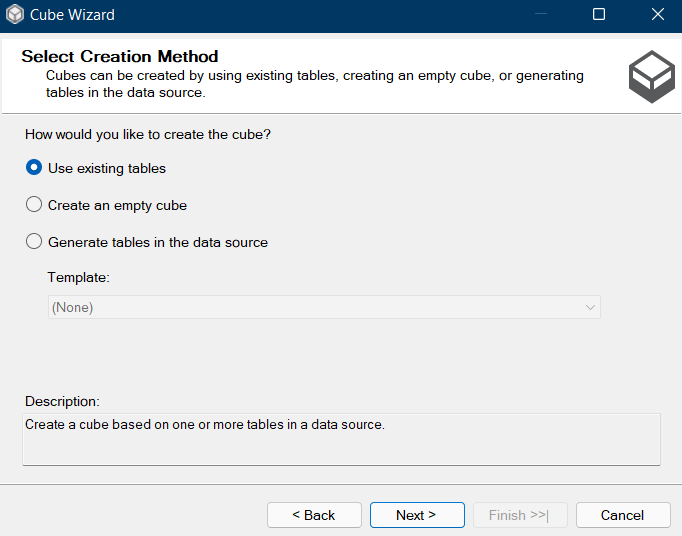
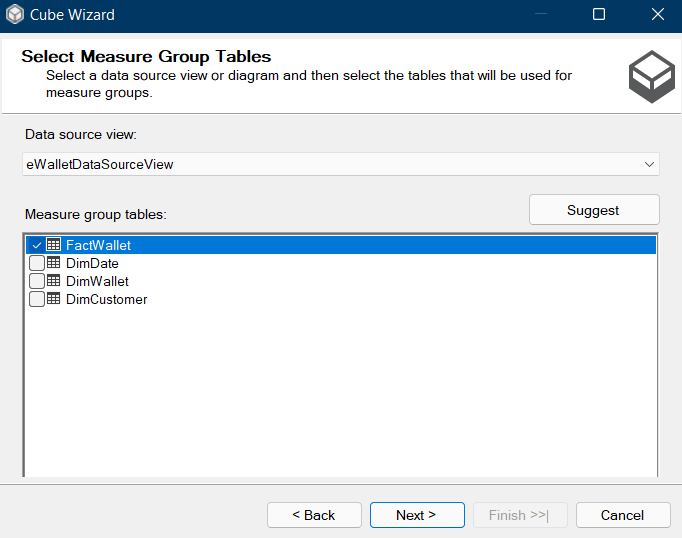
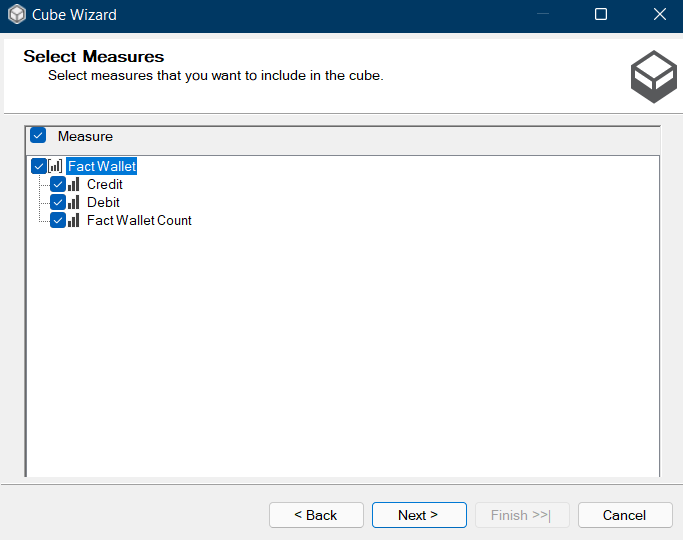
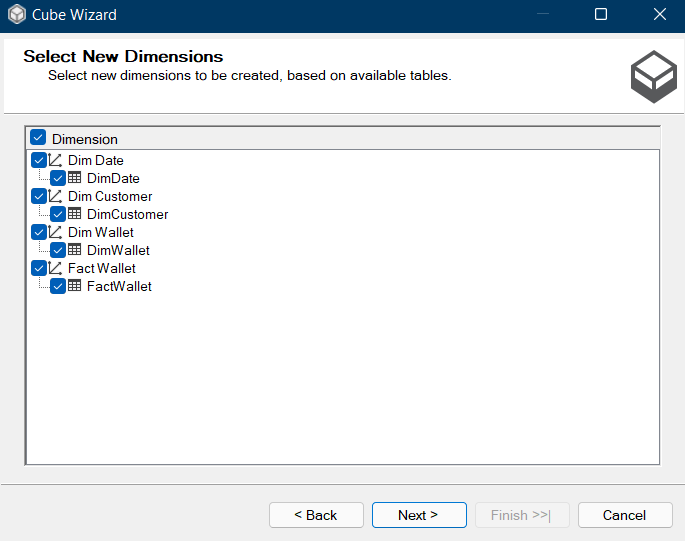
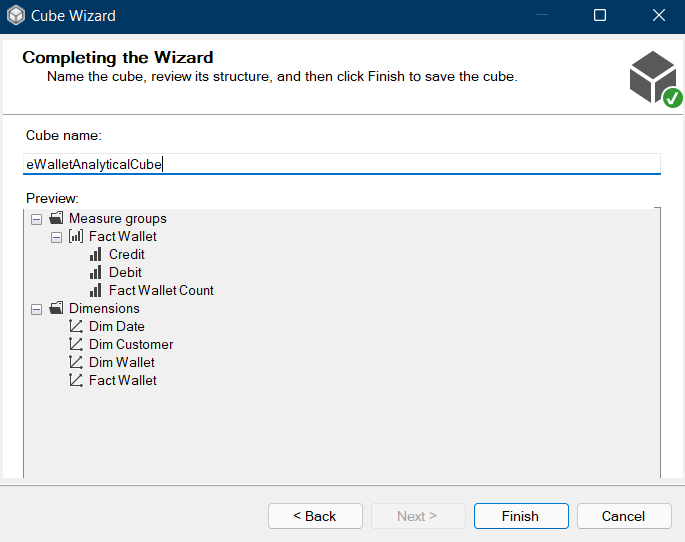
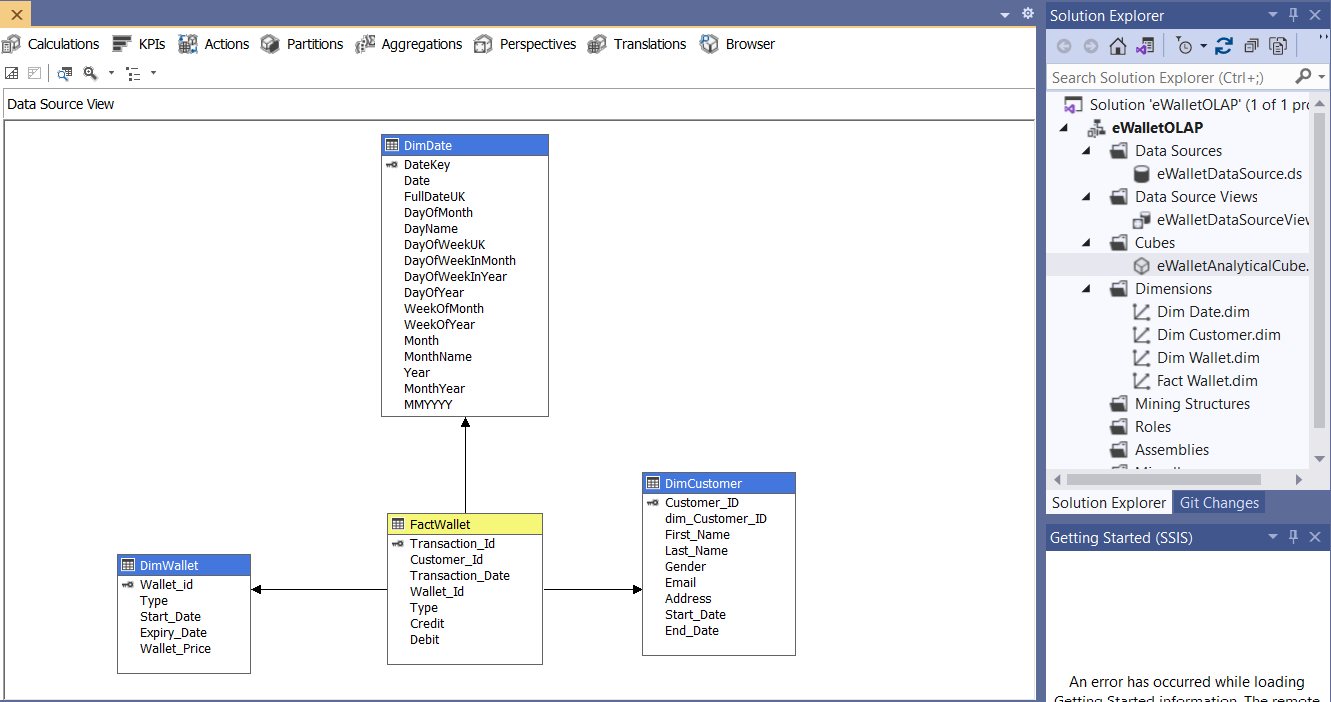
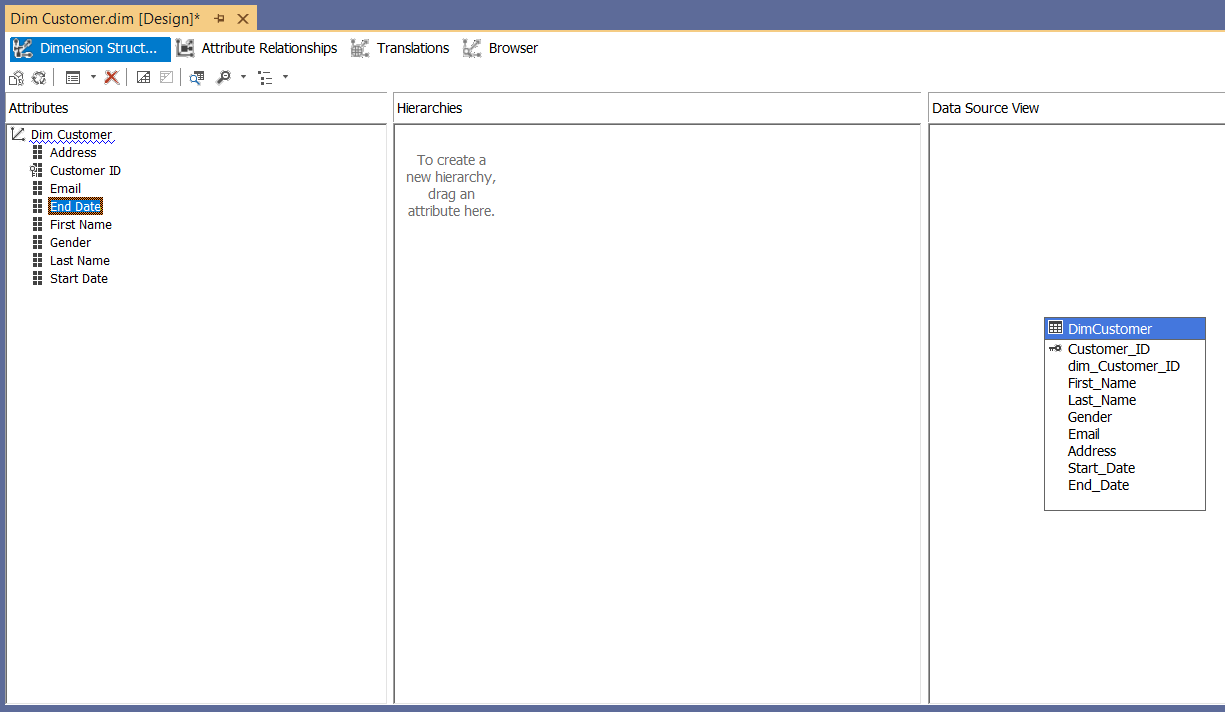
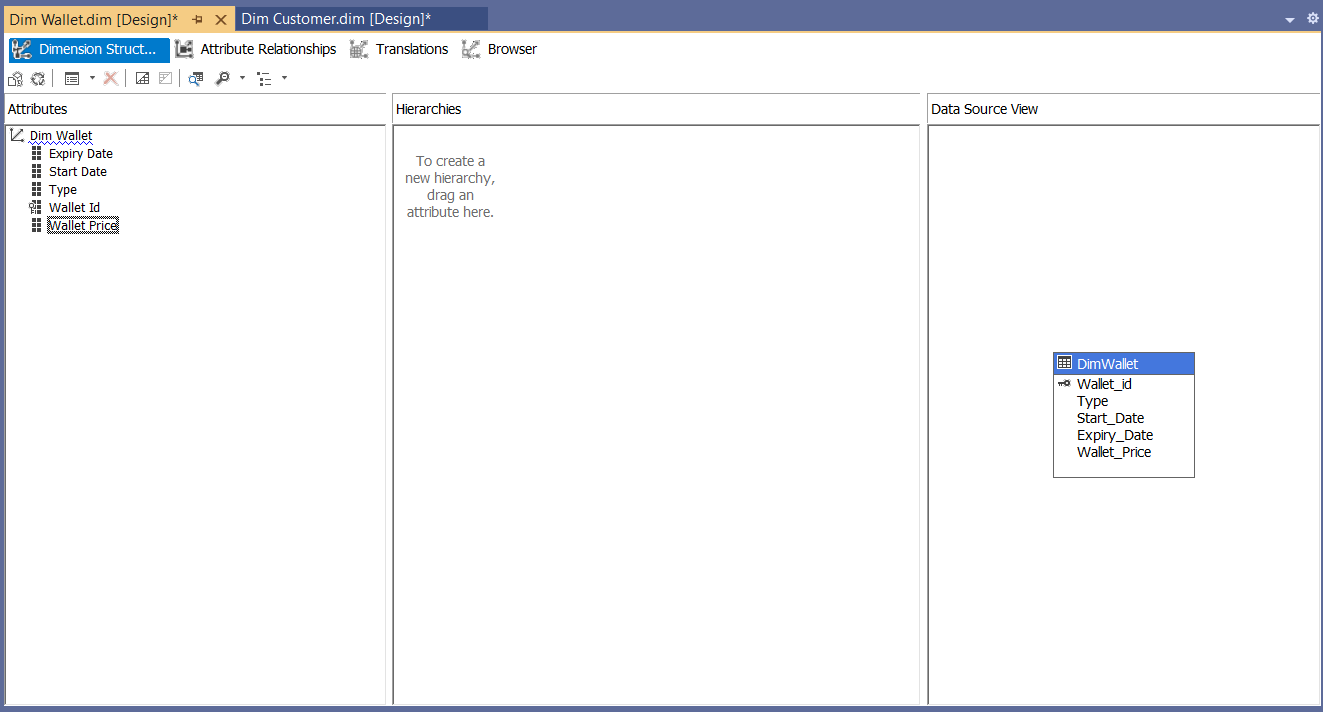
**ETL operations:**

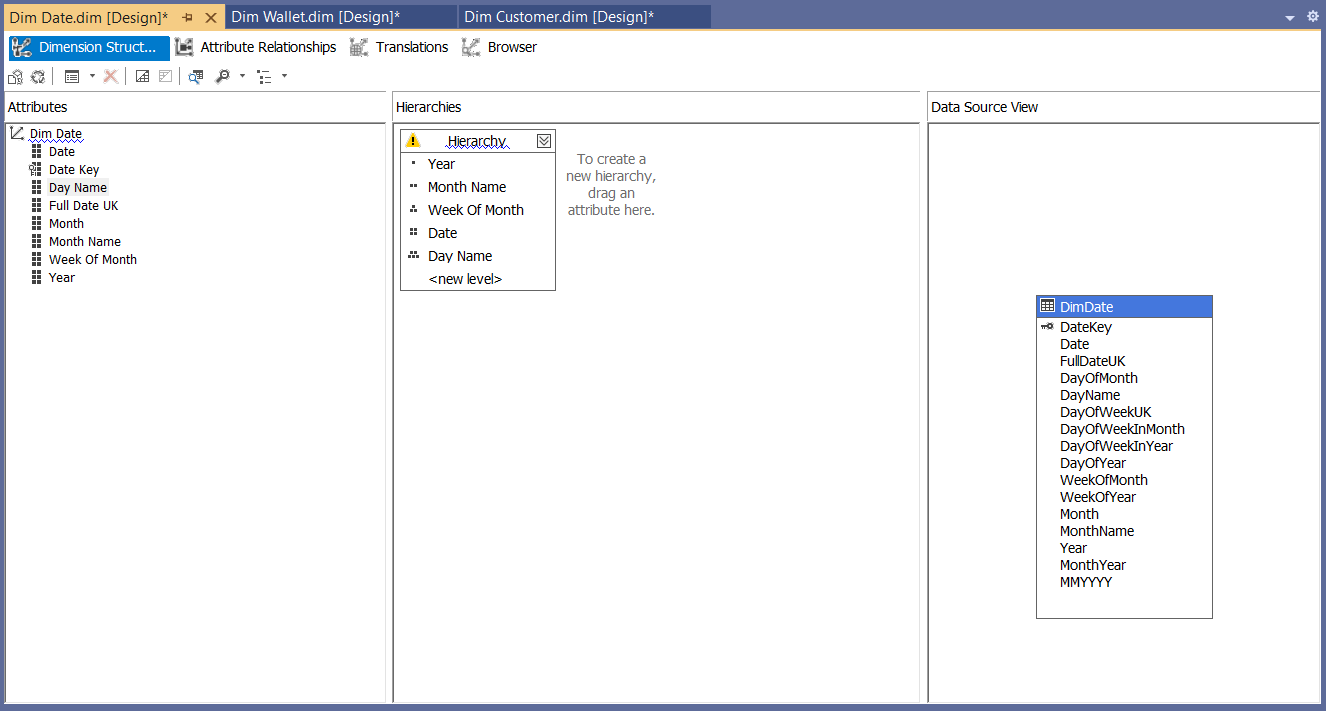
1. Create a Integration Services Project named "eWalletETL" on Visual Studio
2. Drag and Drop a Data Flow Task and rename it to "Load DimCustomer"



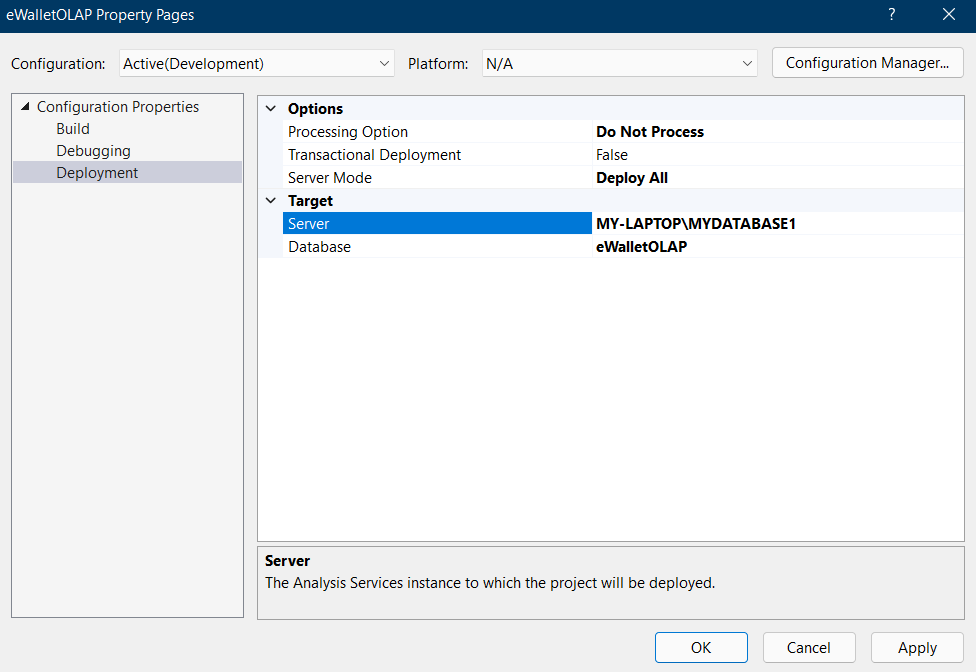
1. Double click on this Data Flow Task. It takes us to the Data Flow tab. Drag and drop "Excel Source" and "OLE DB Destination" from the left tool box. Connect the source to the destination.  
   
2. Double click on the Excel Source and add a new Excel connection Manager by browsing and selecting the excel sheet. Select the appropriate sheet name and click on Preview for verifying the connection.  
   
3. Double click on OLE DB Destination. Create a new connection by providing the SQL Server instance name, and selecting the correct database and table name.  
   
4. Click on Mappings to verify the correct input of the values from the excel sheet  
   
5. Similarly create new data flow tasks to load other tables of the database. And connect them on the Control Flow page.  
   
6. Click on Start in the control flow menu. After successful execution the output window shows the following log:  
   
7. Now we can check the inserted data in the SQL Server database.   
   DimCustomer:  
     
     
   DimDate:  
     
     
   DimWallet  
     
     
   FactWallet  
   

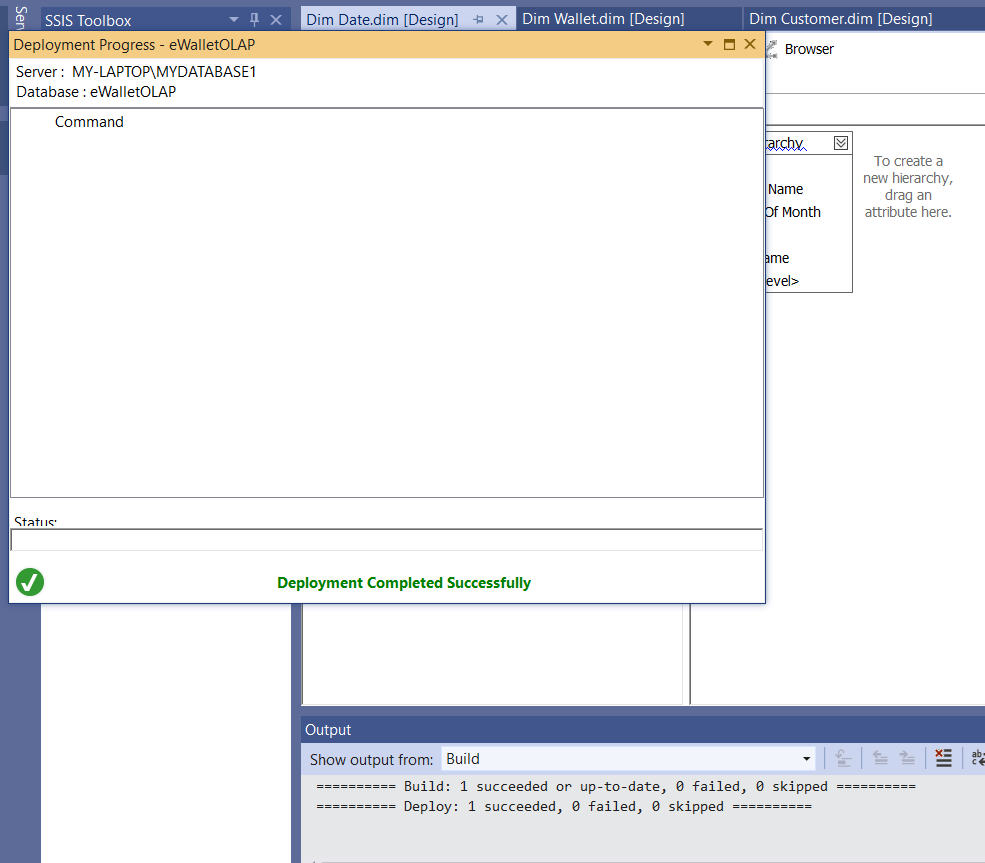
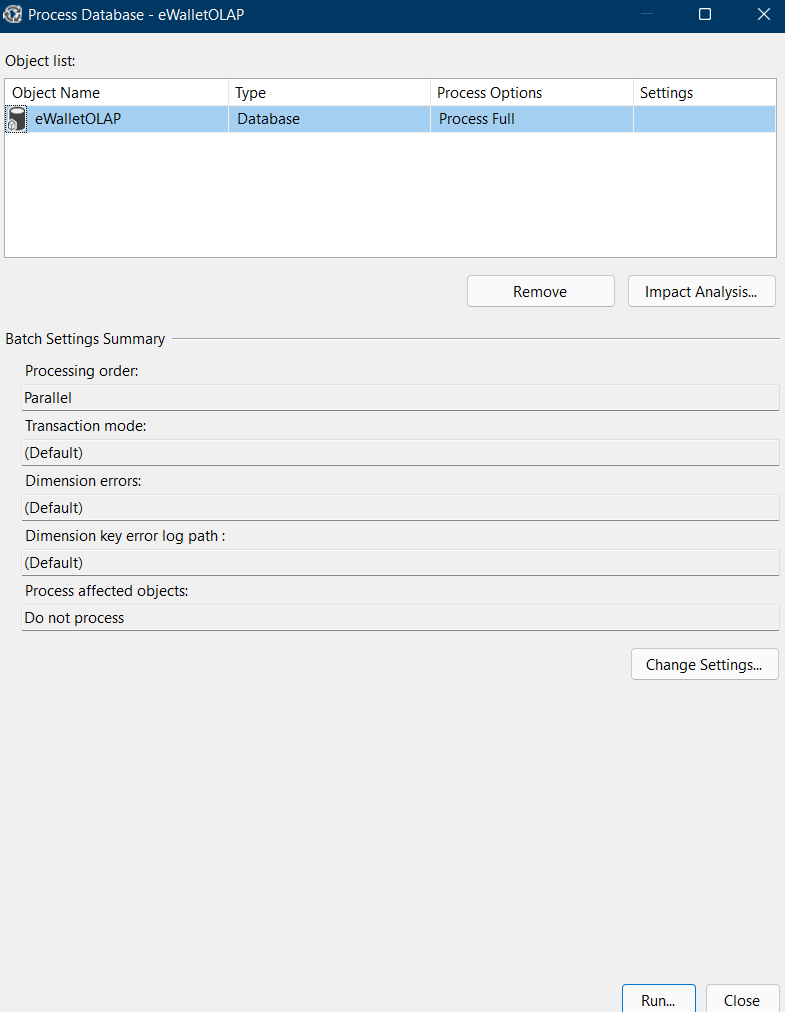
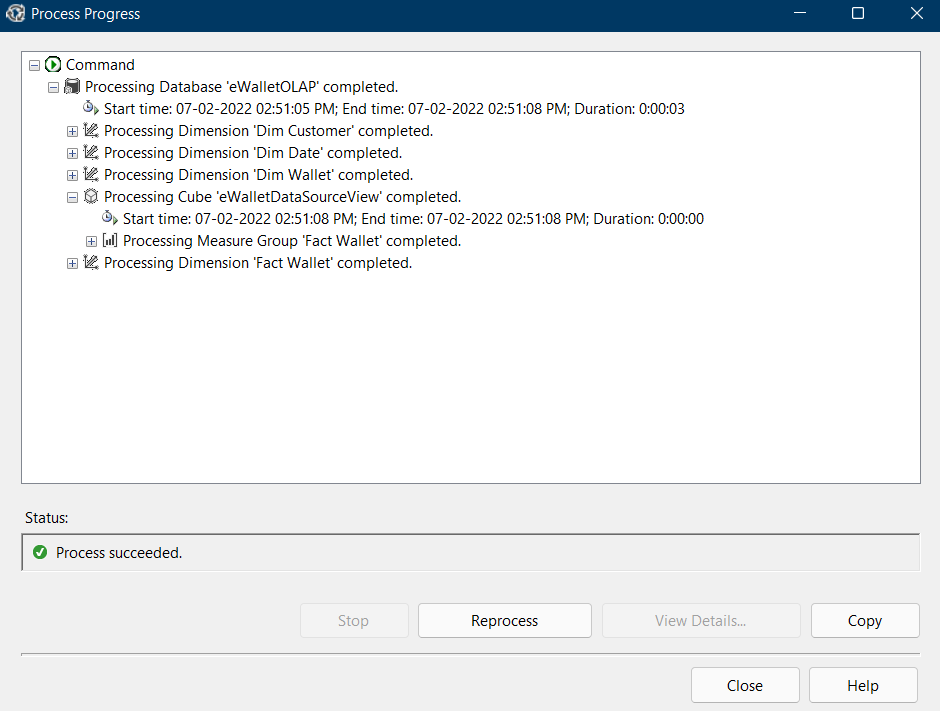
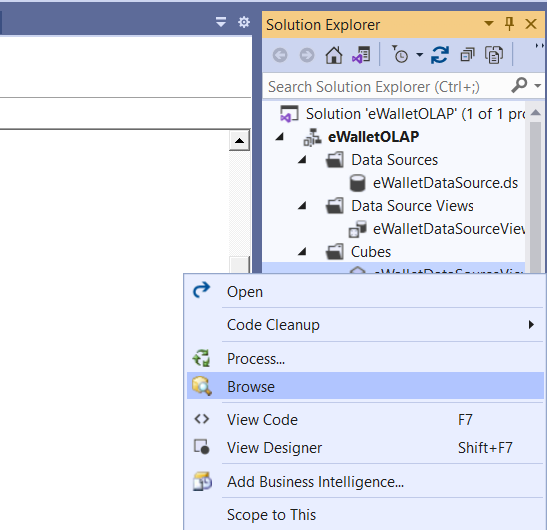
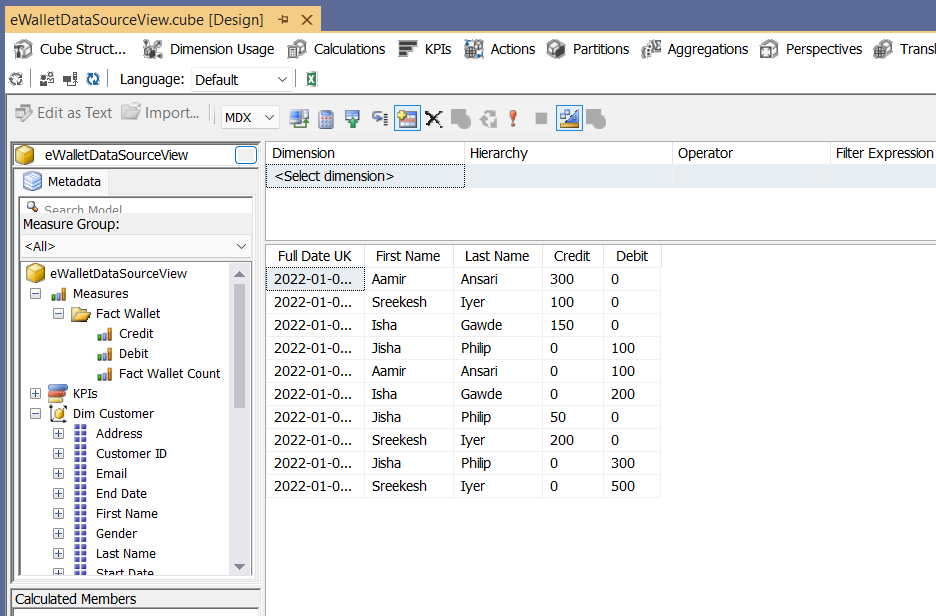
**Cube Setup:**

1. Create a new "Analysis Services Multidimensional and Data Mining Project" with the name "eWalletOLAP" on Visual Studio.
2. In Solution Explorer, Right click on Data Source -> Click New Data Source  
   
3. Click on Next. Click on New Button and create a new SQL Server Connection.   
   
4. Select Connection created in Data Connections-> Click Next. Select Option Inherit. Assign Data Source Name -> Click Finish  
   
5. In the Solution Explorer, Right Click on Data Source View -> Click on New Data Source View.  
   
6. Click Next. Select Relational Data Source we have created previously-> Click Next  
   
7. First move your Fact Table to the right side to include in the objects list. Select FactWallet table -> Click on Arrow Button to move the selected object to the Right Pane.  
   
8. Now to add dimensions which are related to your Fact Table. Select Fact Table in Right Pane (FactWallet) -> Click On Add Related Tables  
   
9. It will add all associated dimensions to your Fact table as per relationship specified in your SQL DW (Sales\_DW). Click Next.  
   
10. Assign appropriate name (eWalletDataSourceView) -> Click Finish  
    
11. Double click on the data source view we created.  
    
12. In Solution Explorer -> Right Click on Cube-> Click New Cube  
    
13. Click Next. Select Option Use existing Tables -> Click Next.   
    
14. Select Fact Table Name from Measure Group Tables (FactWallet) -> Click Next  
    
15. Choose Measures from the List which you want to place in your Cube --> Click Next  
    
16. Select All Dimensions here which are associated with your Fact Table-> Click Next  
    
17. Assign Cube Name (eWalletAnalyticalCube) -> Click Finish  
    
18. Now your Cube is ready, you can see the newly created cube and dimensions added in your solution explorer.  
    
19. In Solution Explorer, double click on dimension Dim Customer -> Drag and Drop first name, last name, and all other useful columns from Table in Data Source View and Add in Attribute Pane at left side.  
    
20. Do the same for DimWallet table also.  
    
21. Double click On Dim Date dimension -> Drag and Drop Fields from Table shown in Data Source View to Attributes-> Drag and Drop attributes from leftmost pane of attributes to middle pane of Hierarchy.

Drag fields in sequence from Attributes to Hierarchy window.  


1. To Deploy the cube, right click on project name eWalletOLAP -> Click Properties.
2. Set Deployment Properties First

In Configuration Properties, Select Deployment-> Assign Your SQL Server Instance Name Where Analysis Services Is Installed (MY-LAPTOP\MYDATABASE1) (Machine Name\Instance Name) -> Choose Deployment Mode Deploy All as of now ->Select Processing Option Do Not Process -> Click OK  


1. In Solution Explorer, right click on Project Name (eWalletOLAP) -- > Click Deploy. Once Deployment will finish, you can see the message Deployment Completed in deployment Properties window.  
   
2. In Solution Explorer, right click on Project Name (eWalletOLAP) -- > Click Process. Click on Run button to process the Cube.  
   
3. Once processing is complete, you can see Status as Process Succeeded -->Click Close to close both the open windows for processing one after the other.  
   
4. Browse the Cube for Analysis:  
   In Solution Explorer, right click on Cube Name (eWalletAnalyticalCube) -- > Click Browse  
   
5. OLAP Querying using Cube:  
   Drag and drop measures into Detail fields, & Drag and Drop Dimension Attributes in Row Field or Column fields. Then click on "Execute query…". This generates the following MDX Query and displays the output:  
     
   SELECT NON EMPTY { [Measures].[Credit], [Measures].[Debit] } ON COLUMNS, NON EMPTY { ([Dim Date].[Full Date UK].[Full Date UK].ALLMEMBERS \* [Dim Customer].[First Name].[First Name].ALLMEMBERS \* [Dim Customer].[Last Name].[Last Name].ALLMEMBERS ) } DIMENSION PROPERTIES MEMBER\_CAPTION, MEMBER\_UNIQUE\_NAME ON ROWS FROM [eWalletDataSourceView] CELL PROPERTIES VALUE, BACK\_COLOR, FORE\_COLOR, FORMATTED\_VALUE, FORMAT\_STRING, FONT\_NAME, FONT\_SIZE, FONT\_FLAGS  
   

**OLAP Operations:**

**Regular query**

select c.First\_Name, c.Last\_Name, fw.Type, fw.Credit, fw.Debit, d.Date, d.Year, d.MonthName, d.DayOfMonth

from FactWallet as fw, DimDate as d, DimCustomer as c

where fw.Transaction\_Date = d.DateKey and fw.Customer\_Id = c.Customer\_ID ;

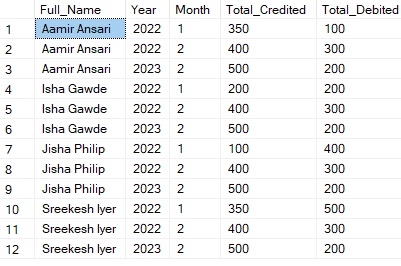
**Drill Down**

SELECT c.First\_Name +' '+ c.Last\_Name AS Full\_Name, d.Year, d.Month, sum(fw.Credit) AS Total\_Credited, sum(fw.Debit) AS Total\_Debited

FROM FactWallet AS fw, DimDate AS d, DimCustomer AS c

WHERE fw.Transaction\_Date = d.DateKey and fw.Customer\_Id = c.Customer\_ID

GROUP BY c.First\_Name +' '+ c.Last\_Name, d.Year, d.Month;



**Roll Up**

SELECT c.First\_Name +' '+ c.Last\_Name AS Full\_Name, d.Year, sum(fw.Credit) AS Total\_Credited, sum(fw.Debit) AS Total\_Debited

FROM FactWallet AS fw, DimDate AS d, DimCustomer AS c

WHERE fw.Transaction\_Date = d.DateKey and fw.Customer\_Id = c.Customer\_ID

GROUP BY c.First\_Name +' '+ c.Last\_Name, d.Year;



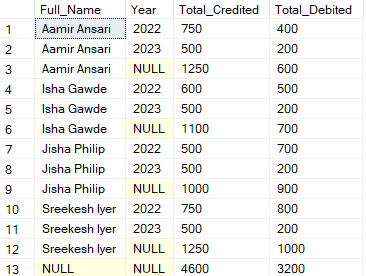
We can also use the preexisting ROLLUP function, which will give a more elaborate set of aggregates.

SELECT c.First\_Name +' '+ c.Last\_Name AS Full\_Name, d.Year, sum(fw.Credit) AS Total\_Credited, sum(fw.Debit) AS Total\_Debited

FROM FactWallet AS fw, DimDate AS d, DimCustomer AS c

WHERE fw.Transaction\_Date = d.DateKey and fw.Customer\_Id = c.Customer\_ID

GROUP BY ROLLUP(c.First\_Name +' '+ c.Last\_Name, d.Year);



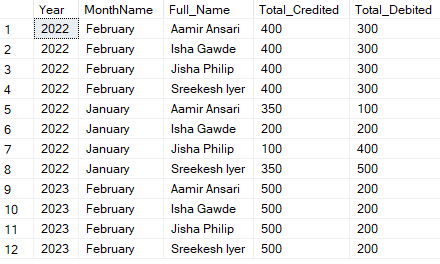
**Pivot**

SELECT d.Year,d.MonthName, c.First\_Name +' '+ c.Last\_Name AS Full\_Name, sum(fw.Credit) AS Total\_Credited, sum(fw.Debit) AS Total\_Debited

FROM FactWallet AS fw, DimDate AS d, DimCustomer AS c

WHERE fw.Transaction\_Date = d.DateKey and fw.Customer\_Id = c.Customer\_ID

GROUP BY d.Year, d.MonthName, c.First\_Name +' '+ c.Last\_Name;



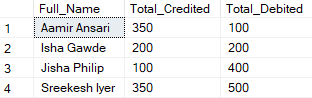
**Slice**

SELECT c.First\_Name +' '+ c.Last\_Name AS Full\_Name, sum(fw.Credit) AS Total\_Credited, sum(fw.Debit) AS Total\_Debited

FROM FactWallet AS fw, DimDate AS d, DimCustomer AS c

WHERE fw.Transaction\_Date = d.DateKey and fw.Customer\_Id = c.Customer\_ID and d.Year=2022 and d.MonthName='January'

GROUP BY c.First\_Name +' '+ c.Last\_Name;



**Dice**

SELECT c.First\_Name +' '+ c.Last\_Name AS Full\_Name, d.Year, sum(fw.Credit) AS Total\_Credited

FROM FactWallet AS fw, DimDate AS d, DimCustomer AS c

WHERE fw.Transaction\_Date = d.DateKey and fw.Customer\_Id = c.Customer\_ID

and d.Year>=2022 and d.Year<=2023 and c.Customer\_ID >=101 and c.Customer\_ID <103

GROUP BY c.First\_Name +' '+ c.Last\_Name, d.Year;



**Conclusion:**

Thus we have learnt about basics of data warehousing, and design schemas. We also learnt about ETL and OLAP operations and applied them on our data warehouse as a case study on eWallet.