

SSIS Exploration Document

Inventory and Sales Management System

Project business intelligence

Avishay Aknin

**CONTENT**

[1 Document Management 3](#_Toc161651523)

[1.1 Project Scope & Objective 3](#_Toc161651524)

[1.2 Project Content 3](#_Toc161651525)

[2 Technical Specifications 4](#_Toc161651526)

[2.1 Prerequisites – the involved systems during the process 4](#_Toc161651527)

[2.2 Solution Architecture 4](#_Toc161651528)

[3 Functional Specification 4](#_Toc161651529)

[3.1 Source to Target and ERD models 4](#_Toc161651530)

[3.2 Detailed description of ETL Process 5](#_Toc161651531)

[3.3 in the DWH Tables + History tables 17](#_Toc161651532)

# Document Management

# Project Scope & Objective

The purpose of the project is to provide Data Warehouse Solution Creation from AdventureWorks2019 Database for AvishayDB, A company that supplies products and conducts business transactions.

This project aims to establish a comprehensive BI solution transferring data from the AdventureWorks2019 Database for SSIS\_DB. The solution will contain summarized data tables, with a focus on sales data, employee records, customer information, product details.

Moreover, the project will solve the following issues:

* + - Providing a unified approach to organized and well-suited data for significant decision-making scenarios.
    - Integrating data from different sources to be further used by reporting activities. The data is updated in real time, which makes it conveniently used for automated actions.

# Project Content

In this project, we will build a Data Mart that will include information about many aspects of a real-world business order detailing from end-to-end perspective.

1. Data Cleaning and Preparation: Before the analysis starts, we filter our data via data cleaning method and preparation to verify their quality and consistency.
2. Main summary tables that will be built for the company's demands:
   * **Sales\_Fact\_table**– Contains information about all the orders, including dates, pricing and quantity per customer from the transactions of the transactional database.

* + **Stores\_Dim**– Information about the company's Stores**.**
  + **Customer\_Dim**– Information about the company's customers.
  + **Products\_dim**- Information about the products that are sold by the company.
  + **Calendar\_dim** - Information about date of transaction .

1. The project will contain measures that will lead into action to the achievement of the project's goal:

##### Sales Transaction orders Department:

The Orders Department will focus on the sales-related data, when the order took place, identified by the customer that made the order, the store that activated the order. This department will contain a full description of the purchase including the productID, its price and the total amount per product in the order.

The department gathers information every time that a new order enters to the system and help the user to follow over the new transactions.

##### Store Department:

Within this department, stores are tasked with initiating orders for specific customers, with each order being documented in the Sales\_Fact\_table table. Relevant stores details are housed in the Stores\_Dim table. Each store is distinguished by a unique identifier (StoreId), establishing a direct connection to the Sales\_Fact\_table table for efficient order management.

##### Customer Department:

A customer can make a new order, all the data about the customer will be stored inside the dimCustomers table – including an identifier (CustomerID) that is connected to the Sales\_Fact\_table table.

##### Production Department:

A table that contains all the manufactured product in the company, every product has a description about it. It includes an identifier (ProductID) which is connected to Sales\_Fact\_table Table and will take part inside the transaction process.

# Technical Specifications

# Prerequisites – the involved systems during the process

|  |  |
| --- | --- |
| **System / Process** | **Explanation** |
| SQL Server | Operational DB – tables – data (SQL Files) |
| SSIS | ETL processes using SSIS in Visual Studio |
| Data Refreshing | Refreshing processes through an attribute of Employees in SSMS |

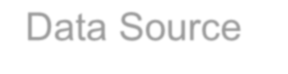
# Source to Target and ERD models



DWH



ETL



# Detailed description of ETL Process

1. **Master Package-**

**Stage One: Data Deletion**

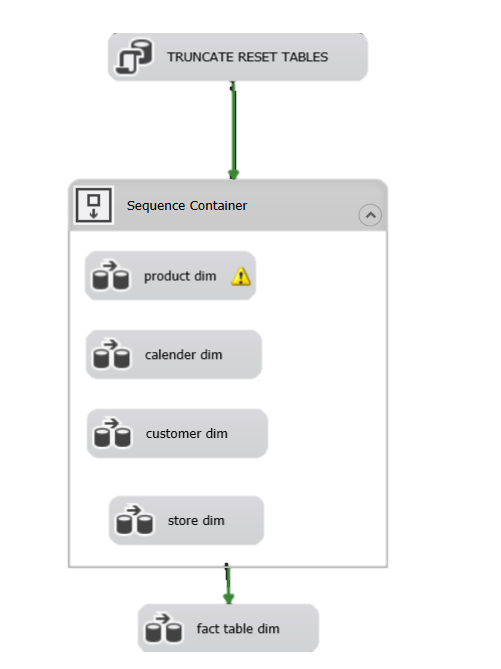
The initial phase involves clearing existing data from dimension and FACT tables to establish a clean environment for subsequent operations. This is achieved by connecting to the database, executing SQL tasks to truncate or delete data from the tables, and ensuring seamless interaction between the SSIS package and the database.

**Stage Two: Data Manipulation**

In this stage, data manipulation occurs, focusing on loading data into dimension tables. A container is introduced for organizational clarity, within which "Data Flow Tasks" are created for each dimension. These tasks extract data from source views and load it into corresponding dimension tables, with field mappings ensuring data alignment.

**Stage Three: Updating the FACT Table**

The final stage revolves around updating the FACT table with processed data from the dimensions. A "Data Flow Task" is employed to transfer data from source views containing all dimensions and the FACT table. Field mappings are established to integrate data seamlessly into the FACT table, maintaining data integrity.

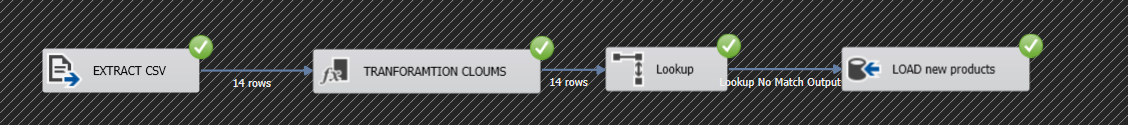


1. **CSV TO DB -**

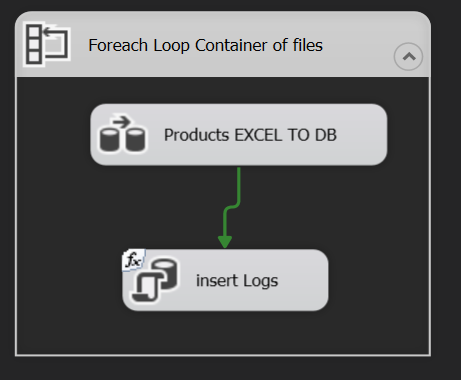
**Product table – insert just the new products**

Loading the csv files: Retailers products(such as spring , tnuva) into Products tables.

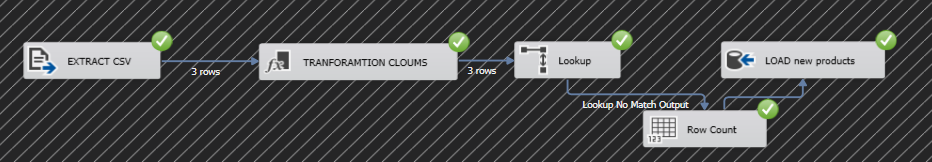
The load action will load ONLY new records that did not appear in the 'products' table before to avoid a case that we load duplicated data. All the new records will be counted.

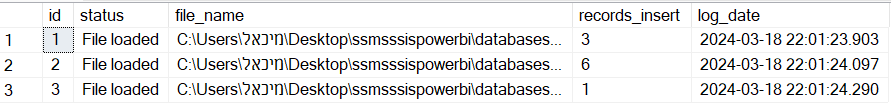


In this loop there is an exchange of the files that contain items of products in Excel files that belong to each company - such as Castro, Tnuva, etc.



In addition, I built documentation of the logs of all the files, if there are new details there will be documentation in the database "LogTable"





3.

**DB TO DWH -**

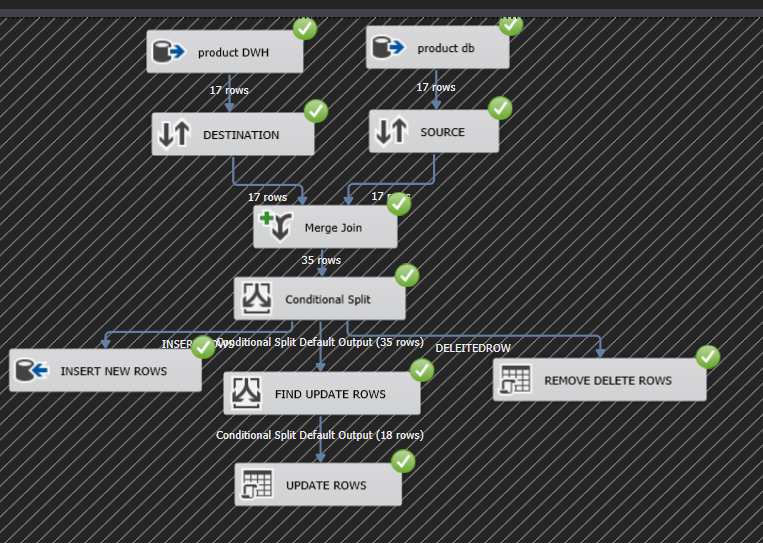
**Product table – insert, delete, update just the relevant products rows**

Packeage aim:

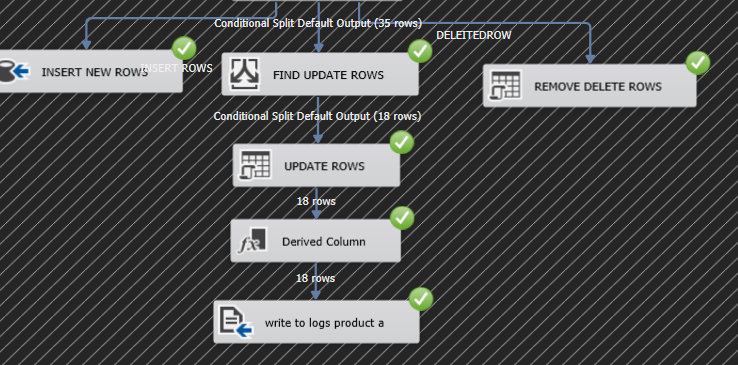
Ensure synchronization of data between a database (db) table and a data warehouse (dwh) table efficiently and accurately using SQL Server Integration Services (SSIS).

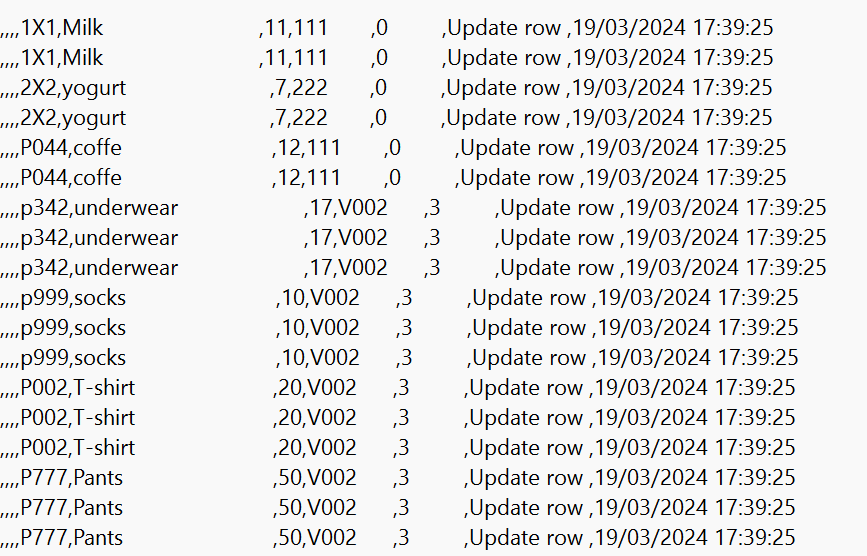
Levels Explained:

* Database and Data Warehouse Setup:
* Create tables in both the database (db) and data warehouse (dwh).
* Data Comparison and Sorting: Employ a Merge Join tool to compare data between the db and dwh tables. Utilize a Conditional Split tool to categorize changes such as new data, updates, or deletions of data.
* Detailed Data Analysis: Conduct in-depth analysis of existing data to identify any discrepancies between the db and dwh tables.
* Verify if the data in both tables aligns accurately.



Furthermore, I have implemented a feature whereby all updated products are logged into a text file. This log file contains a comprehensive list of products that underwent updates on any given day.



This it the logs products.txt file

4.

### The next step is to deploy the Project

The deployed path : /SSISDB/BI33/PRODPROJ\_SSIS\_DB Into SSMS.

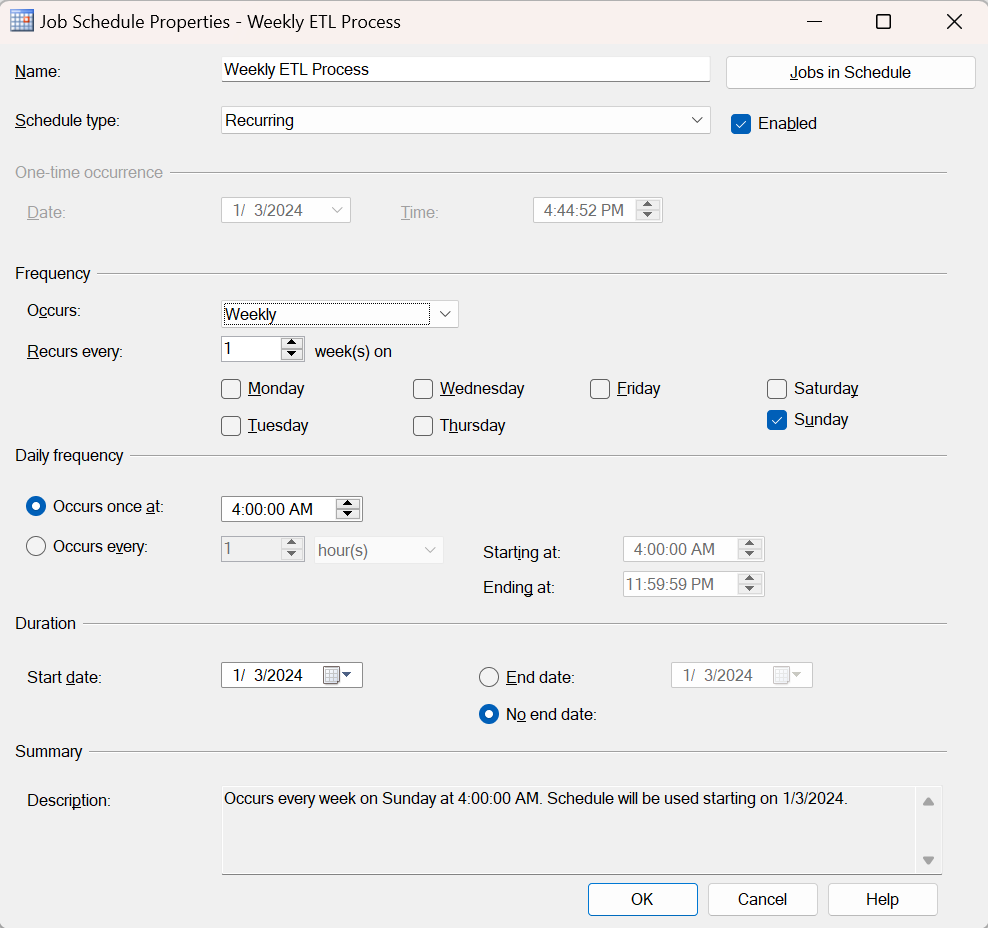
After that I create a job called MrrLoadingJob which will call the next job in line until all the jobs will be done.

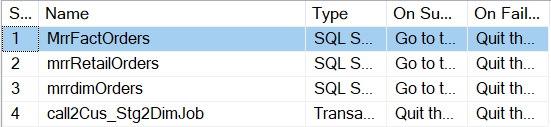
**The jobs are separated into small parts because if an error will occur, we want to detect it in the specific job and be more efficient instead of loading massive amount of data each time**.

This is the chronological order of the jobs:

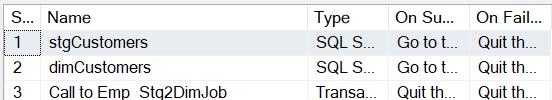
MrrLoadingJob → Cus\_Stg2DimJob → Emp\_Stg2DimJob → Prd\_Stg2DimJob → Rtl\_Stg2DimJob →

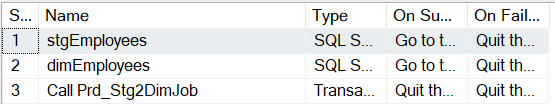
Str\_Stg2DimJob → FactOrders\_Stg2DWHJob → FactRetailOrders\_Stg2DWHJob All the Extract-Transform-Load process will happen every Sunday at 04:00 AM.

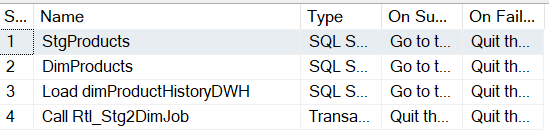


MRRLoadingJob steps:

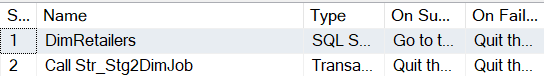
Cus\_Stg2DimJob steps:

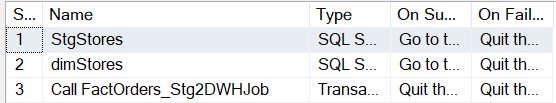


Emp\_Stg2DimJob steps:

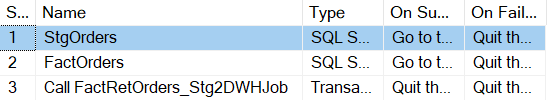
Prd\_Stg2DimJob steps:

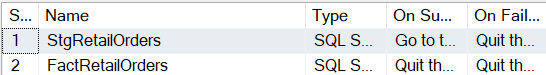
Rtl\_Stg2DimJob steps:



Str\_Stg2DimJob steps:

FactOrders\_Stg2DWHJOB steps:



FactRetailOrders\_Stg2DWHJOB steps:

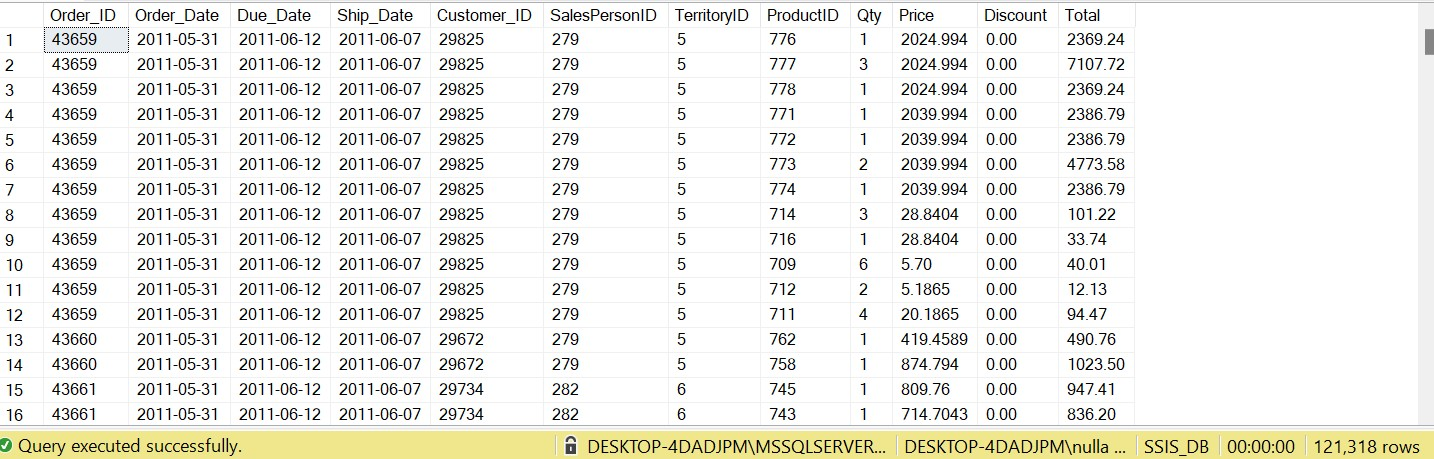
# in the DWH Tables + History tables

Tables in the DWH:

* + - FactOrders:

In the FactOrders table we have the following columns:

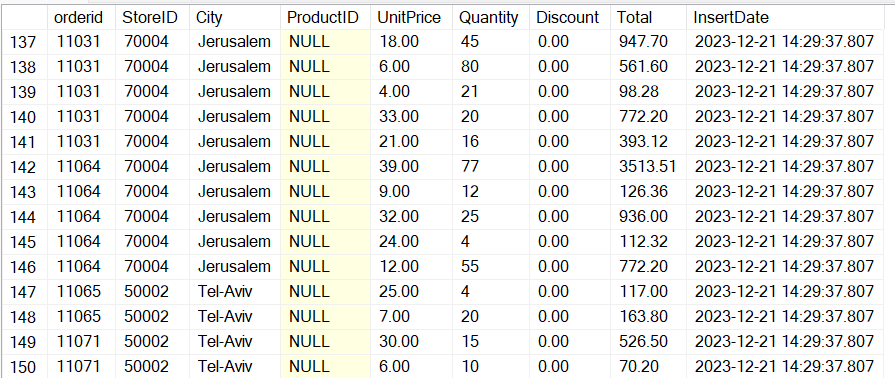
1. Order\_ID: The ID of the order.
2. Order\_Date: The date that the order was recorded.
3. Due\_Date: The date that the order is due to the customer.
4. Ship\_Date: The date that the order was shipped to the customer.
5. Customer\_ID: The ID of the customer. Foreign Key to DimCustomers table.
6. SalesPersonID: The ID of the employee that created the order. Foreign Key to DimEmployees table.
7. TerritoryID: The ID of the place that the order took place at.
8. ProductID: The ID of the purchased product, Foreign Key to DimProducts Table.
9. Qty: The quantity of the selected product related to the specific order.
10. Price: The price of each unit of the purchased product.
11. Discount: The percentage of the discount in the given order for the given product.
12. Total: The total amount that the order was cost per product inside the order, including the unitprice, discount, quantity and an additional VAT payment in a rate of 17%.



* + FactRetailOrders:

In the FactRetailOrders table we have the following columns:

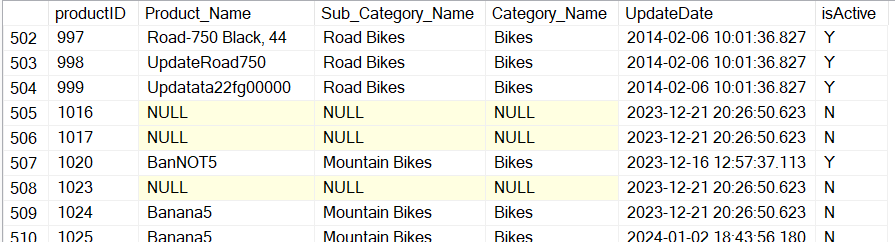
1. OrderID: The ID of the order.
2. Store\_ID: The ID of the store that made the order. Foreign Key to DimStores table
3. City: The city that the order took place at.
4. ProductID: The ID of the purchased product, Foreign Key to DimProducts Table.
5. UnitPrice: The price of each unit of the purchased product.
6. Quantity: The quantity of the selected product related to the specific order.
7. Discount: The percentage of the discount in the given order for the given product.
8. Total: The total amount that the order was cost per product inside the order, including the unitprice, discount, quantity and an additional VAT payment in a rate of 17%.
9. InsertDate: The date that the order was inserted to the system.



* + DimProducts:

In the DimProducts table we have the following columns:

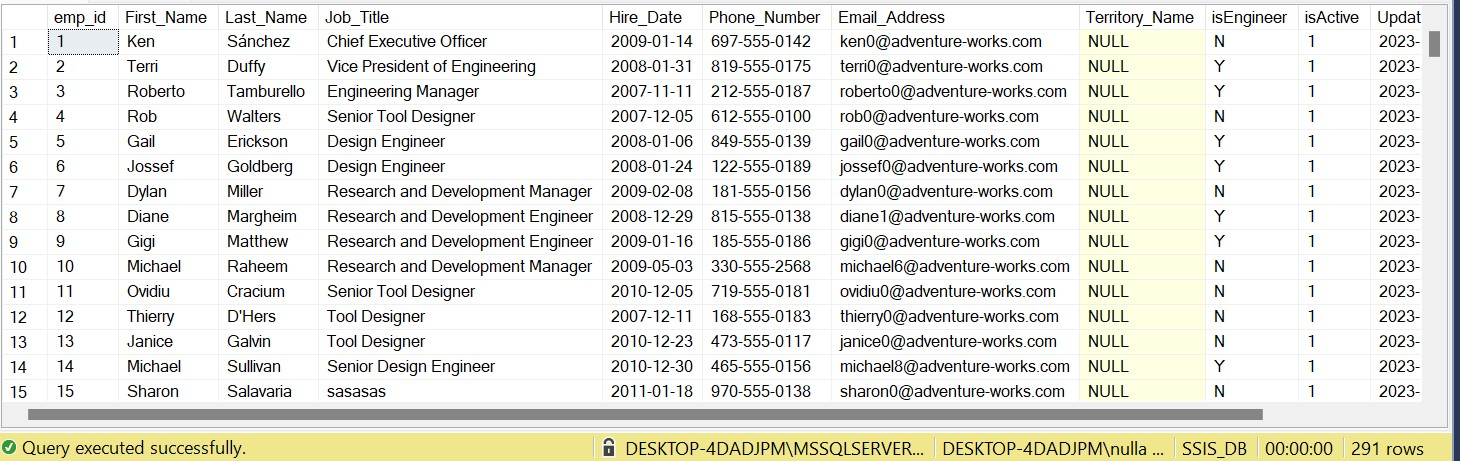
1. productID: Behaves as a primary key of the table. Represents the unique ID of the product.
2. Product\_Name: The name of the product.
3. Sub\_Category\_Name: The name of the subcategory that the product related to.
4. Category\_Name: The name of the category that the product related to.
5. UpdateDate: A field that shows if a product has been deleted from the original database or not.
6. isActive: A field that shows if the product is still active for sales or not.



* + DimEmployees:

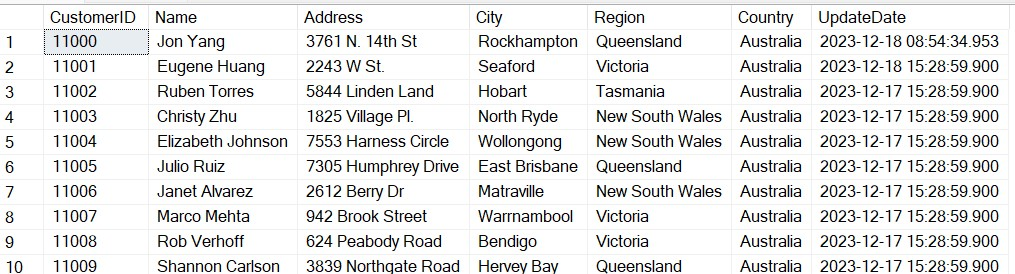
In the DimEmployees table we have the following columns:

1. emp\_id: Behaves as a primary key of the table, Represents the unique ID of the employee.
2. First\_Name: The first name of the employee.
3. Last\_Name: The last name of the employee.
4. Job\_Title: The job title of the employee.
5. Hire\_Date: The date that the employee was hired to the company.
6. Phone\_Number: The employee's phone number.
7. Email\_Address: The employee's email address.
8. Territory\_Name: The territory that the employee was assigned to.
9. isEngineer: A flag column, checks if the employee's title include Engineer field or not.
10. isActive: A flag column that displays if the employee is still in the company or not.
11. UpdateDate: The date that the table was updated per employee.

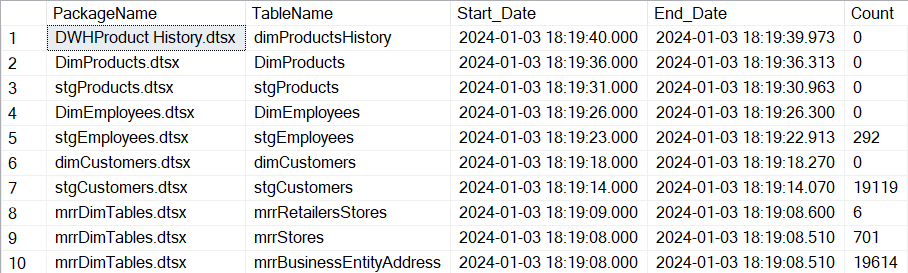


* + DimCustomers:

In the DimCustomers table we have the following columns:

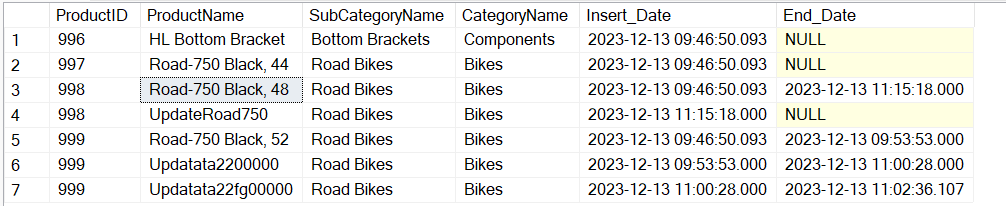
1. Customer\_ID: Behaves as a primary key of the table, Represents the unique ID of customer.
2. Name: The full name of the customer.
3. Address: The address of the customer.
4. City: The city that the customer lives at.
5. Region: The religion that the customer is related to.
6. Country: The country of the customer.
7. UpdateDate: Column that checks when the customer's details were updated most recently.
8. TransferTable:

In the TransferTable table we have the following columns:

1. Packagename: The name of the recorded package.
2. Tablename: The name of the recorded table.
3. Start\_Date: The date that the load started the action.
4. End\_date: The date that the transaction finished the action.
5. Count: The amount of rows that were transferred in the specific transaction.
   * DimProductsHistory:

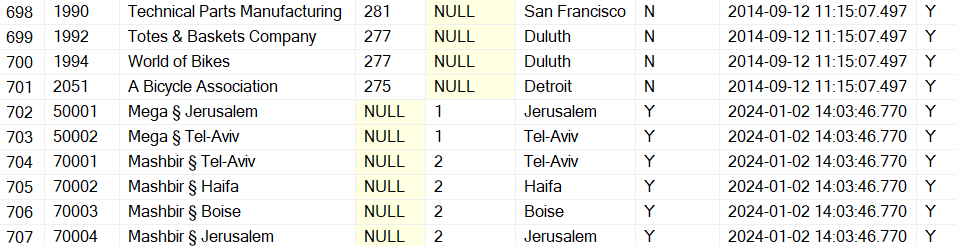
In the DimProductsHistory table we have the following columns:

1. productID: Behaves as a primary key of the table. Represents the unique ID of the product.
2. Product\_Name: The name of the product.
3. Sub\_Category\_Name: The name of the subcategory that the product related to.
4. Category\_Name: The name of the category that the product related to.
5. Insert\_Date: A field that shows when the record was updated in the dimProducts table.
6. End\_Date: A field that shows if a product has been deleted from the original database or not. NULL = Active Product, not null = product that is still active



* + DimStores:

In the DimStores table we have the following columns:

1. storeID: Behaves as a primary key of the table. Represents the unique ID of the store.
2. StoreName: The name of the store.
3. Emp\_id: The id of the employee that related to the store, Represents as a Foreign Key to dimEmployees.
4. RetailerID: The id of the retailer that related to the store, Represents as a Foreign Key to dimRetailers.
5. City: The city that the store is located at.
6. isRetailer: A flag that checks if the related person is an employee or retailer.
7. UpdateDate: A field that shows if a product has been updated from the original database or not.
8. isActive: A field that shows if the store is still active for service or not.

### \*\* Store can't have both Employee and Retailer that related to it.

* + DimRetailers:

In the DimRetailers table we have the following columns:

1. Retailer\_ID: Behaves as a primary key of the table, Represents the unique ID of a retailer.
2. RetailerName: The name of the retailer.
3. RetailerType: The type of the retailer.
4. isActive: A field that shows if the retailer is still active or not.
5. UpdateDate: Column that checks when the retailer's details were updated most recently.