ETL and Data Warehouse Implementation

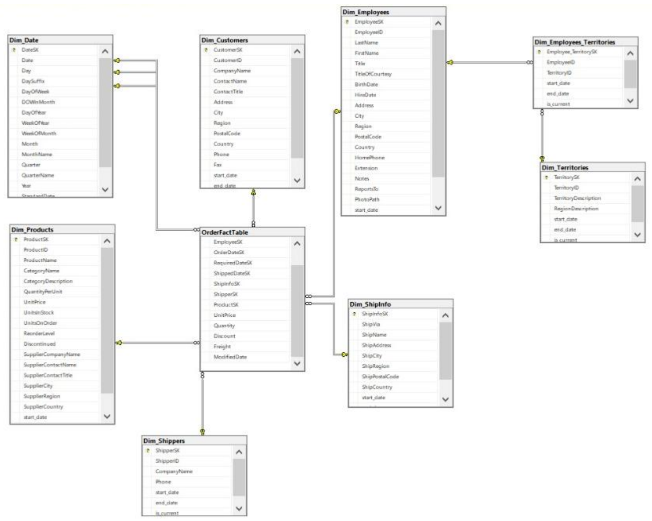
Overview

Following the successful deployment of a fully functional **Online Transaction Processing (OLTP) system** to support business operations, we proceeded with the development of an **Online Analytical Processing (OLAP) data warehouse**. This data warehouse enables efficient **business analysis and reporting, leveraging historical and aggregated data.**

Data Warehouse Modeling.

Data modeling is a critical factor in determining the effectiveness of the **Extract, Transform, Load (ETL)** process within a data warehouse. For the Northwind database, we opted to implement **a snowflake schema due to its distinct advantages tailored to our requirements:**

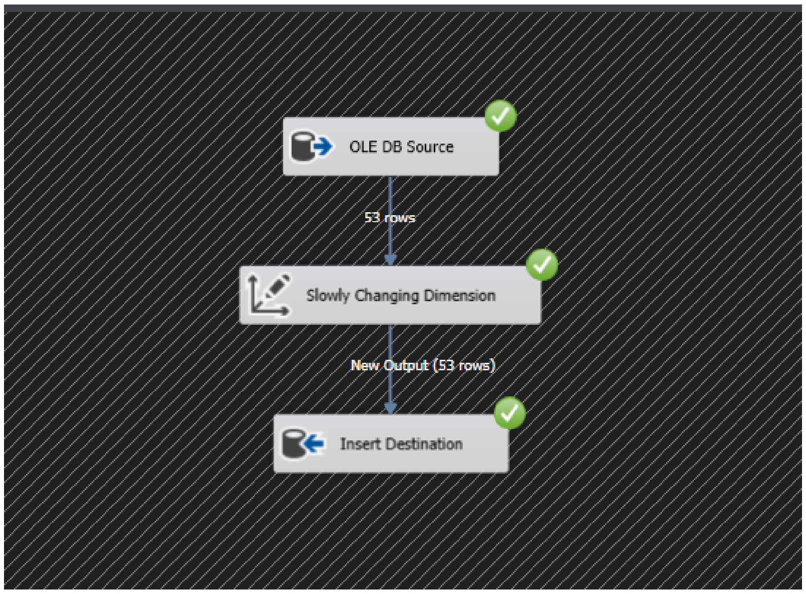
1. **Storage Optimization**: The snowflake schema's normalized structure eliminates **data redundancy, optimizing storage utilization**. This is particularly beneficial as the data warehouse is deployed on ***Microsoft Azure***, where efficient resource allocation directly impacts operational costs.
2. **Capturing Changes Easily**: Normalization inherent in the snowflake schema facilitates accurate and anomaly-free data updates. This ensures seamless handling of insertions, updates, and deletions, maintaining data integrity throughout the warehouse.
3. **Hierarchy** The schema supports **hierarchical relationships** within the data, which aligns with the Northwind dataset's structure. Examples include **geographic hierarchies** (Region → Territory → Employee) and **product hierarchies** (Supplier → Product → Category), enabling granular analysis and reporting.



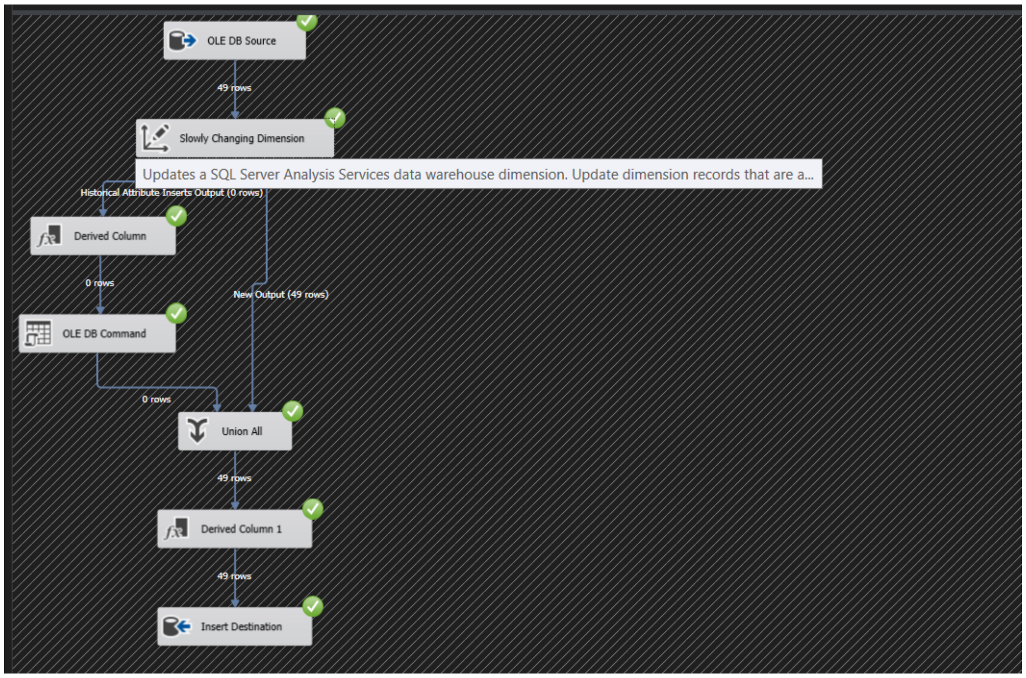
ETL Using SQL Server Integration Services (SSIS)

With the data warehouse modeled and implemented in SQL Server, the ETL process was designed to transfer data from the OLTP system into the data warehouse, adhering to the snowflake schema. The following tables were populated:

- **Dim_Territory (53 Rows)**

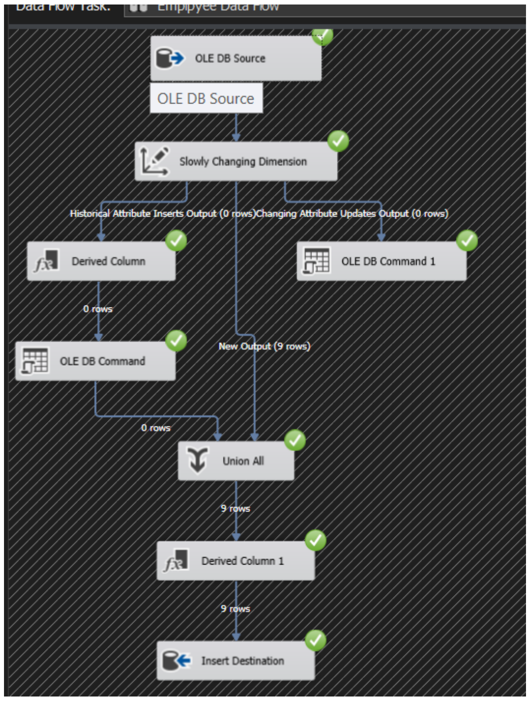


- **Dim_Employee_Territory (49 Rows)**

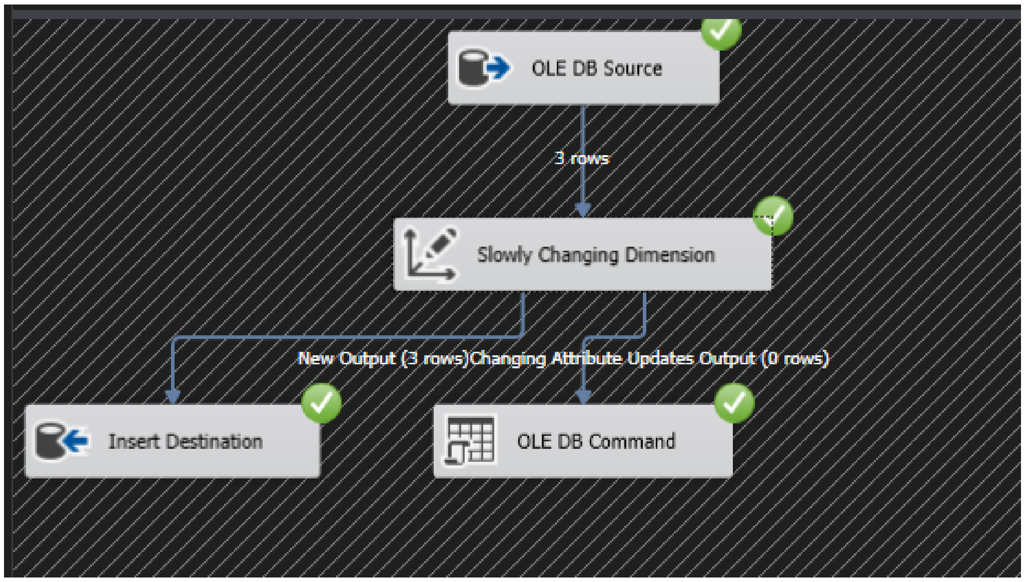


Updates a SQL Server Analysis Services data warehouse dimension. Update dimension records that are a...

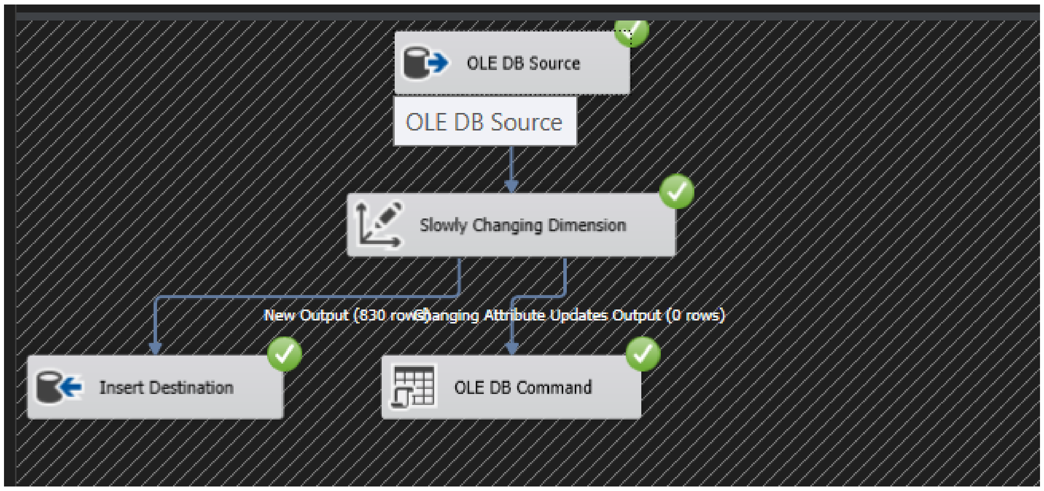
- **Dim_Employee (9 Rows)**



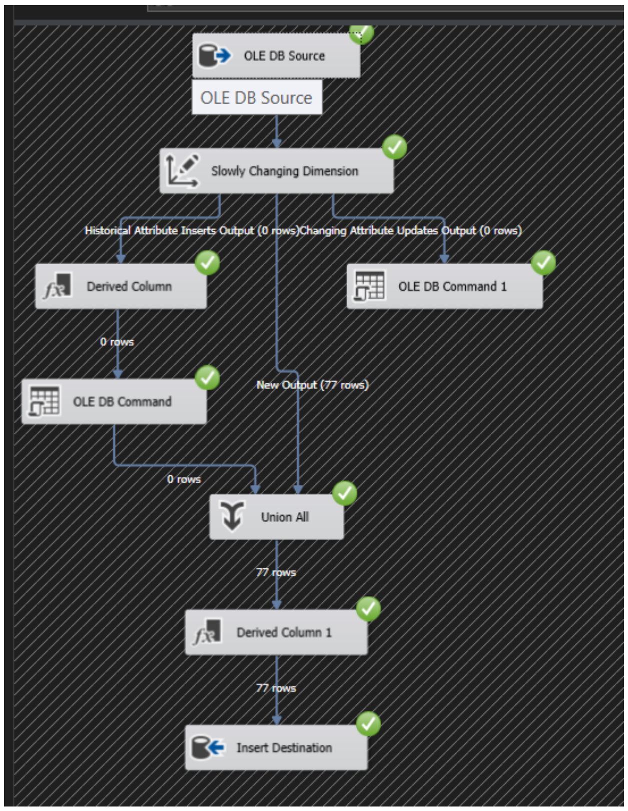
- **Dim_Shipper (3 Rows)**



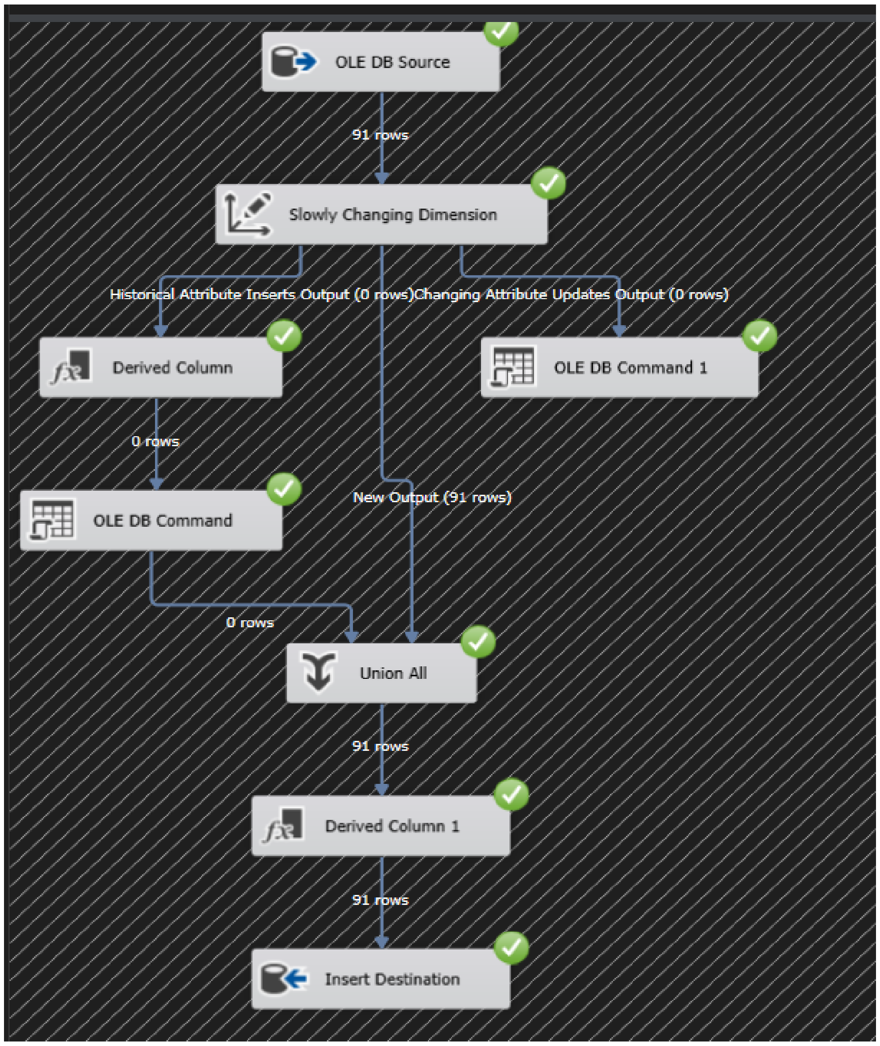
- **Dim_ShipInfo (830 Rows)**



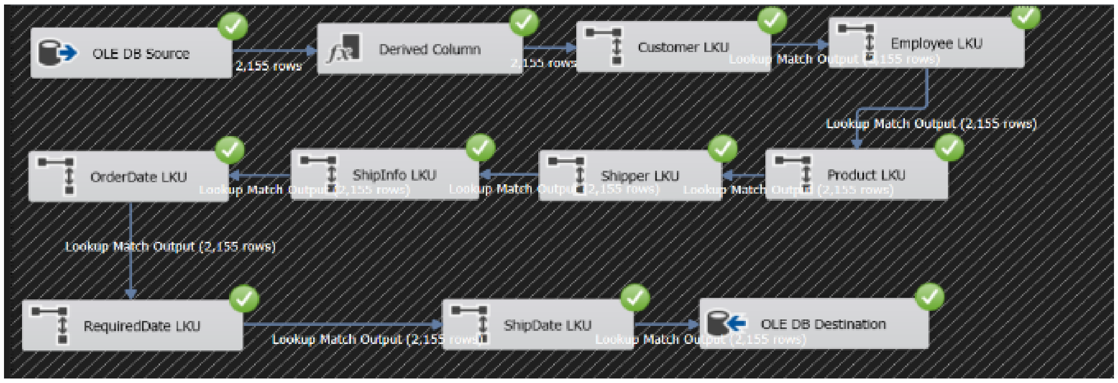
- **Dim_Product(77 Rows)**



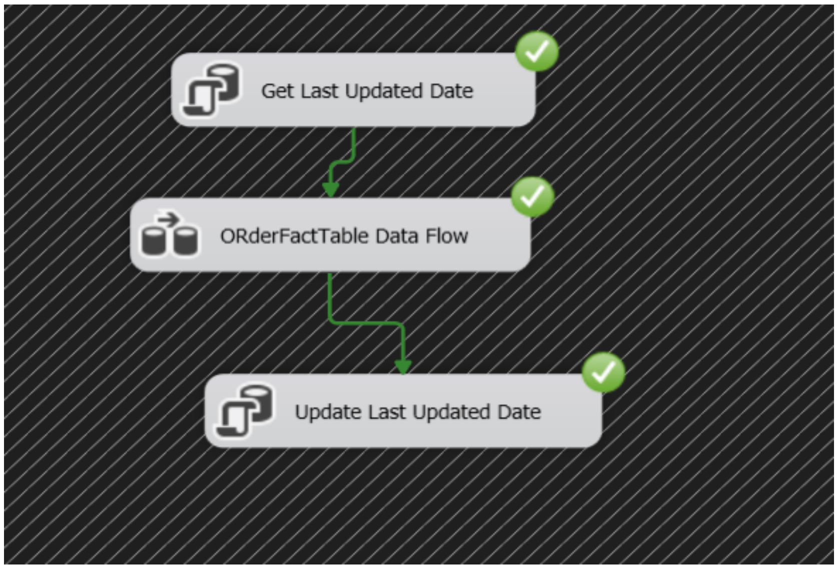
- **Dim_Customer (91 Rows)**



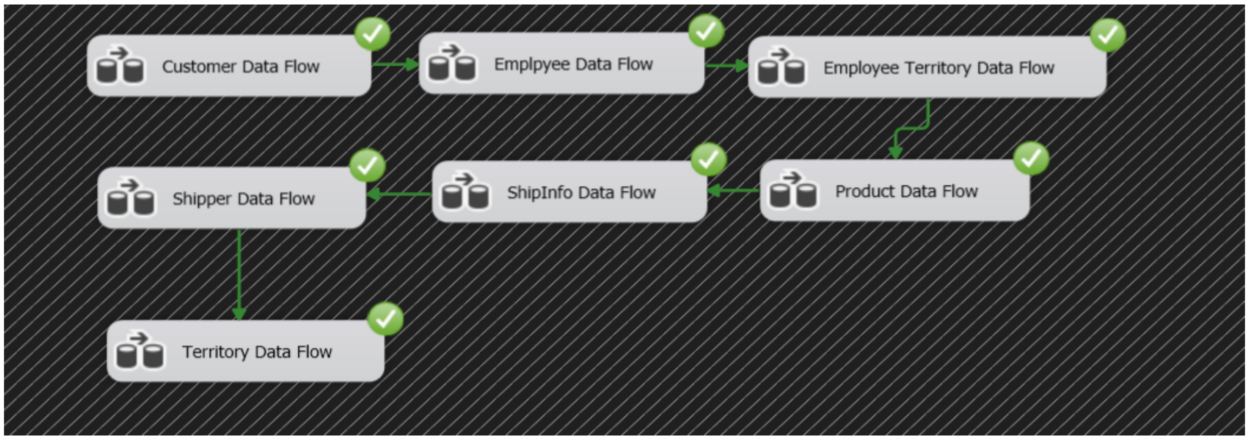
- **FactTable (First Initial Full Load) (2153 Rows)**
  This is the first time the fact table was loaded with 2153.


OLE DB Source
2,155 rows
Derived Column
2,155 rows
Customer LKU
Lookup Match Output (2,155 rows)
Employee LKU
Lookup Match Output (2,155 rows)
Product LKU
Lookup Match Output (2,155 rows)
Shipper LKU
Lookup Match Output (2,155 rows)
ShipInfo LKU
Lookup Match Output (2,155 rows)
OrderDate LKU
Lookup Match Output (2,155 rows)
RequiredDate LKU
Lookup Match Output (2,155 rows)
ShipDate LKU
Lookup Match Output (2,155 rows)
OLE DB Destination

- **FactTable (Incremental Load)**
  To avoid fully offloading and loading the Fact Table, we implemented an incremental load based on the ModifiedDate from the original database.



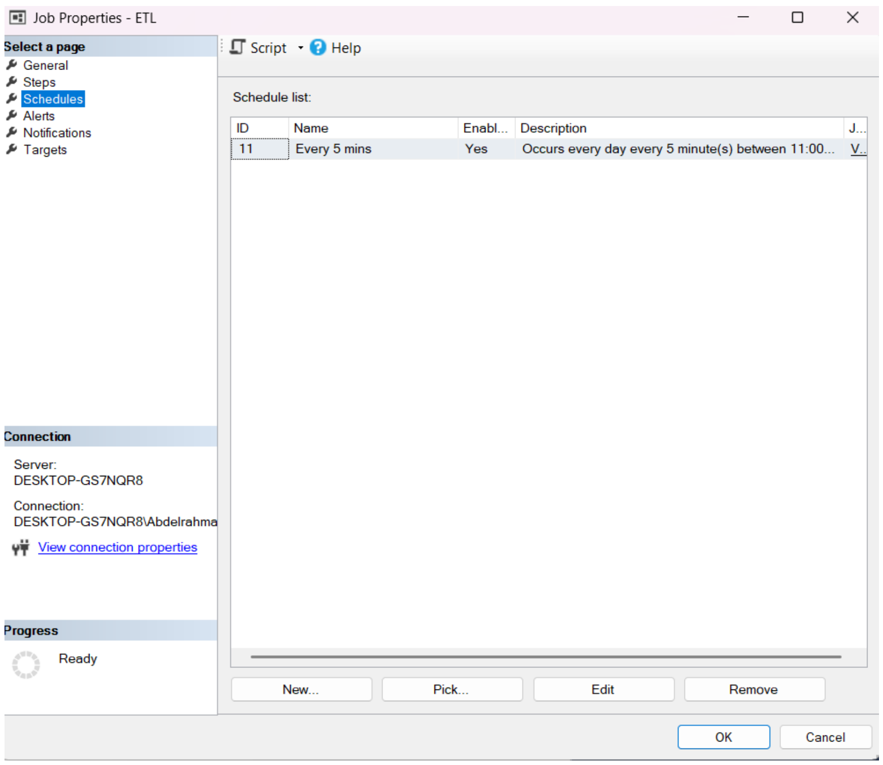
- **Full Dimensions ETL Package**



Features Used for ETL Process using SSIS.

In this section, we are trying to elaborate on what are the features and options that we have used for the pipeline.

1. **Slowly Changing Dimensions (SCD):** By implementing slowly changing dimensions we are able to capture Type 1 and Type 2 changes in the data base. Any record that are added to the database will be dealt with in a one of both ways:
    a. Type 1 Change: Replacing old values with new one (**not keeping historical records**)
    b. Type 2 Change: Adding a new row using start_date, end_date, and a flag column is_current. (**Saving historical records**).
2. **Orphan Data Handling**: By Inserting a record in each dimension with **a surrogate key of -1,** we are able to deal with any missing values in the original database and handling them correctly and efficiently.
3. **Pipeline Automation Using SQL Server Job**: By using SQL Server jobs, we are able to automate the pipeline to be started in **a scheduled time without human interference.**



Schedule list:

ID	Name	Enabl...	Description	J...
11	Every 5 mins	Yes	Occurs every day every 5 minute(s) between 11:00...	V...

Connection

Server:
DESKTOP-GS7NQR8

Connection:
DESKTOP-GS7NQR8\Abdelrahma

View connection properties

Progress

Ready

4. **Incremental Load for Fact Table:** Since the package runs automatically every 5 mins, we can't allow the table to be loaded and offloaded but only new data inserted, update, or deleted from the database should be reflected in the DWH.