Sri Lanka Institute of Information Technology

IT3021 - Data warehousing and Business Intelligence

Year 3 Semester 2

DWBI – Assignment 01

IT22923424 – Jayasekara S.S

Step 1: Data set selection

Data Set Title -Hospital Appointment Management System

This dataset represents a hospital's appointment scheduling and management system over a year, including patient information, doctor & departments, appointment details, treatments, and payments. It simulates the workflow from scheduling to billing. It includes 6 CSV and Excel files (data sources):

- Patient details (CSV file)- Stores patient info like name, gender, DOB, contact, and address
- Doctors (CSV file)- Contains doctor details and links to their department
- Departments (Excel file)- Defines the departments such as Cardiology, Pediatrics, etc.
- Appointments and their statuses (CSV file)- Contains scheduled appointment details for patients with doctors.
- Treatments/services provided (CSV file)- Lists the treatments provided per appointment with cost.
- Payments (Excel file)- Tracks the payment made for each treatment.

This is an OLTP dataset that normalized, with transactional data. It's perfect for this since it, has multiple entity relationships, is suitable for building star/snowflake schemas, can be split into CSV and database sources, has enough data to support cubes, hierarchies, aggregates, ETL, etc.

ER Diagram

Tables	**Attributes**
Patient	patient_id, full_name, gender, date_of_birth, contact_info, address
Doctor	doctor_id, name, specialization, department_id (FK department)
Department	department_id, name
Appointment	appointment_id, patient_id (FK1 patient), doctor_id (FK2 doctor), appointment_date, status (Scheduled/Completed/Cancelled)
Treatment	treatment_id, appointment_id (FK appointment), treatment_type, cost

Payment	payment_id, appointment_id (FK appionment), payment_date, payment_method, amount, payment_status

- A **Patient** can have multiple **Appointments**
- Each **Appointment** is with a **Doctor**
- A **Doctor** belongs to a **Departmen**t
- Each **Appointment** may have **Treatment(s)**
- Each **Treatment** may have one **Payment**

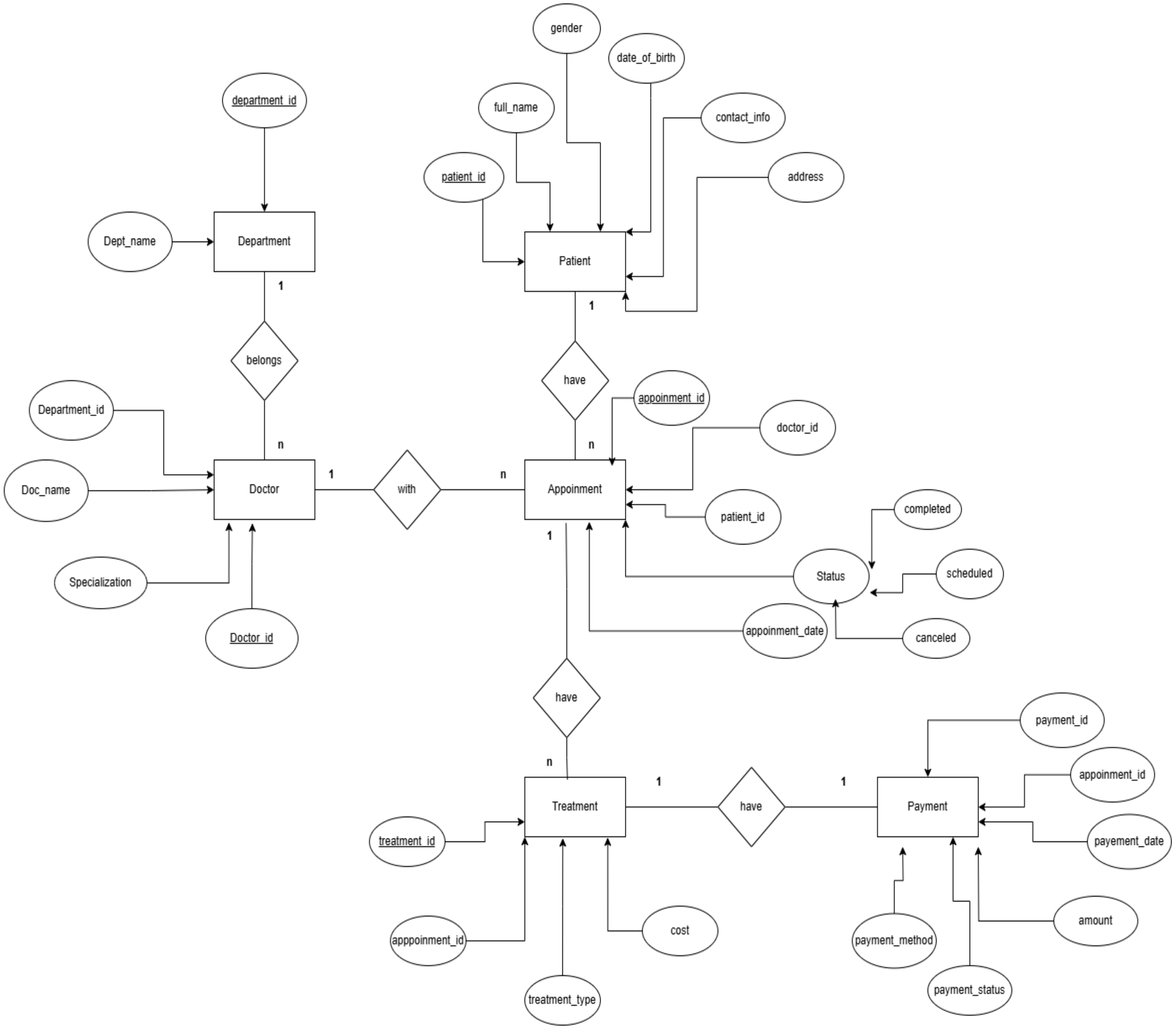


Figure 1:ER Diagram

Step 2: Preparation of data sources

To simulate a real-world data integration environment, the dataset is split into **multiple formats** as required. We have used **at least two** types of data sources as required:

- **Structured tabular files (CSV, Excel)**
- **Semi-structured files (TXT, SQL-like)**

These sources cover **multiple aspects** of hospital operations:

- Patient & appointment lifecycle
- Doctor and department structure
- Billing and treatment

All sources reflect a real-world transactional (OLTP) system. The mix of **CSV, TXT and Excel files** demonstrates integration from diverse systems.

Data Source	**File Name**	**Reason**
Patient Data	Patient.csv (CSV)	CSV is lightweight and widely used for exporting customer records. It simulates data coming from a **front desk registration system** or **online portal export**.
Doctor Data	Doctor.txt (text file)	TXT files are often used for **staff data exchange** between HR systems or between partner hospitals. Separating this helps show integration from semi-structured sources.
Department Data	Department.xlsx (Excel)	Departments rarely change, and Excel is often used by admin teams for **maintaining static master data**.
Appointment Data	Appointment.csv (CSV)	Appointment data changes frequently and is usually logged in **transactional logs or scheduling systems**.

		Using a separate CSV helps simulate such raw, growing data.
Treatment Data	Treatment.csv (CSV)	Treatment data might come from a **billing database export** or **legacy health system**. Using CSV format demonstrates **parsing and cleaning raw backend data**.
Payment Data	Payment.xlsx (Excel)	Finance teams often manage payments in Excel sheets. This shows how DW can integrate **financial records from different business units**

Step 3: Solution architecture

High-Level DW & BI Solution Architecture

Component	**Description**
Data Sources	Mixed-format files (CSV, TXT, Excel) containing raw hospital data like patients, appointments, doctors, etc.
SSIS (ETL Layer)	Used to extract data from each source, apply transformations (cleaning, joining, formatting), and load it into staging and warehouse tables.
Staging Area (SQL Server)	Temporary tables in SQL Server where raw data is initially loaded before further processing.
Data Warehouse (Dimensional Model)	A dimensional model (Star Schema) built using SQL Server to store historical, cleaned, and aggregated data
SSAS (Cube Layer)	Multidimensional cube built on the DW to enable fast analysis with dimensions, measures, and hierarchies.
BI Tools (Reporting)	Tools like Power BI or SSRS are used to generate reports and dashboards based on the DW or cube data.

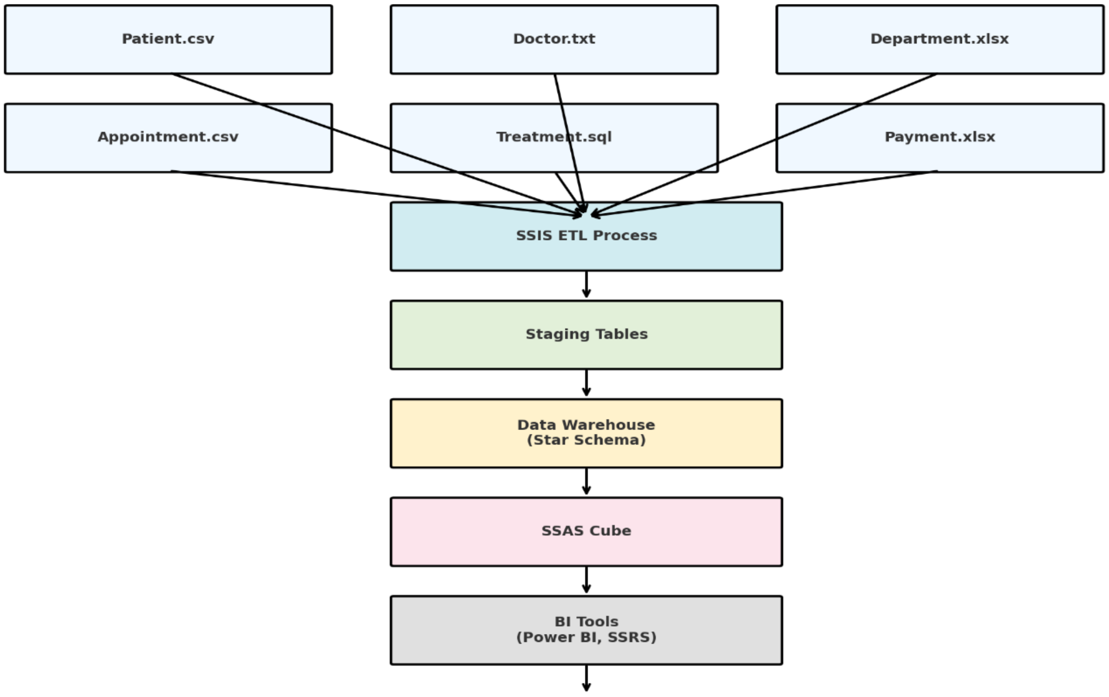


Figure 2: Architectural Diagram

Step 4: Data warehouse design & development

In this step, we designed a **Star Schema** data warehouse model tailored to the Hospital Appointment Management System. The central **fact table**, **FactAppointment**, captures measurable business events such as appointments, treatment costs, and appointment durations. It is surrounded by **dimension tables** that provide descriptive context about each appointment, including **patient, doctor, department, payment method, and date.**

Star Schema

Table Name	**Description**
DimDate	Contains full date breakdown for analytics (ex: day, week, month, quarter, year).
DimDepartment	Contains department info (Department ID,DepartmentName).
DimPatient(SCD)	Describes patient details (PatientID,FullName, Gender, Address).
DimDoctor(SCD)	Describes doctor profile (DoctorID,Name, Specialization, DepartmentID).
DimPayment(SCD)	Stores PaymentID,payment method and status.
FactAppointment	Stores appointment-specific data including AppointmentID, PatientSK, DoctorSK, DepartmentSK, PaymentSK, DateSK, Status, TreatmentCost, and timing details.

Assumptions Made

- A doctor always belongs to **one department**.
- Each appointment has one treatment and one payment.
- Payment method/status may change over time (→ SCD).
- Date values are tracked using a surrogate **date_id**.

Step 5: ETL development

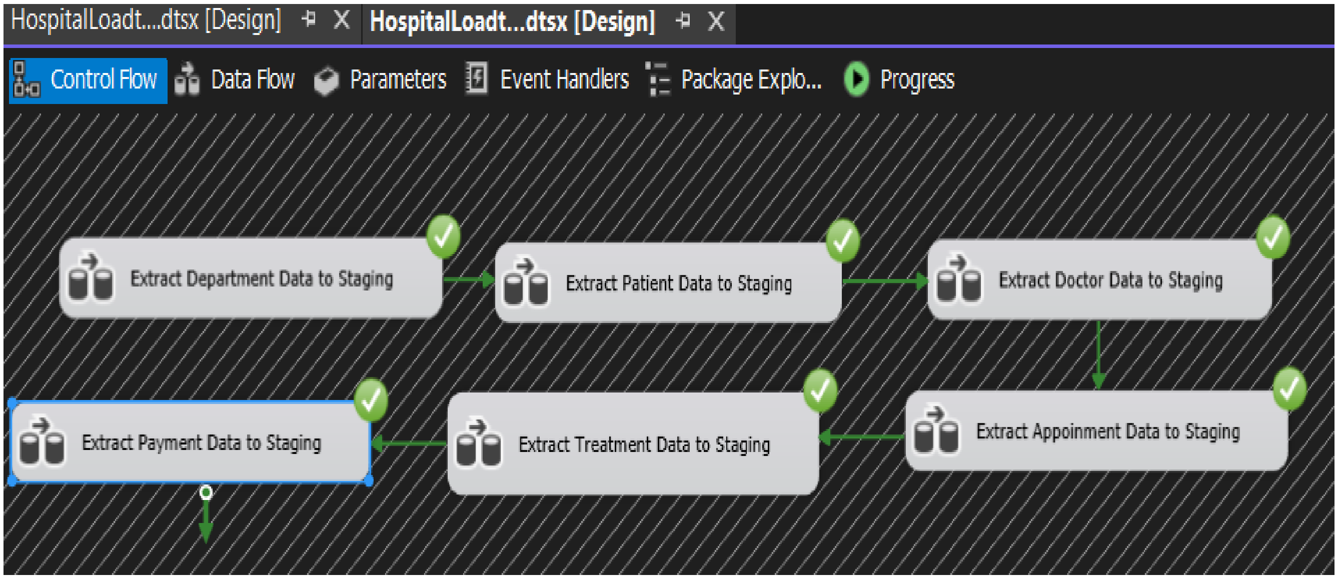
In this step, we designed and implemented the ETL (Extract, Transform, Load) process using **SQL Server Integration Services (SSIS)** to transfer data from various data sources into the data warehouse.

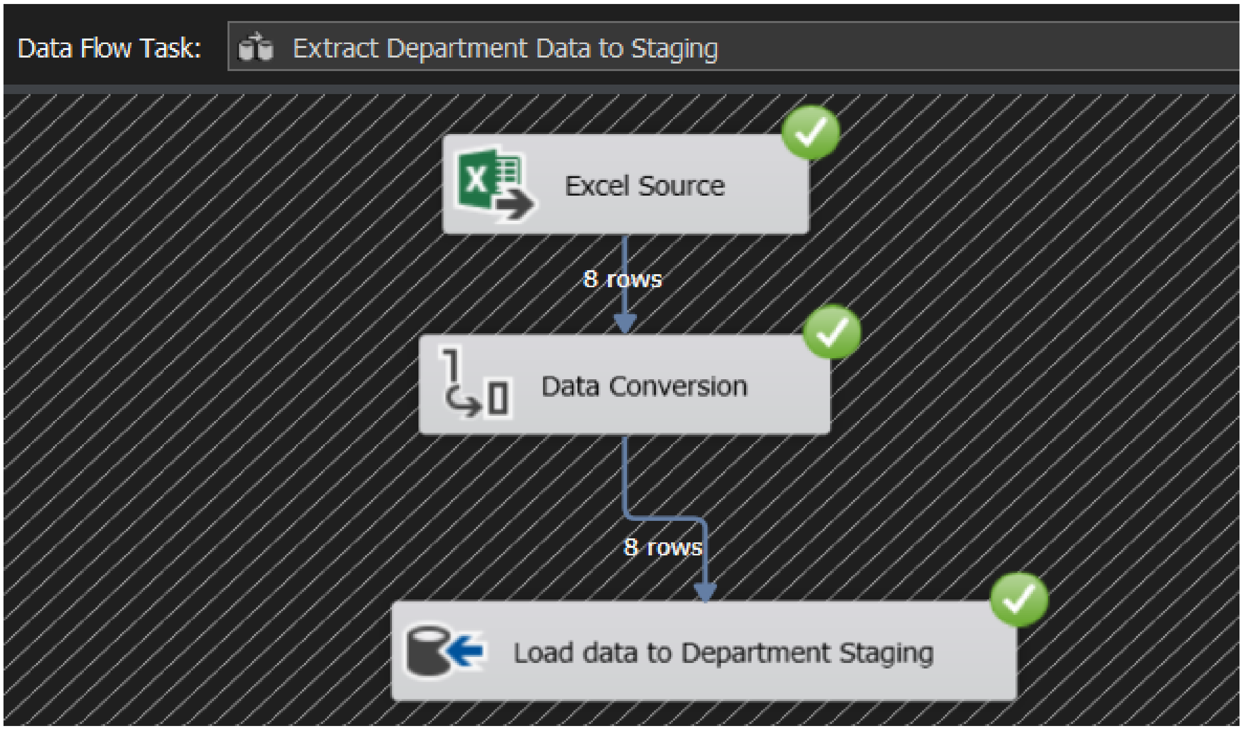
Extract

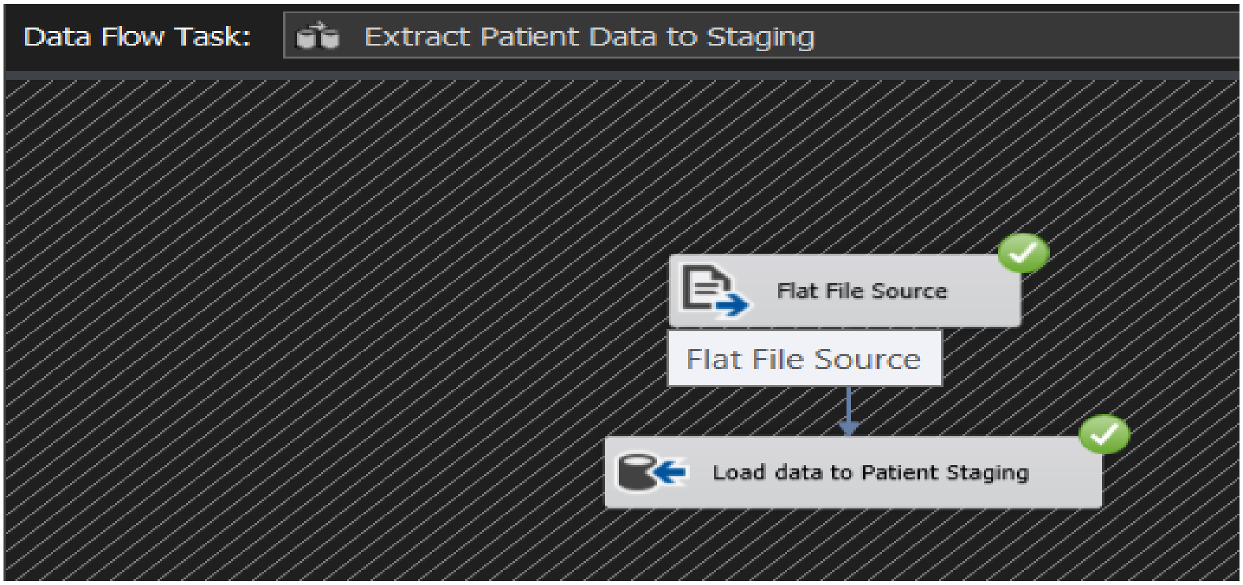
We imported data from multiple formats — CSV (Patient.csv, Appointment.csv, Treatment.csv), TXT (Doctor.txt), and Excel (Payment.xlsx) — into staging tables.

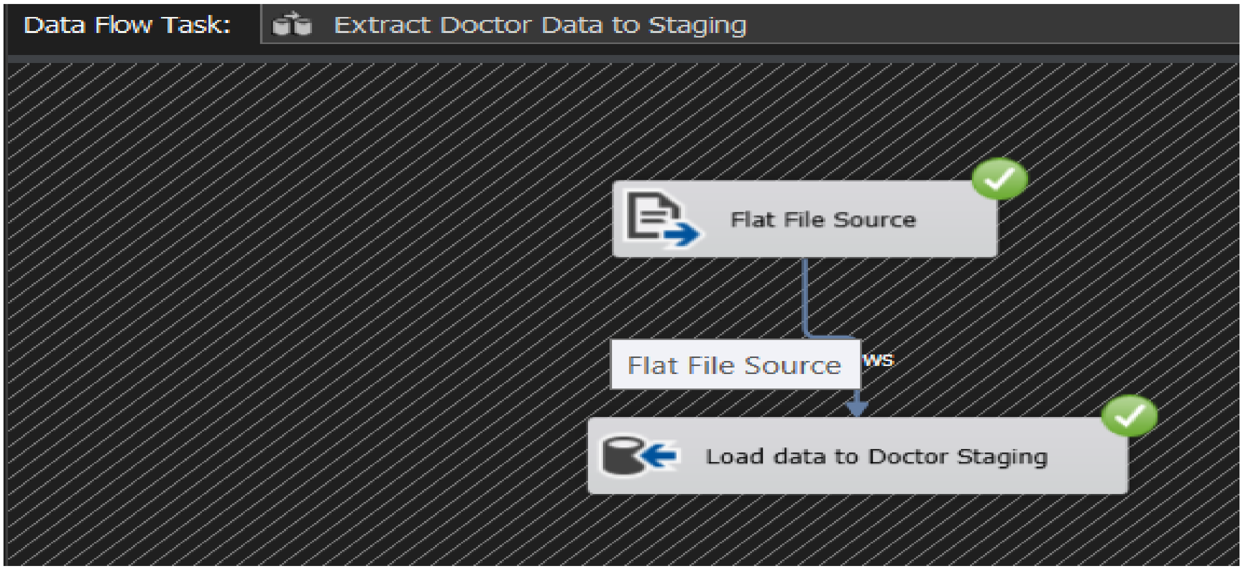
- Each file was loaded into a corresponding **staging table** in SQL Server, such as StagingPatient, StagingDoctor, StagingAppointment, etc., using **Flat File Source** or **Excel Source** components in SSIS.

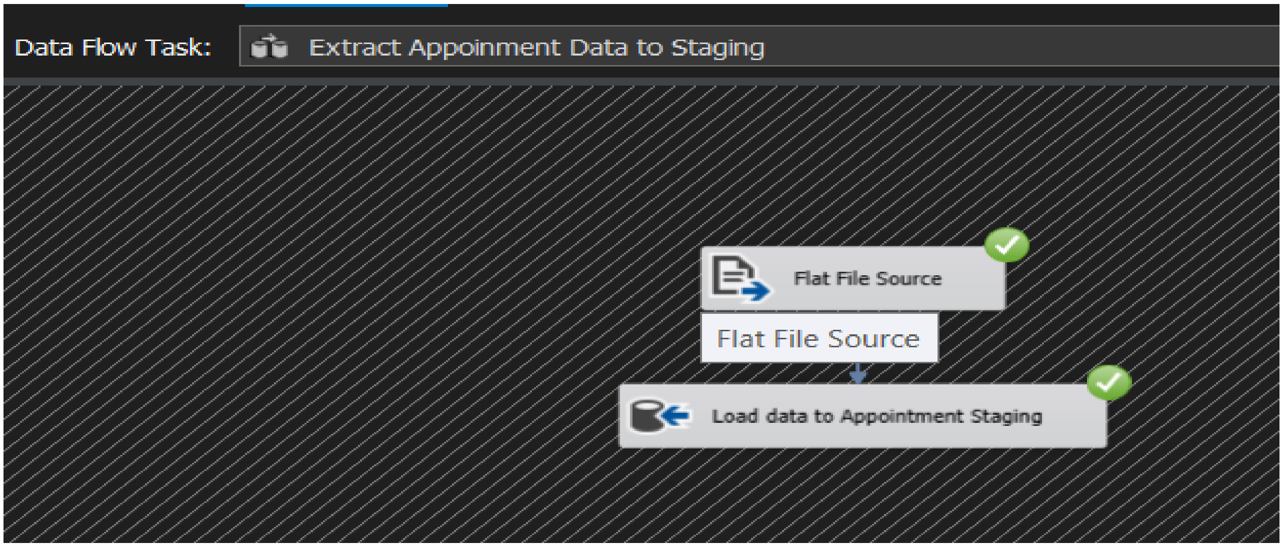
Load to Staging


HospitalLoadt....dtsx [Design]
HospitalLoadt...dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Package Explo...
Progress
Extract Department Data to Staging
Extract Patient Data to Staging
Extract Doctor Data to Staging
Extract Payment Data to Staging
Extract Treatment Data to Staging
Extract Appoinment Data to Staging

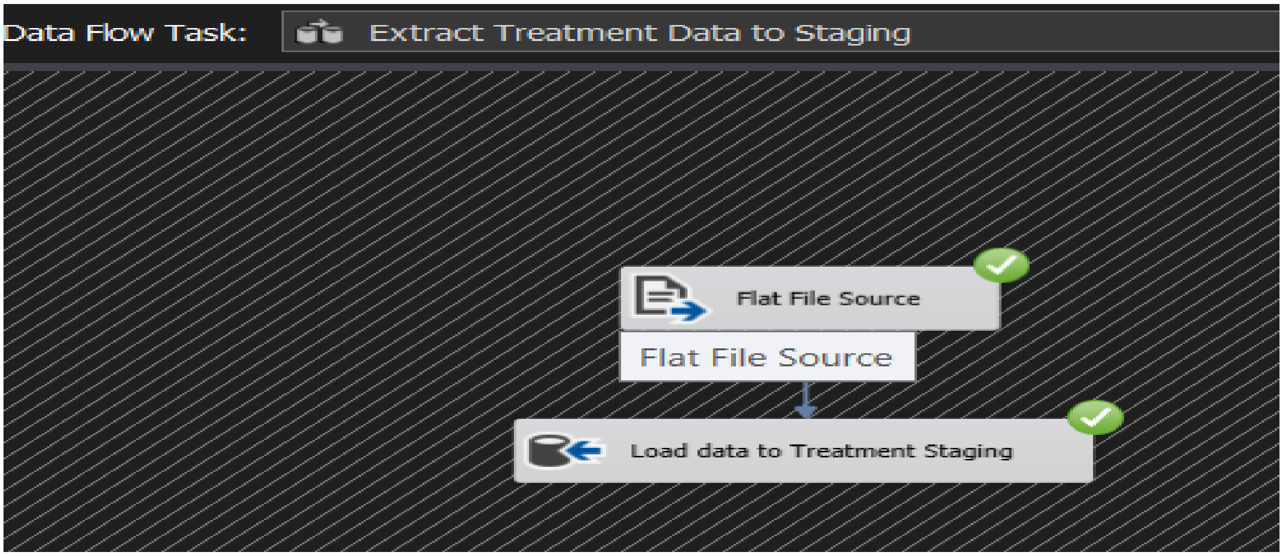

Data Flow Task:
Extract Department Data to Staging
Excel Source
8 rows
Data Conversion
8 rows
Load data to Department Staging


Data Flow Task:
Extract Patient Data to Staging
Flat File Source
Flat File Source
Load data to Patient Staging

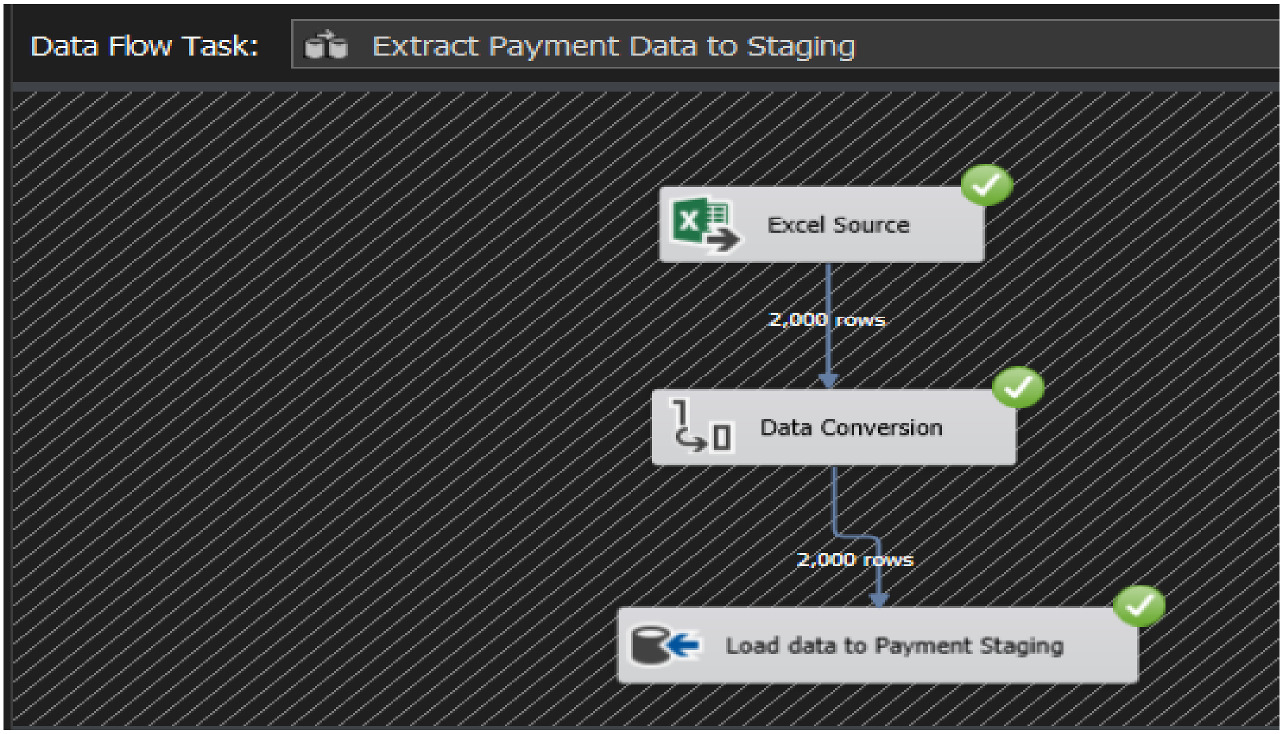

Data Flow Task:
Extract Doctor Data to Staging
Flat File Source
Flat File Source
ws
Load data to Doctor Staging



Data Flow Task: Extract Appoinment Data to Staging



Data Flow Task: Extract Treatment Data to Staging



Data Flow Task: Extract Payment Data to Staging

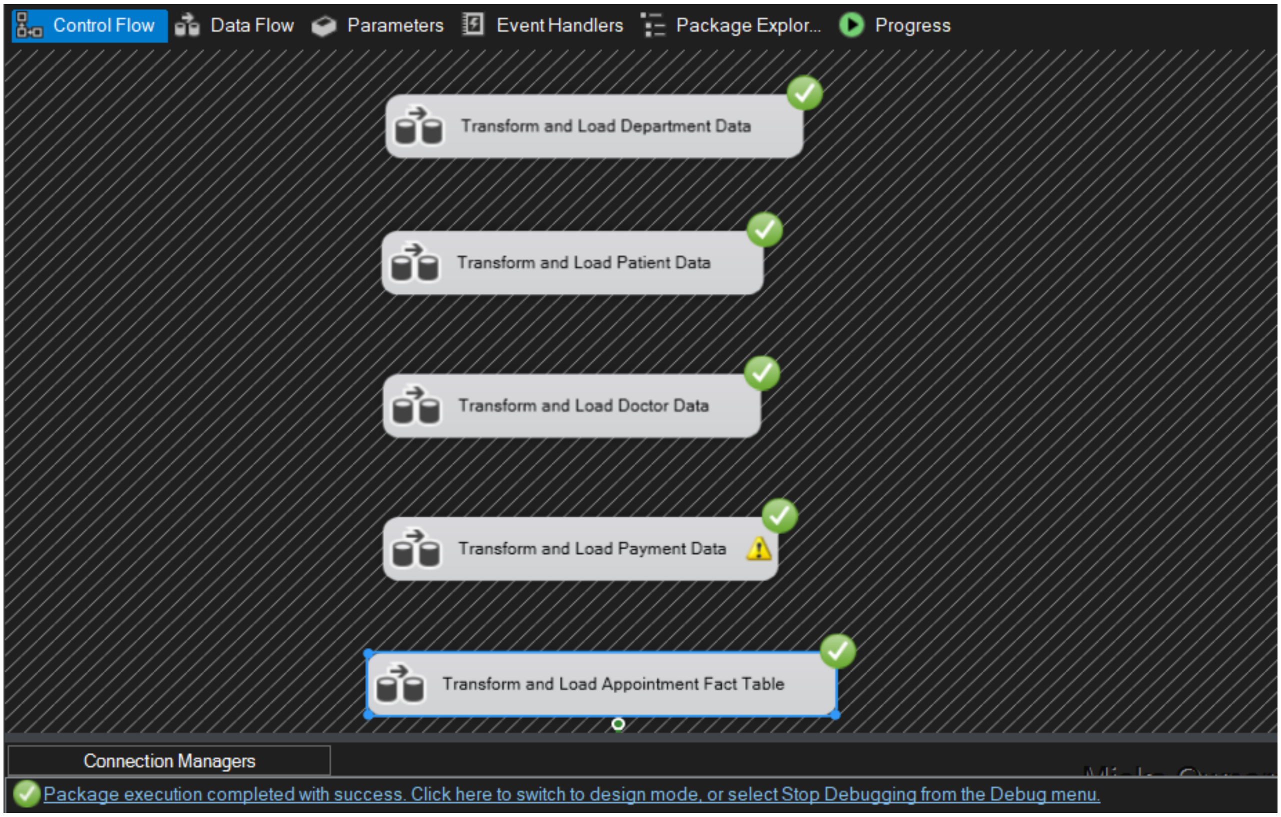
Transform

After extraction, we transformed the data using SSIS components:

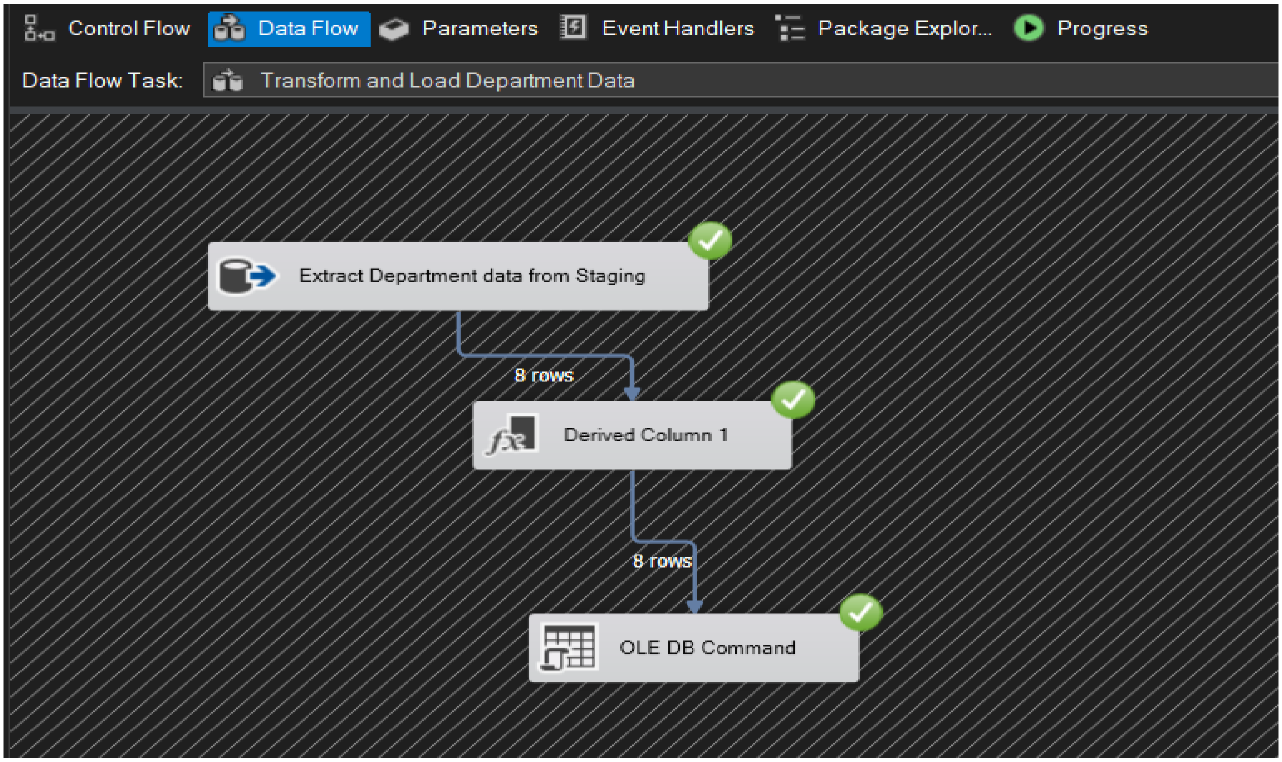
Dimension Tables:

- Applied **Derived Columns** (e.g., concatenating names, standardizing formats).
- Used **Data Conversion** tasks to match the correct data types (e.g., DT_WSTR, DT_DATE).
- Implemented **Slowly Changing Dimension (SCD)** logic for tracking historical changes where needed (e.g., DimDoctor, DimPatient,DimPayment).
- And I Used **Procedure** as well to **using NOT SCD** Dimension for demonstration **(DimDepartment).**

1)DimDepartment Transform

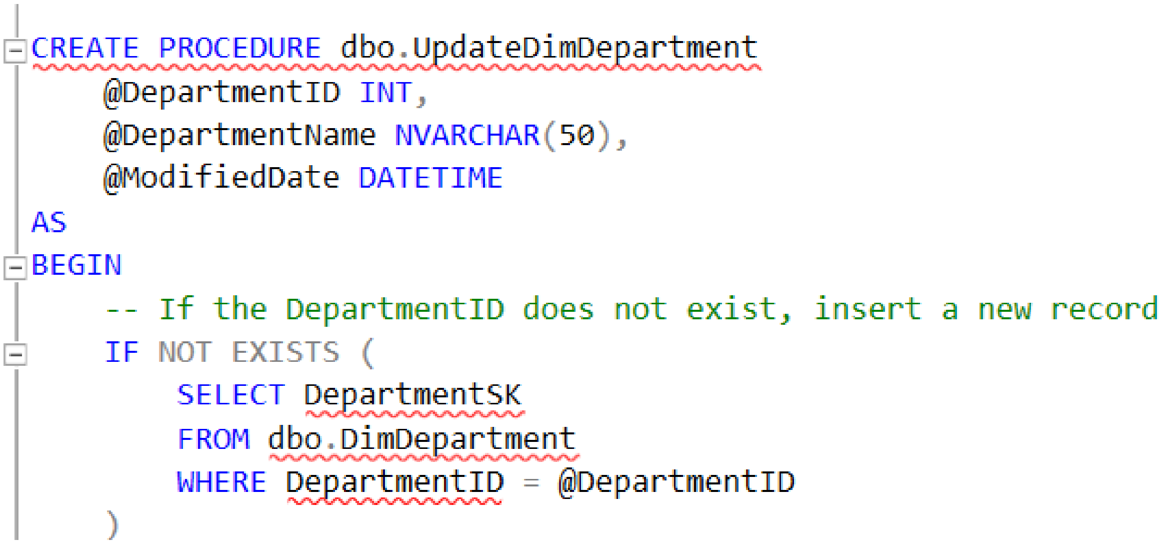

Control Flow
Data Flow
Parameters
Event Handlers
Package Explor...
Progress
Transform and Load Department Data
Transform and Load Patient Data
Transform and Load Doctor Data
Transform and Load Payment Data
Transform and Load Appointment Fact Table
Connection Managers

Package execution completed with success. Click here to switch to design mode, or select Stop Debugging from the Debug menu.

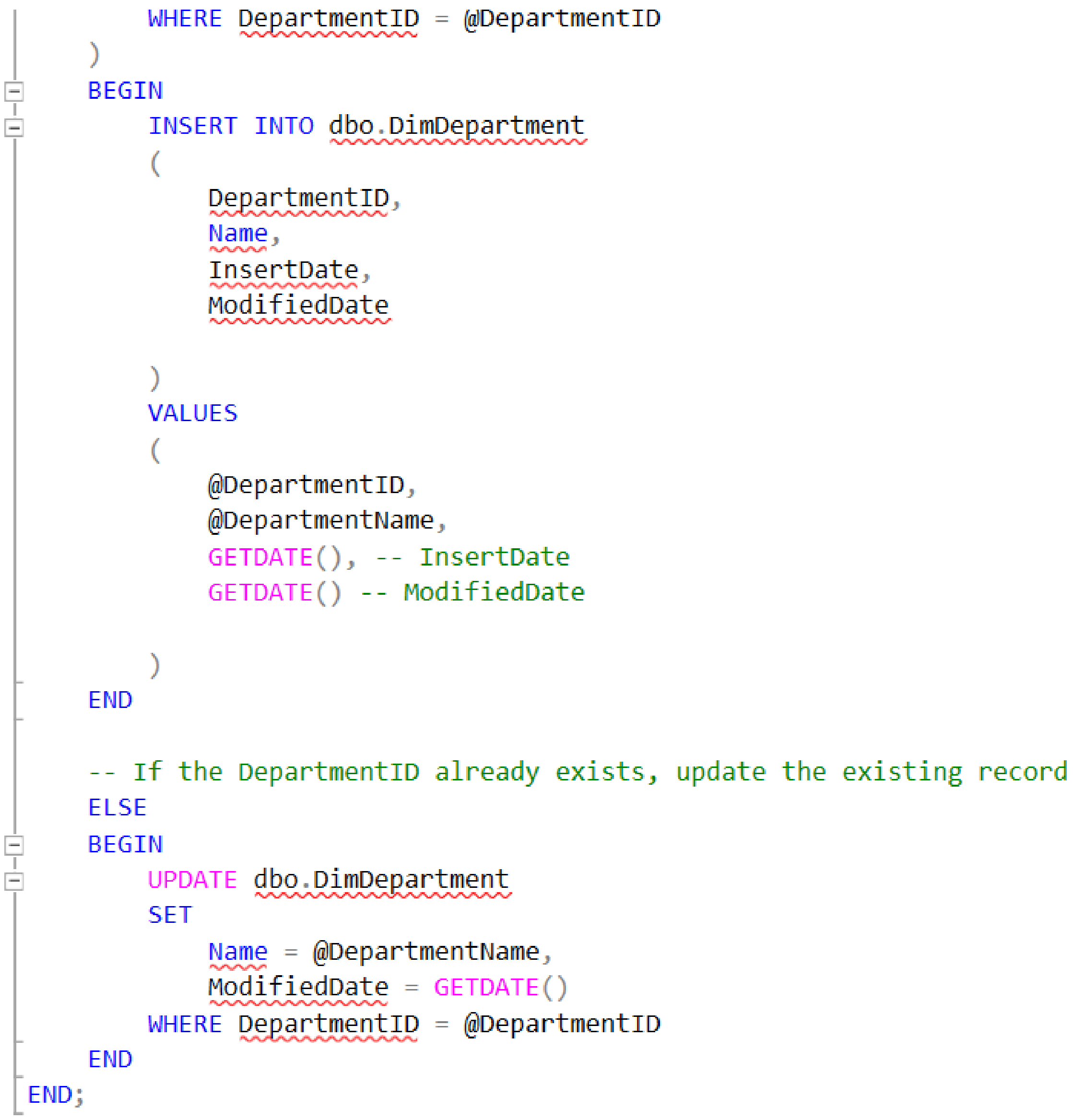


Data Flow Task: Transform and Load Department Data

Procedure Query

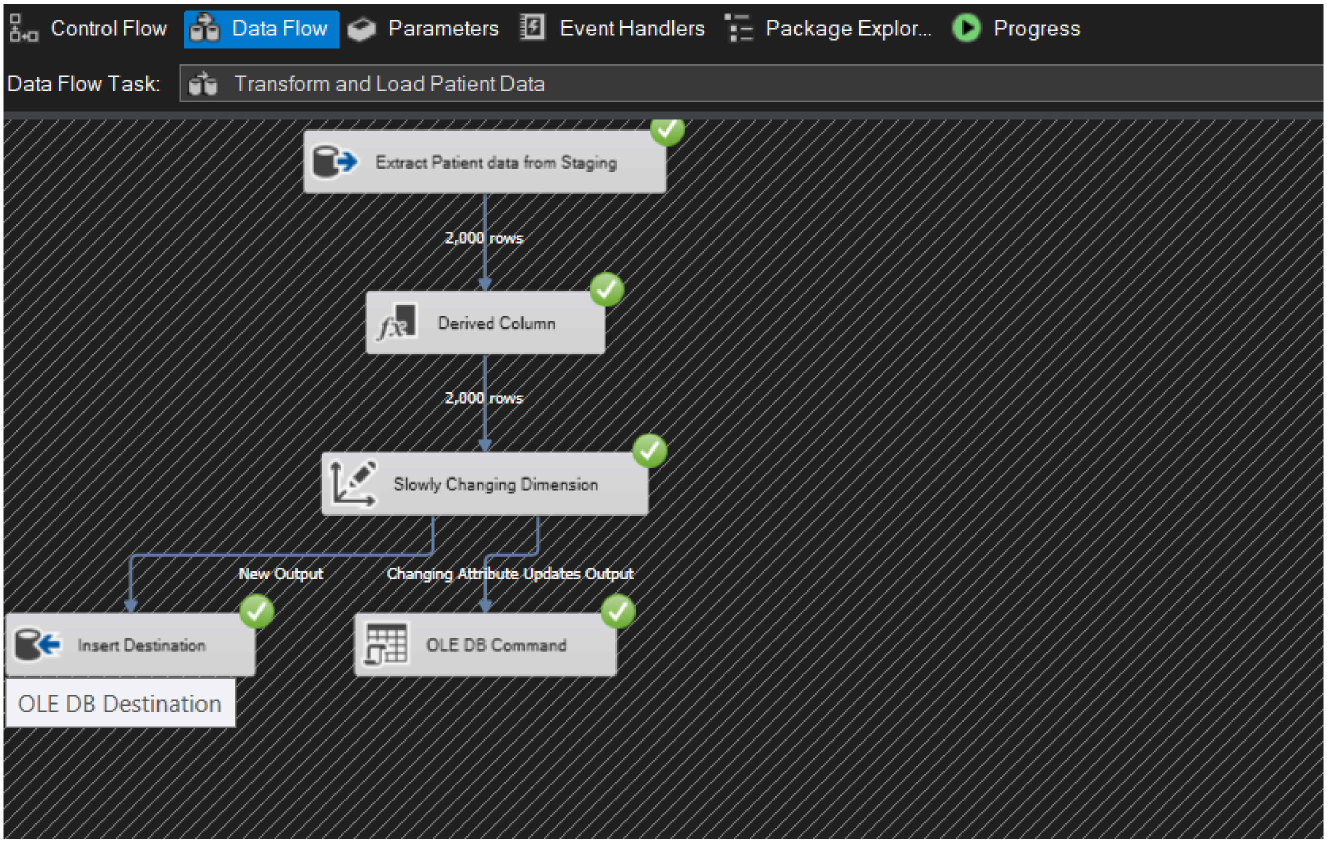


```
CREATE PROCEDURE dbo.UpdateDimDepartment
    @DepartmentID INT,
    @DepartmentName NVARCHAR(50),
    @ModifiedDate DATETIME
AS
BEGIN
    -- If the DepartmentID does not exist, insert a new record
    IF NOT EXISTS (
        SELECT DepartmentSK
        FROM dbo.DimDepartment
        WHERE DepartmentID = @DepartmentID
    )
```

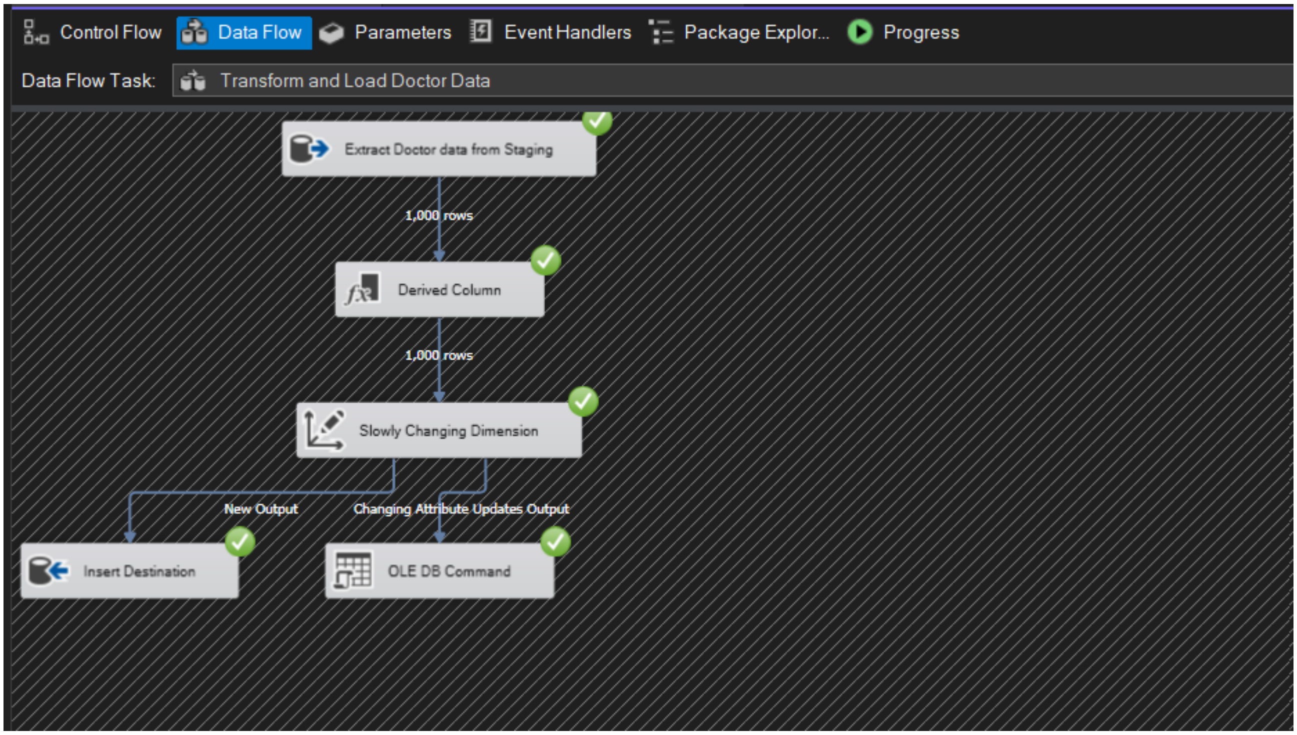


```
        WHERE DepartmentID = @DepartmentID
    )
    BEGIN
        INSERT INTO dbo.DimDepartment
        (
            DepartmentID,
            Name,
            InsertDate,
            ModifiedDate

        )
        VALUES
        (
            @DepartmentID,
            @DepartmentName,
            GETDATE(), -- InsertDate
            GETDATE() -- ModifiedDate

        )
    END

    -- If the DepartmentID already exists, update the existing record
    ELSE
    BEGIN
        UPDATE dbo.DimDepartment
        SET
            Name = @DepartmentName,
            ModifiedDate = GETDATE()
        WHERE DepartmentID = @DepartmentID
    END
END;
```

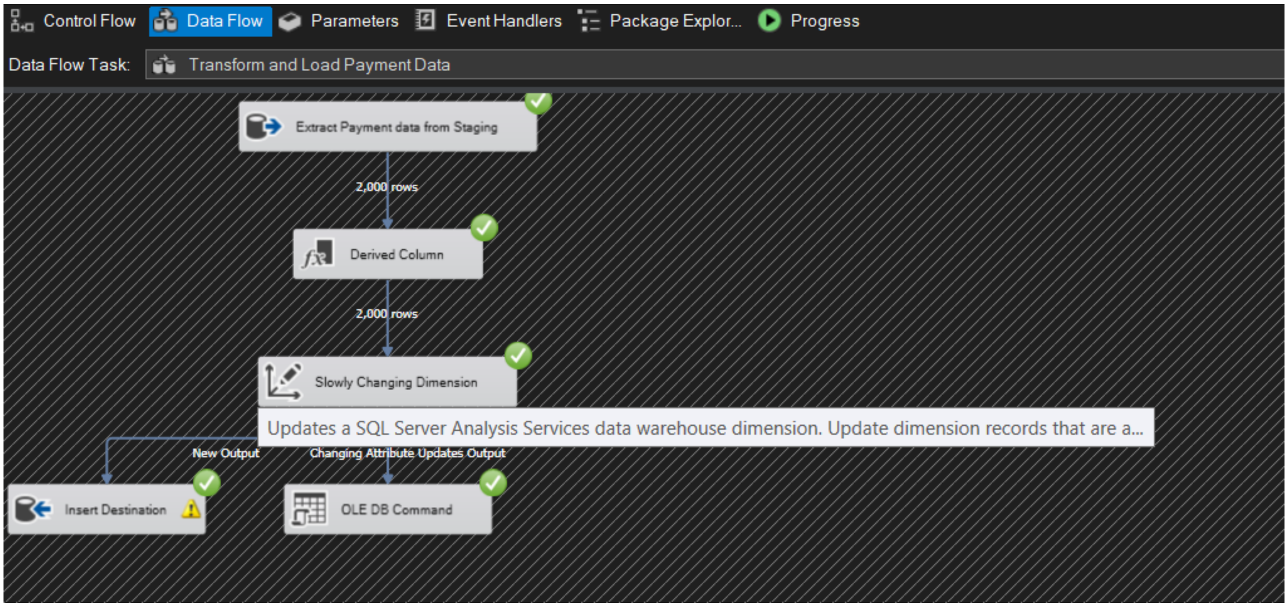
2)DimPatient Transform



3)DimDoctor Transform



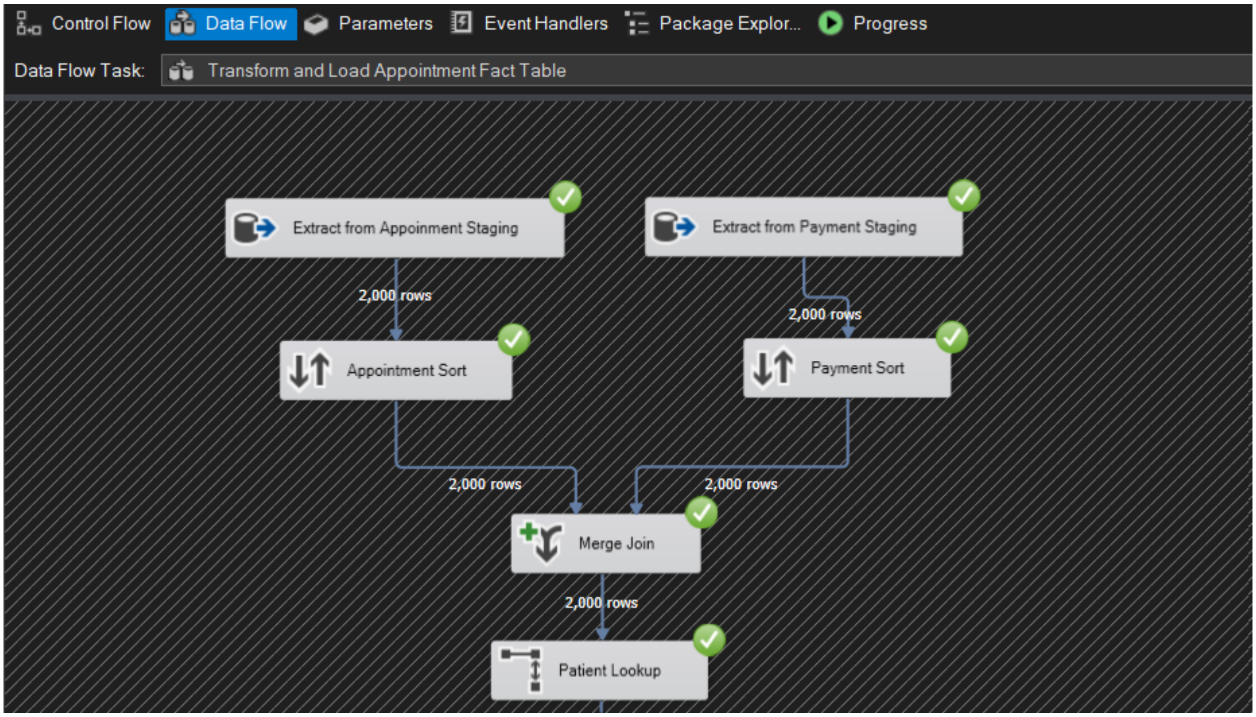
4)DimPayment Transform

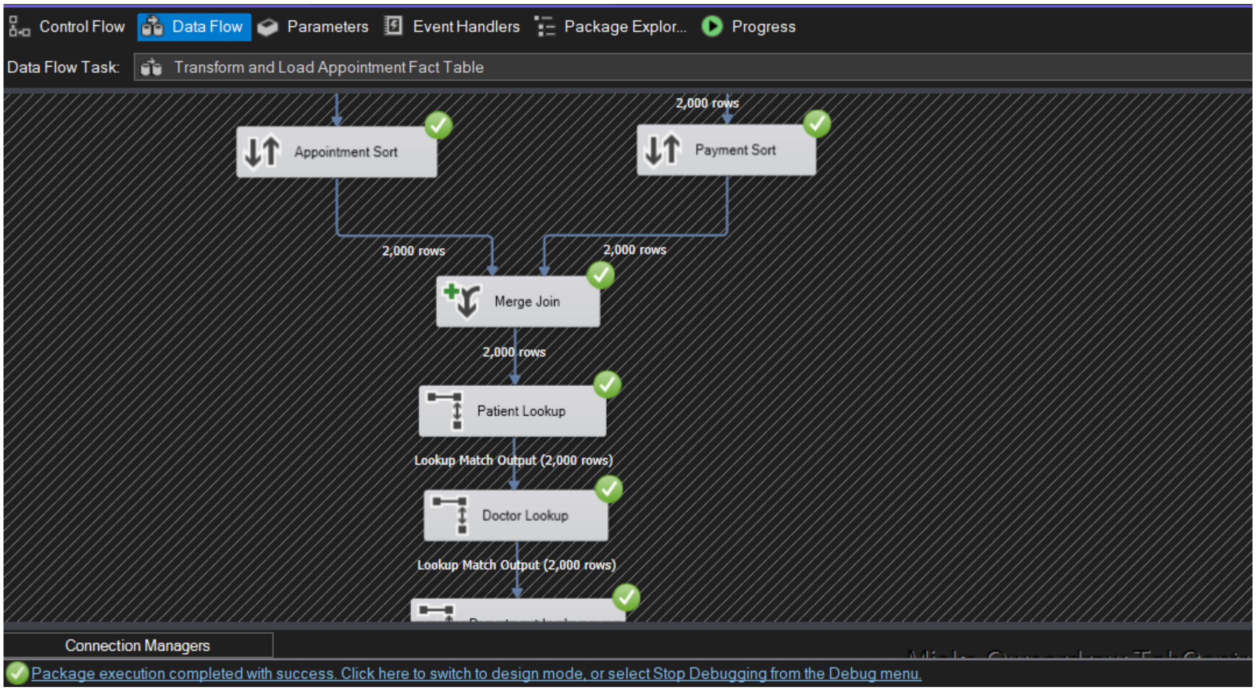


Updates a SQL Server Analysis Services data warehouse dimension. Update dimension records that are a...

Fact Table (FactAppointment):

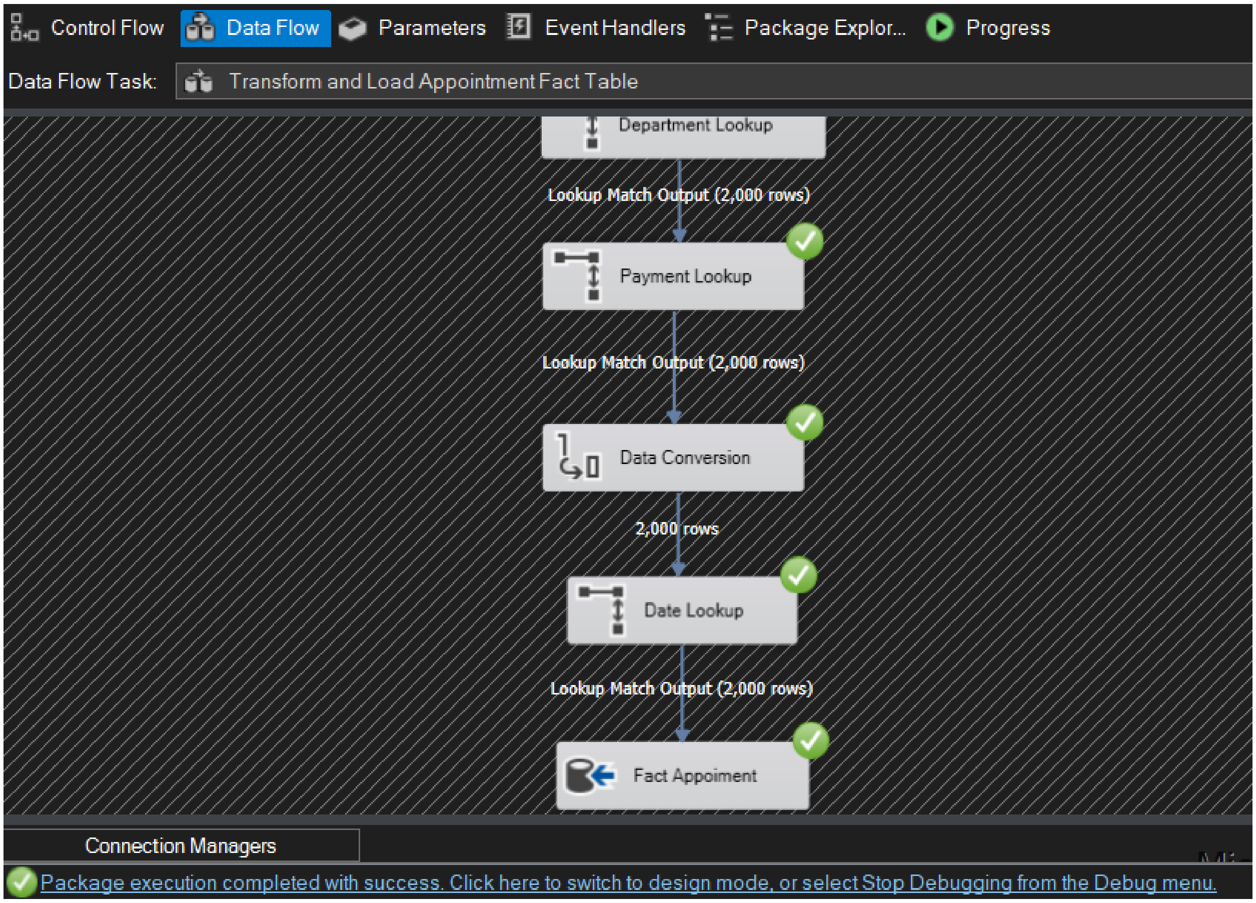
- Merged Appointment and Payment data using **Sort** and **Merge Join** transformations to align common keys (e.g., appointment_id).
- Applied **Lookup** transformations to fetch surrogate keys (SKs) from:
  - DimPatient
  - DimDoctor
  - DimPayment
  - DimDate
  - DimDepartment
- Performed **Derived Column** operations to calculate metrics like:
  - ProcessDurationHours = DATEDIFF(hour, appointment_create_time, appointment_complete_time)
- Converted data types where necessary to match the warehouse schema.





Data Flow Task: Transform and Load Appointment Fact Table

Package execution completed with success. Click here to switch to design mode, or select Stop Debugging from the Debug menu.



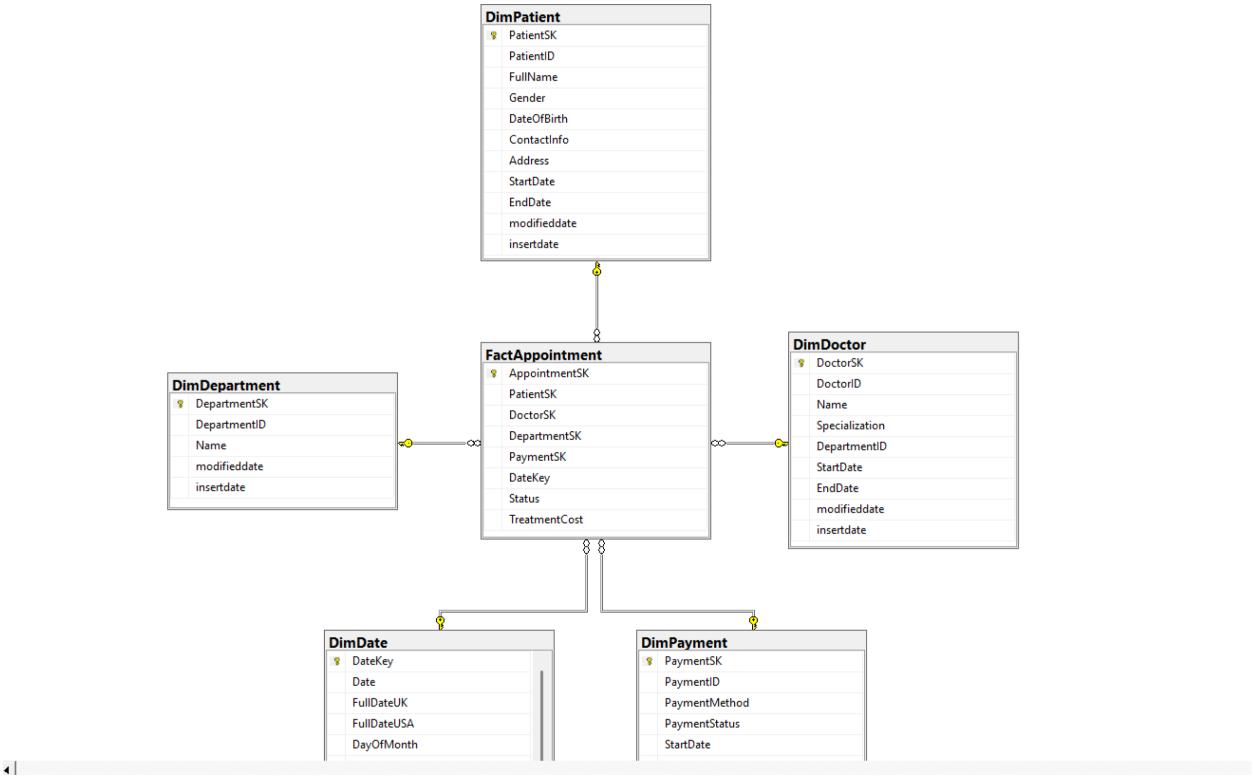
Data Flow Task: Transform and Load Appointment Fact Table

Package execution completed with success. Click here to switch to design mode, or select Stop Debugging from the Debug menu.

3. Load Phase

- The transformed dimension data was loaded into:
    - DimPatient, DimDoctor, DimDepartment, DimPayment, DimDate
- The cleaned and enriched appointment records were loaded into FactAppointment with all surrogate keys and calculated measures.

Dimension Table


DimPatient
PatientSK
PatientID
FullName
Gender
DateOfBirth
ContactInfo
Address
StartDate
EndDate
modifieddate
insertdate
DimDepartment
DepartmentSK
DepartmentID
Name
modifieddate
insertdate
FactAppointment
AppointmentSK
PatientSK
DoctorSK
DepartmentSK
PaymentSK
DateKey
Status
TreatmentCost
DimDoctor
DoctorSK
DoctorID
Name
Specialization
DepartmentID
StartDate
EndDate
modifieddate
insertdate
DimDate
DateKey
Date
FullDateUK
FullDateUSA
DayOfMonth
DimPayment
PaymentSK
PaymentID
PaymentMethod
PaymentStatus
StartDate