Data warehousing and Business Intelligence - IT3021

IT22047724 – Rathnasiri E.M.S.N

Assignment 1

Table of Contents

1. Data Selection 3
2. ER Diagram 4
3. Preparation Of Data Set 5
4. Architecture Solution 8
5. Data Warehouse Design and Development 9
6. ETL Development 14

Data Set Selection

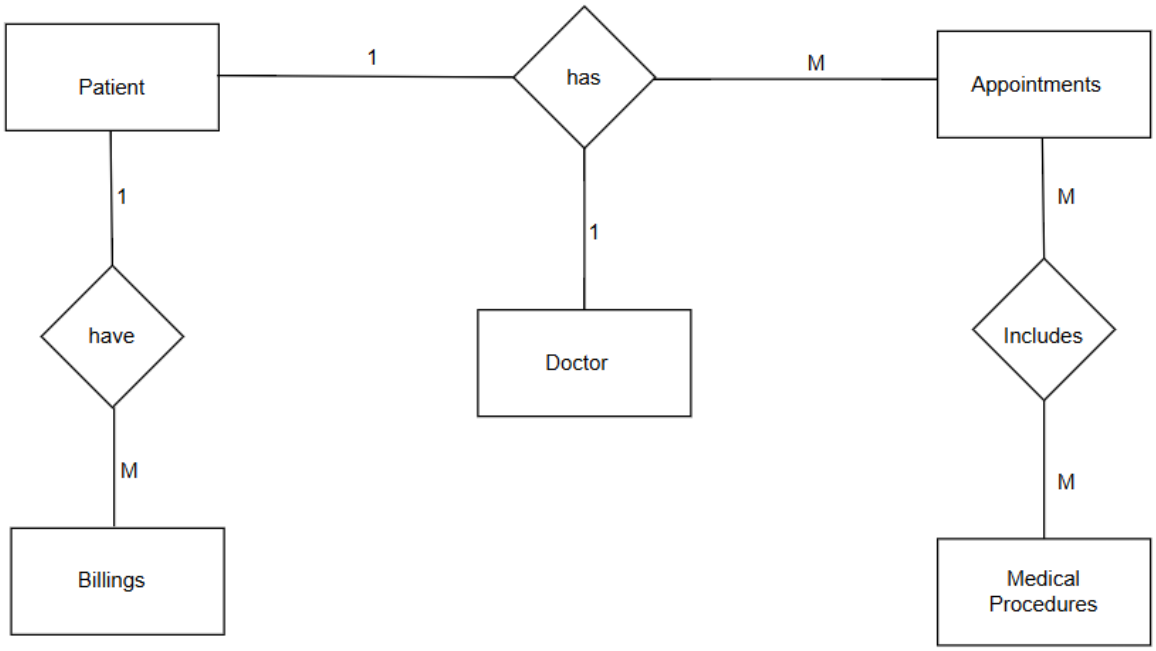
This dataset captures information related to healthcare appointments, billing, doctors, medical procedures, and patients in a hospital or clinic environment. It is modeled after a real-world OLTP (Online Transaction Processing) system where daily transactions such as patient registrations, medical consultations, procedure scheduling, and billing are recorded.

Entities and Their Purpose

1. **Patient**
    - Stores personal and contact details of the patients.
    - Attributes: PatientID, FirstName, LastName, Email
2. **Doctor**
    - Contains professional and contact information of doctors.
    - Attributes: DoctorID, DoctorName, Specialization, DoctorContact
3. **Appointment**
    - Represents a patient's appointment with a doctor.
    - Attributes: AppointmentID, PatientID, DoctorID, Date, Time
4. **Medical Procedure**
    - Holds details about procedures assigned during appointments (e.g., blood tests, scans).
    - Attributes: ProcedureID, ProcedureName, ProcedureType, Cost.
5. **Billing**
    - Tracks the payment information related to medical appointments and procedures.
    - Attributes: ProcedureID, ProcedureName, AppointmentID

Relationships

- A **Patient** can have multiple **Appointments**.
- A **Doctor** can attend to multiple **Appointments**.
- Each **Appointment** can include one or more **MedicalProcedures**.



Screenshots of the 1st two rows of datasets

Patient

PatientID	firstname	lastname	email
919	Anallese	Halla	Anallese.Halla@yopmail.com
535	Shel	Dearborn	Shel.Dearborn@yopmail.com
830	Mara	Shuler	Mara.Shuler@yopmail.com

Appointment

AppointmentID	Date	Time	PatientID	DoctorID
639	4/8/2022	2023-12-2	109	462
404	########	2023-12-2	823	774

Doctor

DoctorID	DoctorName	Specialization	DoctorContact
116	Shell	Infectious disea	.@yopmail.com
752	Regina	Oncologist	.@yopmail.com

Medical Procedures

Procedure	ProcedureName	AppointmentID
432	Kidney transplant	955
574	Allergy testing	701

Billing

InvoiceID	PatientID	Items	Amount
e9609dce-f125-4f65-a067-4760a06e60b6	894	Immunizations	956065
fb593fb6-466a-4dc8-9a61-afbd9d4b5031	448	Cataract surgery	188997

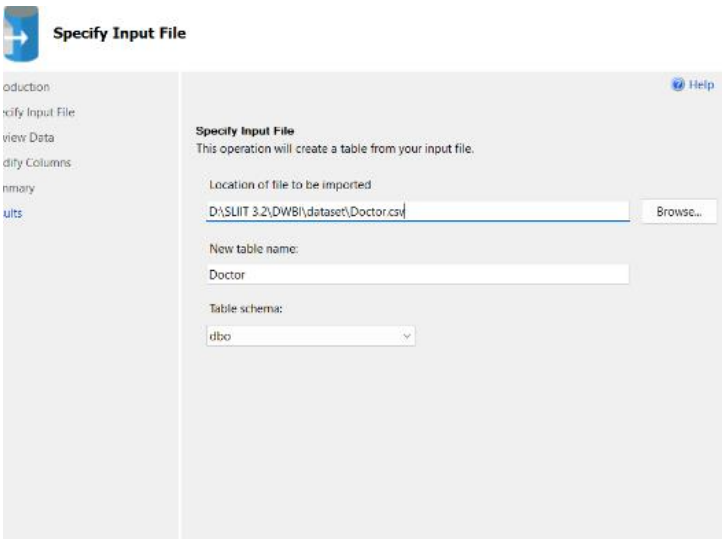
Preparation of Data Sources

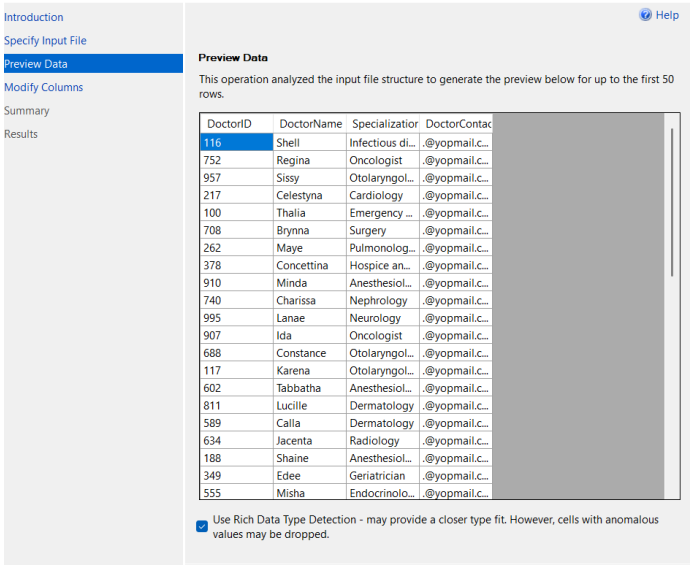
Regarding the data sources, 4 files have been imported in csv format and the remaining file has been imported in text format. They are described below.

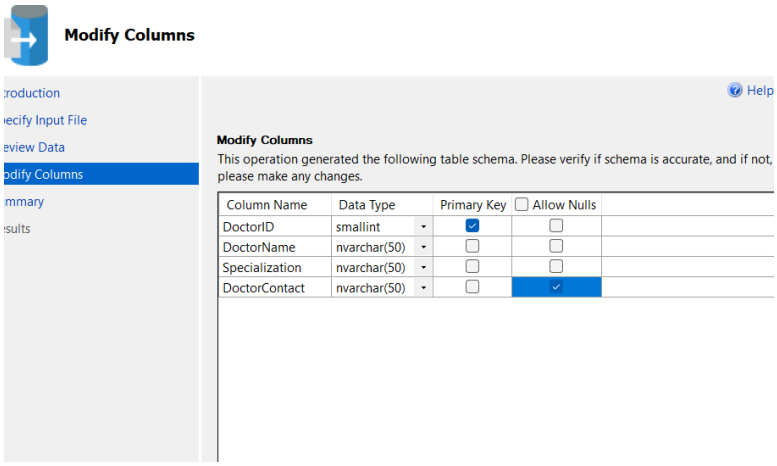
File name	Format	Key Information
Patient.csv	CSV	**Patient demographics:** PatientID, FirstName, LastName, Email
Doctor.csv	CSV	**Doctor details:** DoctorID, DoctorName, Specialization
Appointment.csv	CSV	**Appointment information:** AppointmentID, PatientID, DoctorID, Date, Time
Billing.csv	CSV	**Billing transactions:** InvoiceID, PatientID, DoctorID, Date, Time
MedicalProcedure.txt	Text(Tab)	**Medical Procedure:** ProcedureID, ProcedureName, AppointmentID

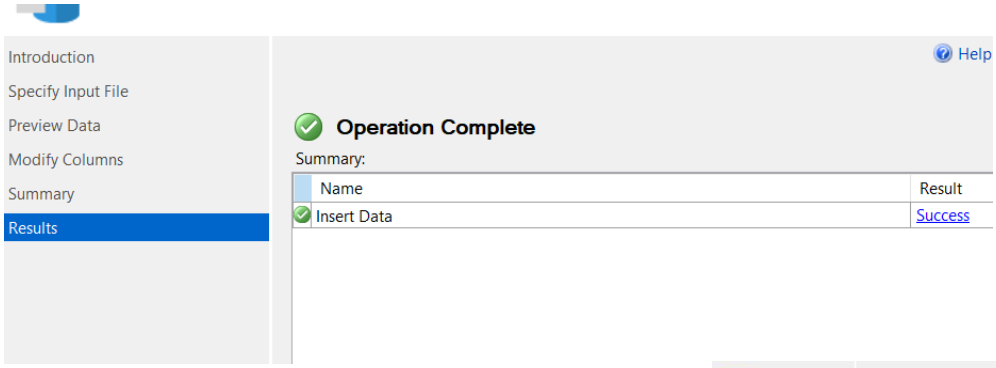
Screenshots of the preparation of datasets in different formats

Doctor.csv:

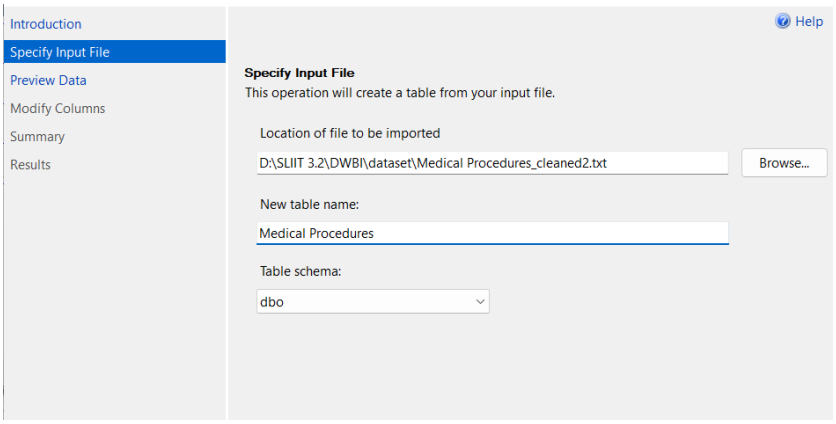


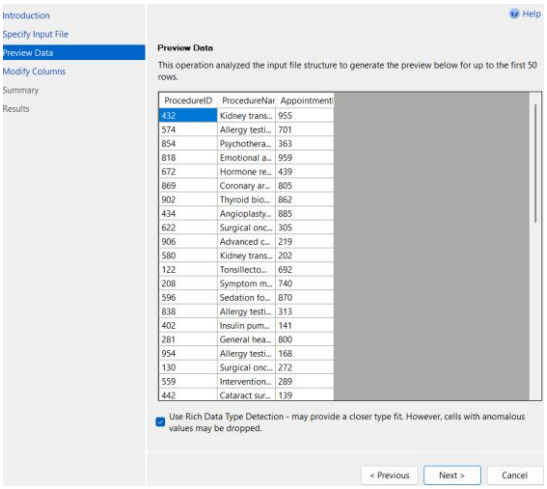


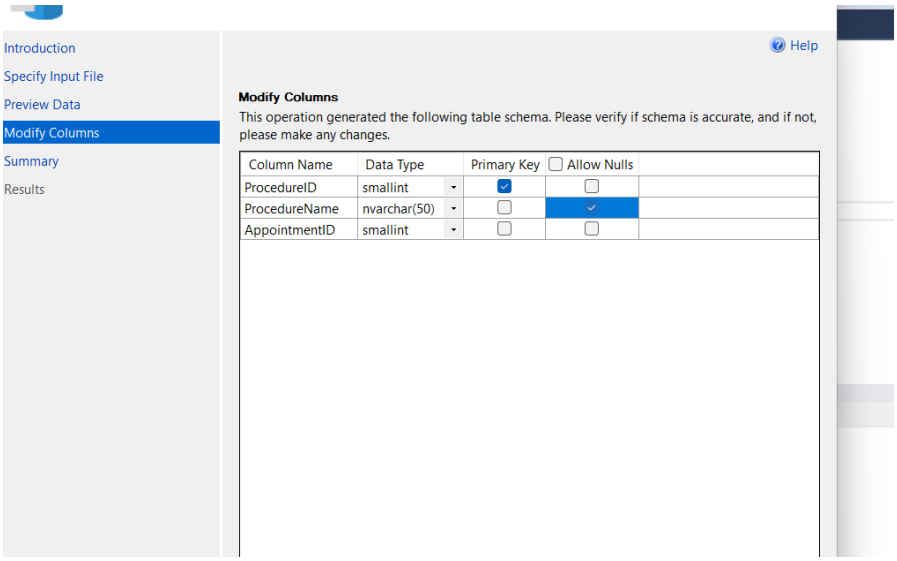


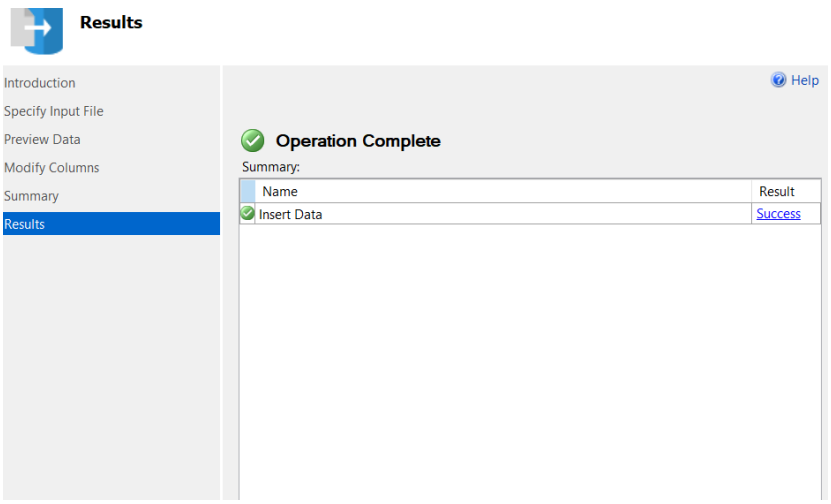


MedicalProcedure.txt



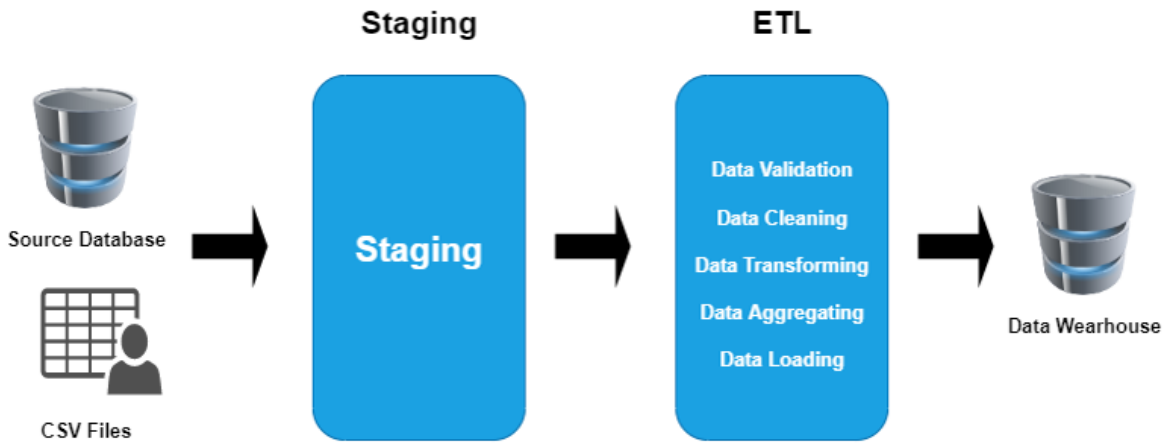






Architecture Solution

This high-level DW & BI solution architecture illustrates the flow of data from sources to insights. It highlights the key components: data sources, ETL processes, the data warehouse, analytical tools, and end-users, showing how data is processed and utilized for business intelligence.



Data Warehouse Design and Development

In this step, a dimensional model was designed and implemented in SQL Server to support analytical queries on the healthcare dataset. A **star schema** was used, with a central **FactAppointment** table surrounded by four dimension tables: DimPatient, DimDoctor, DimProcedure, and DimDate.

The **FactAppointment** table stores transactional data such as appointment ID, billing amount, and timestamps for tracking appointment processing time. It also includes foreign keys to the dimension tables for analysis.

The **DimPatient** table was implemented as a **Slowly Changing Dimension (Type 2)** to track changes in patient information over time (e.g., name or email changes). This enables accurate historical reporting.

All dimension and fact tables were created in SQL Server using appropriate keys and data types. Surrogate keys (auto-increment IDs) were used for dimensions, while natural keys (like AppointmentID) were retained for tracking source data.

This schema enables rich analysis, such as:

- Appointments by doctor specialization
- Revenue by procedure type
- Appointment completion time trends

- Fact Table

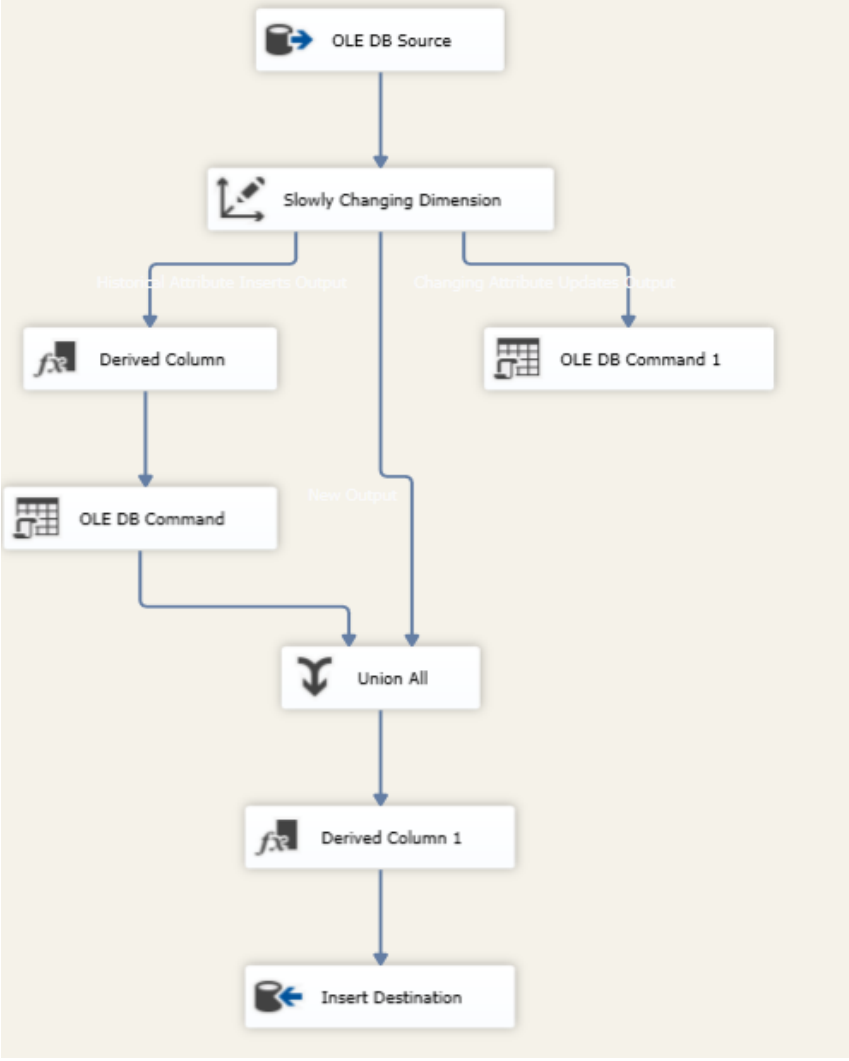
FactAppointment table stores metrics like BillingAmount, AppointmentCount and links to to the dimensions.

```
CREATE TABLE FactAppointment (
    FactAppointmentID INT IDENTITY(1,1) PRIMARY KEY,
    AppointmentID INT,
    PatientKey INT FOREIGN KEY REFERENCES DimPatient(PatientKey),
    DoctorKey INT FOREIGN KEY REFERENCES DimDoctor(DoctorKey),
    ProcedureKey INT FOREIGN KEY REFERENCES DimProcedure(ProcedureKey),
    DateID INT FOREIGN KEY REFERENCES DimDate(DateKey),
    Amount VARCHAR(50),
    accm_txn_create_time DATETIME,
    accm_txn_complete_time DATETIME,
    txn_process_time_hours INT
);
```

- Dimension Tables

DimPatient tracks patient demographic information **(Slowly Changing Dimentions)**

```
CREATE TABLE DimPatient (
    PatientKey INT IDENTITY(1,1) PRIMARY KEY,
    PatientID INT,
    FirstName VARCHAR(100),
    LastName VARCHAR(100),
    Email VARCHAR(100),
    IsCurrent BIT,
    StartDate DATE,
    EndDate DATE
);
```



DimDoctor provides details about doctors and their expertise

```
CREATE TABLE DimDoctor (
    DoctorKey INT IDENTITY(1,1) PRIMARY KEY,
    DoctorID VARCHAR(50),
    DoctorName VARCHAR(100),
    Specialization VARCHAR(100),
    DoctorContact VARCHAR(50)
);
```

DimDate contains the common time dimensions for appointments and billing

```
CREATE TABLE DimDate (
    DateKey INT PRIMARY KEY,          -- Format: YYYYMMDD
    FullDate DATE NOT NULL,
    Day INT,
    Month INT,
    Year INT,
    Quarter INT,
    DayOfWeek INT,                    -- 1=Sunday, 7=Saturday
    DayName VARCHAR(10),
    MonthName VARCHAR(15),
    IsWeekend BIT
);
```

DimMedicalProcedure

```
CREATE TABLE DimProcedure (
    ProcedureKey INT IDENTITY(1,1) PRIMARY KEY,
    ProcedureID VARCHAR(50),
    ProcedureName VARCHAR(100)
);
```

Testing and Validation

These tests were performed to ensure that the schema runs correctly.

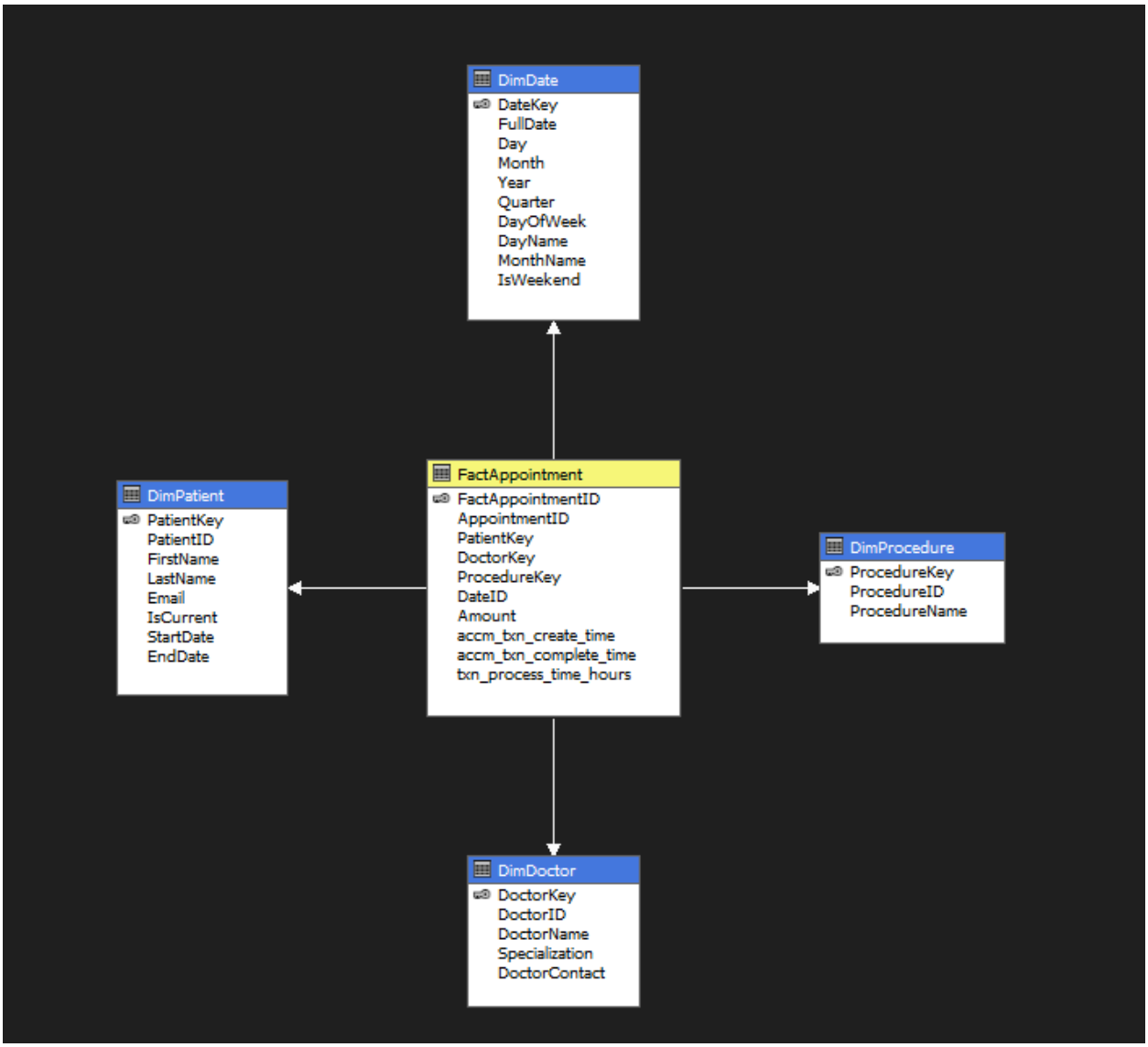
```
select D.Specialization, count(A.AppointmentID) as 'Number of appointments'
from FactAppointment A
JOIN DimDoctor D on A.DoctorKey = D.DoctorKey
GROUP BY D.Specialization
```

Results | Messages

Specialization	Number of appointments
Allergists	2
Anesthesiology	4
Cardiology	2
Critical Care Medicine	1
Dermatology	4
Emergency Medicine	4
Endocrinologist	1
Family Medicine	1
Gastroenterology	4
Geriatrician	4
Hospice and Palliati...	5
Infectious disease	4
Internists	5
Nephrology	1
Obstetric Anesthesio...	3
Oncologist	5

Relational Diagram

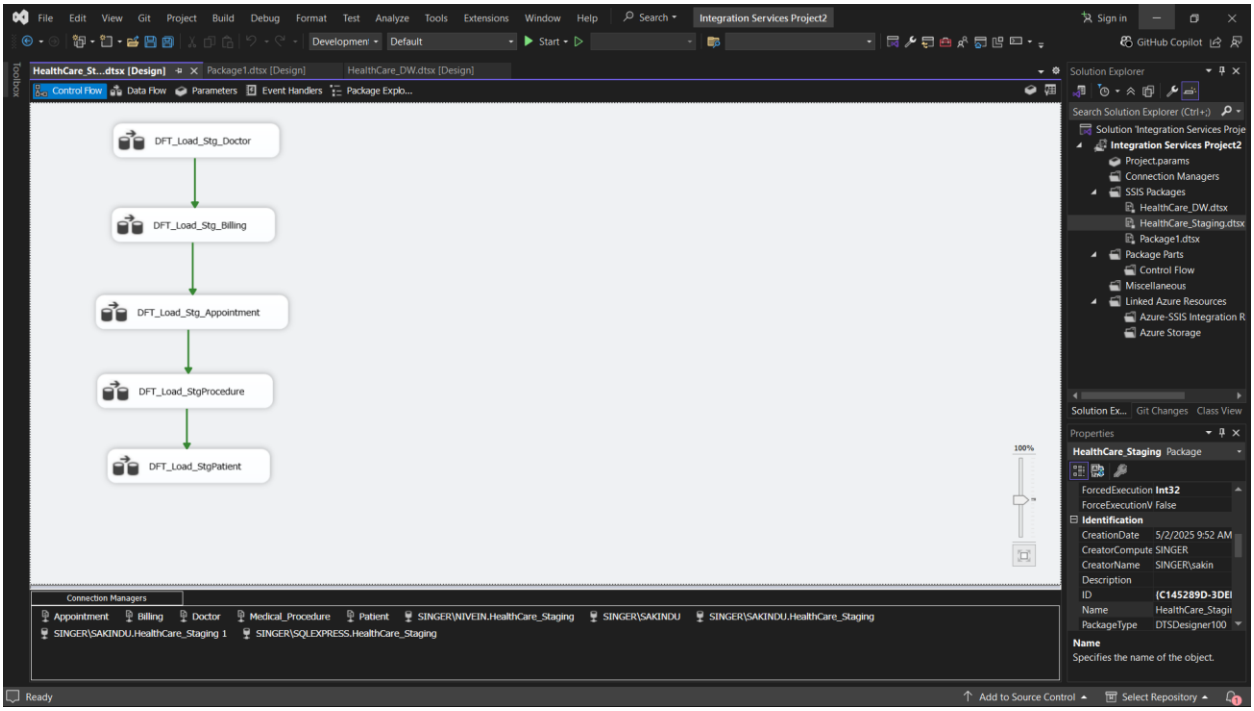
The relational diagram shows how the FactAppointments table connects to the dimension tables- DimPatient, DimDoctor, DimDate and DimMedicalProcedure through foreign key relationships. It illustrates the logical structure of the schema enabling efficient query and analysis.



ETL Development

Extract Phase

- Created SSIS package called Healthcare_Staging.dtsx.
- Extracted data from various sources (CSV and Text Files) using Flat File Source.



SQLQuery8.sql - SI...(SINGER\sakin (64)) SQLQuery7.sql - SI...(S

```sql
CREATE TABLE Stg_Patient (
    PatientID INT,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Email VARCHAR(100)
);

CREATE TABLE Stg_MedicalProcedure (
    ProcedureID VARCHAR(50),
    ProcedureName VARCHAR(100),
    AppointmentID VARCHAR(50)
);

CREATE TABLE Stg_Doctor (
    DoctorID VARCHAR(50),
    DoctorName VARCHAR(100),
    Specialization VARCHAR(100),
    DoctorContact VARCHAR(50)
);

CREATE TABLE Stg_Billing (
    InvoiceID VARCHAR(50),
    PatientID INT,
    Items VARCHAR(100),
    Amount VARCHAR(50)
);

CREATE TABLE Stg_Appointment (
    AppointmentID VARCHAR(50),
    Date DATE,
    Time VARCHAR(50),
    PatientID INT,
    DoctorID VARCHAR(50)
);
```

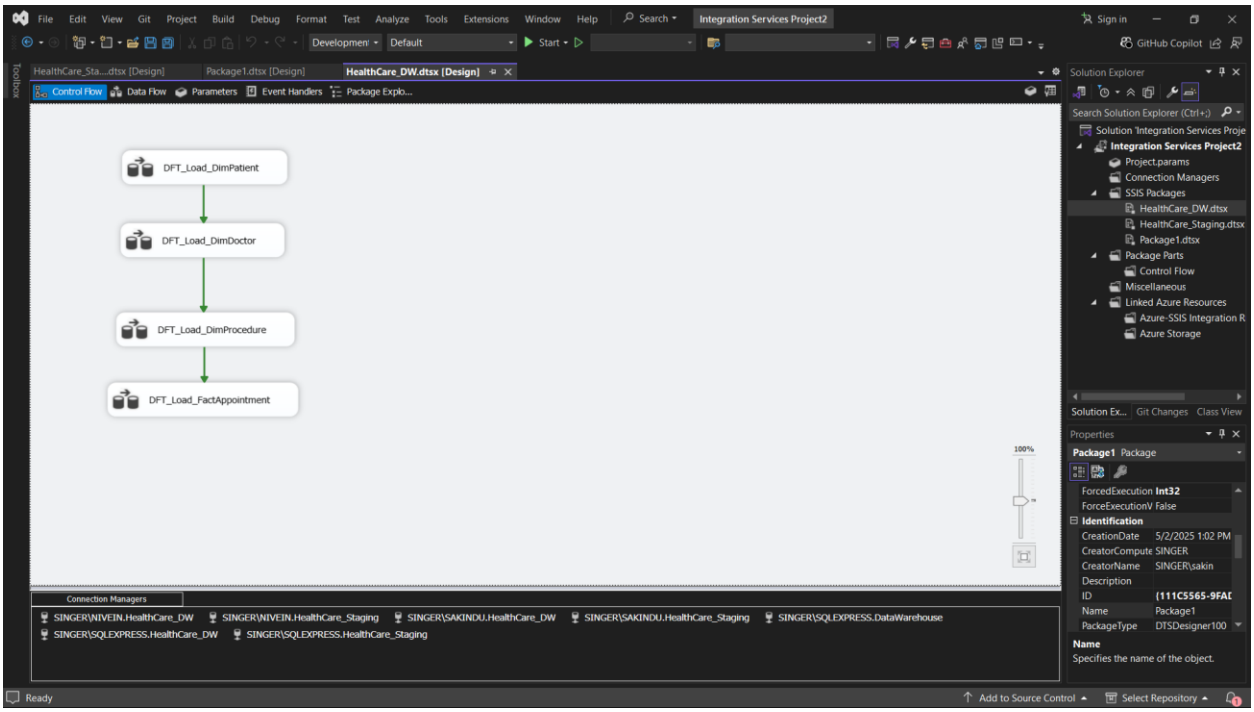
Transformation Phase

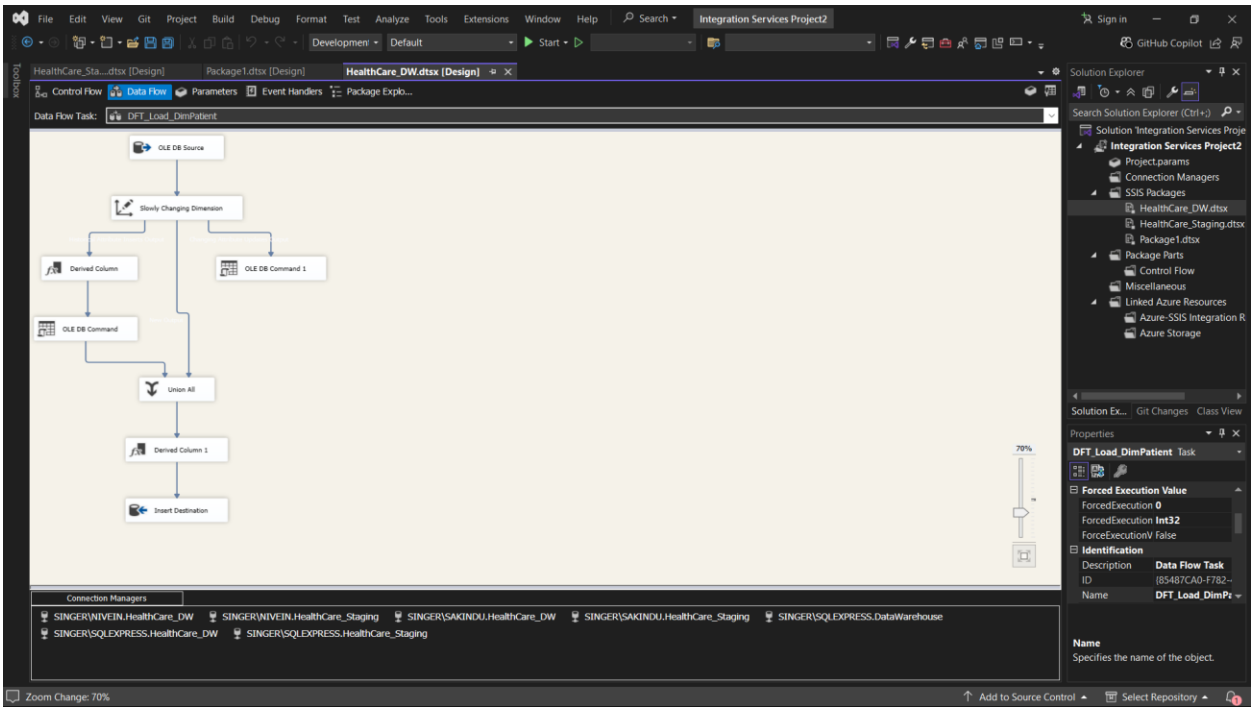
- Validated staged data using SQL queries
    - Checked for missing or NULL values in key tables.
    - Removed duplicate records from staging tables.

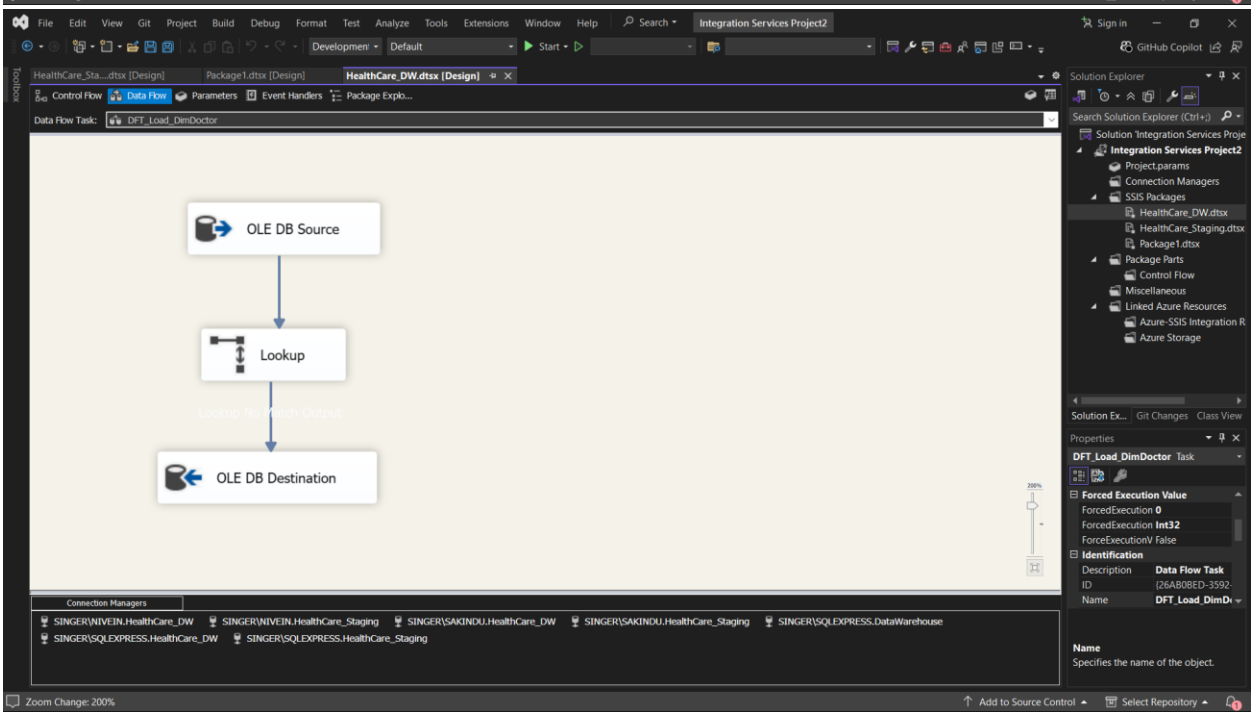
```
SELECT * FROM Staging_Billing WHERE PatientID NOT IN (SELECT PatientID FROM Staging_Patient);
SELECT * FROM Staging_MedicalProcedures s WHERE ProcedureName IS NULL;
SELECT ProcedureID, COUNT(*) FROM Staging_MedicalProcedures GROUP BY ProcedureID HAVING COUNT(*) > 1;
```

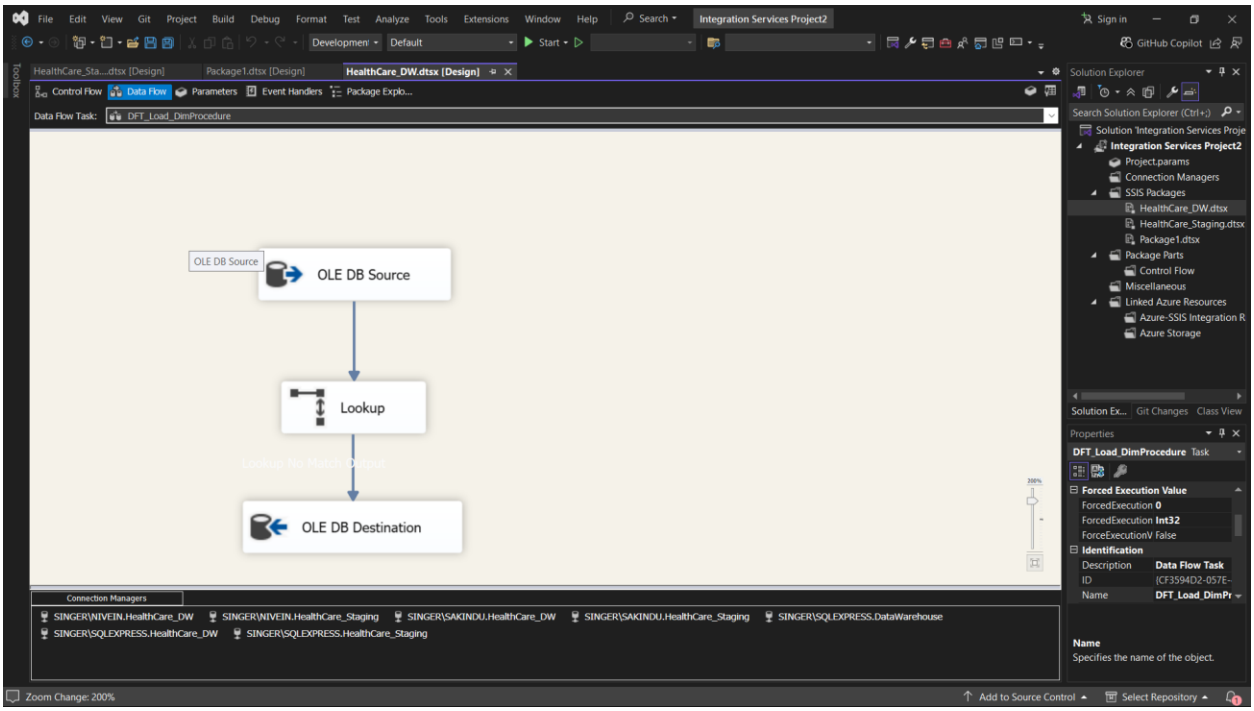
Load Phase

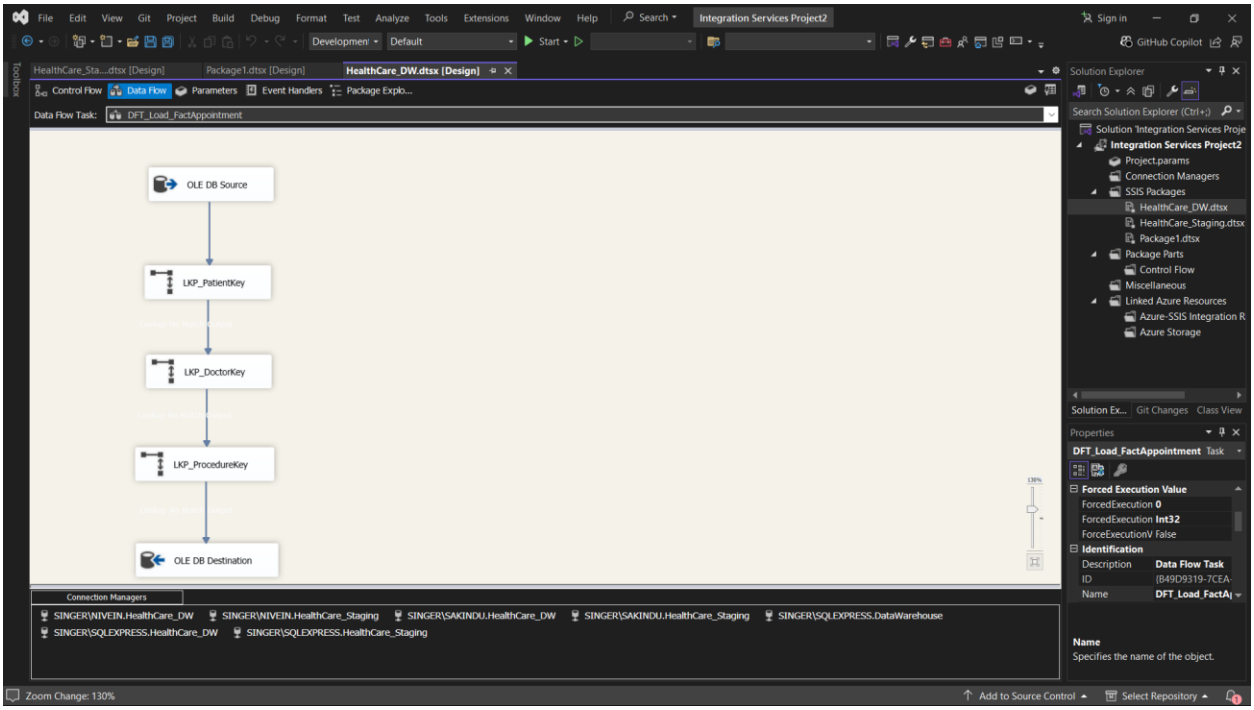
- Created SSIS Package called Healthcare_DW.dtsx for Data Warehouse Loading.
- Created HealthCare_DW database and created the tables for patient, doctor, appointments, medical procedure and billing.
- Set up the OLE DB source connections to extract data from staging.
- Mapped transformed staging data into HealthCare_DW tables using OLE DB Destination.
- Executed package to load warehouse tables.




File Edit View Git Project Build Debug Format Test Analyze Tools Extensions Window Help
Integration Services Project2
HealthCare_DW.dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Package Explo...
Data Flow Task:
DFT_Load_DimPatient
OLE DB Source
Slowly Changing Dimension
Derived Column
OLE DB Command 1
OLE DB Command
Union All
Derived Column 1
Insert Destination
Connection Managers
SINGER\NIVEIN.HealthCare_DW
SINGER\NIVEIN.HealthCare_Staging
SINGER\SAKINDU.HealthCare_DW
SINGER\SAKINDU.HealthCare_Staging
SINGER\SQLEXPRESS.DataWarehouse
SINGER\SQLEXPRESS.HealthCare_DW
SINGER\SQLEXPRESS.HealthCare_Staging
Solution Explorer
Integration Services Project2
Project.params
Connection Managers
SSIS Packages
HealthCare_DW.dtsx
HealthCare_Staging.dtsx
Package1.dtsx
Package Parts
Control Flow
Miscellaneous
Linked Azure Resources
Azure-SSIS Integration R
Azure Storage
Properties
DFT_Load_DimPatient Task
Forced Execution Value
ForcedExecution 0
ForcedExecution Int32
Identification
Description Data Flow Task
Name DFT_Load_DimPa
Name
Specifies the name of the object.
Zoom Change: 70%
Add to Source Control
Select Repository


File Edit View Git Project Build Debug Format Test Analyze Tools Extensions Window Help
Integration Services Project2
HealthCare_DW.dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Package Explo...
Data Flow Task:
DFT_Load_DimDoctor
OLE DB Source
Lookup
OLE DB Destination
Connection Managers
SINGER\NIVEIN.HealthCare_DW
SINGER\NIVEIN.HealthCare_Staging
SINGER\SAKINDU.HealthCare_DW
SINGER\SAKINDU.HealthCare_Staging
SINGER\SQLEXPRESS.DataWarehouse
SINGER\SQLEXPRESS.HealthCare_DW
SINGER\SQLEXPRESS.HealthCare_Staging
Solution Explorer
Integration Services Project2
Project.params
Connection Managers
SSIS Packages
HealthCare_DW.dtsx
HealthCare_Staging.dtsx
Package1.dtsx
Package Parts
Control Flow
Miscellaneous
Linked Azure Resources
Azure-SSIS Integration R
Azure Storage
Properties
DFT_Load_DimDoctor Task
Forced Execution Value
ForcedExecution 0
ForcedExecution Int32
Identification
Description Data Flow Task
Name DFT_Load_DimDo
Name
Specifies the name of the object.
Zoom Change: 200%
Add to Source Control
Select Repository


OLE DB Source
OLE DB Source
Lookup
Lookup No Match Output
OLE DB Destination
Data Flow Task: DFT_Load_DimProcedure
Zoom Change: 200%

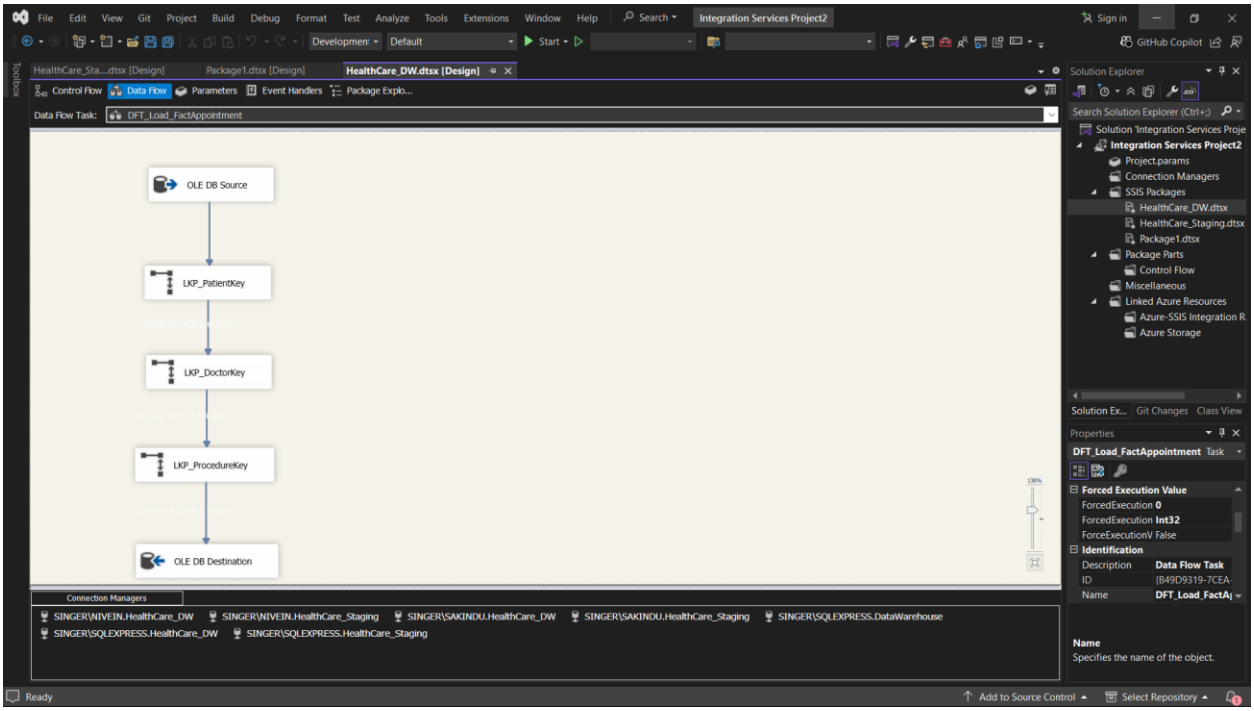

OLE DB Source
LKP_PatientKey
LKP_DoctorKey
LKP_ProcedureKey
OLE DB Destination
Data Flow Task: DFT_Load_FactAppointment
Zoom Change: 130%

ETL Development – Accumulating Fact Tables

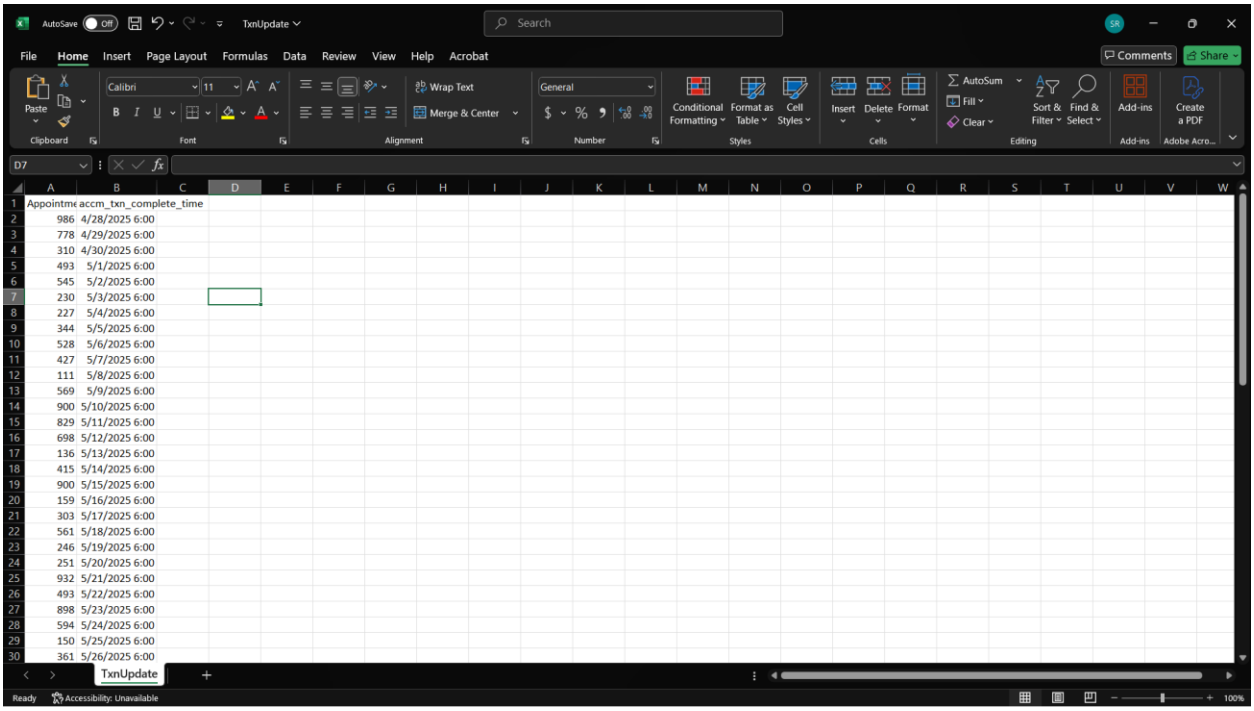
- Created fact table called Staging_FactAppintments

```
CREATE TABLE FactAppointment (
    FactAppointmentID INT IDENTITY(1,1) PRIMARY KEY,
    AppointmentID INT,
    PatientKey INT FOREIGN KEY REFERENCES DimPatient(PatientKey),
    DoctorKey INT FOREIGN KEY REFERENCES DimDoctor(DoctorKey),
    ProcedureKey INT FOREIGN KEY REFERENCES DimProcedure(ProcedureKey)
    DateID INT FOREIGN KEY REFERENCES DimDate(DateKey),
    Amount VARCHAR(50),
    accm_txn_create_time DATETIME,
    accm_txn_complete_time DATETIME,
    txn_process_time_hours INT
);
```

- In the HealthCare_DW package the stage appointments were extracted using OLE DB Source and mapped into FactAppointments using OLE DB Destination.



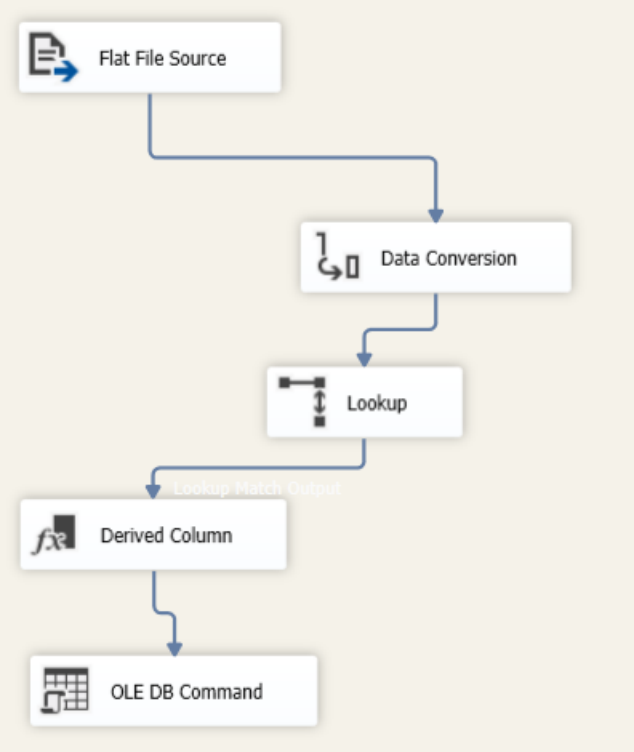
- Prepared a transaction completion dataset called TxnCompletionUpdates and stored completion timestamps in a CSV file.



Appointme	accm_txn_complete_time
986	4/28/2025 6:00
778	4/29/2025 6:00
310	4/30/2025 6:00
493	5/1/2025 6:00
545	5/2/2025 6:00
230	5/3/2025 6:00
227	5/4/2025 6:00
344	5/5/2025 6:00
528	5/6/2025 6:00
427	5/7/2025 6:00
111	5/8/2025 6:00
569	5/9/2025 6:00
900	5/10/2025 6:00
829	5/11/2025 6:00
698	5/12/2025 6:00
136	5/13/2025 6:00
415	5/14/2025 6:00
900	5/15/2025 6:00
159	5/16/2025 6:00
303	5/17/2025 6:00
561	5/18/2025 6:00
246	5/19/2025 6:00
251	5/20/2025 6:00
932	5/21/2025 6:00
493	5/22/2025 6:00
898	5/23/2025 6:00
594	5/24/2025 6:00
150	5/25/2025 6:00
361	5/26/2025 6:00

- Created a SSIS package called AccumulatingFactTable
- Used Lookup Transformation to match transactions in FactAppointments.

- Applied OLE DB Command to update accm_txn_complete_time.
- Derived Column Transformation for txn_process_time_hours Calculation
- Used DATEDIFF (hour, accm_txn_create_time, accm_txn_complete_time) in SSIS.
- Updated the fact table with transaction processing duration.


Flat File Source
Data Conversion
Lookup
Lookup Match Output
Derived Column
OLE DB Command