Sri Lanka Institute of Information Technology

Assignment I

Data Warehouse & Business Intelligence

2022

Submitted By:

Sandeepanie W.D.N

IT20263294

Contents

Data set selection & Preparation

Step 1 – Data set Selection

The selected data source is a collection of transactional data. The link to the source data set is mentioned below:

https://www.kaggle.com/datasets/joniarroba/noshowappointments

Modifications were done accordingly to the data set derived from the source. This data set can be utilized to derive insightful information regarding the medical appointments in a certain medical center. Stakeholders such as physicians and patients are involved in the system. Patients can book one or more than one appointment to their preferred physician. Physicians are pre allocated to the rooms in wards. Same room can be assigned to several physicians in different time slots. Further, the data regarding the ratings and reviews of appointments which were published by patients are also included in the dataset.

Step 2 – Preparation of Data sources

The two main sources are listed below:
SQL Database

One text file – Patients Locations Data

Also, the below mentioned **CSV files** were imported to the SQL source database.

Appointments Transaction Data
Physicians Details
Patients Data
Allocated Rooms Data
Ratings and Reviews Data
Appointment Due Date Data

Description of the data set:

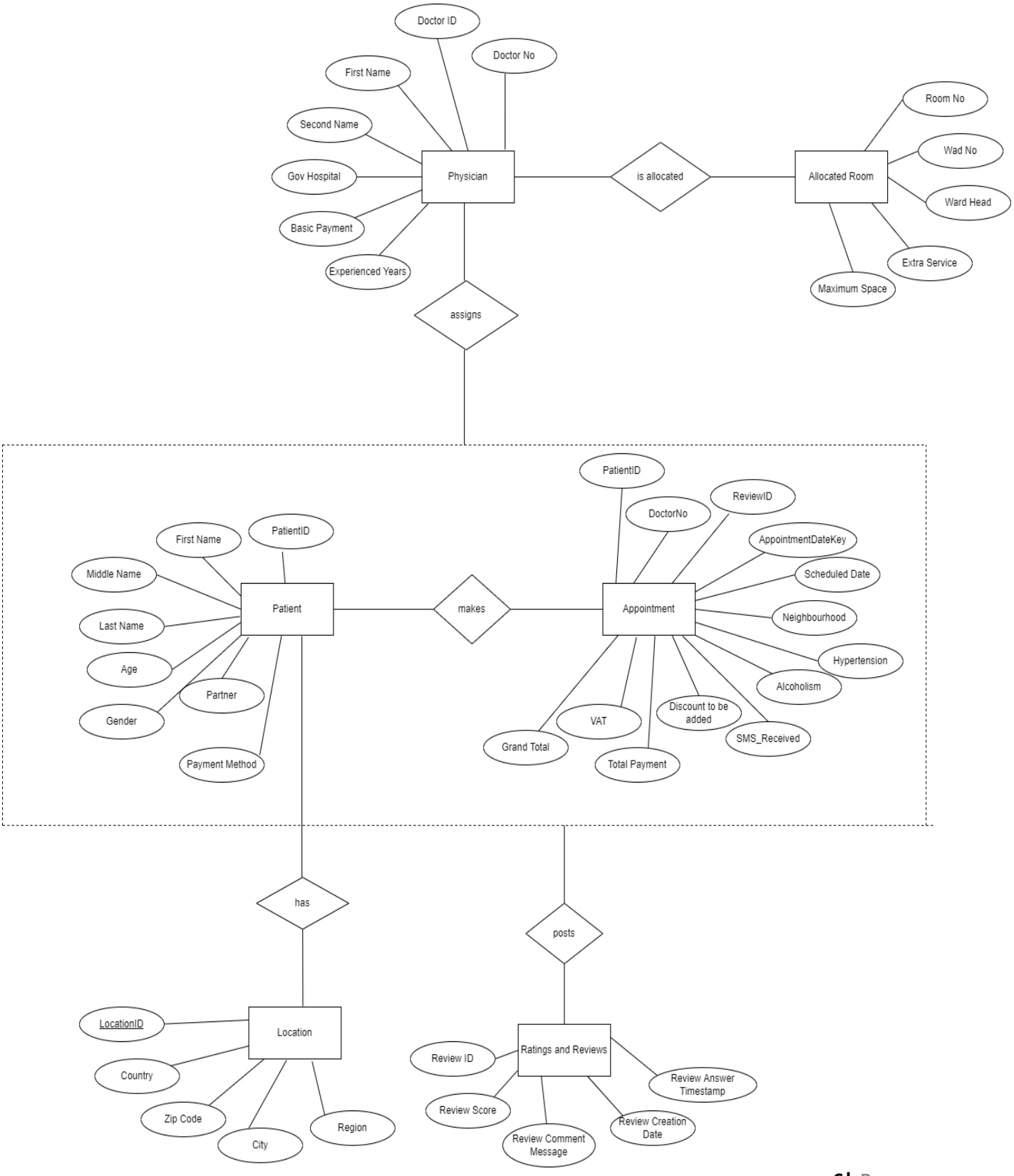
Table name	Column name	Data type	Description
Appointments	appointment_id	int	summary of appointments
	patient_id	int	
	review_id	int	
	doctor_no	int	
	scheduled_date	datetime	
	neighbourhood	nvarchar(MAX)	
	hypertension	bit	
	alcoholism	bit	
	SMS_received	bit	
	Discount_to_be_Added	int	
	Total_Payment	int	
	VAT	float	
	Grand_Total	float	
Patients	patient_id	int	Details of promotion campaigns
	first name	nvarchar(50)	

	middle name	nvarchar(50)	
	last name	nvarchar(50)	
	age	int	
	gender	nvarchar(50)	
	partner	bit	
	payment_method	nvarchar(50)	
Reviews	review_id	int	Customer Details
	review score	int	
	review comment message	nvarchar(MAX)	
	review creation date	datetime	
	review answer timestamp	datetime	
Physicians	doctor_no	int	Details of Physicians
	doctor_id	varchar(50)	
	first name	nvarchar(50)	
	second name	nvarchar(50)	
	govhospital	nvarchar(MAX)	
	basic payment	float	
	experienced years	int	
Allocated Rooms	room no	int	Details of allocated rooms
	ward no	int	
	ward head	nvarchar(50)	
	extra service	nvarchar(MAX)	
	minimum space	int	

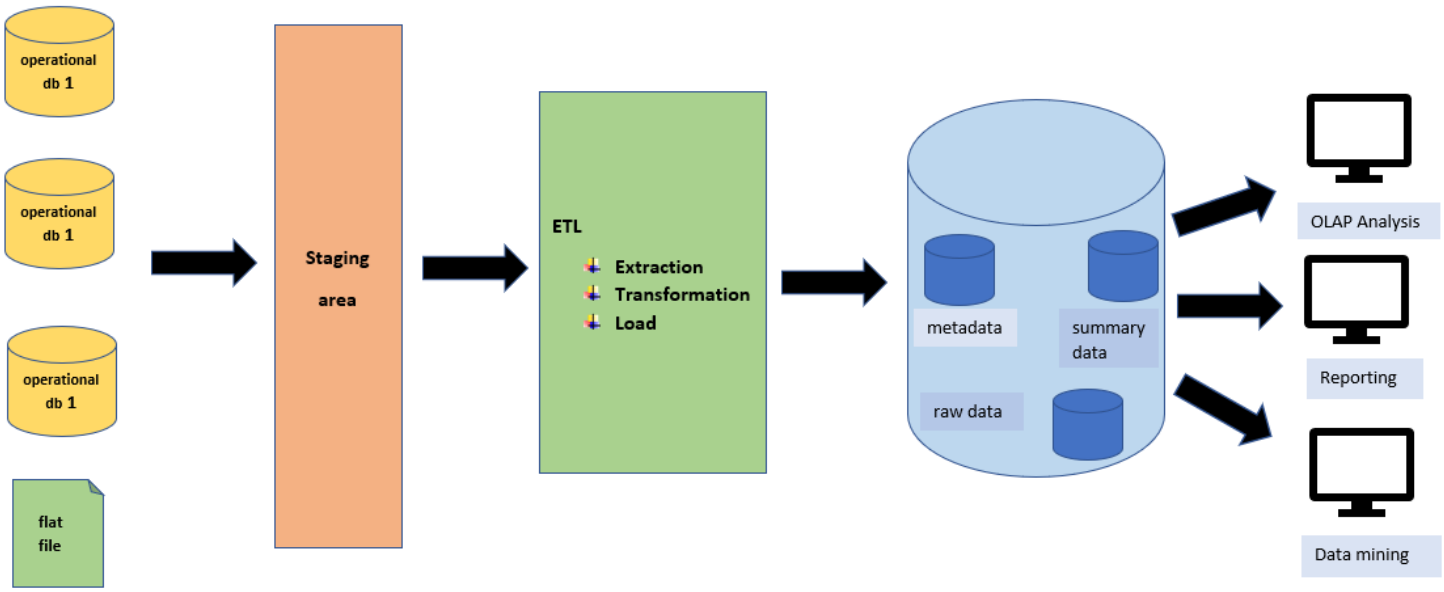
Step 3 – Solution Architecture

ER Diagram

This diagram shows the connection between the entities in the data set



Solution Architecture



First the intention is to transform the gathered row data to a SSIS package (Staging). Although it is optional, truncating all data into a backup package ensures the safety of data. For each source data table in the source database, we should have separate staging data table. After the staging layer the below mentioned staging tables are created:

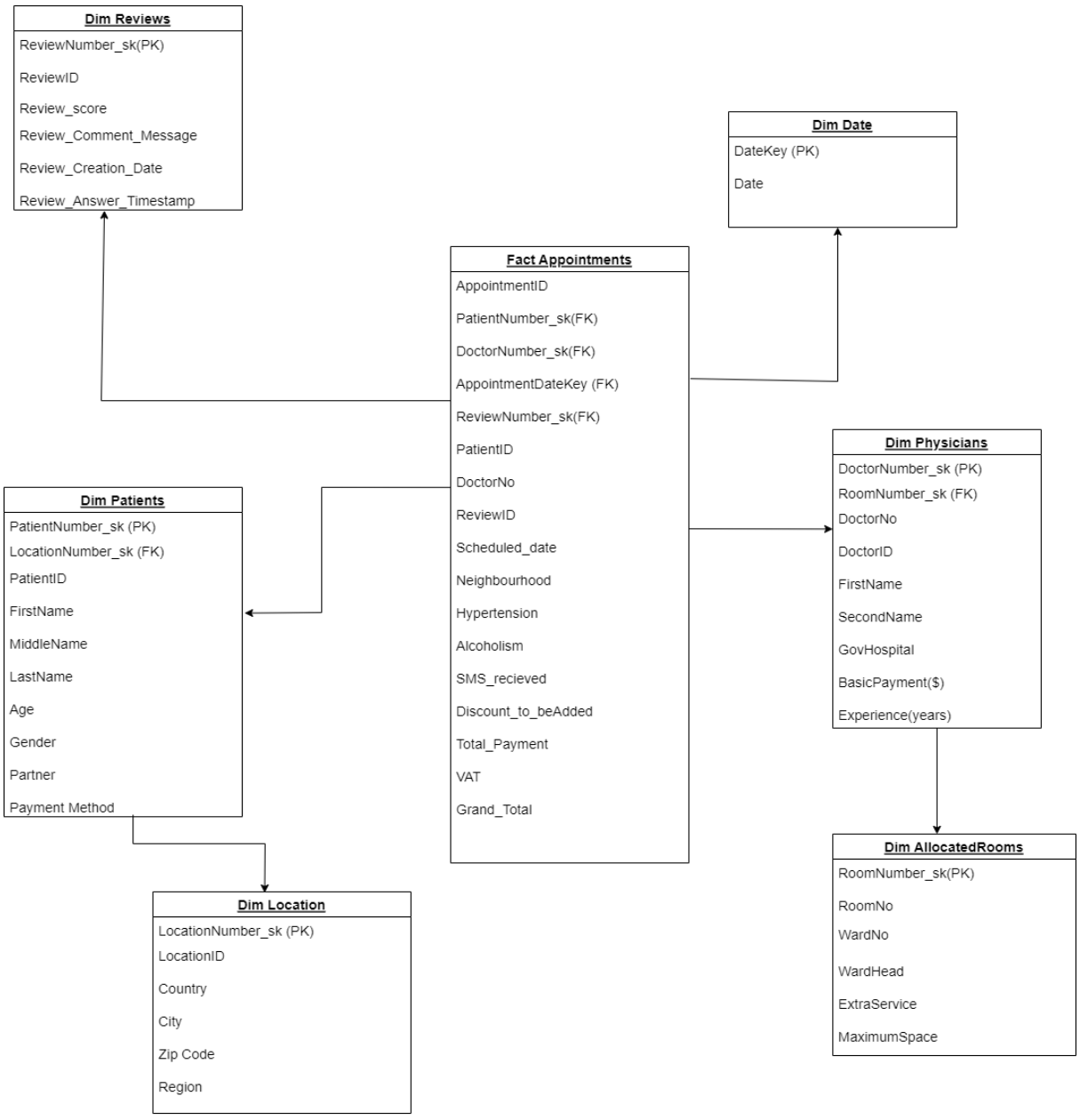
1. Appointments Transaction Staging
2. Physicians Staging
3. Patients Staging
4. Patients Locations Staging
1. Allocated Rooms Staging
1. Ratings and Reviews Staging

Source Table	Data Flow Task	OLE DB Destination Task	Staging Table	Event Handler SQL Task
dbo.Physicians	Extract Physicians Data to staging	Load data to **dbo.Stg.Physicians** Staging table	**dbo.Stg.Physicians**	Truncate Physicians Staging Table
dbo.Patients	Extract Patients Data to staging	Load data to **dbo.Stg.Patients** Staging table	**dbo.Stg.Patients**	Truncate Patients Staging Table
dbo.Patients Locations	Extract Patients Locations Data to Staging	Load data to **dbo.Stg.PatientLocations** Staging table	**dbo.Stg.PatientLocations**	Truncate Patients Locations Staging Table
dbo.Allocated Rooms	Extract Allocated Rooms Data to Staging	Load data to **dbo.Stg.AllocatedRooms** Staging table	**dbo.Stg.AllocatedRooms**	Truncate Allocated Rooms Staging Table
dbo.Ratings and Reviews	Extract Ratings and Reviews Data to Staging	Load data to **dbo.Stg.RatingsandReviews** Staging table	**dbo.Stg.RatingsandReviews**	Truncate Ratings and Reviews Staging Table

Next staged tables are profiled, and aggregations are performed when necessary. As the next step data is transformed and loaded. After completing the described stages, data is tested and validated and the Datawarehouse is created.

After the warehouse is created BI results such as OLAP analysis, Reports, Data visualization, Data mining can be obtained as results after further modifications.

Step 4: Data warehouse design and development



Snowflake schema is used to design the Datawarehouse design. There is one fact table and there are four different dimensional tables. Also, the appointments per patient was considered as the grain when designing.

Assumptions.

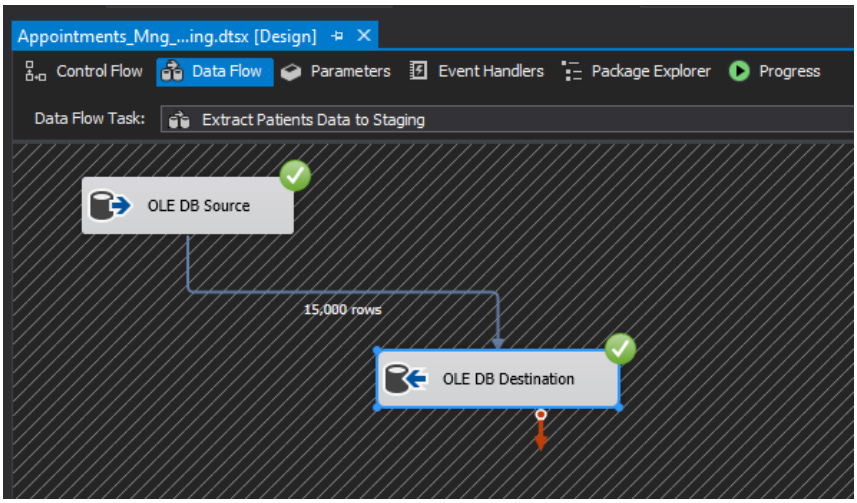
Patients table was considered as a slowly changing dimension.

Step 5 - ETL Development

As the first step data was extracted from the sources (DB source & text file). For every extraction, data flow tasks were used, and data was extracted from the source to the staging table. Then for every staging table a truncate table was created. All the data flow tasks were joined as shown below at the end:

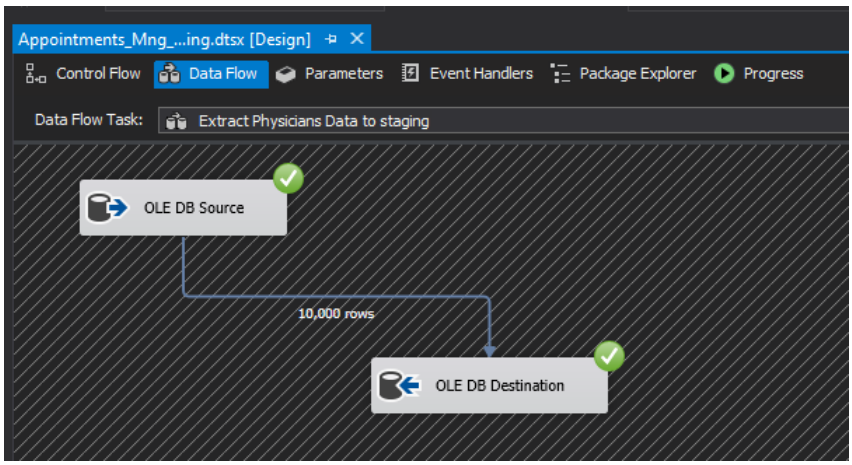
Screenshots of all the data sources that were staged, and truncate tables created are attached below:

Staging Patients Details



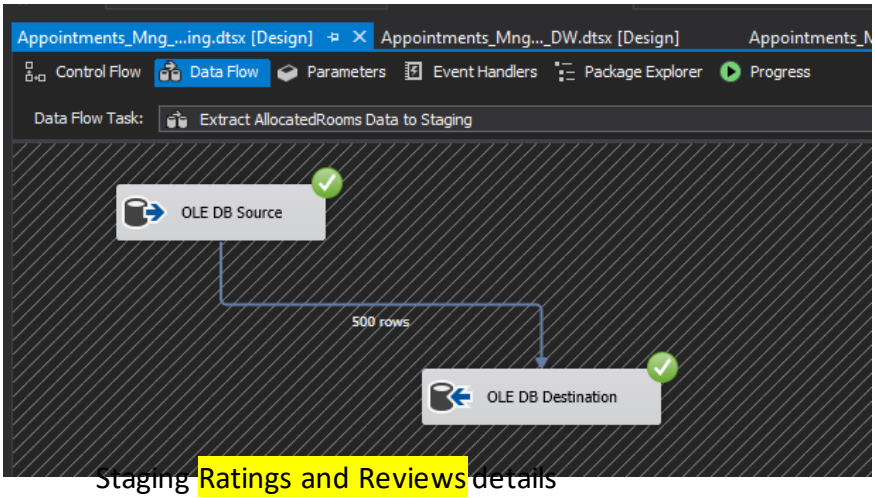
Patients' data is extracted from the Patients table in source database and inserted to Patients Staging Table

Staging Physicians Details



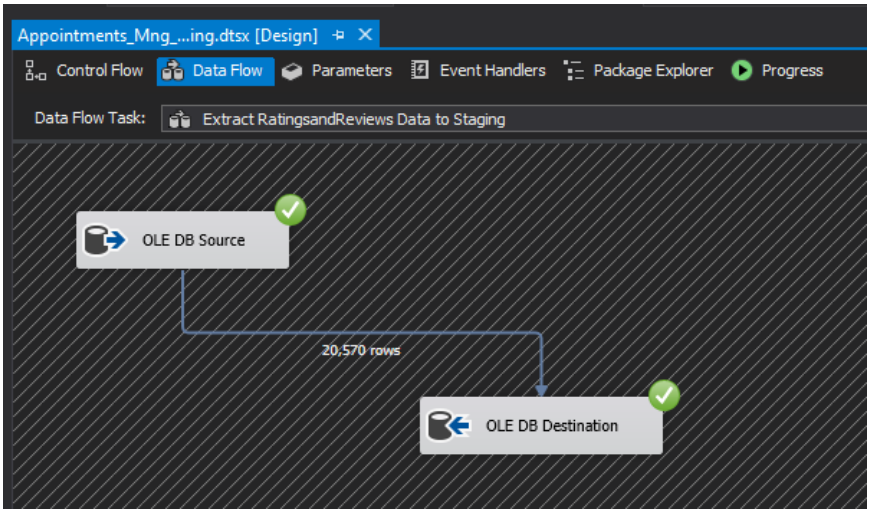
Physicians' data is extracted from the Physicians table in source database and inserted to Physicians Staging Table

Staging Allocated Rooms Details



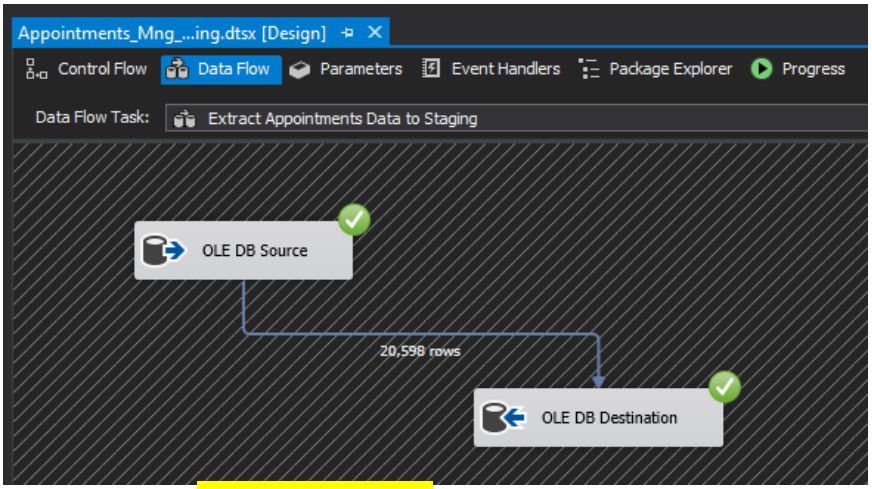
Allocated Rooms data is extracted from the Allocated Rooms table in source database and inserted to Allocated Rooms Staging Table

Staging Ratings and Reviews details



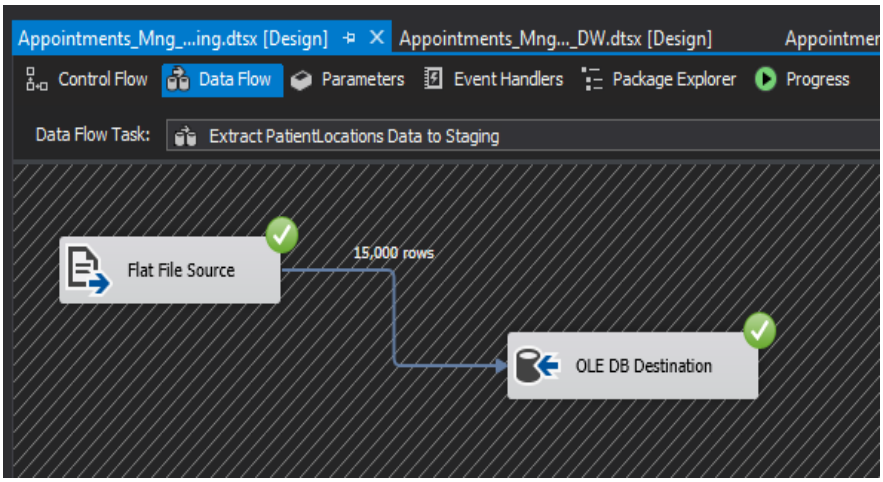
Ratings and Reviews data is extracted from the Ratings and Reviews table in source database and inserted to Ratings and Reviews Staging Table

Staging Appointments details



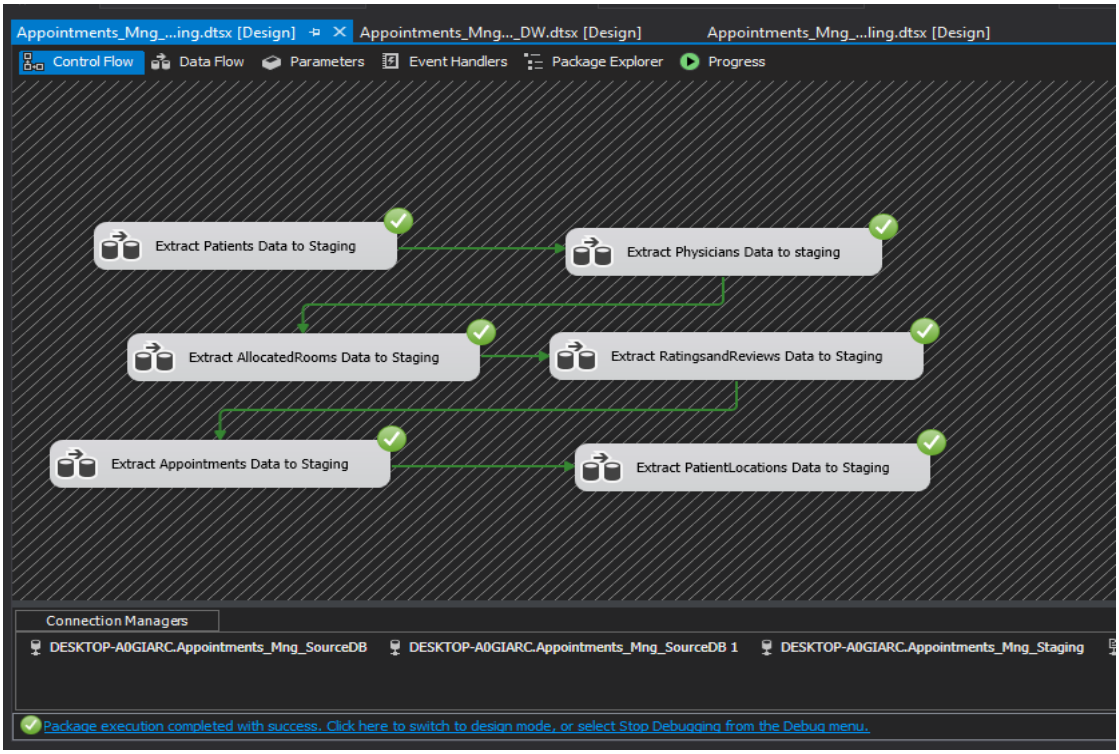
Appointments data is extracted from the Appointments table in source database and inserted to Appointments Staging Table

Staging Patients Locations details



Patients Locations data is extracted from the Patients Locations text file and inserted to Patients Locations Staging Table

After following the above steps and executing: (Executing Staging Package as itself)

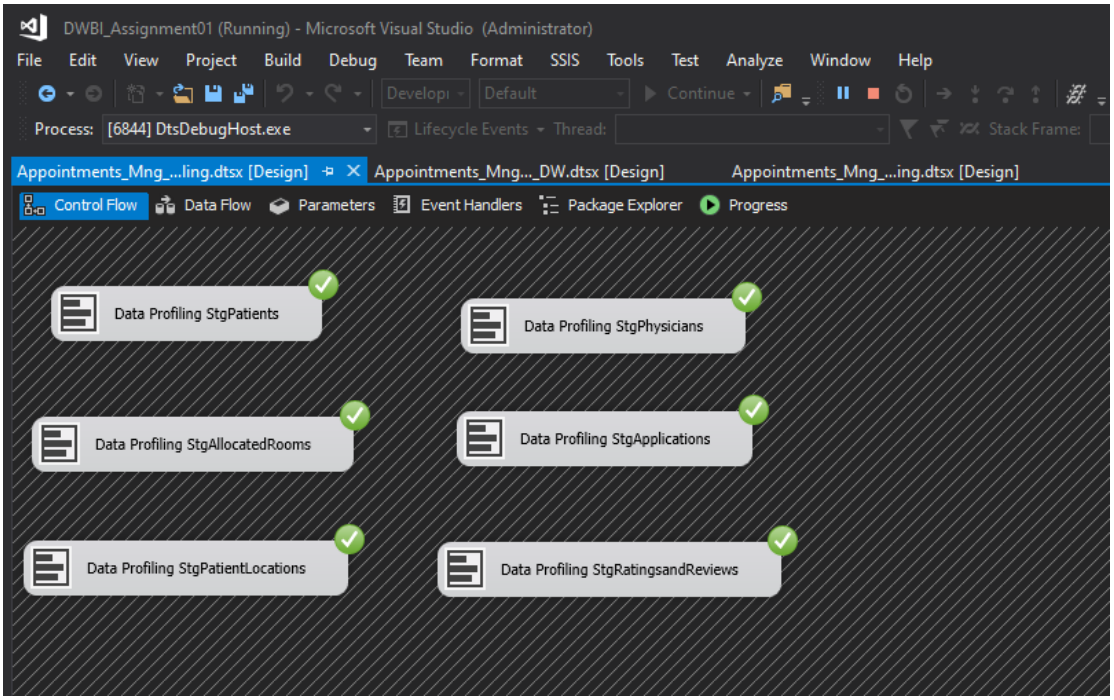


When we run the created package several times, the staging tables get repeatedly loaded without truncating the data already available in the table. As a solution for this we configure EVENT HANDLERS.

truncate table dbo.<table_name>

Next step is data profiling, and it is done as shown below:

Loaded staging tables are used to analyze the way data should look like and the types of transformations needed to perform on data.

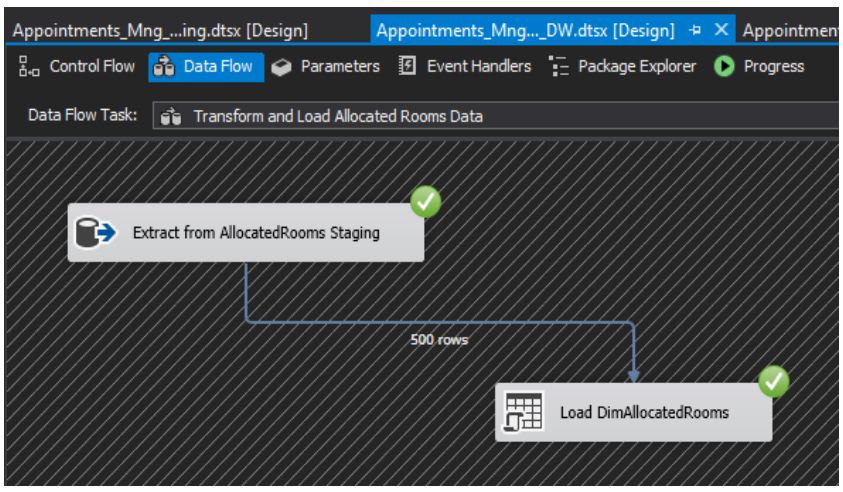


Every staging table is profiled and saved in a selected location.

Then ETL pipeline is used to read data from staging area tables and update corresponding data warehouse tables. Between staging area tables and source area tables there's one to one mapping; Subsequently between staging area and warehouse tables only the updated rows of data are inserted. (Incremental load).

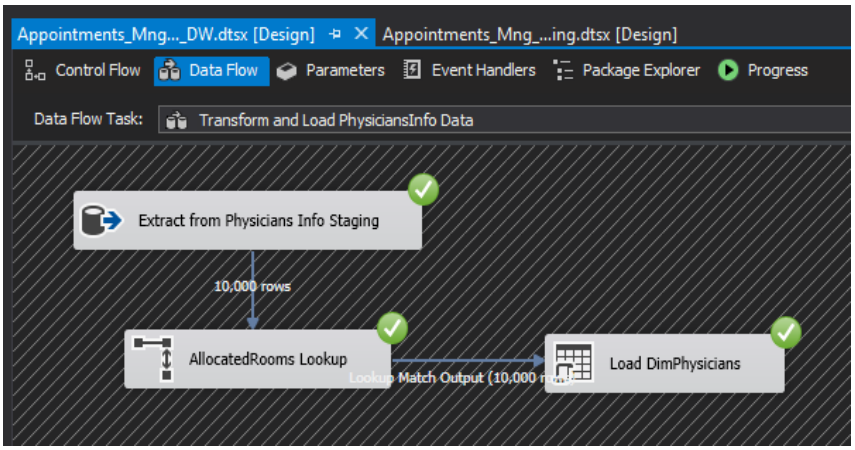
The below depicted order should be concerned when loading data from staging to warehouse due to the dependencies in between tables.

1. Data was loaded from the Allocated Rooms Staging table ➔ Allocated Rooms Dimension.



Extracted from Allocated Rooms Staging table and Loaded to Allocated Rooms Dimension table in DW package

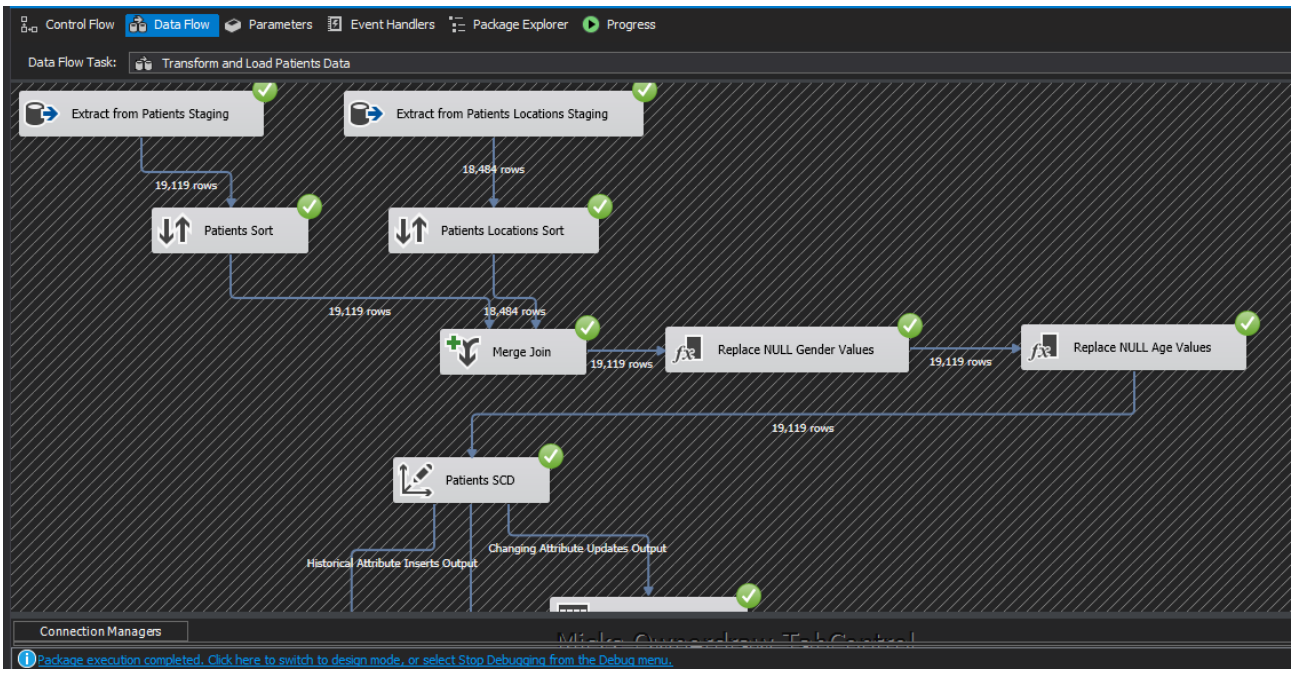
2 Data was loaded from the Physicians Staging table ➔ Physicians Dimension.

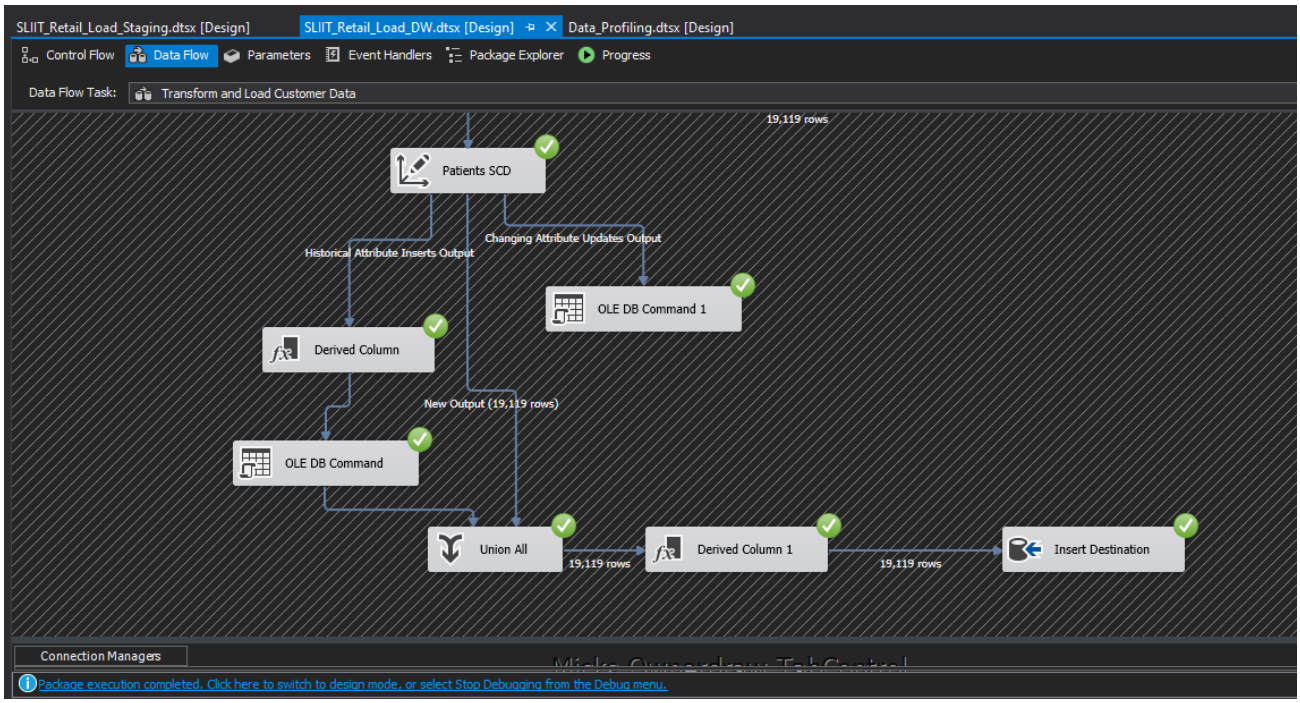


Extracted from Physicians Staging table and Loaded to Physicians Dimension table in DW package

3 Data was loaded from the Patients Locations Staging table ➔ Patients Locations Dimension

Data was loaded from the Patients Staging table ➔ Patients Dimension





As mentioned earlier under assumptions, Patients Info Dimension was considered as a slowly changing dimension. After merging patients table and patients' locations table two data cleansing steps were applied.

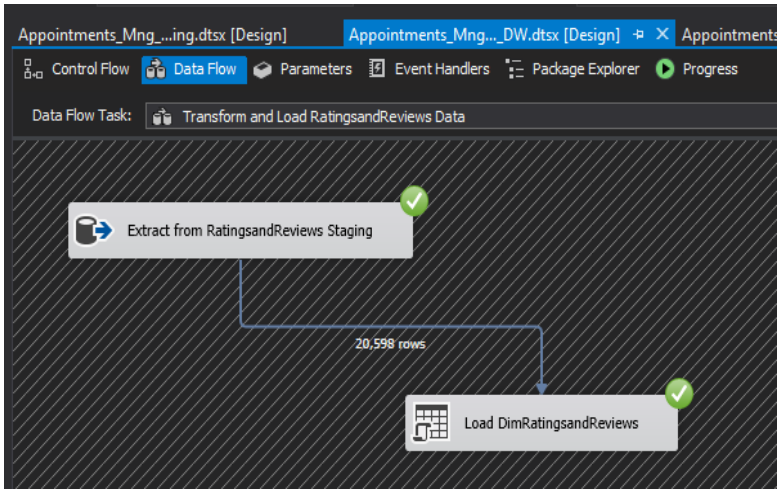
Here, null values of gender column and age column are replaced by suitable values

The below mentioned columns were set as changing attributes:

1. Last Name
2. Age
3. Gender
4. Partner
5. Payment Method

After extracting data from the Patients staging table, it was sorted according to the PatientID and as it was identified as a slowly changing dimension and loaded into Patients Info Dimension table

4 Data was loaded from the Ratings and Reviews Staging table ➔ Ratings and Reviews Dimension



Extracted from Ratings and Reviews Staging table and Loaded to Ratings and Reviews Dimension table in DW package

5 The query used to create the date dimension is mentioned below:

```
BEGIN TRY
	DROP TABLE [dbo].[DimDate]
END TRY

BEGIN CATCH
	/*No Action*/
END CATCH

/*************************************************************************************/

CREATE TABLE  [dbo].[DimDate]
	(	[DateKey] INT primary key,
		[Date] DATETIME,
		[FullDateUK] CHAR(10), -- Date in dd-MM-yyyy format
		[FullDateUSA] CHAR(10),-- Date in MM-dd-yyyy format
		[DayOfMonth] VARCHAR(2), -- Field will hold day number of Month
		[DaySuffix] VARCHAR(4), -- Apply suffix as 1st, 2nd ,3rd etc
		[DayName] VARCHAR(9), -- Contains name of the day, Sunday, Monday
		[DayOfWeekUSA] CHAR(1),-- First Day Sunday=1 and Saturday=7
		[DayOfWeekUK] CHAR(1),-- First Day Monday=1 and Sunday=7
		[DayOfWeekInMonth] VARCHAR(2), --1st Monday or 2nd Monday in Month
		[DayOfWeekInYear] VARCHAR(2),
		[DayOfQuarter] VARCHAR(3),
		[DayOfYear] VARCHAR(3),
		[WeekOfMonth] VARCHAR(1),-- Week Number of Month
		[WeekOfQuarter] VARCHAR(2), --Week Number of the Quarter
```

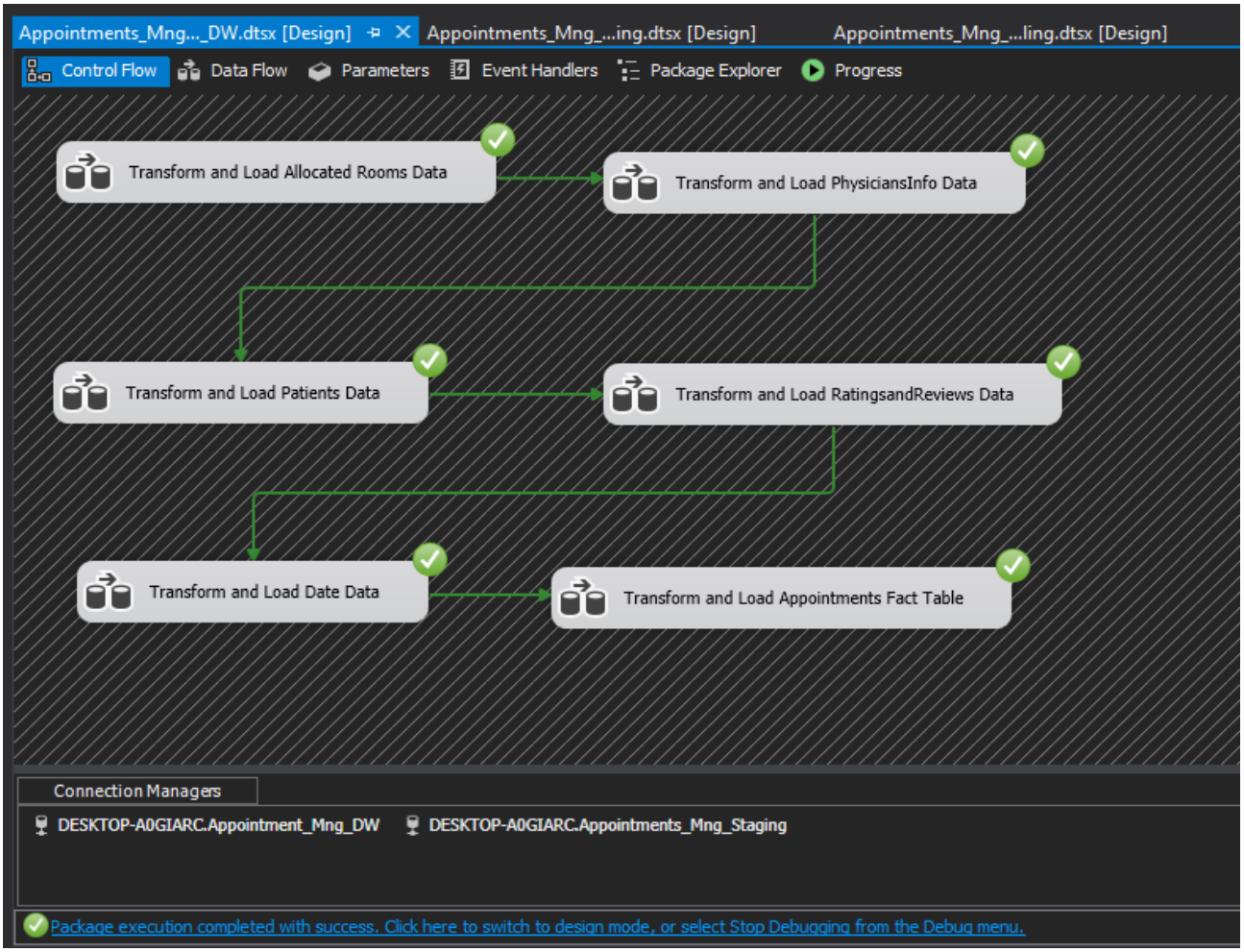
```
		[WeekOfYear] VARCHAR(2),--Week Number of the Year
		[Month] VARCHAR(2), --Number of the Month 1 to 12
		[MonthName] VARCHAR(9),--January, February etc
		[MonthOfQuarter] VARCHAR(2),-- Month Number belongs to Quarter
		[Quarter] CHAR(1),
		[QuarterName] VARCHAR(9),--First,Second..
		[Year] CHAR(4),-- Year value of Date stored in Row
		[YearName] CHAR(7), --CY 2012,CY 2013
		[MonthYear] CHAR(10), --Jan-2013,Feb-2013
		[MMYYYY] CHAR(6),
		[FirstDayOfMonth] DATE,
		[LastDayOfMonth] DATE,
		[FirstDayOfQuarter] DATE,
		[LastDayOfQuarter] DATE,
		[FirstDayOfYear] DATE,
		[LastDayOfYear] DATE,
		[IsHolidaySL] BIT,-- Flag 1=National Holiday, 0-No National Holiday
		[IsWeekday] BIT,-- 0=Week End ,1=Week Day
		[HolidaySL] VARCHAR(50),--Name of Holiday in US
		[isCurrentDay] int, -- Current day=1 else = 0
		[isDataAvailable] int, -- data available for the day = 1, no data
available for the day = 0
		[isLatestDataAvailable] int
	)
GO


/*************************************************************************************
*****/
--Specify Start Date and End date here
--Value of Start Date Must be Less than Your End Date

DECLARE @StartDate DATETIME = '01/01/1990' --Starting value of Date Range
DECLARE @EndDate DATETIME = '01/01/2099' --End Value of Date Range

--Temporary Variables To Hold the Values During Processing of Each Date of Year
DECLARE
	@DayOfWeekInMonth INT,
	@DayOfWeekInYear INT,
	@DayOfQuarter INT,
	@WeekOfMonth INT,
	@CurrentYear INT,
	@CurrentMonth INT,
	@CurrentQuarter INT

/*Table Data type to store the day of week count for the month and year*/
DECLARE @DayOfWeek TABLE (DOW INT, MonthCount INT, QuarterCount INT, YearCount INT)

INSERT INTO @DayOfWeek VALUES (1, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (2, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (3, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (4, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (5, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (6, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (7, 0, 0, 0)

--Extract and assign various parts of Values from Current Date to Variable

DECLARE @CurrentDate AS DATETIME = @StartDate
SET @CurrentMonth = DATEPART(MM, @CurrentDate)
SET @CurrentYear = DATEPART(YY, @CurrentDate)
SET @CurrentQuarter = DATEPART(QQ, @CurrentDate)
```

```
/*************************************************************************************
*****/
--Proceed only if Start Date(Current date ) is less than End date you specified above

WHILE @CurrentDate < @EndDate
BEGIN

/*Begin day of week logic*/

         /*Check for Change in Month of the Current date if Month changed then
          Change variable value*/
       IF @CurrentMonth != DATEPART(MM, @CurrentDate)
       BEGIN
              UPDATE @DayOfWeek
              SET MonthCount = 0
              SET @CurrentMonth = DATEPART(MM, @CurrentDate)
       END

        /* Check for Change in Quarter of the Current date if Quarter changed then
change
         Variable value*/

       IF @CurrentQuarter != DATEPART(QQ, @CurrentDate)
       BEGIN
              UPDATE @DayOfWeek
              SET QuarterCount = 0
              SET @CurrentQuarter = DATEPART(QQ, @CurrentDate)
       END

        /* Check for Change in Year of the Current date if Year changed then change
         Variable value*/


       IF @CurrentYear != DATEPART(YY, @CurrentDate)
       BEGIN
              UPDATE @DayOfWeek
              SET YearCount = 0
              SET @CurrentYear = DATEPART(YY, @CurrentDate)
       END

        -- Set values in table data type created above from variables

       UPDATE @DayOfWeek
       SET
              MonthCount = MonthCount + 1,
              QuarterCount = QuarterCount + 1,
              YearCount = YearCount + 1
       WHERE DOW = DATEPART(DW, @CurrentDate)

       SELECT
              @DayOfWeekInMonth = MonthCount,
              @DayOfQuarter = QuarterCount,
              @DayOfWeekInYear = YearCount
       FROM @DayOfWeek
       WHERE DOW = DATEPART(DW, @CurrentDate)

/*End day of week logic*/


/* Populate Your Dimension Table with values*/

       INSERT INTO [dbo].[DimDate]
       SELECT
```

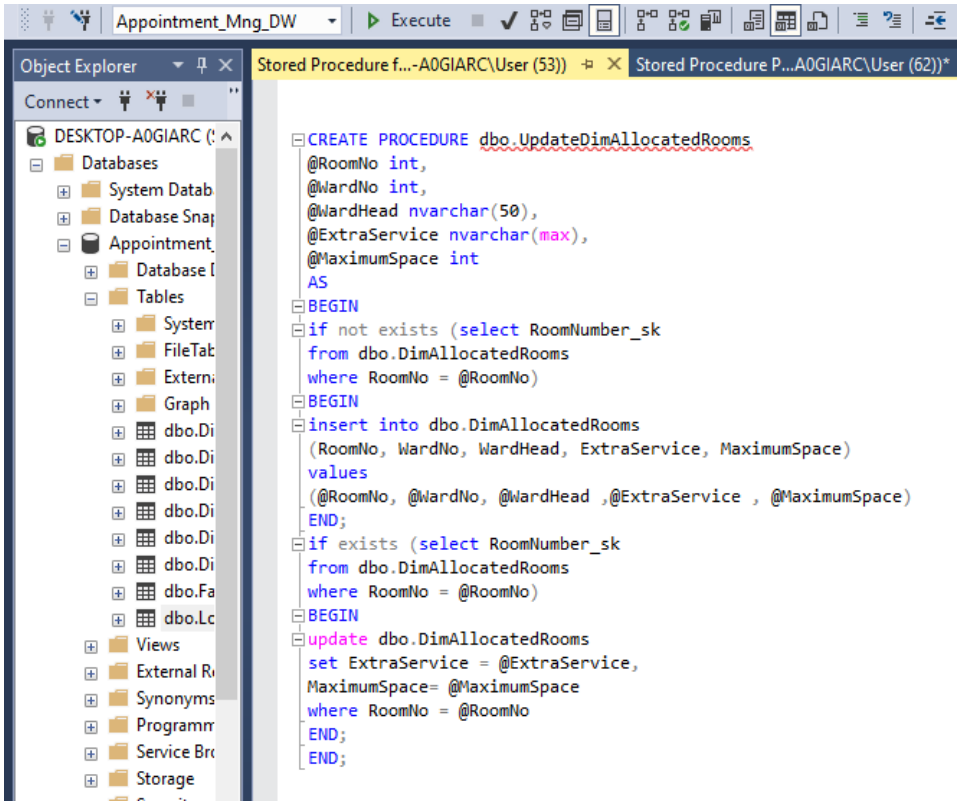
```
CONVERT (char(8),@CurrentDate,112) as DateKey,
@CurrentDate AS Date,
CONVERT (char(10),@CurrentDate,103) as FullDateUK,
CONVERT (char(10),@CurrentDate,101) as FullDateUSA,
DATEPART(DD, @CurrentDate) AS DayOfMonth,
--Apply Suffix values like 1st, 2nd 3rd etc..
CASE
       WHEN DATEPART(DD,@CurrentDate) IN (11,12,13)
       THEN CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'th'
       WHEN RIGHT(DATEPART(DD,@CurrentDate),1) = 1
       THEN CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'st'
       WHEN RIGHT(DATEPART(DD,@CurrentDate),1) = 2
       THEN CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'nd'
       WHEN RIGHT(DATEPART(DD,@CurrentDate),1) = 3
       THEN CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'rd'
       ELSE CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'th'
       END AS DaySuffix,

DATENAME(DW, @CurrentDate) AS DayName,
DATEPART(DW, @CurrentDate) AS DayOfWeekUSA,

-- check for day of week as Per US and change it as per UK format
CASE DATEPART(DW, @CurrentDate)
       WHEN 1 THEN 7
       WHEN 2 THEN 1
       WHEN 3 THEN 2
       WHEN 4 THEN 3
       WHEN 5 THEN 4
       WHEN 6 THEN 5
       WHEN 7 THEN 6
       END
       AS DayOfWeekUK,

@DayOfWeekInMonth AS DayOfWeekInMonth,
@DayOfWeekInYear AS DayOfWeekInYear,
@DayOfQuarter AS DayOfQuarter,
DATEPART(DY, @CurrentDate) AS DayOfYear,
DATEPART(WW, @CurrentDate) + 1 - DATEPART(WW, CONVERT(VARCHAR,
DATEPART(MM, @CurrentDate)) + '/1/' + CONVERT(VARCHAR,
DATEPART(YY, @CurrentDate))) AS WeekOfMonth,
(DATEDIFF(DD, DATEADD(QQ, DATEDIFF(QQ, 0, @CurrentDate), 0),
@CurrentDate) / 7) + 1 AS WeekOfQuarter,
DATEPART(WW, @CurrentDate) AS WeekOfYear,
DATEPART(MM, @CurrentDate) AS Month,
DATENAME(MM, @CurrentDate) AS MonthName,
CASE
       WHEN DATEPART(MM, @CurrentDate) IN (1, 4, 7, 10) THEN 1
       WHEN DATEPART(MM, @CurrentDate) IN (2, 5, 8, 11) THEN 2
       WHEN DATEPART(MM, @CurrentDate) IN (3, 6, 9, 12) THEN 3
       END AS MonthOfQuarter,
DATEPART(QQ, @CurrentDate) AS Quarter,
CASE DATEPART(QQ, @CurrentDate)
       WHEN 1 THEN 'First'
       WHEN 2 THEN 'Second'
       WHEN 3 THEN 'Third'
       WHEN 4 THEN 'Fourth'
       END AS QuarterName,
DATEPART(YEAR, @CurrentDate) AS Year,
'CY ' + CONVERT(VARCHAR, DATEPART(YEAR, @CurrentDate)) AS YearName,
LEFT(DATENAME(MM, @CurrentDate), 3) + '-' + CONVERT(VARCHAR,
DATEPART(YY, @CurrentDate)) AS MonthYear,
RIGHT('0' + CONVERT(VARCHAR, DATEPART(MM, @CurrentDate)),2) +
```

```
                CONVERT(VARCHAR, DATEPART(YY, @CurrentDate)) AS MMYYYY,
                CONVERT(DATETIME, CONVERT(DATE, DATEADD(DD, - (DATEPART(DD,
                @CurrentDate) - 1), @CurrentDate))) AS FirstDayOfMonth,
                CONVERT(DATETIME, CONVERT(DATE, DATEADD(DD, - (DATEPART(DD,
                (DATEADD(MM, 1, @CurrentDate)))), DATEADD(MM, 1,
                @CurrentDate)))) AS LastDayOfMonth,
                DATEADD(QQ, DATEDIFF(QQ, 0, @CurrentDate), 0) AS FirstDayOfQuarter,
                DATEADD(QQ, DATEDIFF(QQ, -1, @CurrentDate), -1) AS LastDayOfQuarter,
                CONVERT(DATETIME, '01/01/' + CONVERT(VARCHAR, DATEPART(YY,
                @CurrentDate))) AS FirstDayOfYear,
                CONVERT(DATETIME, '12/31/' + CONVERT(VARCHAR, DATEPART(YY,
                @CurrentDate))) AS LastDayOfYear,
                NULL AS IsHolidaySL,
                CASE DATEPART(DW, @CurrentDate)
                       WHEN 1 THEN 0
                       WHEN 2 THEN 1
                       WHEN 3 THEN 1
                       WHEN 4 THEN 1
                       WHEN 5 THEN 1
                       WHEN 6 THEN 1
                       WHEN 7 THEN 0
                       END AS IsWeekday,
                NULL AS HolidaySL, (case when @CurrentDate = convert(date, sysdatetime())
then 1 else 0 end), 0, 0

       SET @CurrentDate = DATEADD(DD, 1, @CurrentDate)
END

/*****************************************************************************************
*****/

/*****************************************************************************************
**/

SELECT * FROM [dbo].[DimDate]
```

6 After loading data to all the dimensions and the fact table:

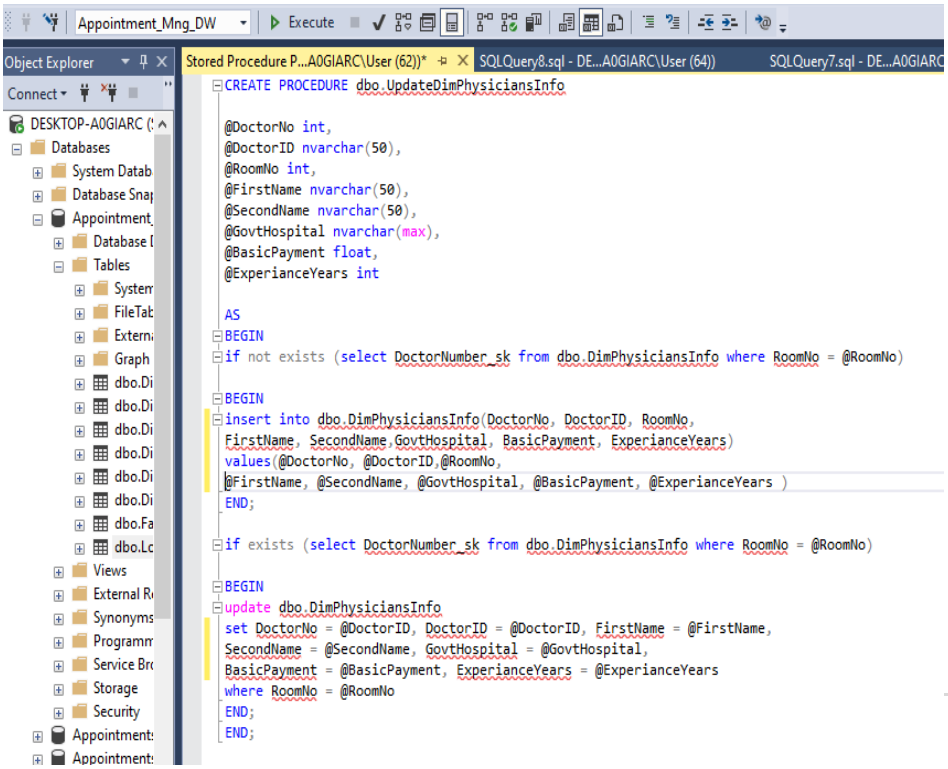


The update procedure used to update Allocated Rooms is attached below:



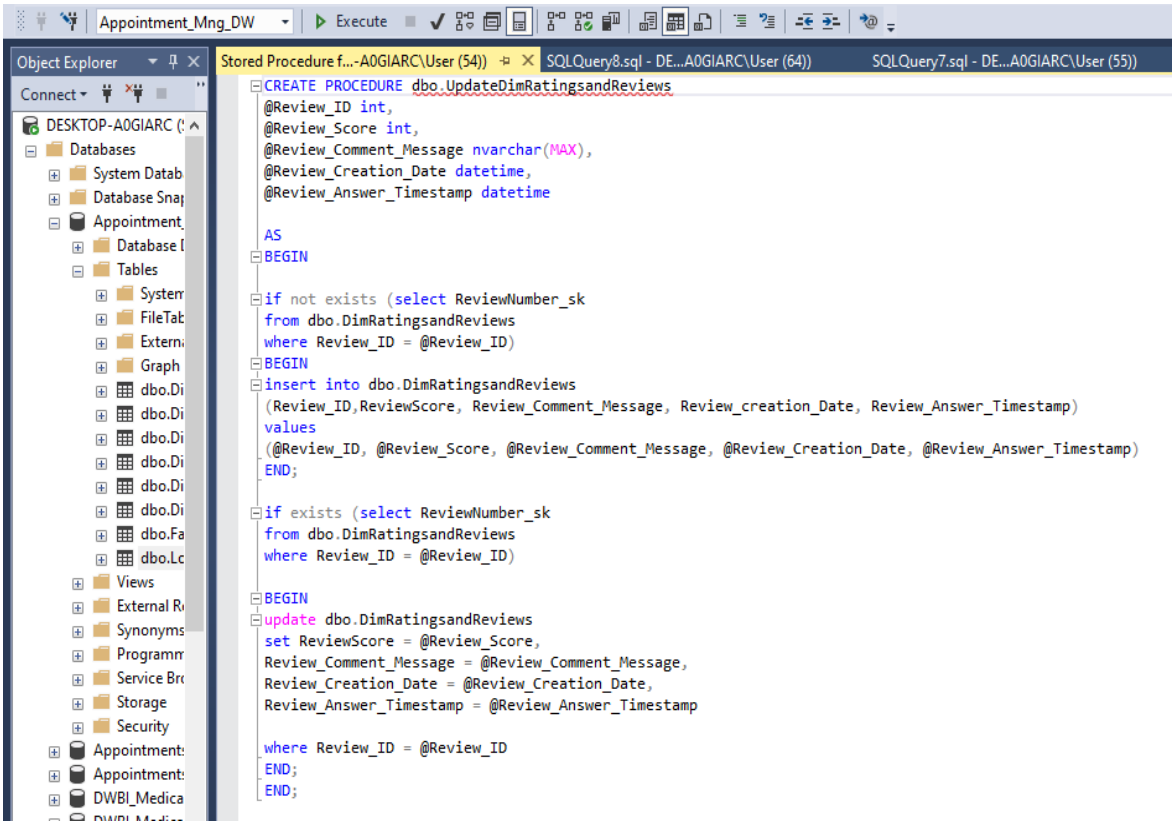
```
CREATE PROCEDURE dbo.UpdateDimAllocatedRooms
@RoomNo int,
@WardNo int,
@WardHead nvarchar(50),
@ExtraService nvarchar(max),
@MaximumSpace int
AS
BEGIN
if not exists (select RoomNumber_sk
from dbo.DimAllocatedRooms
where RoomNo = @RoomNo)
BEGIN
insert into dbo.DimAllocatedRooms
(RoomNo, WardNo, WardHead, ExtraService, MaximumSpace)
values
(@RoomNo, @WardNo, @WardHead ,@ExtraService , @MaximumSpace)
END;
if exists (select RoomNumber_sk
from dbo.DimAllocatedRooms
where RoomNo = @RoomNo)
BEGIN
update dbo.DimAllocatedRooms
set ExtraService = @ExtraService,
MaximumSpace= @MaximumSpace
where RoomNo = @RoomNo
END;
END;
```

The update procedure used to update Physicians is attached below:



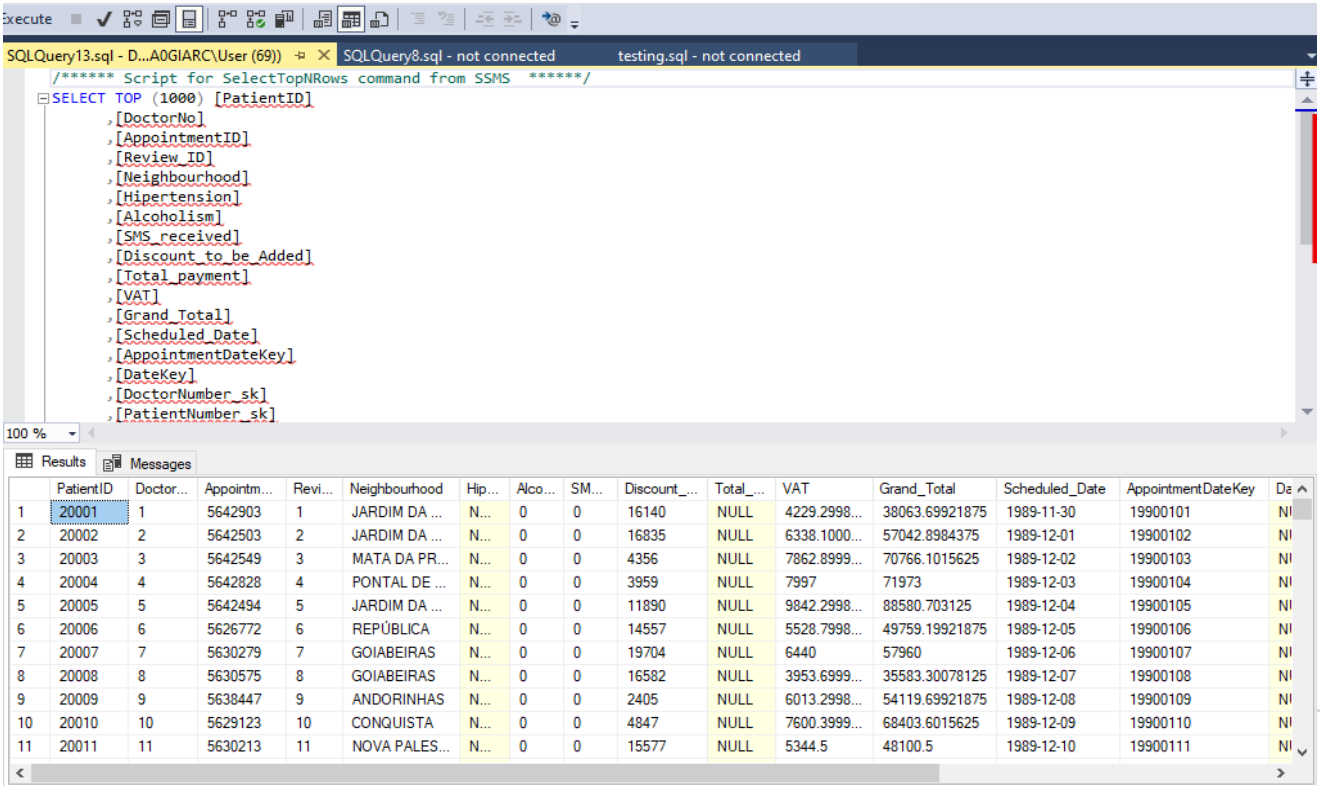
```
CREATE PROCEDURE dbo.UpdateDimPhysiciansInfo

@DoctorNo int,
@DoctorID nvarchar(50),
@RoomNo int,
@FirstName nvarchar(50),
@SecondName nvarchar(50),
@GovtHospital nvarchar(max),
@BasicPayment float,
@ExperianceYears int

AS
BEGIN
if not exists (select DoctorNumber_sk from dbo.DimPhysiciansInfo where RoomNo = @RoomNo)

BEGIN
insert into dbo.DimPhysiciansInfo(DoctorNo, DoctorID, RoomNo,
FirstName, SecondName,GovtHospital, BasicPayment, ExperianceYears)
values(@DoctorNo, @DoctorID,@RoomNo,
@FirstName, @SecondName, @GovtHospital, @BasicPayment, @ExperianceYears )
END;

if exists (select DoctorNumber_sk from dbo.DimPhysiciansInfo where RoomNo = @RoomNo)

BEGIN
update dbo.DimPhysiciansInfo
set DoctorNo = @DoctorID, DoctorID = @DoctorID, FirstName = @FirstName,
SecondName = @SecondName, GovtHospital = @GovtHospital,
BasicPayment = @BasicPayment, ExperianceYears = @ExperianceYears
where RoomNo = @RoomNo
END;
END;
```

The update procedure used to update Ratings and Reviews is attached below:



```
CREATE PROCEDURE dbo.UpdateDimRatingsandReviews
@Review_ID int,
@Review_Score int,
@Review_Comment_Message nvarchar(MAX),
@Review_Creation_Date datetime,
@Review_Answer_Timestamp datetime

AS
BEGIN

if not exists (select ReviewNumber_sk
from dbo.DimRatingsandReviews
where Review_ID = @Review_ID)
BEGIN
insert into dbo.DimRatingsandReviews
(Review_ID,ReviewScore, Review_Comment_Message, Review_creation_Date, Review_Answer_Timestamp)
values
(@Review_ID, @Review_Score, @Review_Comment_Message, @Review_Creation_Date, @Review_Answer_Timestamp)
END;

if exists (select ReviewNumber_sk
from dbo.DimRatingsandReviews
where Review_ID = @Review_ID)

BEGIN
update dbo.DimRatingsandReviews
set ReviewScore = @Review_Score,
Review_Comment_Message = @Review_Comment_Message,
Review_Creation_Date = @Review_Creation_Date,
Review_Answer_Timestamp = @Review_Answer_Timestamp

where Review_ID = @Review_ID
END;
END;
```

Please find below a screenshot of the final fact table:



```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP (1000) [PatientID]
      ,[DoctorNo]
      ,[AppointmentID]
      ,[Review_ID]
      ,[Neighbourhood]
      ,[Hipertension]
      ,[Alcoholism]
      ,[SMS_received]
      ,[Discount_to_be_Added]
      ,[Total_payment]
      ,[VAT]
      ,[Grand_Total]
      ,[Scheduled_Date]
      ,[AppointmentDateKey]
      ,[DateKey]
      ,[DoctorNumber_sk]
      ,[PatientNumber_sk]
```

	PatientID	Doctor...	Appointm...	Revi...	Neighbourhood	Hip...	Alco...	SM...	Discount_...	Total_...	VAT	Grand_Total	Scheduled_Date	AppointmentDateKey	Da
1	20001	1	5642903	1	JARDIM DA ...	N...	0	0	16140	NULL	4229.2998...	38063.69921875	1989-11-30	19900101	NI
2	20002	2	5642503	2	JARDIM DA ...	N...	0	0	16835	NULL	6338.1000...	57042.8984375	1989-12-01	19900102	NI
3	20003	3	5642549	3	MATA DA PR...	N...	0	0	4356	NULL	7862.8999...	70766.1015625	1989-12-02	19900103	NI
4	20004	4	5642828	4	PONTAL DE ...	N...	0	0	3959	NULL	7997	71973	1989-12-03	19900104	NI
5	20005	5	5642494	5	JARDIM DA ...	N...	0	0	11890	NULL	9842.2998...	88580.703125	1989-12-04	19900105	NI
6	20006	6	5626772	6	REPÚBLICA	N...	0	0	14557	NULL	5528.7998...	49759.19921875	1989-12-05	19900106	NI
7	20007	7	5630279	7	GOIABEIRAS	N...	0	0	19704	NULL	6440	57960	1989-12-06	19900107	NI
8	20008	8	5630575	8	GOIABEIRAS	N...	0	0	16582	NULL	3953.6999...	35583.30078125	1989-12-07	19900108	NI
9	20009	9	5638447	9	ANDORINHAS	N...	0	0	2405	NULL	6013.2998...	54119.69921875	1989-12-08	19900109	NI
10	20010	10	5629123	10	CONQUISTA	N...	0	0	4847	NULL	7600.3999...	68403.6015625	1989-12-09	19900110	NI
11	20011	11	5630213	11	NOVA PALES...	N...	0	0	15577	NULL	5344.5	48100.5	1989-12-10	19900111	NI

As we are not maintaining a history of data, we only need to have the updated records in data warehouse. Either we can do a full load, or we can check for the unique data and update them.

In this assignment, I've used both these methods.

1- Allocated Rooms details and Physicians details are loaded by checking the latest values. For that I used two stored procedures. (Mentioned above)

2- After writing the stored procedure, 'Allocated Rooms' table is loaded first using a OLE DB Command task. Then the loaded table is merged with Physicians table by using a Look up instance.

3-When loading Ratings and Reviews table, I used a stored procedure. (Mentioned above)

4-Patients table

The column Total is calculated the following way

Grand Total = Total Payment + VAT – Discount to be Added

VAT	Grand_Total
4229.2998046875	38063.69921875
6338.10009765625	57042.8984375
7862.89990234375	70766.1015625
7997	71973
9842.2998046875	88580.703125
5528.7998046875	49759.19921875
6440	57960
3953.69995117188	35583.30078125
6013.2998046875	54119.69921875
7600.39990234375	68403.6015625
5344.5	48100.5

ETL development-Accumulating fact tables

[developed files are attached herewith]