INTRODUCTION

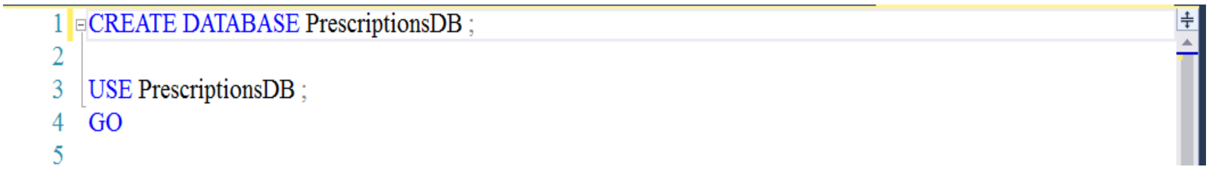
Three datasets were provided for analysis. Prescription, Drugs and Medical Practice csv files. These datasets are related and consists of information on how drugs are prescribed by medical practices in Bolton.

PART 1

Database creation

I created the database PrescriptionsDB using the CREATE DATABASE statement. Used the USE statement to make use of the Prescription database.

Figure 1: Creation of Database



```
CREATE DATABASE PrescriptionsDB ;

USE PrescriptionsDB ;
GO

```

Table Creation

I created the tables for the PrescriptionsDB database by importing the flat files into my SSMS interface. Right clicked on the database on the object explorer, then clicking on Task then import flat files.

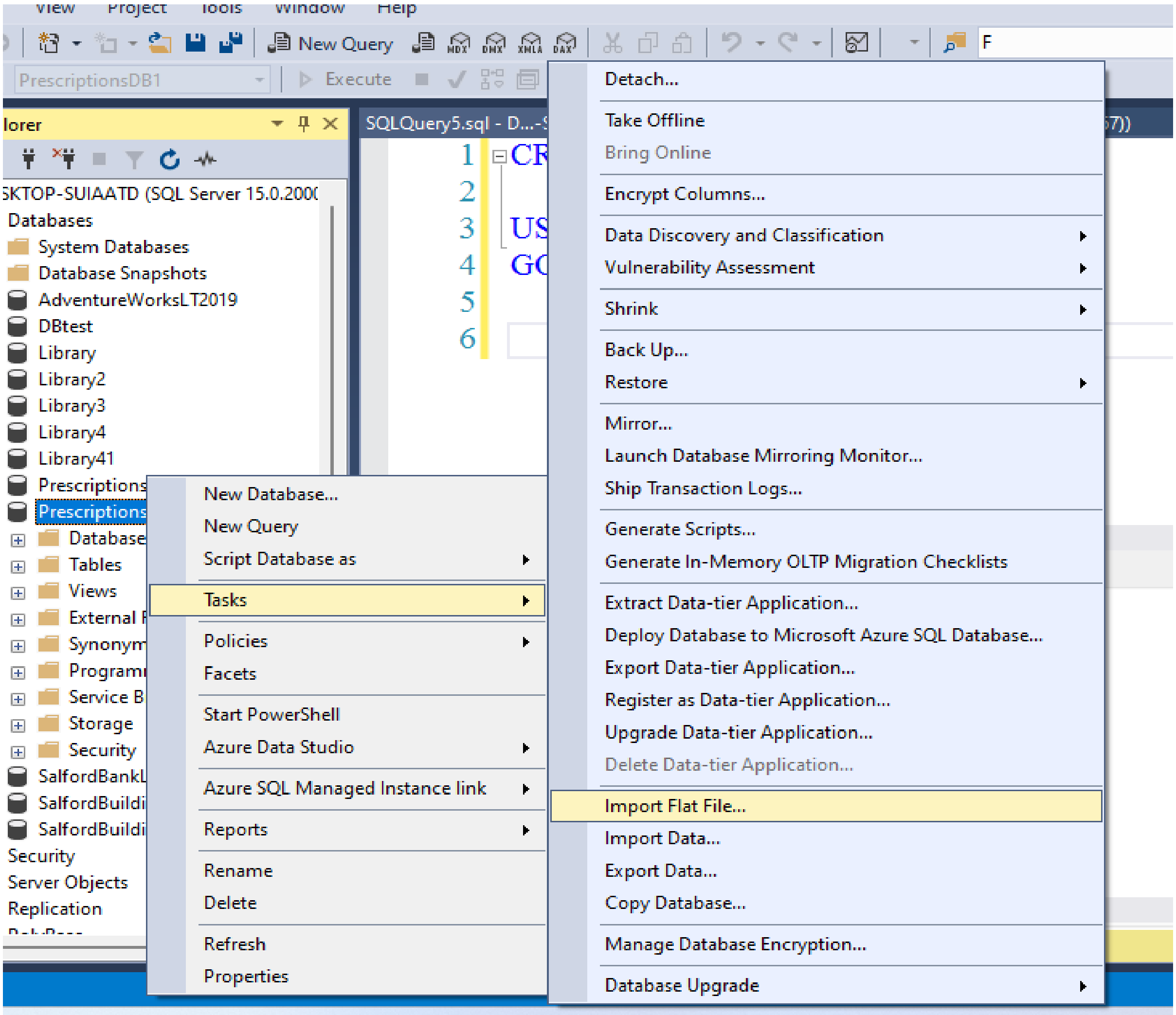
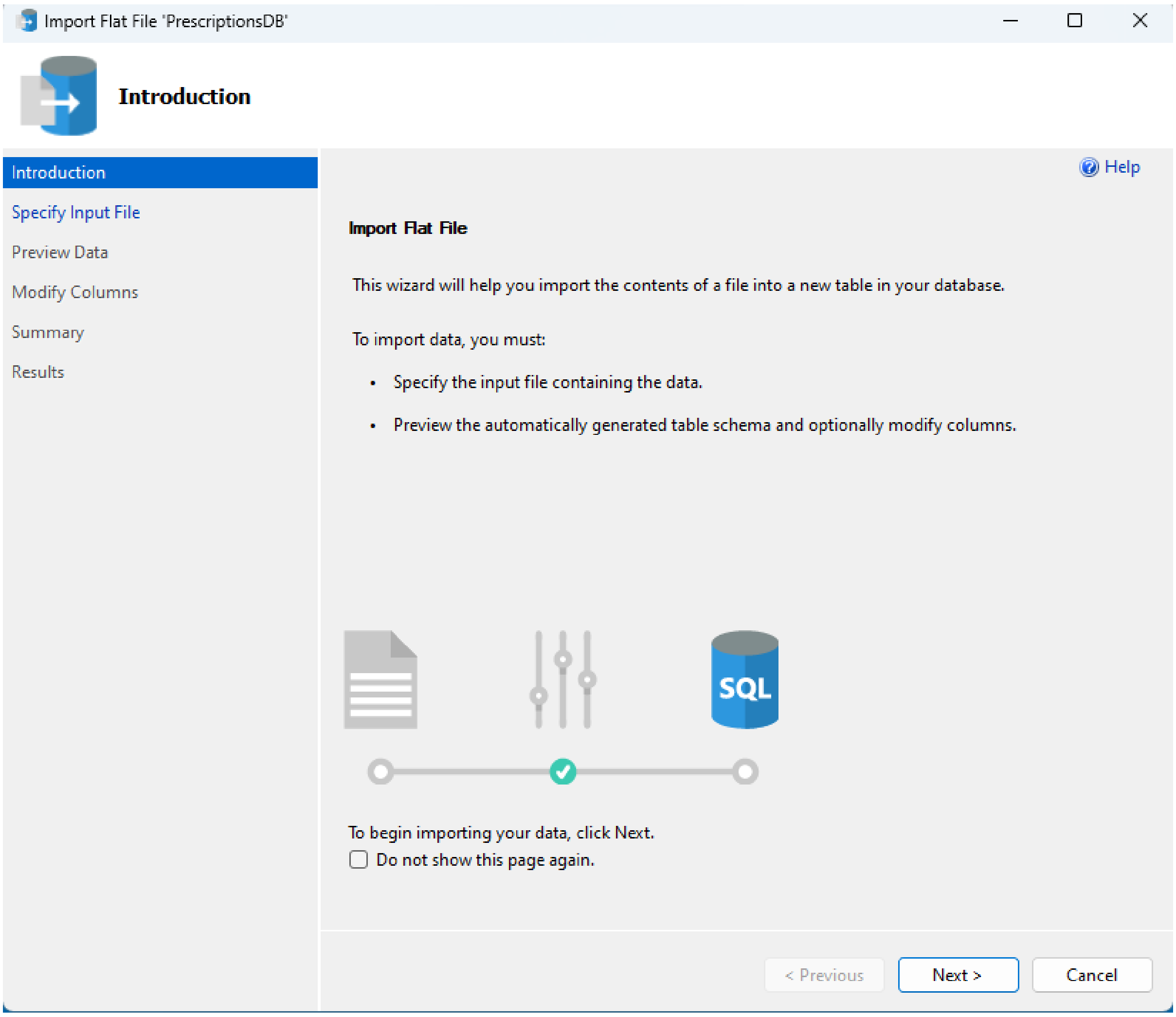


Figure 2: Importing flat files

After clicking on the import file the windows below **(Figure 3)** popped up, showing the introduction page



Import Flat File

This wizard will help you import the contents of a file into a new table in your database.

To import data, you must:

- Specify the input file containing the data.
- Preview the automatically generated table schema and optionally modify columns.

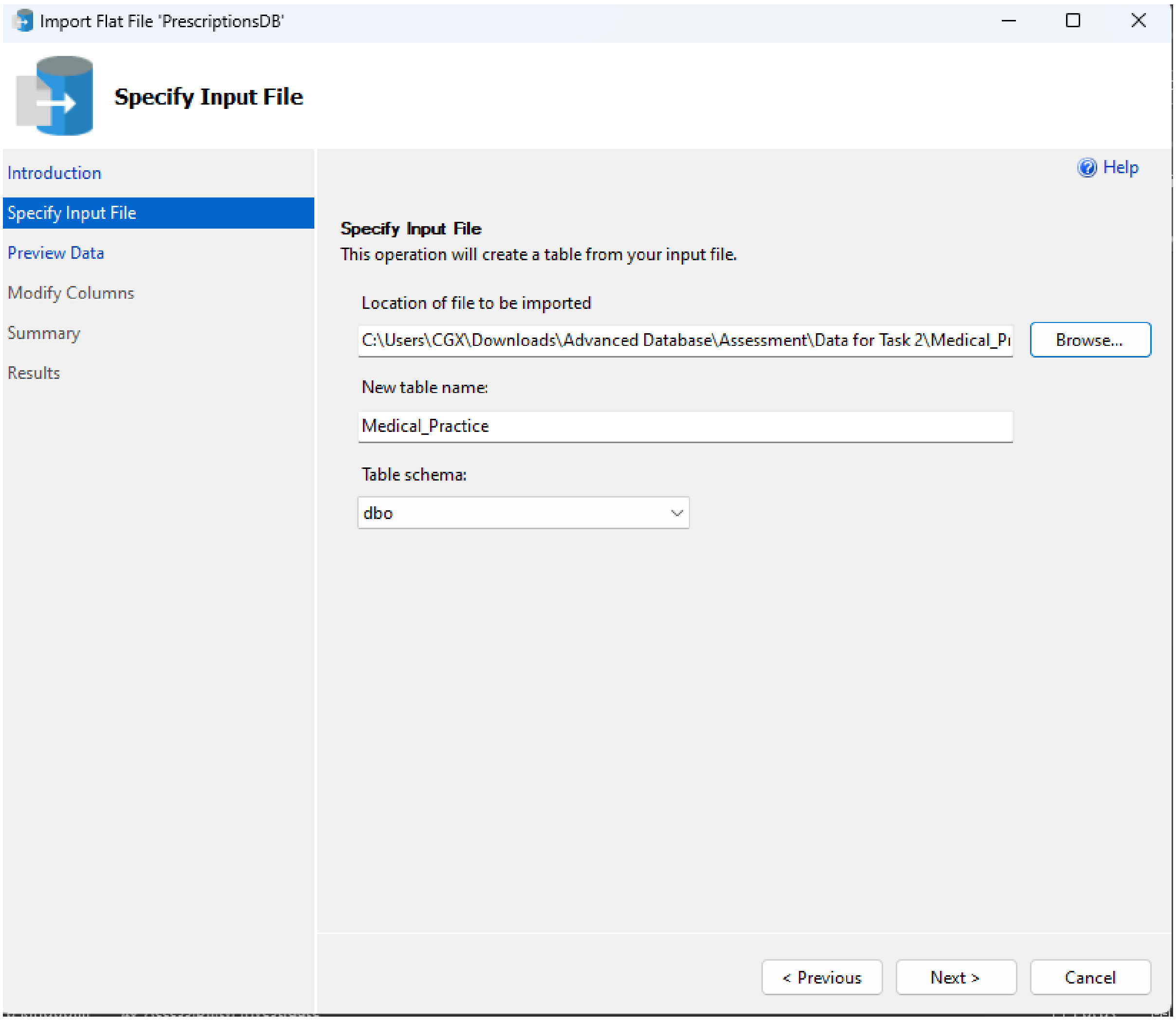
To begin importing your data, click Next.

□ Do not show this page again.

Figure 3: Introduction page

Medical_Practice

I clicked on next which then led me to the specify input file page. I clicked the browse button which allowed me to locate the file path of the files I imported. I also imputed the name of my tables and left the table schema as default(dbo).



Import Flat File 'PrescriptionsDB'

Specify Input File

This operation will create a table from your input file.

Location of file to be imported

C:\Users\CGX\Downloads\Advanced Database\Assessment\Data for Task 2\Medical_Pr

New table name:

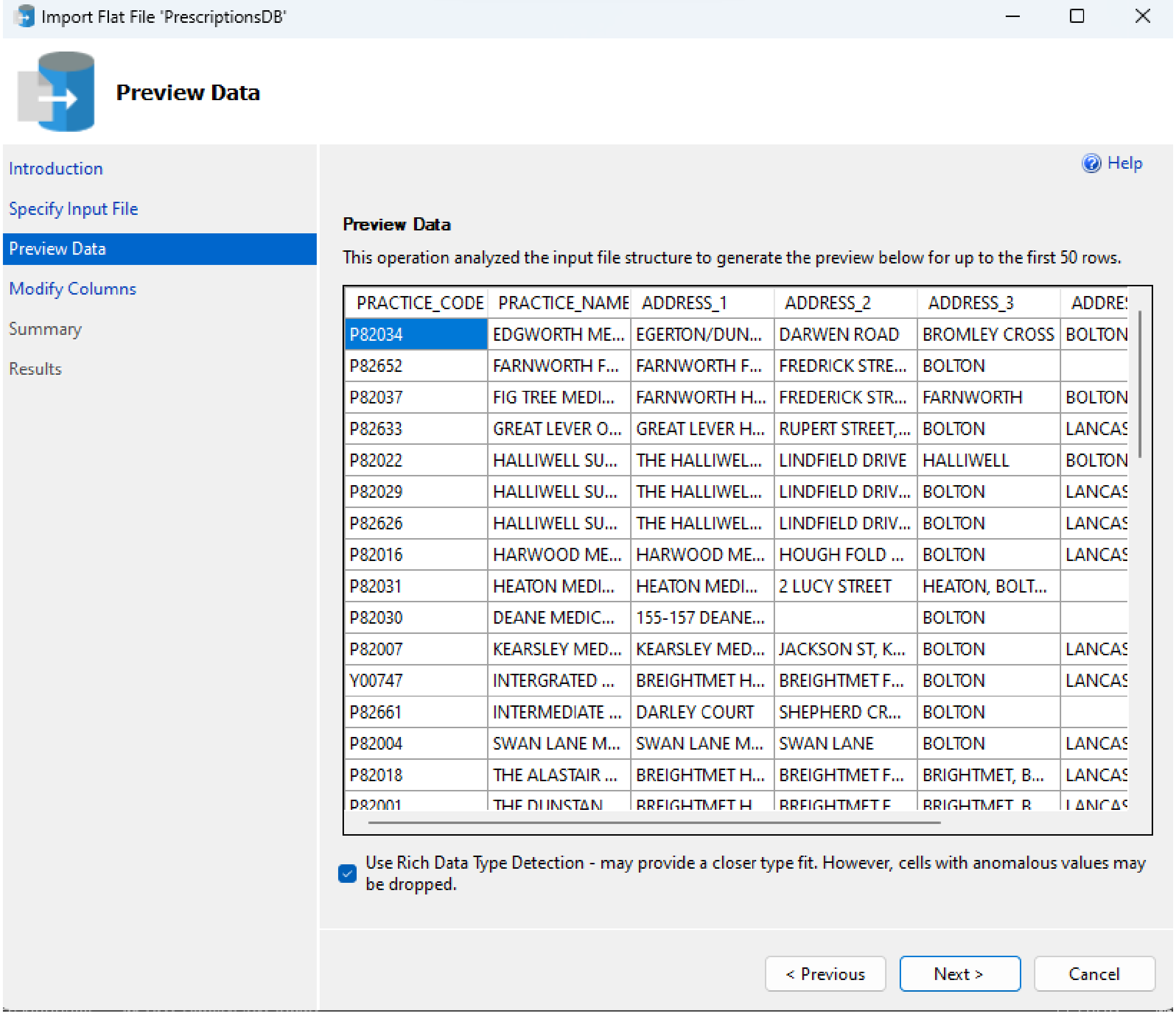
Medical_Practice

Table schema:

dbo

Figure 4: Specify input file

After specifying the filepath for Medical_Practice as the file to import. Then clicked on next and was led to the Preview data page. Here I was able to preview the first 50 rows of the Medical Practice data, view the data types and have an idea of the length of the character of the various columns.



Import Flat File 'PrescriptionsDB'

Preview Data

Preview Data

This operation analyzed the input file structure to generate the preview below for up to the first 50 rows.

PRACTICE_CODE	PRACTICE_NAME	ADDRESS_1	ADDRESS_2	ADDRESS_3	ADDRE
P82034	EDGWORTH ME...	EGERTON/DUN...	DARWEN ROAD	BROMLEY CROSS	BOLTON
P82652	FARNWORTH F...	FARNWORTH F...	FREDRICK STRE...	BOLTON	
P82037	FIG TREE MEDI...	FARNWORTH H...	FREDERICK STR...	FARNWORTH	BOLTON
P82633	GREAT LEVER O...	GREAT LEVER H...	RUPERT STREET,...	BOLTON	LANCAS
P82022	HALLIWELL SU...	THE HALLIWEL...	LINDFIELD DRIVE	HALLIWELL	BOLTON
P82029	HALLIWELL SU...	THE HALLIWEL...	LINDFIELD DRIV...	BOLTON	LANCAS
P82626	HALLIWELL SU...	THE HALLIWEL...	LINDFIELD DRIV...	BOLTON	LANCAS
P82016	HARWOOD ME...	HARWOOD ME...	HOUGH FOLD ...	BOLTON	LANCAS
P82031	HEATON MEDI...	HEATON MEDI...	2 LUCY STREET	HEATON, BOLT...	
P82030	DEANE MEDIC...	155-157 DEANE...		BOLTON	
P82007	KEARSLEY MED...	KEARSLEY MED...	JACKSON ST, K...	BOLTON	LANCAS
Y00747	INTERGRATED ...	BREIGHTMET H...	BREIGHTMET F...	BOLTON	LANCAS
P82661	INTERMEDIATE ...	DARLEY COURT	SHEPHERD CR...	BOLTON	
P82004	SWAN LANE M...	SWAN LANE M...	SWAN LANE	BOLTON	LANCAS
P82018	THE ALASTAIR ...	BREIGHTMET H...	BREIGHTMET F...	BRIGHTMET, B...	LANCAS
P82001	THE DUNSTAN	BREIGHTMET H	BREIGHTMET F	BRIGHTMET, B	LANCAS

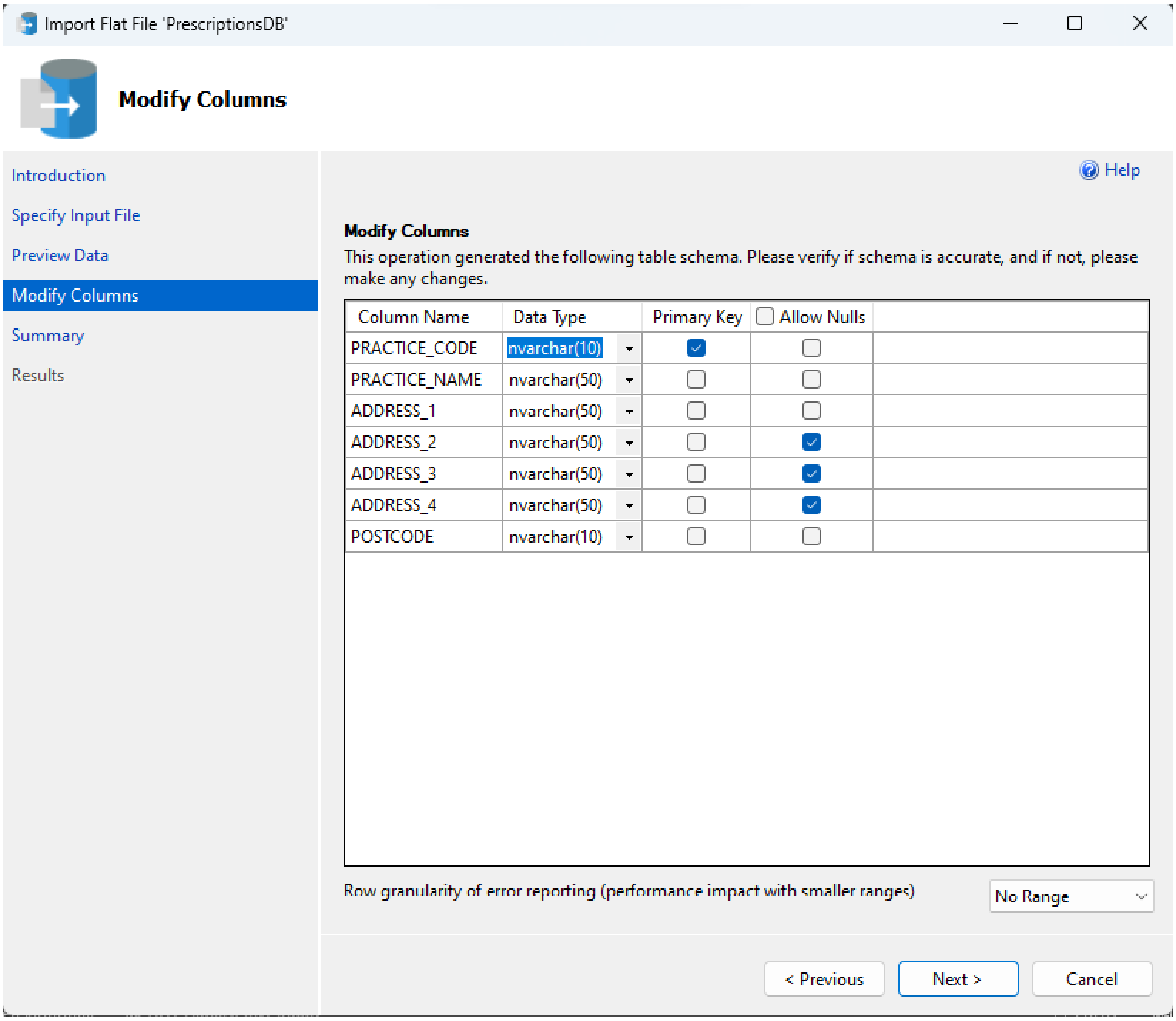
☑ Use Rich Data Type Detection - may provide a closer type fit. However, cells with anomalous values may be dropped.

Figure 5: Preview Data

I modified the columns in my data, PRACTICE_CODE contains unique characters for each row with no repetition , hence I made it the primary key.

I left ADDRESS_2, ADDRESS_3, ADDRESS_4 as NULL because they are used to collect secondary address information so they are not compulsory since the primary information in ADDRESS_1 and POSTCODE is filled and this is enough information.

I changed the data type of both the PRACTICE_CODE and POSTCODE from nvarchar(50) to nvarchar(10) because the length of the characters of these columns do not require that much storage.



Modify Columns

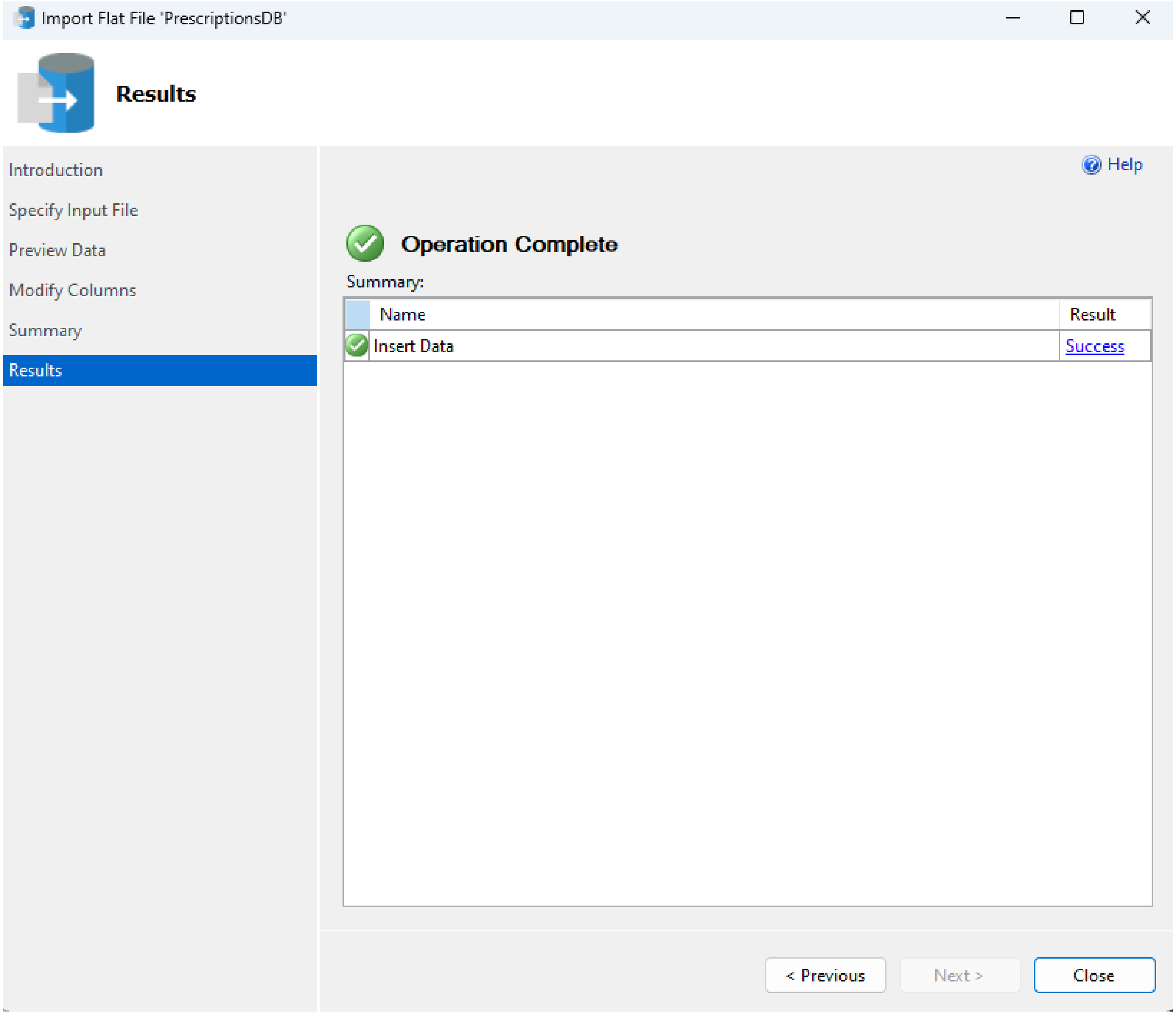
This operation generated the following table schema. Please verify if schema is accurate, and if not, please make any changes.

Column Name	Data Type	Primary Key	□ Allow Nulls
PRACTICE_CODE	nvarchar(10)	☑	□
PRACTICE_NAME	nvarchar(50)	□	□
ADDRESS_1	nvarchar(50)	□	□
ADDRESS_2	nvarchar(50)	□	☑
ADDRESS_3	nvarchar(50)	□	☑
ADDRESS_4	nvarchar(50)	□	☑
POSTCODE	nvarchar(10)	□	□

Row granularity of error reporting (performance impact with smaller ranges) No Range

Figure 6: Modifying Columns

Clicked on next and my file was imported successfully.



Operation Complete

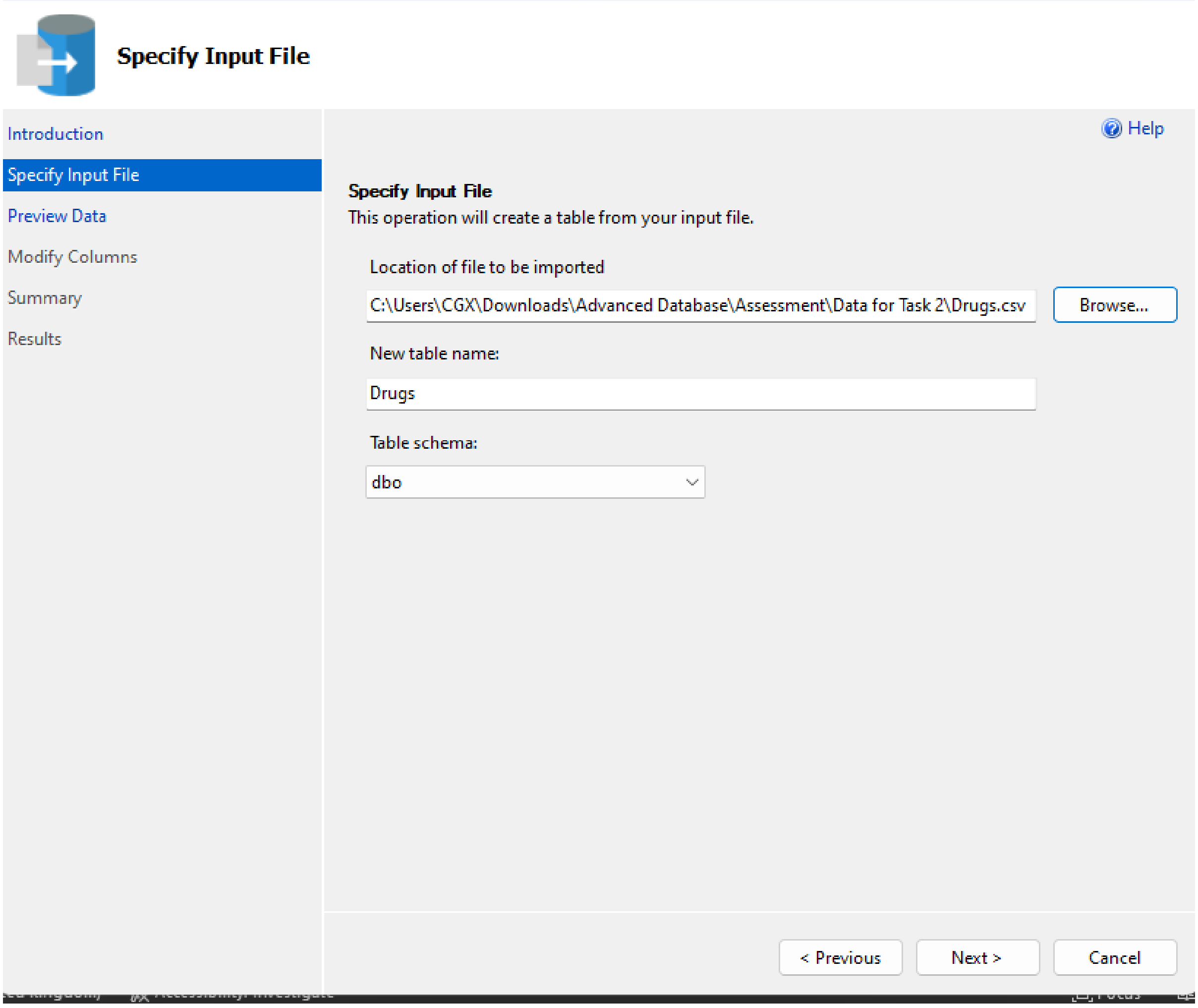
Summary:

Name	Result
✓ Insert Data	Success

Figure 7: Result of files imported

DRUGS

I clicked the browse button in **Figure 8** which allowed me to locate the Drugs filepath. Imputed the Drugs as the name of my table and left the table schema as default(dbo).



Specify Input File

Specify Input File

This operation will create a table from your input file.

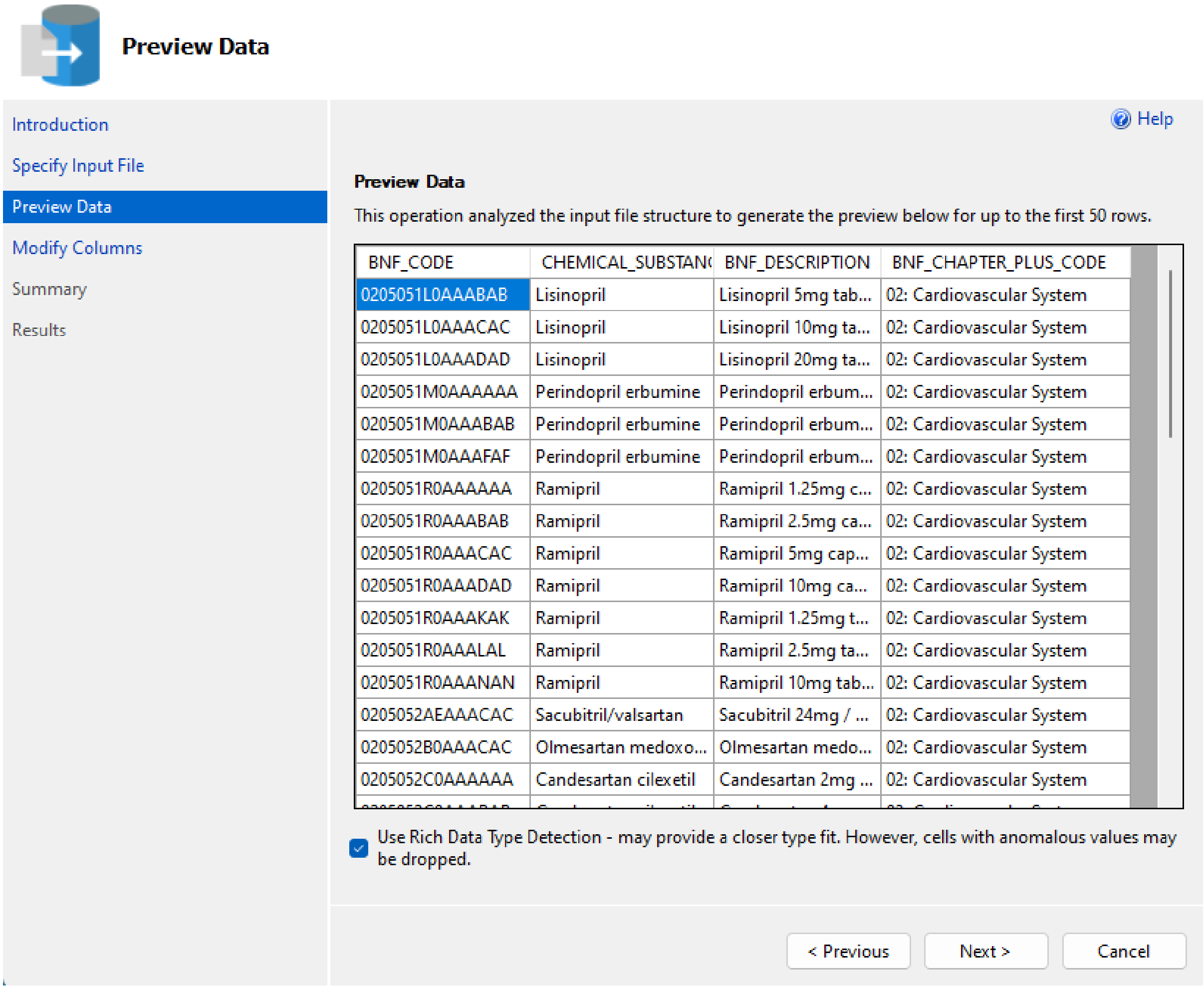
Location of file to be imported: C:\Users\CGX\Downloads\Advanced Database\Assessment\Data for Task 2\Drugs.csv

New table name: Drugs

Table schema: dbo

Figure 8: Specify Input file

After specifying the Drugs as the file to import. Then clicked on next and was led to the Preview data page. Here I was able to preview the first 50 rows of the Drugs data, view the data types and have an idea of the length of the character of the various columns.



Preview Data

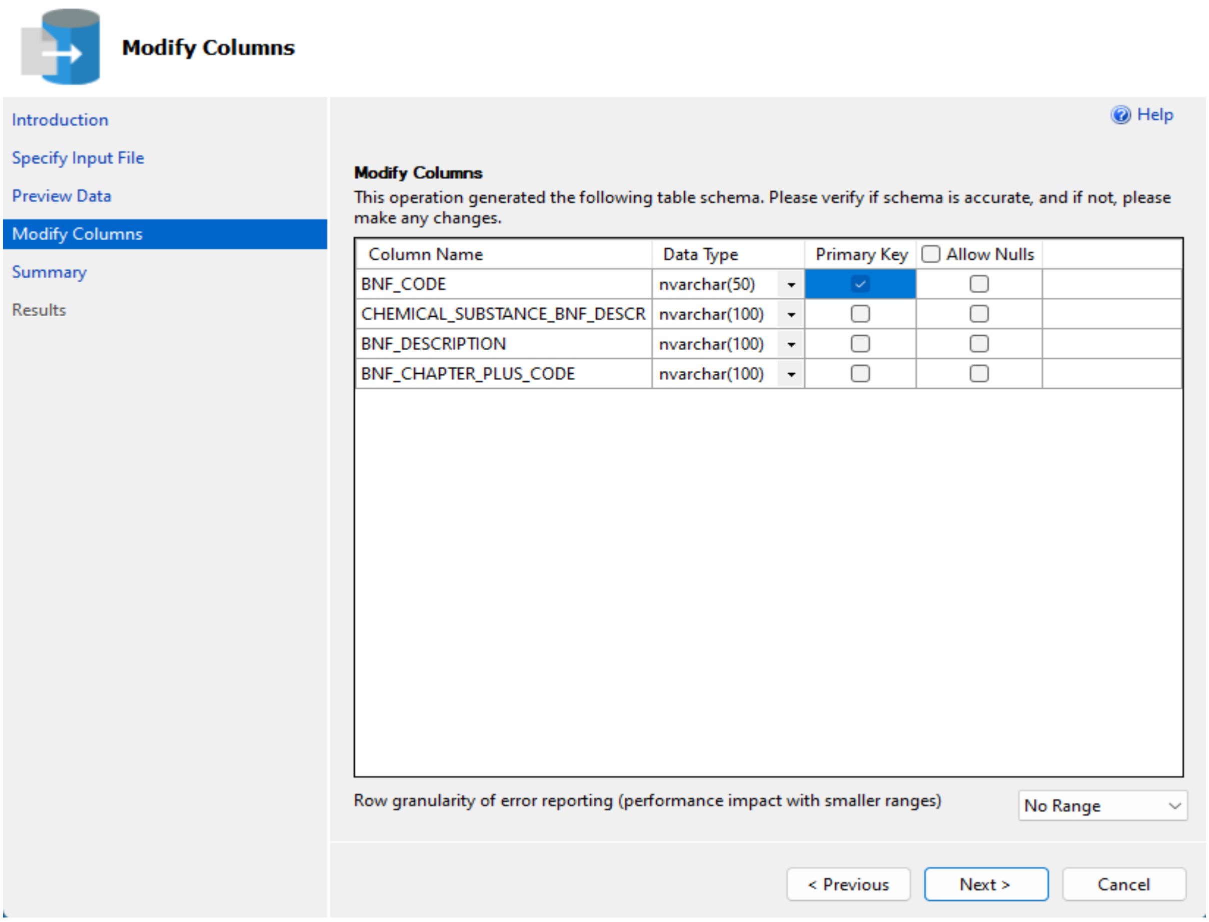
This operation analyzed the input file structure to generate the preview below for up to the first 50 rows.

BNF_CODE	CHEMICAL_SUBSTAN(	BNF_DESCRIPTION	BNF_CHAPTER_PLUS_CODE
0205051L0AAABAB	Lisinopril	Lisinopril 5mg tab...	02: Cardiovascular System
0205051L0AAACAC	Lisinopril	Lisinopril 10mg ta...	02: Cardiovascular System
0205051L0AAADAD	Lisinopril	Lisinopril 20mg ta...	02: Cardiovascular System
0205051M0AAAAAA	Perindopril erbumine	Perindopril erbum...	02: Cardiovascular System
0205051M0AAABAB	Perindopril erbumine	Perindopril erbum...	02: Cardiovascular System
0205051M0AAAFAF	Perindopril erbumine	Perindopril erbum...	02: Cardiovascular System
0205051R0AAAAAA	Ramipril	Ramipril 1.25mg c...	02: Cardiovascular System
0205051R0AAABAB	Ramipril	Ramipril 2.5mg ca...	02: Cardiovascular System
0205051R0AAACAC	Ramipril	Ramipril 5mg cap...	02: Cardiovascular System
0205051R0AAADAD	Ramipril	Ramipril 10mg ca...	02: Cardiovascular System
0205051R0AAAKAK	Ramipril	Ramipril 1.25mg t...	02: Cardiovascular System
0205051R0AAALAL	Ramipril	Ramipril 2.5mg ta...	02: Cardiovascular System
0205051R0AAANAN	Ramipril	Ramipril 10mg tab...	02: Cardiovascular System
0205052AEAAACAC	Sacubitril/valsartan	Sacubitril 24mg / ...	02: Cardiovascular System
0205052B0AAACAC	Olmesartan medoxo...	Olmesartan medo...	02: Cardiovascular System
0205052C0AAAAAA	Candesartan cilexetil	Candesartan 2mg ...	02: Cardiovascular System

☑ Use Rich Data Type Detection - may provide a closer type fit. However, cells with anomalous values may be dropped.

Figure 9: Preview of the Drugs dataset

Clicked on next and on this screen I modified the columns in my data. I made BNF_CODE the primary key has this contains unique characters for each row with no repetition. I changed the data type of both the CHEMICAL_SUBSTANCE_BNF_DESCR and BNF_CHAPTER_PLUS_CODE from nvarchar(50) to nvarchar(100) because the length of the characters of these columns requires more storage .



Modify Columns

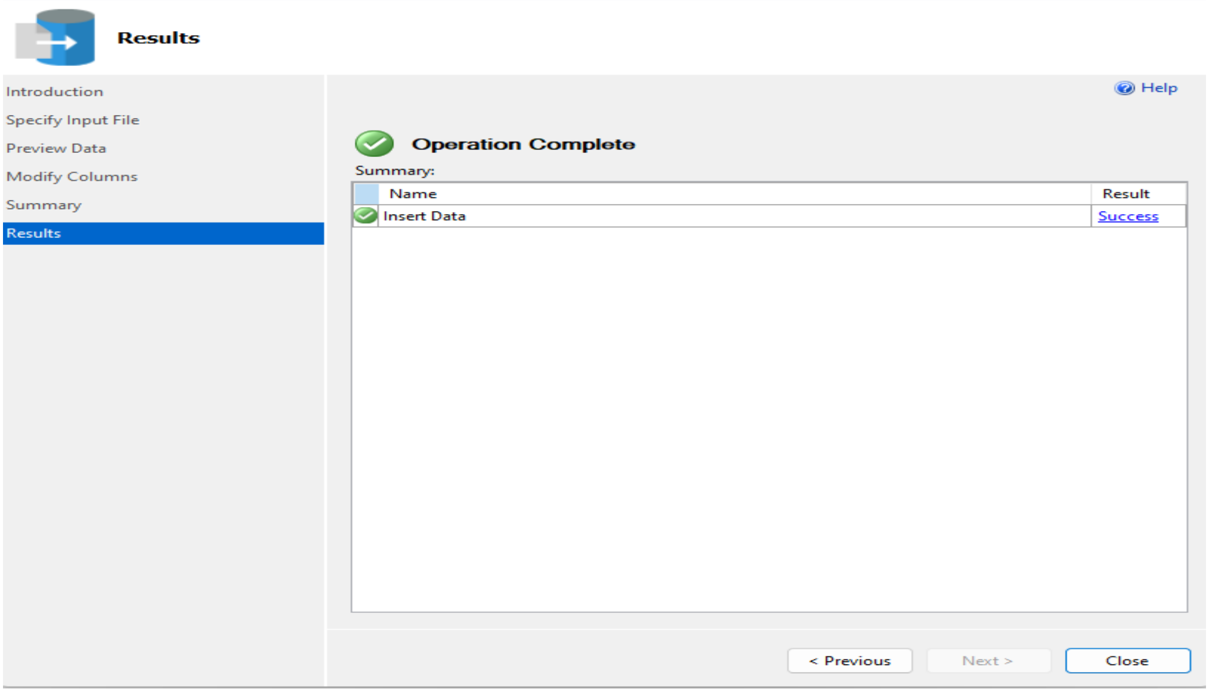
This operation generated the following table schema. Please verify if schema is accurate, and if not, please make any changes.

Column Name	Data Type	Primary Key	□ Allow Nulls
BNF_CODE	nvarchar(50)	☑	□
CHEMICAL_SUBSTANCE_BNF_DESCR	nvarchar(100)	□	□
BNF_DESCRIPTION	nvarchar(100)	□	□
BNF_CHAPTER_PLUS_CODE	nvarchar(100)	□	□

Row granularity of error reporting (performance impact with smaller ranges): No Range

Figure 10: Modify columns

Clicked on next and my file was imported successfully.



✓ Operation Complete

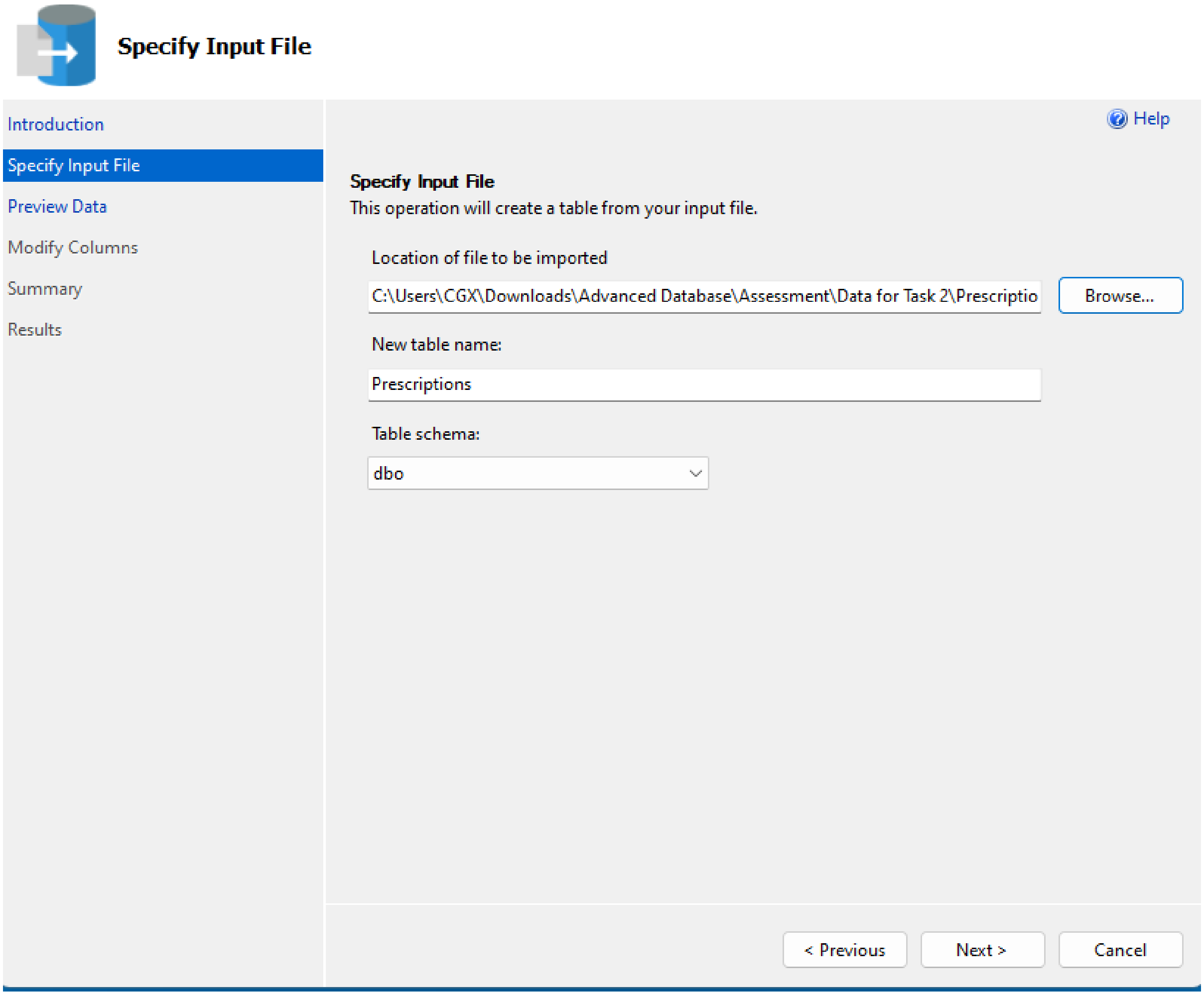
Summary:

Name	Result
✓ Insert Data	Success

Figure 11: Results showing a successful importation

PRESCRIPTION

I clicked the browse button which allowed me to locate the Prescription file path. Imputed the Prescriptions as the name of my table and left the table schema as default(dbo).



Specify Input File

This operation will create a table from your input file.

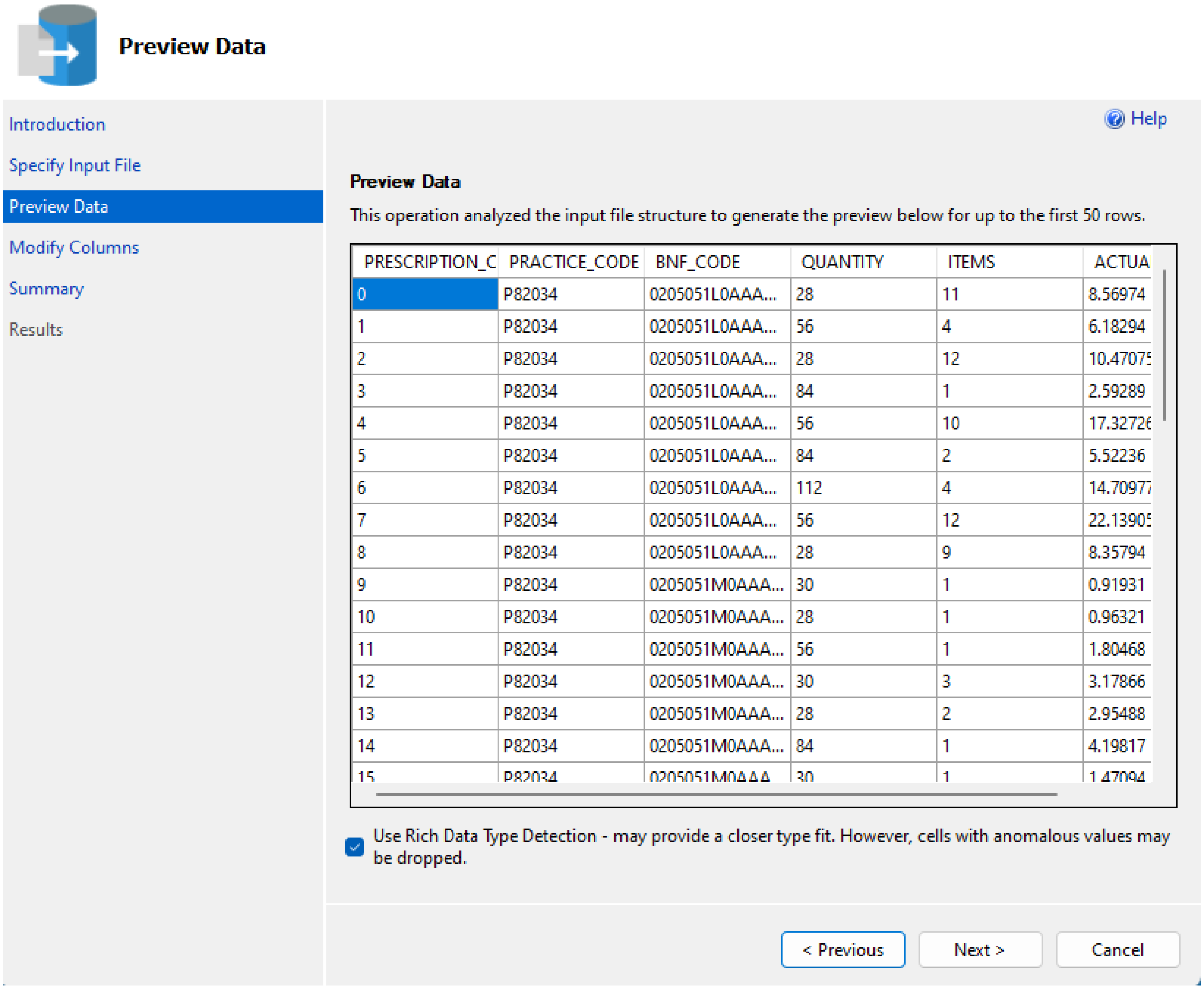
Location of file to be imported: C:\Users\CGX\Downloads\Advanced Database\Assessment\Data for Task 2\Prescriptio

New table name: Prescriptions

Table schema: dbo

Figure 12: Specify Input File

After specifying the Prescription as the file to import. Then clicked on next and was led to the Preview data page. Here I was able to preview the first 50 rows of the Prescription data, view the data types and have an idea of the length of the character of the various columns.



Preview Data

Preview Data

This operation analyzed the input file structure to generate the preview below for up to the first 50 rows.

PRESCRIPTION_C	PRACTICE_CODE	BNF_CODE	QUANTITY	ITEMS	ACTUA
0	P82034	0205051L0AAA...	28	11	8.56974
1	P82034	0205051L0AAA...	56	4	6.18294
2	P82034	0205051L0AAA...	28	12	10.47075
3	P82034	0205051L0AAA...	84	1	2.59289
4	P82034	0205051L0AAA...	56	10	17.32726
5	P82034	0205051L0AAA...	84	2	5.52236
6	P82034	0205051L0AAA...	112	4	14.70977
7	P82034	0205051L0AAA...	56	12	22.13905
8	P82034	0205051L0AAA...	28	9	8.35794
9	P82034	0205051M0AAA...	30	1	0.91931
10	P82034	0205051M0AAA...	28	1	0.96321
11	P82034	0205051M0AAA...	56	1	1.80468
12	P82034	0205051M0AAA...	30	3	3.17866
13	P82034	0205051M0AAA...	28	2	2.95488
14	P82034	0205051M0AAA...	84	1	4.19817
15	P82034	0205051M0AAA	30	1	1.47094

☑ Use Rich Data Type Detection - may provide a closer type fit. However, cells with anomalous values may be dropped.

Figure 13: Preview Data

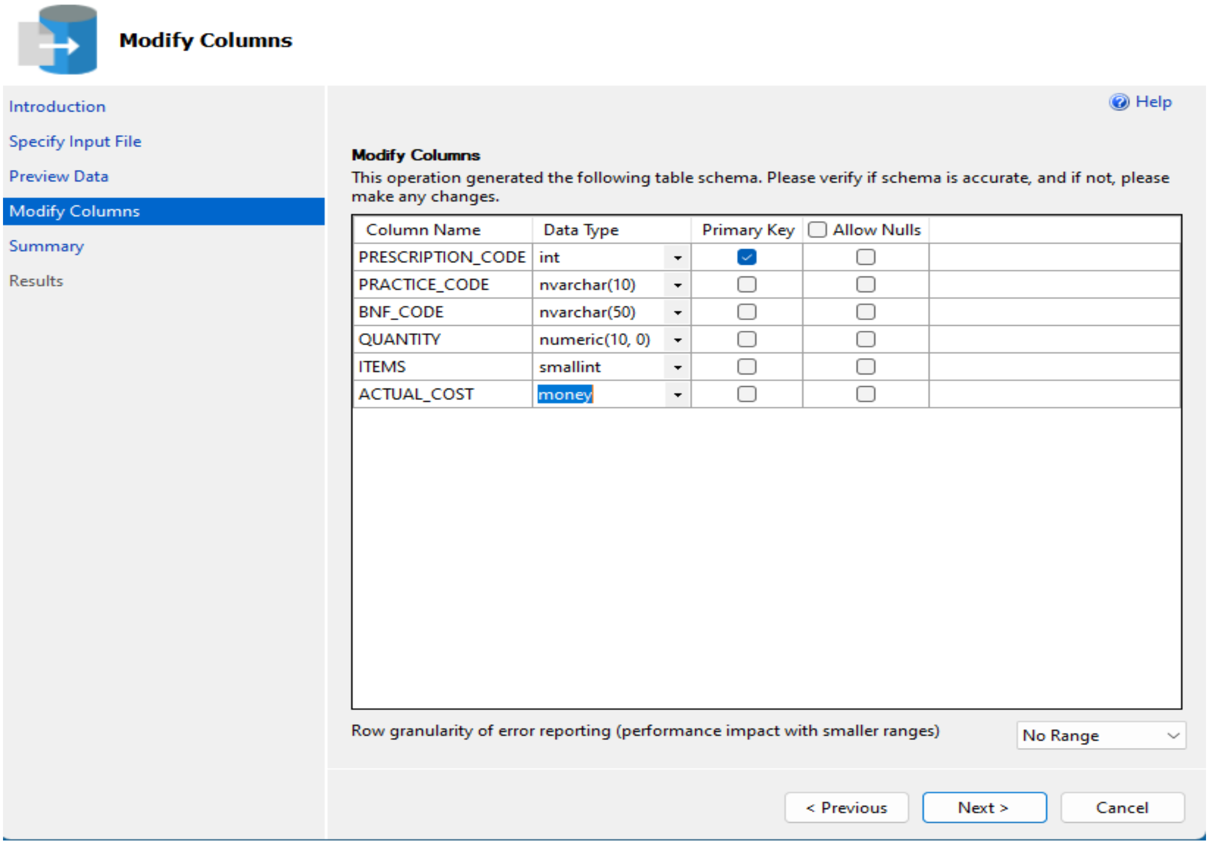
Clicked on next and on this screen I modified the columns in my data. I made PRESCRIPTION_CODE the primary key has this contains unique characters for each row with no repetition and used the data type INT because the column contains whole numbers only.

The PRACTICE_CODE and BNF_CODE has the data type nvarchar because both columns contains both numbers and letters in their characters.

The data type numeric() was used for the column QUANTITY because it contains both whole and decimal numbers.

Item column has the data type small int because they contain whole numbers which are too large to be stores in tinyint but not large enough for small int.

The last column ACTUAL_COST has the data type money because it contains the amount the item costs.



Modify Columns

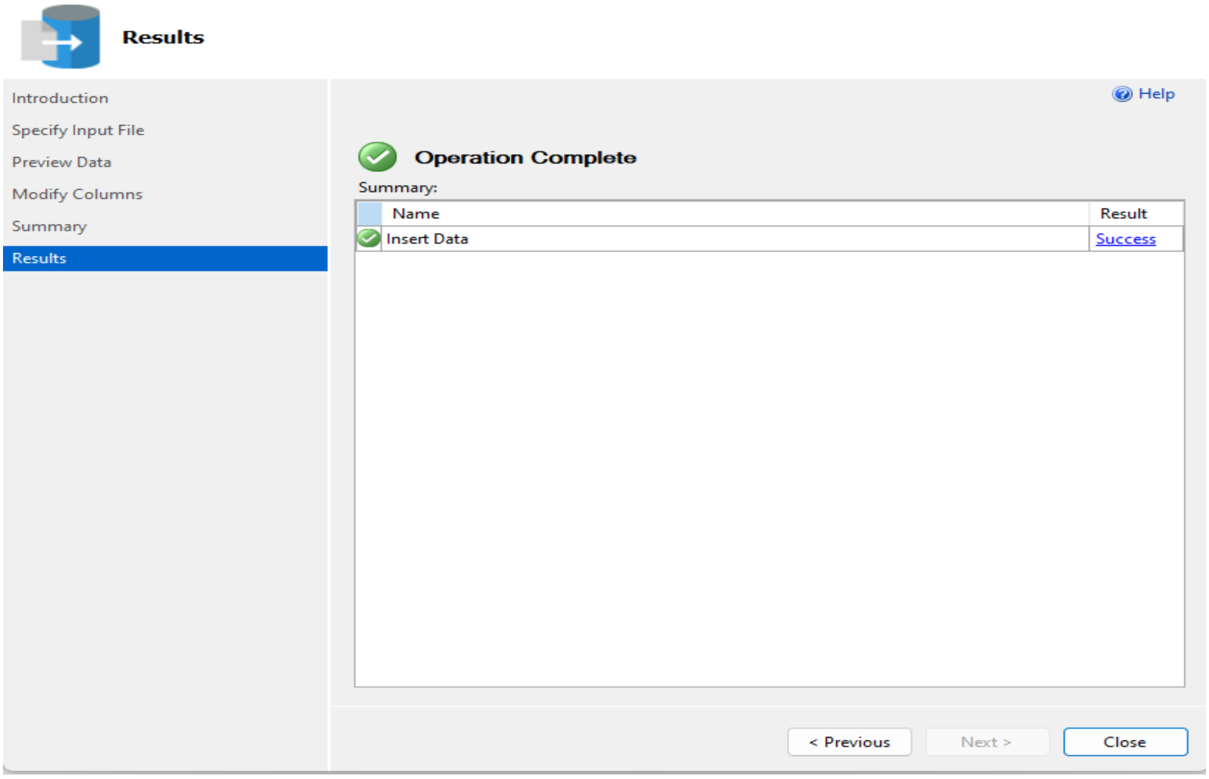
This operation generated the following table schema. Please verify if schema is accurate, and if not, please make any changes.

Column Name	Data Type	Primary Key	☐ Allow Nulls
PRESCRIPTION_CODE	int	☑	☐
PRACTICE_CODE	nvarchar(10)	☐	☐
BNF_CODE	nvarchar(50)	☐	☐
QUANTITY	numeric(10, 0)	☐	☐
ITEMS	smallint	☐	☐
ACTUAL_COST	money	☐	☐

Row granularity of error reporting (performance impact with smaller ranges): No Range

Figure 14: Modify Column

Clicked on next and my file was imported successfully.



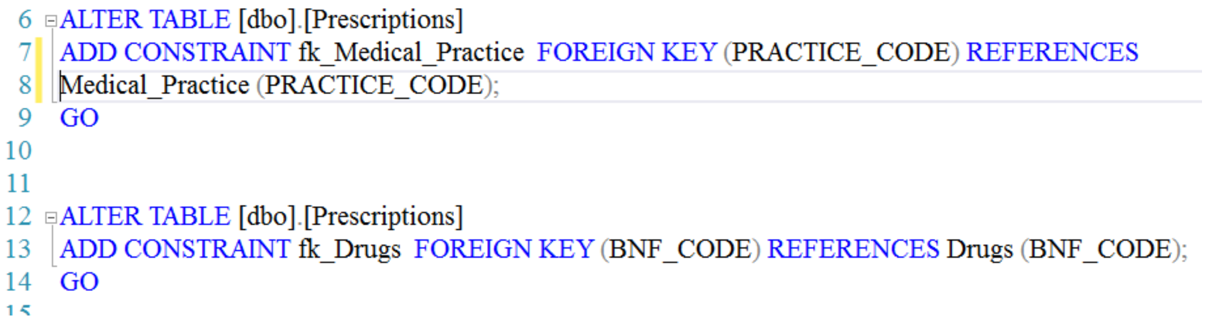
Operation Complete

Summary:

Name	Result
✓ Insert Data	Success

Figure 15: Results showing a successful import of data

After successfully importing all three tables I added the foreign key constraint to PRACTICE CODE and BNF_CODE column in the Prescription table using the ALTER statement as shown in Figure 17 below;



```
ALTER TABLE [dbo].[Prescriptions]
ADD CONSTRAINT fk_Medical_Practice  FOREIGN KEY (PRACTICE_CODE) REFERENCES
Medical_Practice (PRACTICE_CODE);
GO


ALTER TABLE [dbo].[Prescriptions]
ADD CONSTRAINT fk_Drugs  FOREIGN KEY (BNF_CODE) REFERENCES Drugs (BNF_CODE);
GO
```

Figure 16

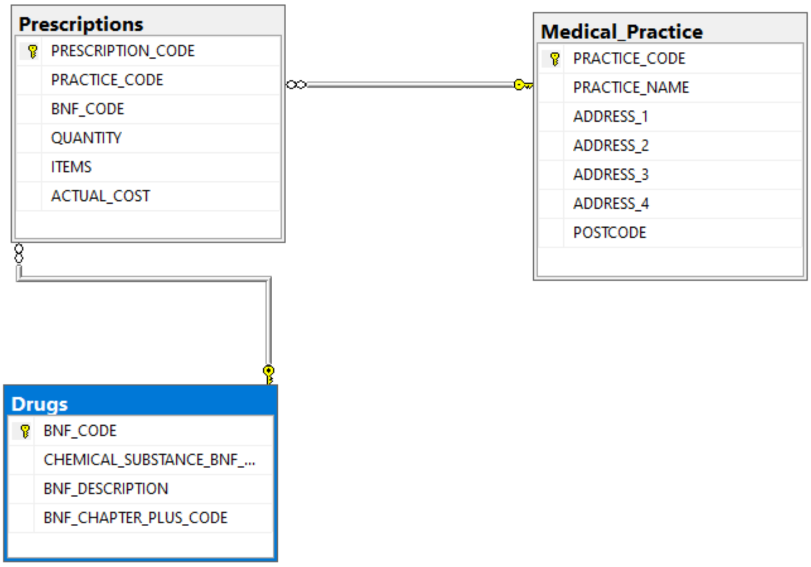
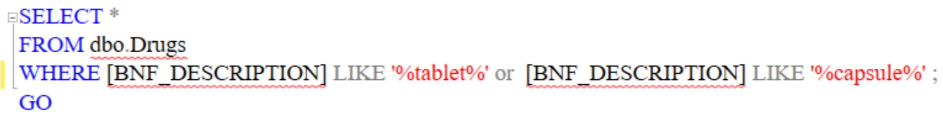


Figure 17: A Database Diagram showing the relationship between the tables

PART 2

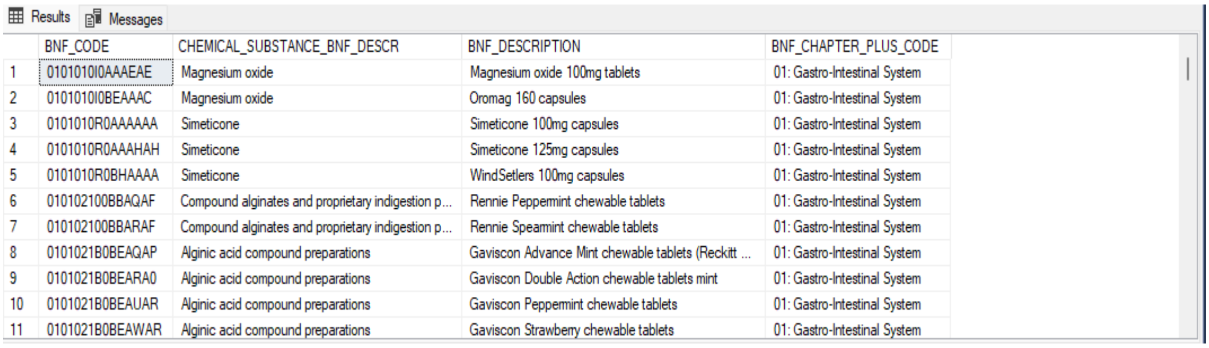
Assuming that every drug that is a tablet or capsule has these words in the BNF_DESCRIPTION column.



```
SELECT *
FROM dbo.Drugs
WHERE [BNF_DESCRIPTION] LIKE '%tablet%' or  [BNF_DESCRIPTION] LIKE '%capsule%' ;
GO
```

Figure **18**

I used the SELECT statement to select rows in each column FROM the DRUGS table where BNF_DESCRIPTION had the word tablets and capsules within its sentences.



	BNF_CODE	CHEMICAL_SUBSTANCE_BNF_DESCR	BNF_DESCRIPTION	BNF_CHAPTER_PLUS_CODE
1	0101010I0AAAEAE	Magnesium oxide	Magnesium oxide 100mg tablets	01: Gastro-Intestinal System
2	0101010I0BEAAAC	Magnesium oxide	Oromag 160 capsules	01: Gastro-Intestinal System
3	0101010R0AAAAAA	Simeticone	Simeticone 100mg capsules	01: Gastro-Intestinal System
4	0101010R0AAAHAH	Simeticone	Simeticone 125mg capsules	01: Gastro-Intestinal System
5	0101010R0BHAAAA	Simeticone	WindSetlers 100mg capsules	01: Gastro-Intestinal System
6	010102100BBAQAF	Compound alginates and proprietary indigestion p...	Rennie Peppermint chewable tablets	01: Gastro-Intestinal System
7	010102100BBARAF	Compound alginates and proprietary indigestion p...	Rennie Speamint chewable tablets	01: Gastro-Intestinal System
8	0101021B0BEAQAP	Alginic acid compound preparations	Gaviscon Advance Mint chewable tablets (Reckitt ...	01: Gastro-Intestinal System
9	0101021B0BEARA0	Alginic acid compound preparations	Gaviscon Double Action chewable tablets mint	01: Gastro-Intestinal System
10	0101021B0BEAUAR	Alginic acid compound preparations	Gaviscon Peppermint chewable tablets	01: Gastro-Intestinal System
11	0101021B0BEAWAR	Alginic acid compound preparations	Gaviscon Strawberry chewable tablets	01: Gastro-Intestinal System

Figure **19**: Result of Query in Figure 18

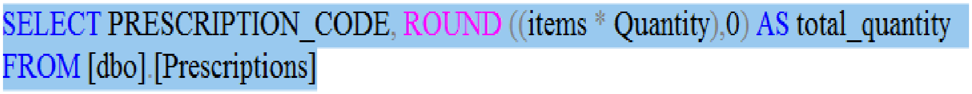
Table 1 above shows the first 11 rows of the result of query in figure 18. Every row in BNF_DESCRIPTION has the either the word tablet, tablets, capsule or capsules

PART 3

I wrote the query in Figure 20 to get the total quantity of each prescription. In the query.

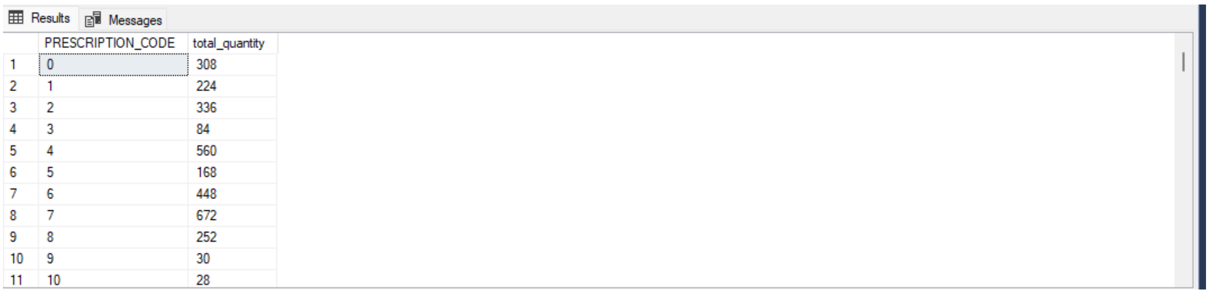
Using the SELECT statement I retrieved PRESCITION_CODE, ITEMS and QUANTITY FROM the PRESCRITIONS table.

To return the total quantity I performed an arithmetic operation multiply the ITEMS and QUANTITY columns. I also used ROUND(), stating zero(0) as the parameter so as to round the total quantity up to the nearest the integer value.



```
SELECT PRESCRIPTION_CODE, ROUND ((items * Quantity),0) AS total_quantity
FROM [dbo].[Prescriptions]
```

Figure **20**



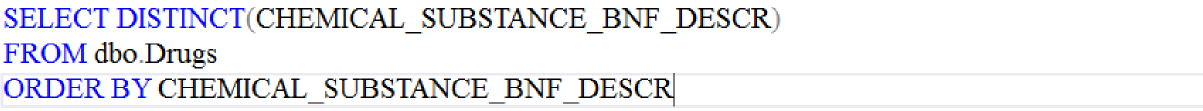
	PRESCRIPTION_CODE	total_quantity
1	0	308
2	1	224
3	2	336
4	3	84
5	4	560
6	5	168
7	6	448
8	7	672
9	8	252
10	9	30
11	10	28

Figure 21: Result of Query in Figure 20

The result in figure 21 shows the prescription code and the total quantity of each prescription.

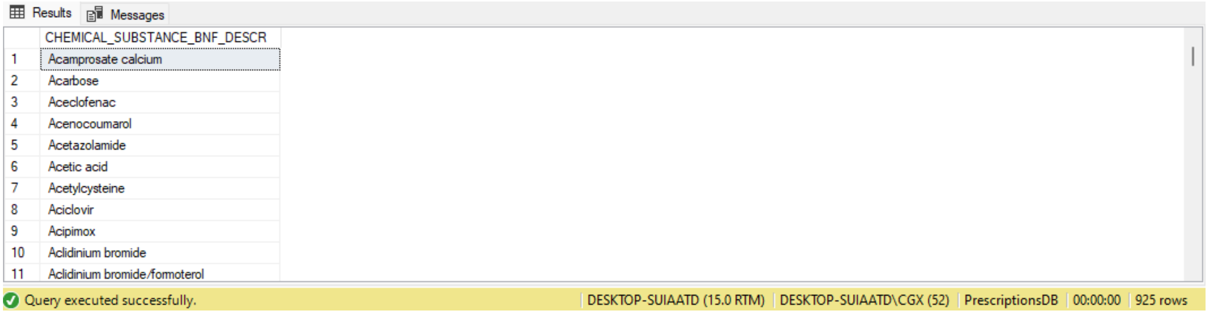
PART 4

The SELECT DISTINCT statement is used to retrieve a distinct list of chemical substances in the CHEMICAL_SUBSTANCE_BNF_DESCR column.



```
SELECT DISTINCT(CHEMICAL_SUBSTANCE_BNF_DESCR)
FROM dbo.Drugs
ORDER BY CHEMICAL_SUBSTANCE_BNF_DESCR
```

Figure 22



	CHEMICAL_SUBSTANCE_BNF_DESCR
1	Acamprosate calcium
2	Acarbose
3	Aceclofenac
4	Acenocoumarol
5	Acetazolamide
6	Acetic acid
7	Acetylcysteine
8	Aciclovir
9	Acipimox
10	Aclidinium bromide
11	Aclidinium bromide/formoterol

Query executed successfully. | DESKTOP-SUIAATD (15.0 RTM) | DESKTOP-SUIAATD\CGX (52) | PrescriptionsDB | 00:00:00 | 925 rows

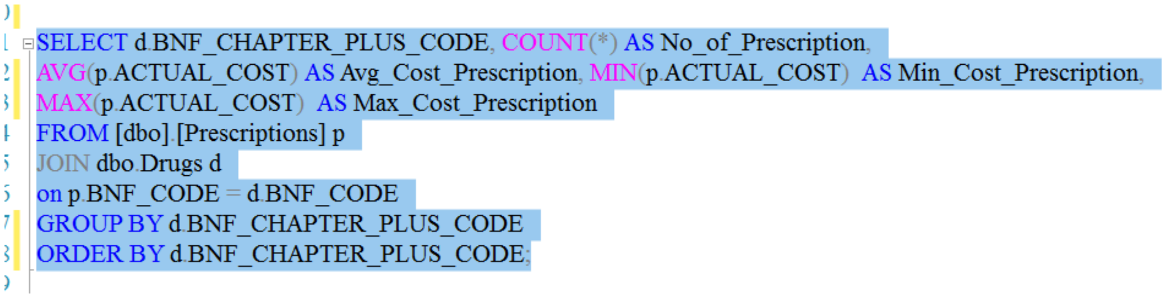
Figure 23: Results of the query in Figure 22

I used the ORDER BY clause to sort the list alphabetically.

PART 5

The query in figure 21 below uses the SELECT statement to return the BNF_CHAPTER_PLUS_CODE.

I used the function COUNT() to retrieve the total number of prescription, AVG() the retrieve the average cost of prescription from the ACTUAL_COST, MIN() to retrieve the minimum cost of prescription from the ACTUAL_COST, MAX() to retrieve the maximum cost of prescription from the ACTUAL_COST

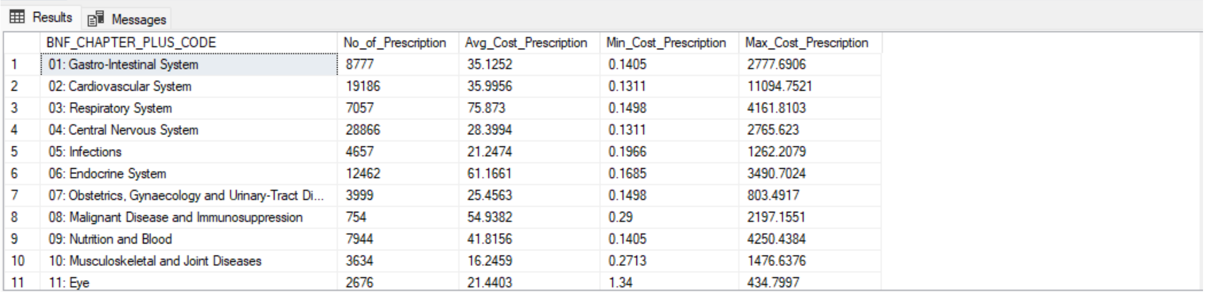


```
SELECT d.BNF_CHAPTER_PLUS_CODE, COUNT(*) AS No_of_Prescription,
AVG(p.ACTUAL_COST) AS Avg_Cost_Prescription, MIN(p.ACTUAL_COST)  AS Min_Cost_Prescription,
MAX(p.ACTUAL_COST)  AS Max_Cost_Prescription
FROM [dbo].[Prescriptions] p
JOIN dbo.Drugs d
on p.BNF_CODE = d.BNF_CODE
GROUP BY d.BNF_CHAPTER_PLUS_CODE
ORDER BY d.BNF_CHAPTER_PLUS_CODE;
```

I joined the Prescriptions and Drugs table on their BNF_CODE column to be able to access the column I needed in both tables.

I used the GROUP BY the BNF_CHAPTER_PLUS_CODE column together. Finally I ordered it by the BNF_CHAPTER_PLUS_CODE.

Figure 24



	BNF_CHAPTER_PLUS_CODE	No_of_Prescription	Avg_Cost_Prescription	Min_Cost_Prescription	Max_Cost_Prescription
1	01: Gastro-Intestinal System	8777	35.1252	0.1405	2777.6906
2	02: Cardiovascular System	19186	35.9956	0.1311	11094.7521
3	03: Respiratory System	7057	75.873	0.1498	4161.8103
4	04: Central Nervous System	28866	28.3994	0.1311	2765.623
5	05: Infections	4657	21.2474	0.1966	1262.2079
6	06: Endocrine System	12462	61.1661	0.1685	3490.7024
7	07: Obstetrics, Gynaecology and Urinary-Tract Di...	3999	25.4563	0.1498	803.4917
8	08: Malignant Disease and Immunosuppression	754	54.9382	0.29	2197.1551
9	09: Nutrition and Blood	7944	41.8156	0.1405	4250.4384
10	10: Musculoskeletal and Joint Diseases	3634	16.2459	0.2713	1476.6376
11	11: Eye	2676	21.4403	1.34	434.7997

Figure 25: Result of query from Figure 24

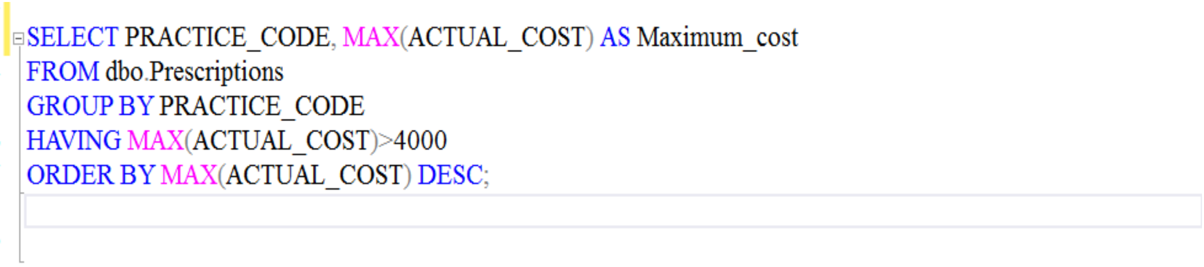
PART 6

In this query I used the SELECT statement to return the PRATICE_CODE, and used the aggregate function MAX() on the ACTUAL_COST column FROM the Prescriptions table.

I used GROUP BY the PRACTICE_CODE column.

The HAVING column is used to filter out any row where the maximum ACTUAL_COST is greater £4000.

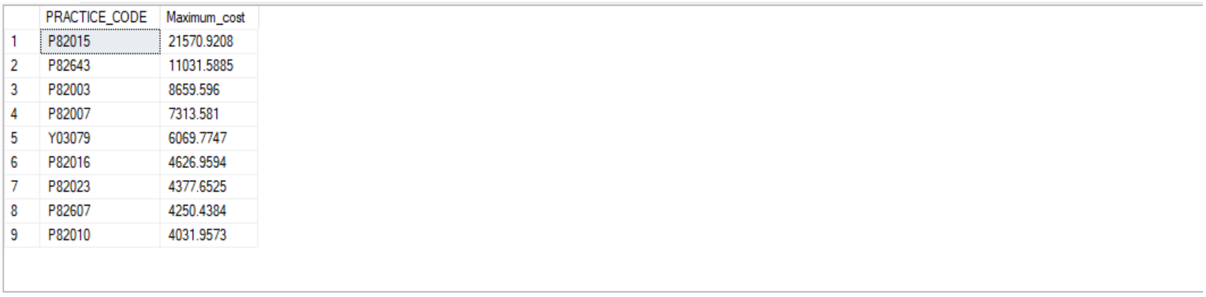
Finally I ordered the table by the maximum cost from lowest to highest



```
SELECT PRACTICE_CODE, MAX(ACTUAL_COST) AS Maximum_cost
FROM dbo.Prescriptions
GROUP BY PRACTICE_CODE
HAVING MAX(ACTUAL_COST)>4000
ORDER BY MAX(ACTUAL_COST) DESC;
```

Figure 26

The



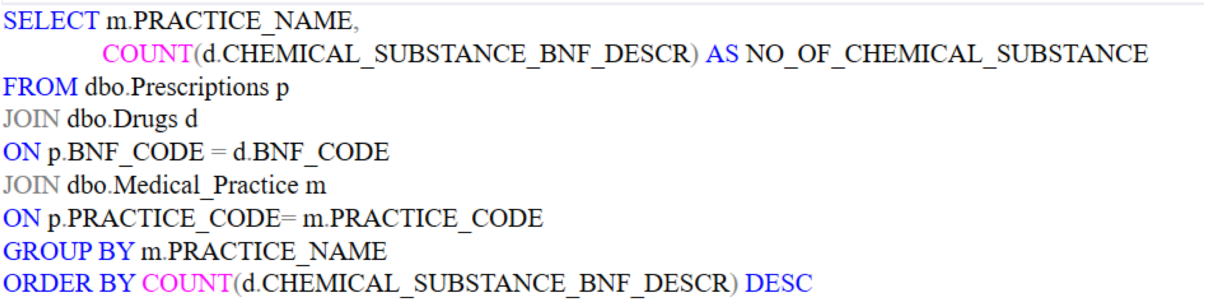
	PRACTICE_CODE	Maximum_cost
1	P82015	21570.9208
2	P82643	11031.5885
3	P82003	8659.596
4	P82007	7313.581
5	Y03079	6069.7747
6	P82016	4626.9594
7	P82023	4377.6525
8	P82607	4250.4384
9	P82010	4031.9573

Figure 27: Result of the Query in Figure 26

PART 7a

The query below was used to get the total number of chemical substance used by each practice.

The SELECT statement returns the PRACTICE_NAME from the Medical_Pratice table and it counts the number of CHEMICAL_SUBSTANCE_BNF_DESCR used from the Drugs table.

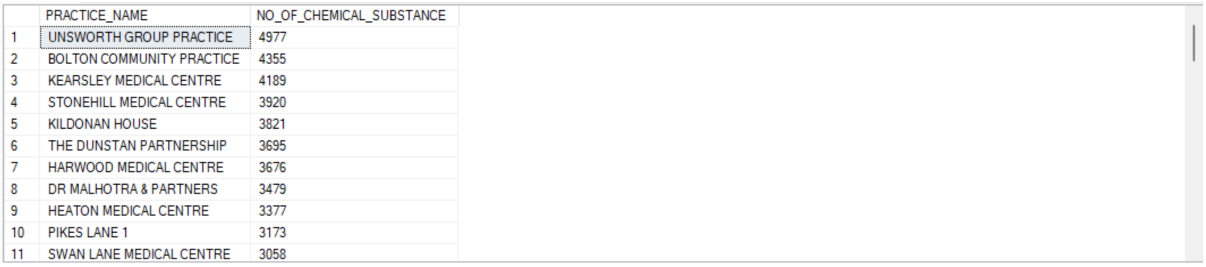


```
SELECT m.PRACTICE_NAME,
        COUNT(d.CHEMICAL_SUBSTANCE_BNF_DESCR) AS NO_OF_CHEMICAL_SUBSTANCE
FROM dbo.Prescriptions p
JOIN dbo.Drugs d
ON p.BNF_CODE = d.BNF_CODE
JOIN dbo.Medical_Practice m
ON p.PRACTICE_CODE= m.PRACTICE_CODE
GROUP BY m.PRACTICE_NAME
ORDER BY COUNT(d.CHEMICAL_SUBSTANCE_BNF_DESCR) DESC
```

Figure 28

Medical_Practice and Drugs table have no foreign keys connecting them together, Using JOIN I combined Prescriptions and Drugs tables on the BNF_CODE column. Then used the JOIN statement to join Medical_Practice to Prescriptions table on the PRACTICE_CODE column which is present in.

I used GROUP BY clause to group the number of chemical substance by the PRACTICE_NAME. Then used ORDER BY to get the practice which uses the most chemical substance .

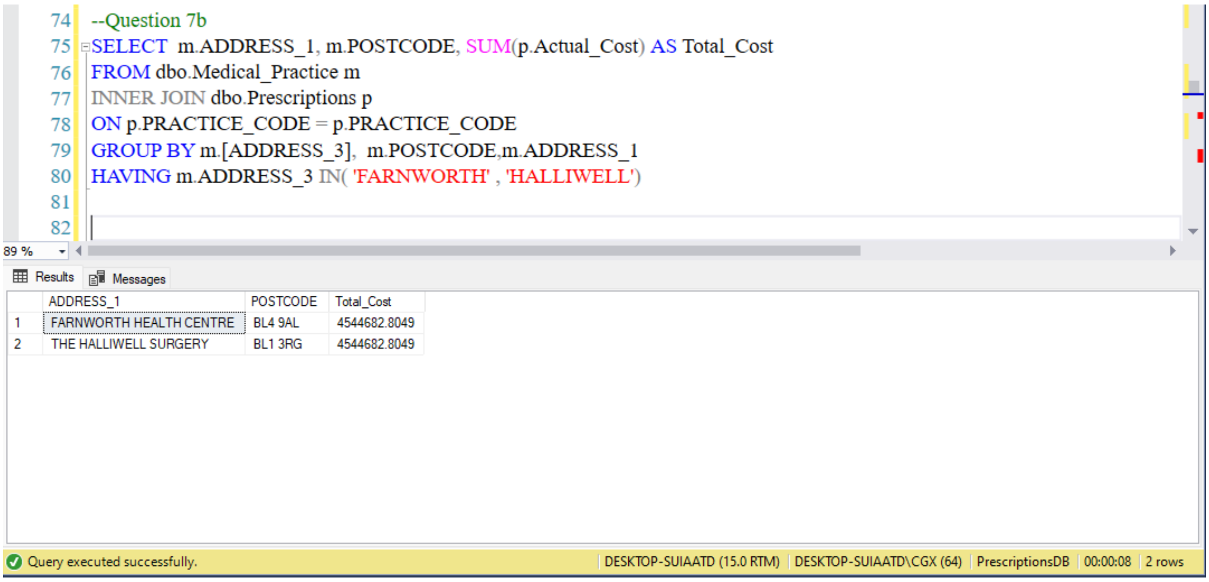


	PRACTICE_NAME	NO_OF_CHEMICAL_SUBSTANCE
1	UNSWORTH GROUP PRACTICE	4977
2	BOLTON COMMUNITY PRACTICE	4355
3	KEARSLEY MEDICAL CENTRE	4189
4	STONEHILL MEDICAL CENTRE	3920
5	KILDONAN HOUSE	3821
6	THE DUNSTAN PARTNERSHIP	3695
7	HARWOOD MEDICAL CENTRE	3676
8	DR MALHOTRA & PARTNERS	3479
9	HEATON MEDICAL CENTRE	3377
10	PIKES LANE 1	3173
11	SWAN LANE MEDICAL CENTRE	3058

Figure 29

PART 7b

The query below shows the total cost of prescription used by medical practice in Farnworth or Halliwell. It is assumed that the Adress_1 and postcode column is are enough to determine the address



```
--Question 7b
SELECT  m.ADDRESS_1, m.POSTCODE, SUM(p.Actual_Cost) AS Total_Cost
FROM dbo.Medical_Practice m
INNER JOIN dbo.Prescriptions p
ON p.PRACTICE_CODE = p.PRACTICE_CODE
GROUP BY m.[ADDRESS_3],  m.POSTCODE,m.ADDRESS_1
HAVING m.ADDRESS_3 IN( 'FARNWORTH' , 'HALLIWELL')
```

	ADDRESS_1	POSTCODE	Total_Cost
1	FARNWORTH HEALTH CENTRE	BL4 9AL	4544682.8049
2	THE HALLIWELL SURGERY	BL1 3RG	4544682.8049

Query executed successfully. | DESKTOP-SUIAATD (15.0 RTM) | DESKTOP-SUIAATD\CGX (64) | PrescriptionsDB | 00:00:08 | 2 rows

Figure 30

Using the SELECT statement to retrieve the ADDRESS_1, POSTCODE and ACTUAL_COST, used the SUM function on ACTUAL_COST to the get Total_Cost.

Used JOIN to combine Medical_Practice and Prescriptions tables on PRACTICE_CODE to be able to retrieve this columns.

GROUP BY the ADDRESS_3, ADDRESS_1 and POSTCODE column.

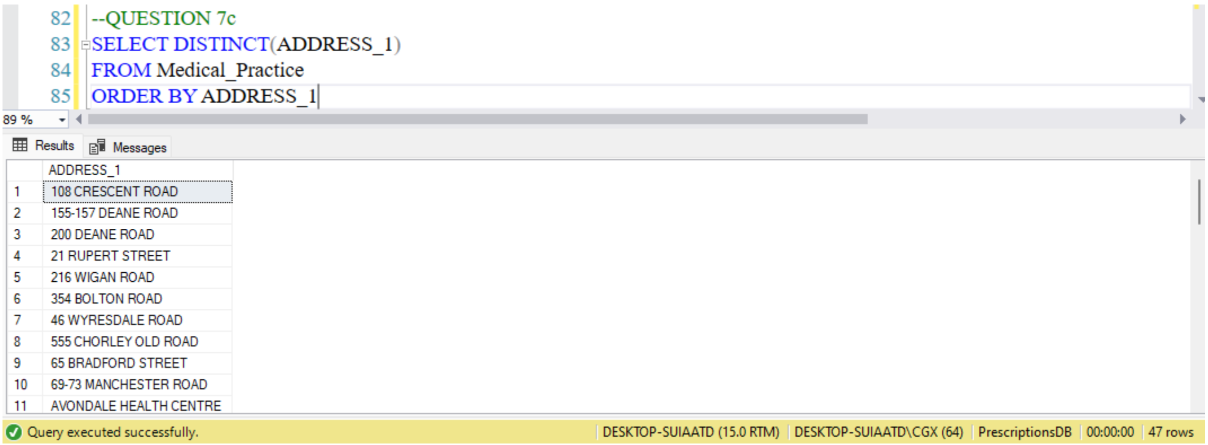
Using HAVING to filter out rows that have Farnworth or Halliwell as their ADRESS_3.

The result shows that both Farnworth or Halliwell spent the same amount on prescription.

PART 7C

The query below shows a distinct list of ADDRESS_1 in the Medical_Practice table.

SELECT DISTINCT statement to retrieve the distinct addresses in the ADRESS_1 column from Medical_Practice table.



```
--QUESTION 7c
SELECT DISTINCT(ADDRESS_1)
FROM Medical_Practice
ORDER BY ADDRESS_1
```

	ADDRESS_1
1	108 CRESCENT ROAD
2	155-157 DEANE ROAD
3	200 DEANE ROAD
4	21 RUPERT STREET
5	216 WIGAN ROAD
6	354 BOLTON ROAD
7	46 WYRESDALE ROAD
8	555 CHORLEY OLD ROAD
9	65 BRADFORD STREET
10	69-73 MANCHESTER ROAD
11	AVONDALE HEALTH CENTRE

```
Query executed successfully.    DESKTOP-SUIAATD (15.0 RTM) | DESKTOP-SUIAATD\CGX (64) | PrescriptionsDB | 00:00:00 | 47 rows
```

Figure 31

The result shows that there are 47 distinct address in the ADDRESS_1 column.

To sort it I used ORDER by ADDRESS_1 in ascending order.

PART 7D

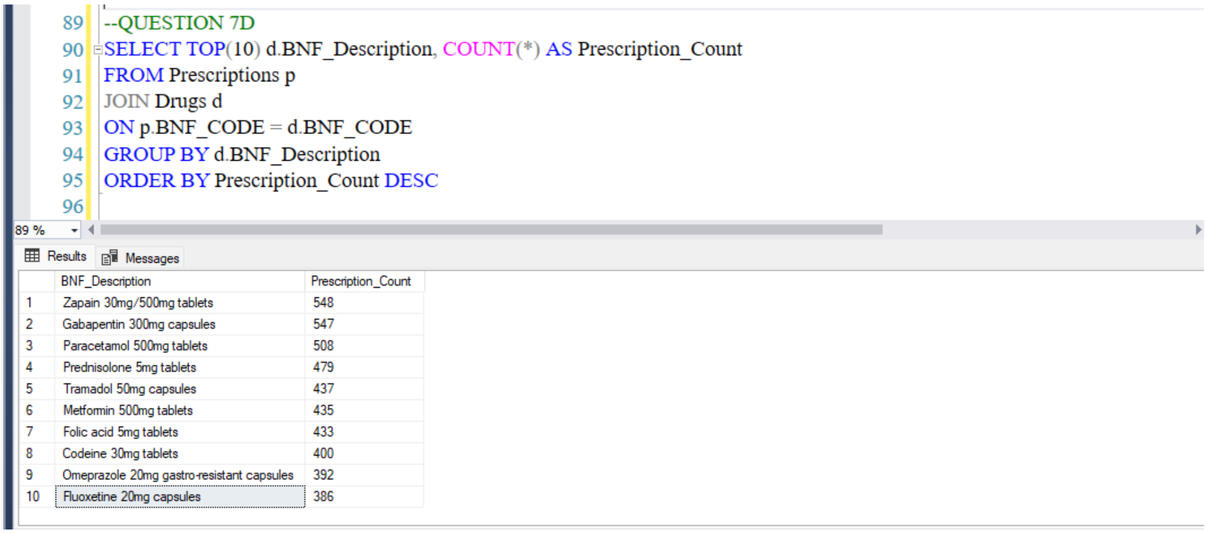
This query shows the top ten Presciption and their BNF_Description.

the SELECT statement to retrieves the BNF_Description.

I used the function COUNT() to retrieve the total number of prescription, and TOP() to retrieve the first ten rows

I joined the Prescriptions and Drugs table on their BNF_CODE column to be able to access the column I needed in both tables.

GROUP BY the BNF_Description column and using ORDER BY to sort the table by the Prescription_Count in the order of highest to lowest.



```sql
--QUESTION 7D
SELECT TOP(10) d.BNF_Description, COUNT(*) AS Prescription_Count
FROM Prescriptions p
JOIN Drugs d
ON p.BNF_CODE = d.BNF_CODE
GROUP BY d.BNF_Description
ORDER BY Prescription_Count DESC
```

	BNF_Description	Prescription_Count
1	Zapain 30mg/500mg tablets	548
2	Gabapentin 300mg capsules	547
3	Paracetamol 500mg tablets	508
4	Prednisolone 5mg tablets	479
5	Tramadol 50mg capsules	437
6	Metformin 500mg tablets	435
7	Folic acid 5mg tablets	433
8	Codeine 30mg tablets	400
9	Omeprazole 20mg gastro-resistant capsules	392
10	Fluoxetine 20mg capsules	386

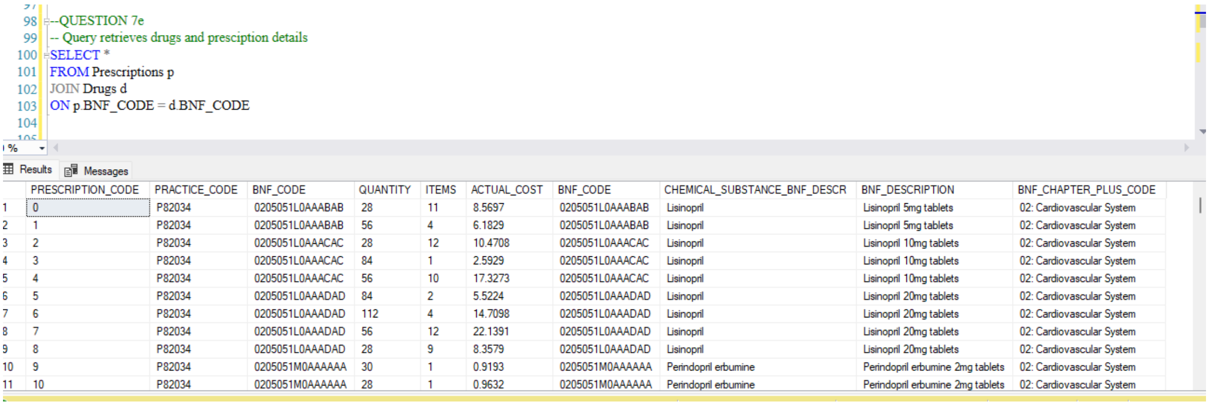
Figure 32

The result shows that Zapain 30mg/500mg tablets is the highest prescription count with 548 counts.

PART 7e

This query retrieves the all drugs and its prescription details

Using SELECT statement to retrieve all the rows FROM Prescriptions JOIN Drugs ON BNF_CODE column.



```sql
--QUESTION 7e
-- Query retrieves drugs and prescription details
SELECT *
FROM Prescriptions p
JOIN Drugs d
ON p.BNF_CODE = d.BNF_CODE
```

	PRESCRIPTION_CODE	PRACTICE_CODE	BNF_CODE	QUANTITY	ITEMS	ACTUAL_COST	BNF_CODE	CHEMICAL_SUBSTANCE_BNF_DESCR	BNF_DESCRIPTION	BNF_CHAPTER_PLUS_CODE
1	0	P82034	0205051L0AAABAB	28	11	8.5697	0205051L0AAABAB	Lisinopril	Lisinopril 5mg tablets	02: Cardiovascular System
2	1	P82034	0205051L0AAABAB	56	4	6.1829	0205051L0AAABAB	Lisinopril	Lisinopril 5mg tablets	02: Cardiovascular System
3	2	P82034	0205051L0AAACAC	28	12	10.4708	0205051L0AAACAC	Lisinopril	Lisinopril 10mg tablets	02: Cardiovascular System
4	3	P82034	0205051L0AAACAC	84	1	2.5929	0205051L0AAACAC	Lisinopril	Lisinopril 10mg tablets	02: Cardiovascular System
5	4	P82034	0205051L0AAACAC	56	10	17.3273	0205051L0AAACAC	Lisinopril	Lisinopril 10mg tablets	02: Cardiovascular System
6	5	P82034	0205051L0AAADAD	84	2	5.5224	0205051L0AAADAD	Lisinopril	Lisinopril 20mg tablets	02: Cardiovascular System
7	6	P82034	0205051L0AAADAD	112	4	14.7098	0205051L0AAADAD	Lisinopril	Lisinopril 20mg tablets	02: Cardiovascular System
8	7	P82034	0205051L0AAADAD	56	12	22.1391	0205051L0AAADAD	Lisinopril	Lisinopril 20mg tablets	02: Cardiovascular System
9	8	P82034	0205051L0AAADAD	28	9	8.3579	0205051L0AAADAD	Lisinopril	Lisinopril 20mg tablets	02: Cardiovascular System
10	9	P82034	0205051M0AAAAAA	30	1	0.9193	0205051M0AAAAAA	Perindopril erbumine	Perindopril erbumine 2mg tablets	02: Cardiovascular System
11	10	P82034	0205051M0AAAAAA	28	1	0.9632	0205051M0AAAAAA	Perindopril erbumine	Perindopril erbumine 2mg tablets	02: Cardiovascular System

Figure 33

CONCLUSION

In conclusion, using SQL I was able to get insight in making analysis from the three csv files provided. This was achieved using system function, SELECT statement, GROUPBY, ORDER BY, WHERE . I was able to see trends and patterns within the datasets. This helped understands how drugs are been prescribed by medical practice in Bolton