SeeSharp

Project Report

Adarsh Ravi
Aeshna Kapoor
Bhavika Bhagaria
Shreyas Krishna
12-14-2018

Table of Contents

1. Revision History 2

2. Datasets Overview
    2.1 Open Payment Data 3
        2.1.1 General Spend 3
        2.1.2 Research Spend 3
        2.1.3 Ownership Spend 3

3. Implementation
    3.1 Open Payments 3
    3.2 Ownership Spend Model 4
    3.3 Data Integration 5

4. Visualizations
    4.1 Investment v Return 8
    4.2 Investment by State 9
    4.3 Manufacturer Invested In 9
    4.4 Investment by Country 10
    4.5 Terms of Interest 10
    4.6 Investment-Value Trend Line 11

5. Analysis and Reporting
    5.1 Analysis to retrieve top 10 earners 11
    5.2 Report 13

1. Revision History

11/12/2018 version 1

- Detailed an overview of the various datasets provided
- Basic plan created for use of open payments dataset

11/26/2018 version 2

- Created data model for ownership spend data
- Created summary table for ownership spend data

11/28/2018 version 2

- Outlined plan for use of Tax returns dataset
- Created summary tables for tax returns

12/05/2018 version 3

- Normalized ownership spend dataset
- Created SSIS package to perform data integration

12/08/2018 version 4

- Created visualizations on tableau
- Created SSRS report

12/10/2018 version 5

- Edited SSRS report
- Added visualizations and reports to document

2. Dataset Overview

2.1 Open Payments

Open Payments collects information about financial relationships between healthcare industries and healthcare providers. And this information is made available to the public to facilitate transparency of the healthcare system. This system allows data accessors to make informed financial decisions when it comes to making purchases, providing or obtaining healthcare services.

The following information is made available on the Open Payments website:

2.1.1 General Spend

Includes details of payments made to physicians or teaching hospitals not related to research.

Data captured in the bucket

Here, information about the recipient of each payment is provided that includes recipient type (physician or teaching hospital), name, address, details of the practice and more along with information about the payment such as its nature, purpose, value, date and benefactor details

Example

On 4/1/2014, Surgi-Care Inc paid Brigham And Women's Hospital $5604.53 towards Space Rental or facility fees in Boston, MA.

2.1.2 Research Spend

Includes details of payments made towards research by physicians or teaching hospitals

Data captured in the bucket

Here, along with all the information about the benefactors, recipients, and payments, details of the research towards which payments are made is given. This includes the name of the study, associated drug, context of the research and link to its information, clinical trials and so on. It also provides identifying information of the involved principal investigators

Example

On 8/5/2013, Eli Lilly and Company paid the Teaching Hospital, Presence Hospitals PRV in Urbana, IL for conducting research on the drug EFFIENT. The research was primarily carried out by Principal Investigator, Dennis M Killian whose specializes in Internal Medicine.

2.1.3 Ownership Spend

Includes details of ownership and investment interest of physicians.

Data captured in the bucket

This bucket holds along with the identifying information of physicians, details of the yearly investment made by a physician on a particular GPO or Applicable manufacture, his/her value of interest and whether the ownership is held by them or an immediate family member

Example

In 2013, Geoffrey Ennis Clark invested $43,581,486 in Braintree Laboratories, Inc in Massachusetts.

3. Implementation

3.1 Open Payments

Combining Open Payments datasets for different years

The Open Payments website provides data for years 2013 through 2017. Each type of the dataset i.e., the research, ownership and general payments can be combined in the following ways to obtain a comprehensive overview allowing trend analysis and more:

a. General Spend

Datasets can be combined using "Physician Profiled ID" or "Teaching Hospital ID" to find out what physicians or hospitals were paid the most over the years.

b. Ownership Spend

Data here can be combined using the *"Physician Profile ID"* as an identifying key. This allows us to obtain useful information like physicians who have made the most investment over the 5-year period, how much value of interest they've obtained from their investments and more.

c. Research Spend

Data sheets here can be combined using *"Teaching Hospital ID"* as an identifying key to obtain information such as total investments made in each teaching hospital to check trends of fall and rise on investment

3.2 Model

Fig 1 shows the model for the data present in the ownership spend (Similar models can be created for research and general spend). The dataset is normalized into fact and dictionary tables to provide a clean and comprehensive look at the data provided

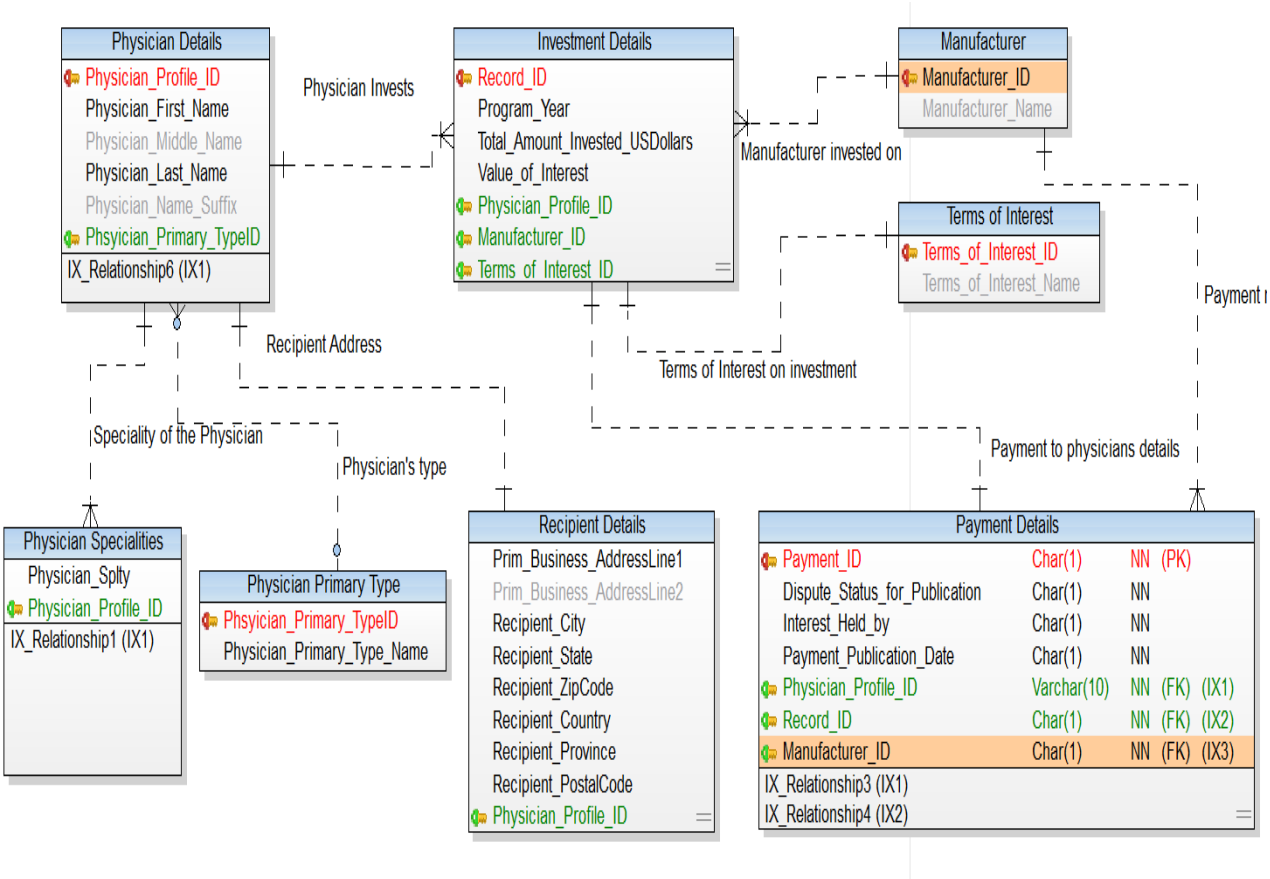


Fig1: Ownership spend data model

3.3 Data Integration

Using SQL Server Integration Services, we perform data integration, moving data from the staging table into separate meaningful tables.

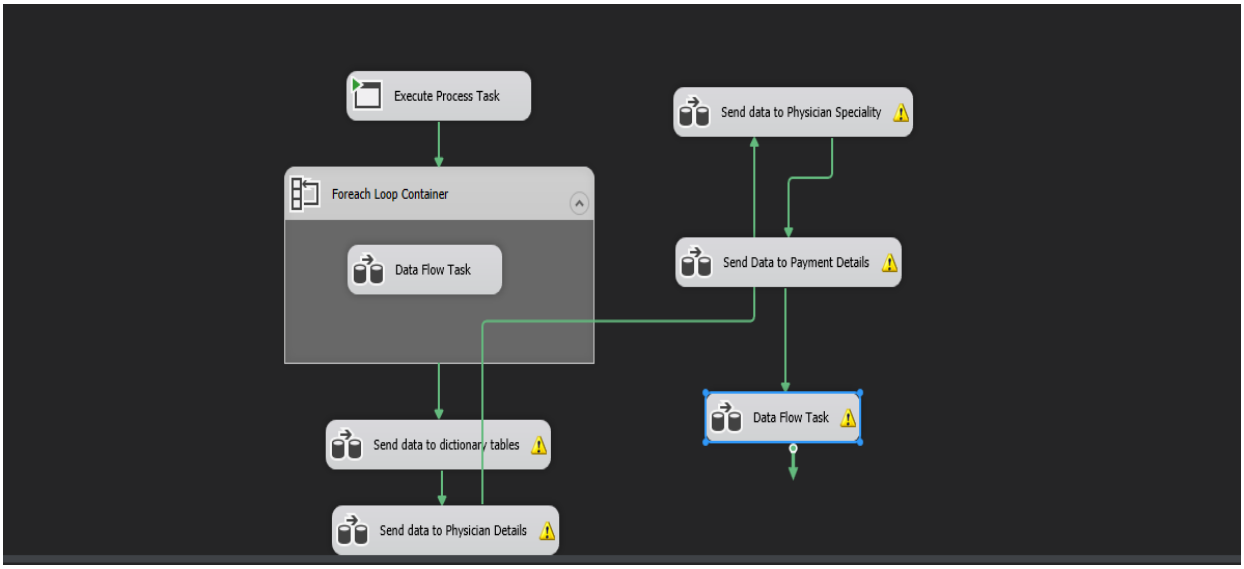


Fig 2: Control Flow

First using Python, we explored cleaned the data. And in order to use the Python script we used Execute Process Task component of SSIS.

```
# This code is to check & remove any irregularities in the data from the file which
causes the SSIS to not able to
# identify the structure of the data properly

import pandas as pd
from pandas import Series, DataFrame
import csv
import shutil
import os

path = "H:\PGYR"
extension = "csv"
fileList = os.listdir(path)

for file in fileList:
    fileName, fileExtension = os.path.splitext(file)
    if fileExtension == ".csv":
        if "OP_DTL_OWNRSHP_PGYR" in fileName:
            with open(path + '\\' + file, "rU") as f:
                reader = csv.reader(f, delimiter=',', dialect="excel")
                rows = list(reader)

                for row in rows:
                    if ";" in row[20]:
                        row[20] = row[20].replace(';', ",")
                    if ";" in row[6]:
                        row[6] = row[6].replace(';', ":")

            with open(path + '\\' + file, mode="w") as outFile:
                writer = csv.writer(outFile, delimiter=";")
                writer.writerows(rows)
```

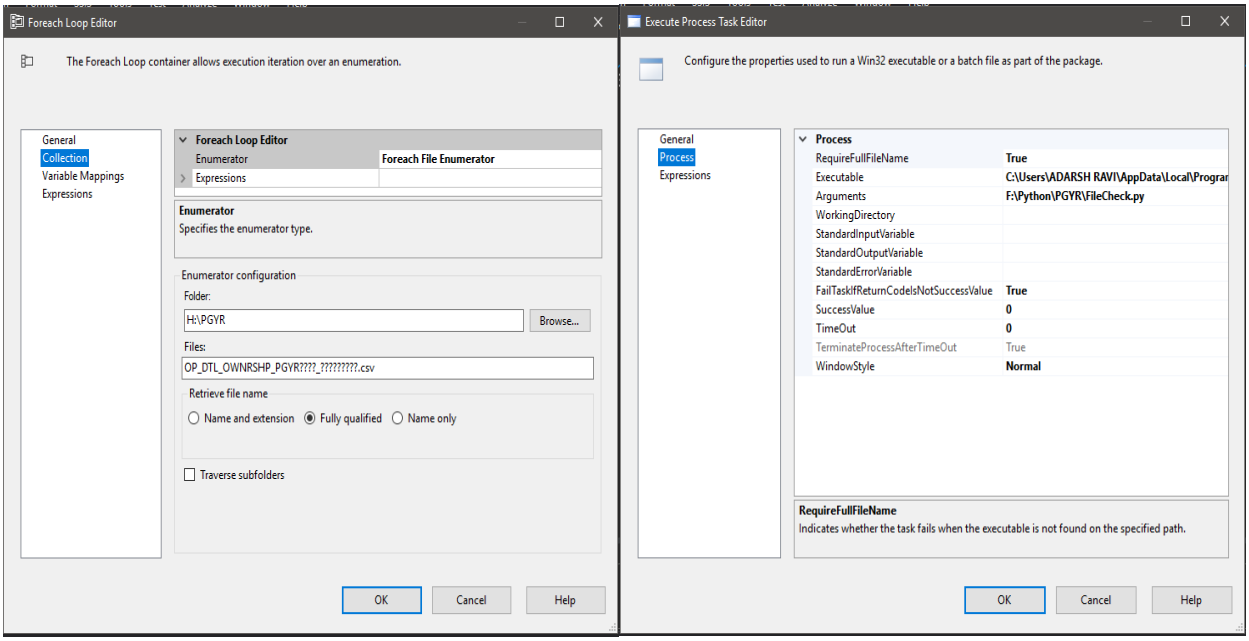


Fig 3: Set up

Next, we populated all the dictionary tables from the staging table by first multicastig the Stagibg table data and then grouping the data and sending the data to the appropriate tables.

Using the below function, we have split the data in the physician speciality column into a table with the various specialities.

```
CREATE FUNCTION [dbo].[ParseValues]
(@String varchar(8000), @Delimiter varchar(10) )
RETURNS @RESULTS TABLE (ID int identity(1,1), Val varchar(8000))
AS
BEGIN
DECLARE @Value varchar(100)
WHILE @String is not null
BEGIN
SELECT @Value=CASE WHEN PATINDEX('%'+@Delimiter+'%',@String) >0 THEN
LEFT(@String,PATINDEX('%'+@Delimiter+'%',@String)-1) ELSE @String END, @String=CASE WHEN
PATINDEX('%'+@Delimiter+'%',@String) >0 THEN
SUBSTRING(@String,PATINDEX('%'+@Delimiter+'%',@String)+LEN(@Delimiter),LEN(@String)) ELSE NULL END
INSERT INTO @RESULTS (Val)
SELECT @Value
END
RETURN
END
```

Table1: Code to create Speciality Tables

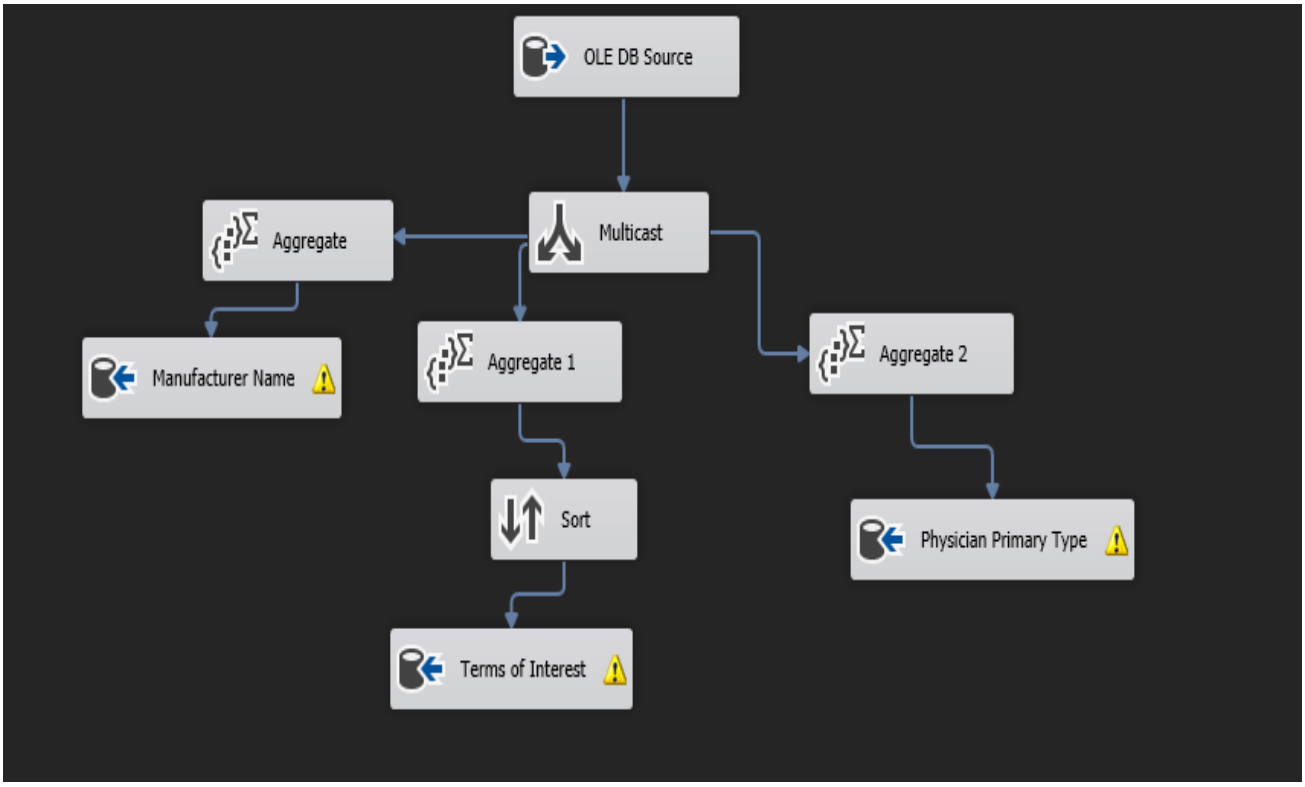
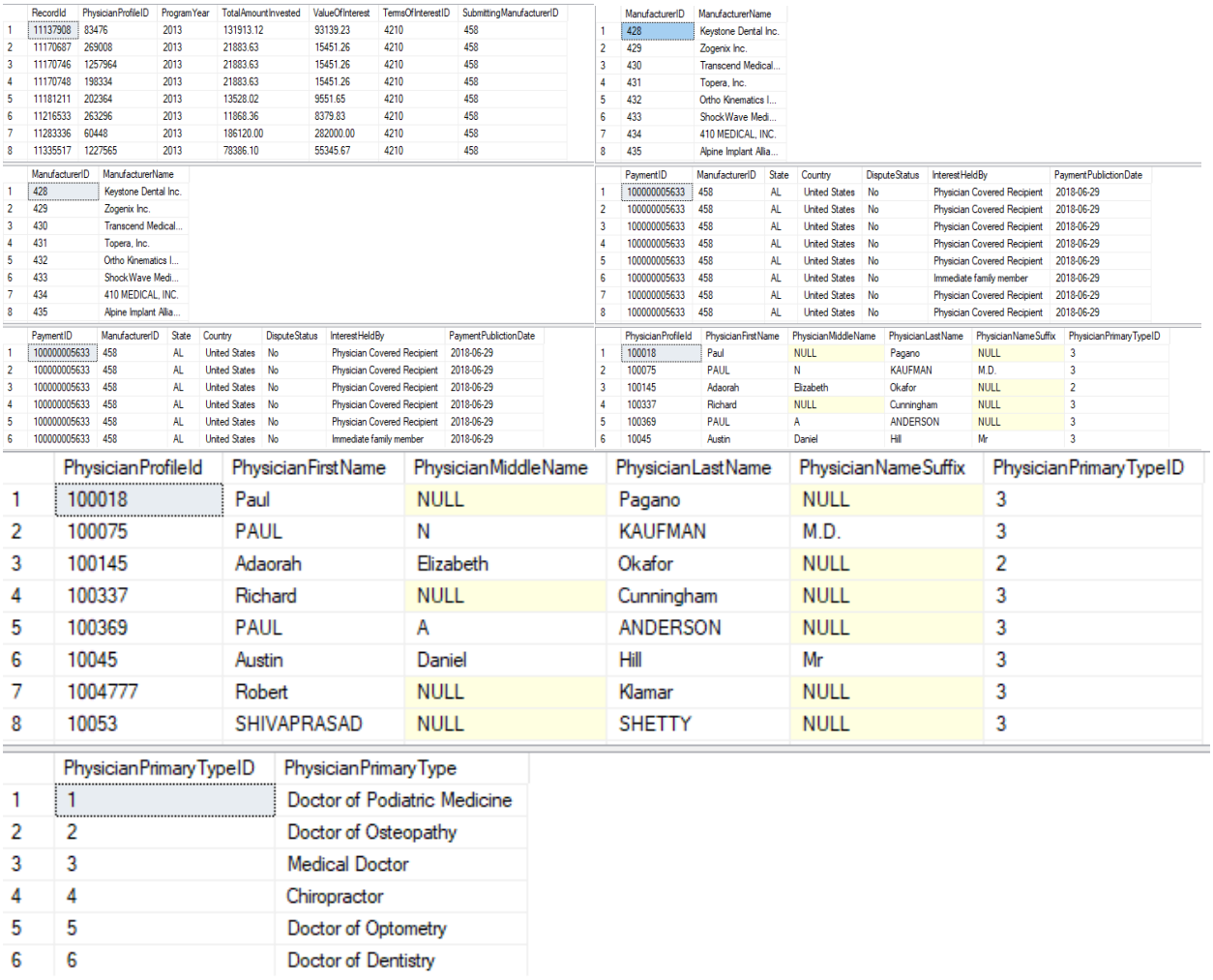


Fig 4: Data Flow

The rest of the data flows we have mapped the columns from the staging tables to their respective tables. In fig 5, is a snapshot of all the tables:



	RecordId	PhysicianProfileID	ProgramYear	TotalAmountInvested	ValueOfInterest	TermsOfInterestID	SubmittingManufacturerID
1	11137908	83476	2013	131913.12	93139.23	4210	458
2	11170687	269008	2013	21883.63	15451.26	4210	458
3	11170746	1257964	2013	21883.63	15451.26	4210	458
4	11170748	198334	2013	21883.63	15451.26	4210	458
5	11181211	202364	2013	13528.02	9551.65	4210	458
6	11216533	263296	2013	11868.36	8379.83	4210	458
7	11283336	60448	2013	186120.00	282000.00	4210	458
8	11335517	1227565	2013	78386.10	55345.67	4210	458

	ManufacturerID	ManufacturerName
1	428	Keystone Dental Inc.
2	429	Zogenix Inc.
3	430	Transcend Medical...
4	431	Topera, Inc.
5	432	Ortho Kinematics I...
6	433	ShockWave Medi...
7	434	410 MEDICAL, INC.
8	435	Alpine Implant Allia...

	ManufacturerID	ManufacturerName
1	428	Keystone Dental Inc.
2	429	Zogenix Inc.
3	430	Transcend Medical...
4	431	Topera, Inc.
5	432	Ortho Kinematics I...
6	433	ShockWave Medi...
7	434	410 MEDICAL, INC.
8	435	Alpine Implant Allia...

	PaymentID	ManufacturerID	State	Country	DisputeStatus	InterestHeldBy	PaymentPublictionDate
1	100000005633	458	AL	United States	No	Physician Covered Recipient	2018-06-29
2	100000005633	458	AL	United States	No	Physician Covered Recipient	2018-06-29
3	100000005633	458	AL	United States	No	Physician Covered Recipient	2018-06-29
4	100000005633	458	AL	United States	No	Physician Covered Recipient	2018-06-29
5	100000005633	458	AL	United States	No	Physician Covered Recipient	2018-06-29
6	100000005633	458	AL	United States	No	Immediate family member	2018-06-29
7	100000005633	458	AL	United States	No	Physician Covered Recipient	2018-06-29
8	100000005633	458	AL	United States	No	Physician Covered Recipient	2018-06-29

	PaymentID	ManufacturerID	State	Country	DisputeStatus	InterestHeldBy	PaymentPublictionDate
1	100000005633	458	AL	United States	No	Physician Covered Recipient	2018-06-29
2	100000005633	458	AL	United States	No	Physician Covered Recipient	2018-06-29
3	100000005633	458	AL	United States	No	Physician Covered Recipient	2018-06-29
4	100000005633	458	AL	United States	No	Physician Covered Recipient	2018-06-29
5	100000005633	458	AL	United States	No	Physician Covered Recipient	2018-06-29
6	100000005633	458	AL	United States	No	Immediate family member	2018-06-29

	PhysicianProfileId	PhysicianFirstName	PhysicianMiddleName	PhysicianLastName	PhysicianNameSuffix	PhysicianPrimaryTypeID
1	100018	Paul	NULL	Pagano	NULL	3
2	100075	PAUL	N	KAUFMAN	M.D.	3
3	100145	Adaorah	Elizabeth	Okafor	NULL	2
4	100337	Richard	NULL	Cunningham	NULL	3
5	100369	PAUL	A	ANDERSON	NULL	3
6	10045	Austin	Daniel	Hill	Mr	3

	PhysicianProfileId	PhysicianFirstName	PhysicianMiddleName	PhysicianLastName	PhysicianNameSuffix	PhysicianPrimaryTypeID
1	100018	Paul	NULL	Pagano	NULL	3
2	100075	PAUL	N	KAUFMAN	M.D.	3
3	100145	Adaorah	Elizabeth	Okafor	NULL	2
4	100337	Richard	NULL	Cunningham	NULL	3
5	100369	PAUL	A	ANDERSON	NULL	3
6	10045	Austin	Daniel	Hill	Mr	3
7	1004777	Robert	NULL	Klamar	NULL	3
8	10053	SHIVAPRASAD	NULL	SHETTY	NULL	3

	PhysicianPrimaryTypeID	PhysicianPrimaryType
1	1	Doctor of Podiatric Medicine
2	2	Doctor of Osteopathy
3	3	Medical Doctor
4	4	Chiropractor
5	5	Doctor of Optometry
6	6	Doctor of Dentistry

Fig 5: Tables

4. Visualizations

4.1

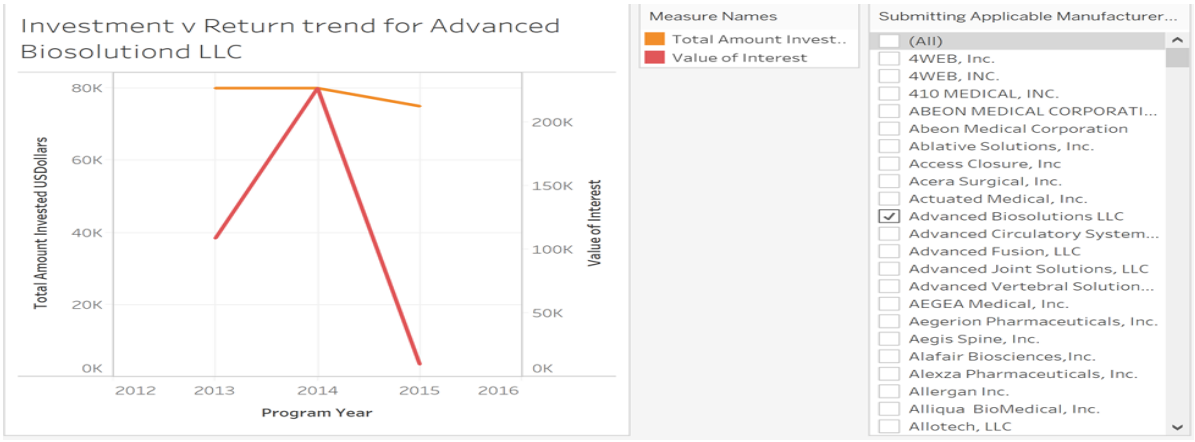


Fig 6: Investment v Return

4.2

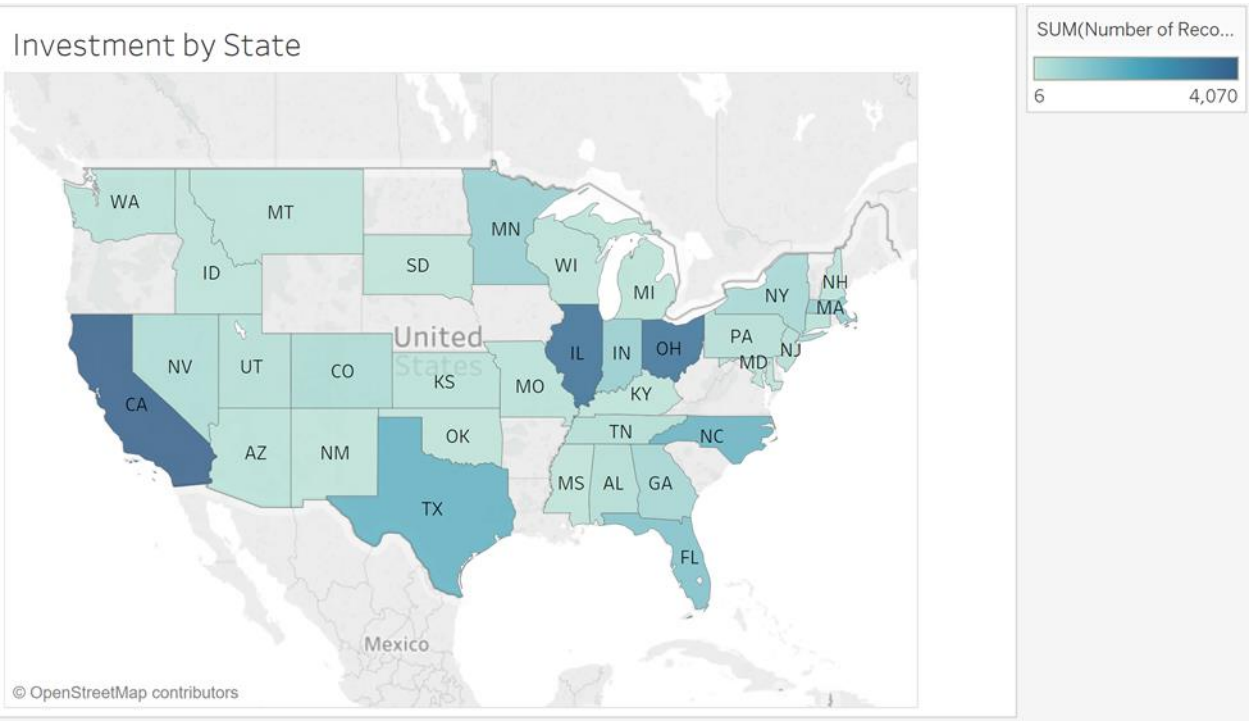


Fig 7: Investment by State

4.3

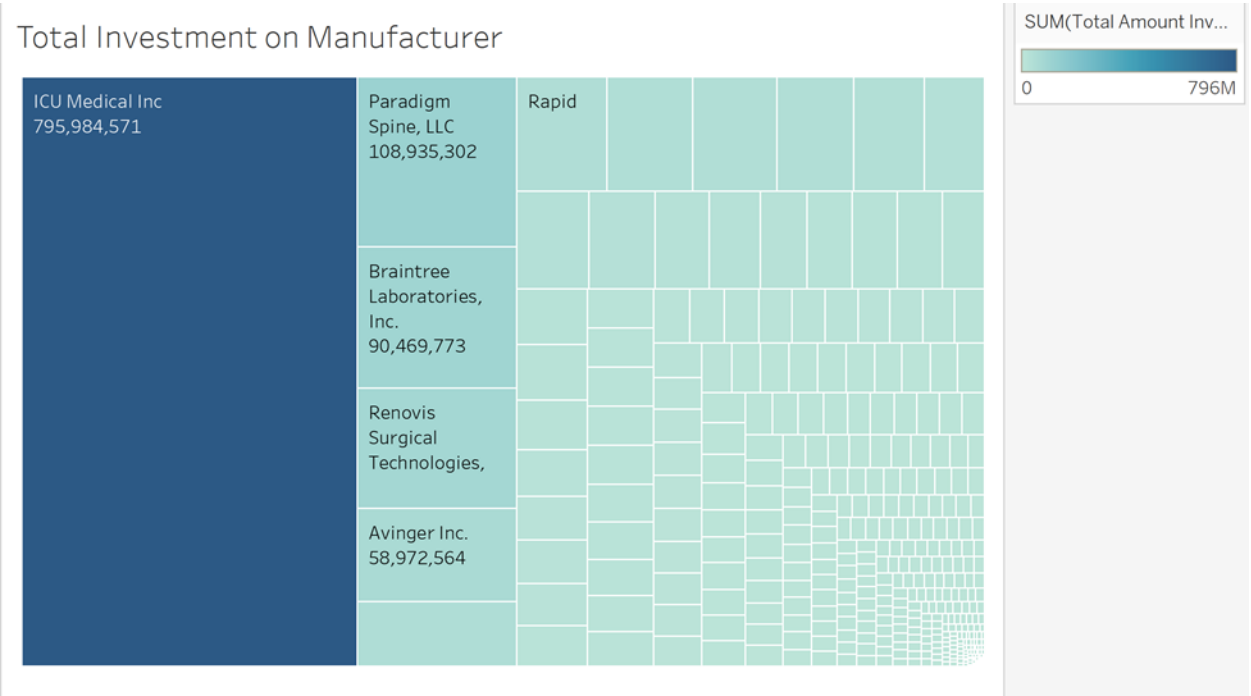


Fig 8: Investment on Manufacturer

4.4

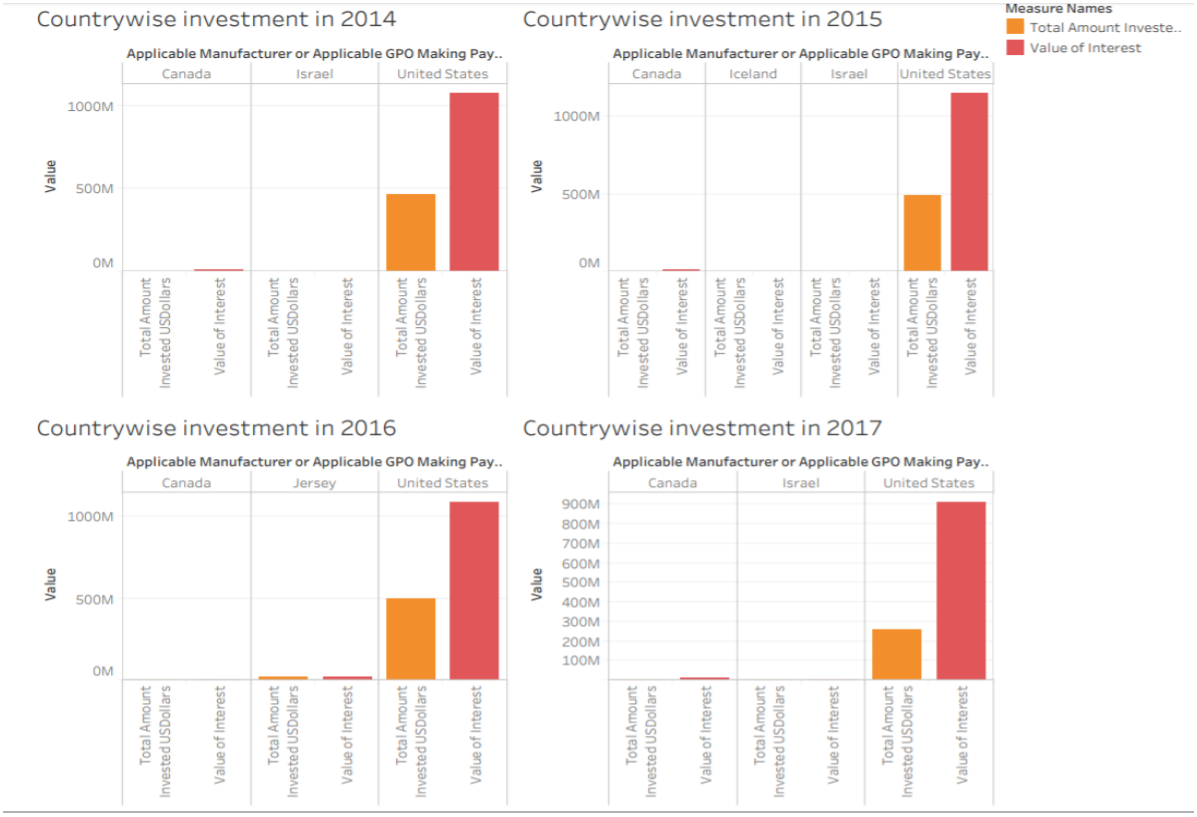


Fig 9: Country wise Investment

4.5

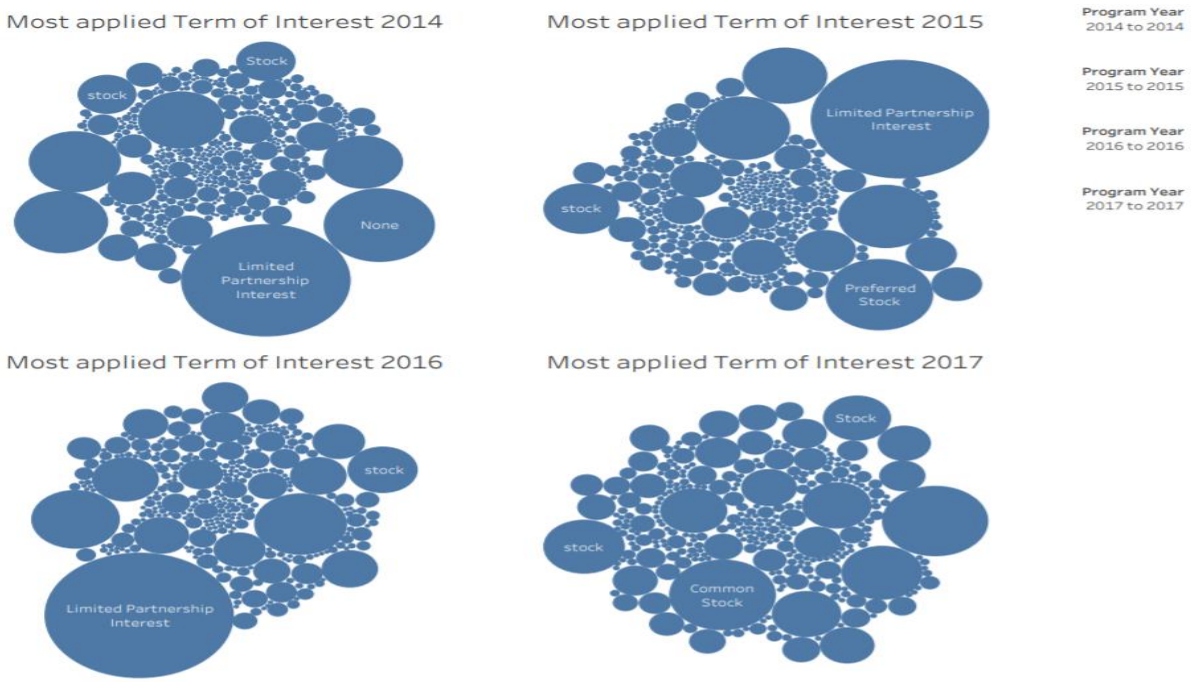


Fig 10: Terms of Interest

4.6

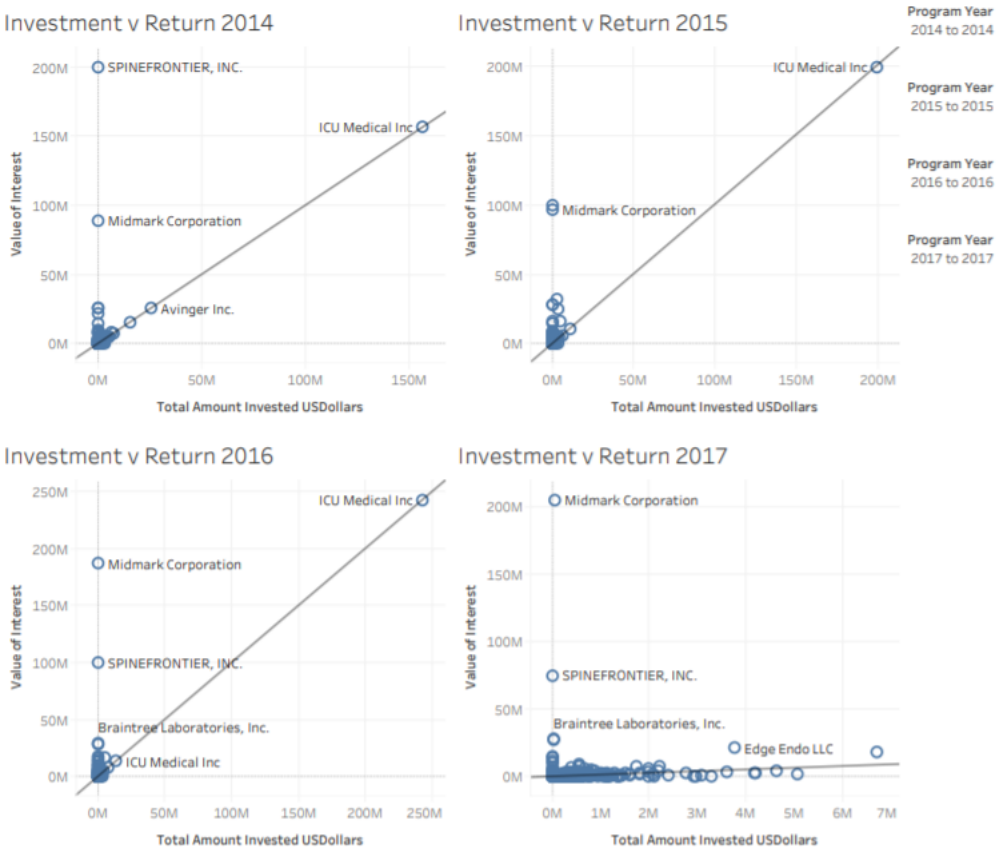


Fig 11: Investment-Value Trend Line

5. Analysis and Reporting

5.1 Analysis to retrieve top 10 earners

The following python code was used to retrieve the top earning physicians in the period 2013-2017.

```
import matplotlib.pyplot as plt
import seaborn as sns
import pyodbc as pdb
import prettytable
import pandas.io.sql as psql
import pandas as pd
from pandas import DataFrame

conn = pdb.connect('Driver={SQL Server};'
                   'Server=ADARSH\ADARSH;'
                   'Database=PGYR;'
                   'Trusted_Connection=yes;')

cursor = conn.cursor()
cursor.execute("select TOP 10 Physician_Profile_ID,
Physician_First_Name, Physician_Last_Name,  "
               "CONVERT(varchar, CAST(SUM(Value_of_Interest) as
Money), 1) Value_of_Interest "
               "from OP_DTL_OWNRSHP_PGYR_Final "
               "group by Physician_Profile_ID, Physician_First_Name,
Physician_Last_Name "
               "order by SUM(Value_of_Interest) desc")
```

```
x = prettytable.PrettyTable(["Physician Profile ID", "Physician First
Name", "Physician Last Name", "Value Of Interest"])
x.align["Physician Profile ID"] = "l"
x.align["Physician First Name"] = "l"
x.align["Physician Last Name"] = "l"
for row in cursor:
   x.add_row(row)

print(x)
qry = "select Physician_Profile_ID, Program_Year,
SUM(CONVERT(numeric(18,2),Value_of_Interest)) as Value_of_Interest " \
     "from OP_DTL_OWNRSHP_PGYR " \
     "where Physician_Profile_ID in (select TOP 10 Physician_Profile_ID " \
     "from OP_DTL_OWNRSHP_PGYR_Final " \
     "group by Physician_Profile_ID, Physician_First_Name,
Physician_Last_Name " \
     "order by SUM(Value_of_Interest) desc) " \
     "group by Physician_Profile_ID, Program_Year " \
     "order by Physician_Profile_ID, Program_Year"

df = DataFrame(psql.read_sql_query(qry, conn))

years = df["Program_Year"].unique()

PhysicianIds = sorted(df["Physician_Profile_ID"].unique())

pd.options.mode.chained_assignment = None

for ID in PhysicianIds:
   df_filter = df[df["Physician_Profile_ID"] == ID]
   for year in years:
      found = False
      for index, row in df_filter.iterrows():
         if row["Program_Year"] == year:
            found = True
            break
         else:
            found = False
      if not found:
         df_filter.loc[index+1] = [ID, year, 0]
   VoI = list(df_filter["Value_of_Interest"])
   sns.lineplot(x=years, y=VoI, label=ID, linestyle='-')

plt.ylabel("Value of Interest (in 100,000,000)")
plt.xlabel("Year")
plt.title("Top 10 Physicians")
plt.legend(title="Physician Profile ID")
plt.show()
```

Table 2: Python code to retrieve top 10 earners

Now, below is an example of a summary table consisting of the top 10 physicians who've earned the most value of interest in the period of 2013-2017.

No.	Profile ID	Physician Name	Cumulative Payment ($)
1	1315076	George Lopez	726,343,199
2	1004777	Robert Klamar	661,261,304
3	420872	Kingsley Chin	625,000,000
4	240414	Veena Bhat Venkatraman	158,911,293
5	75460	Elizabeth Beeuwkes Baker	151,563,667
6	81653	Nicole M Orr	89,607,096
7	355094	John Steinmann	76,508,183
8	34470	Geoffrey Ennis Clark	66,551,101
9	626803	K Mopper	60,031,553
10	887808	Scott Booher	54,166,033

Table 3: Top 10 earning physicians

Trend analysis done using the above data:

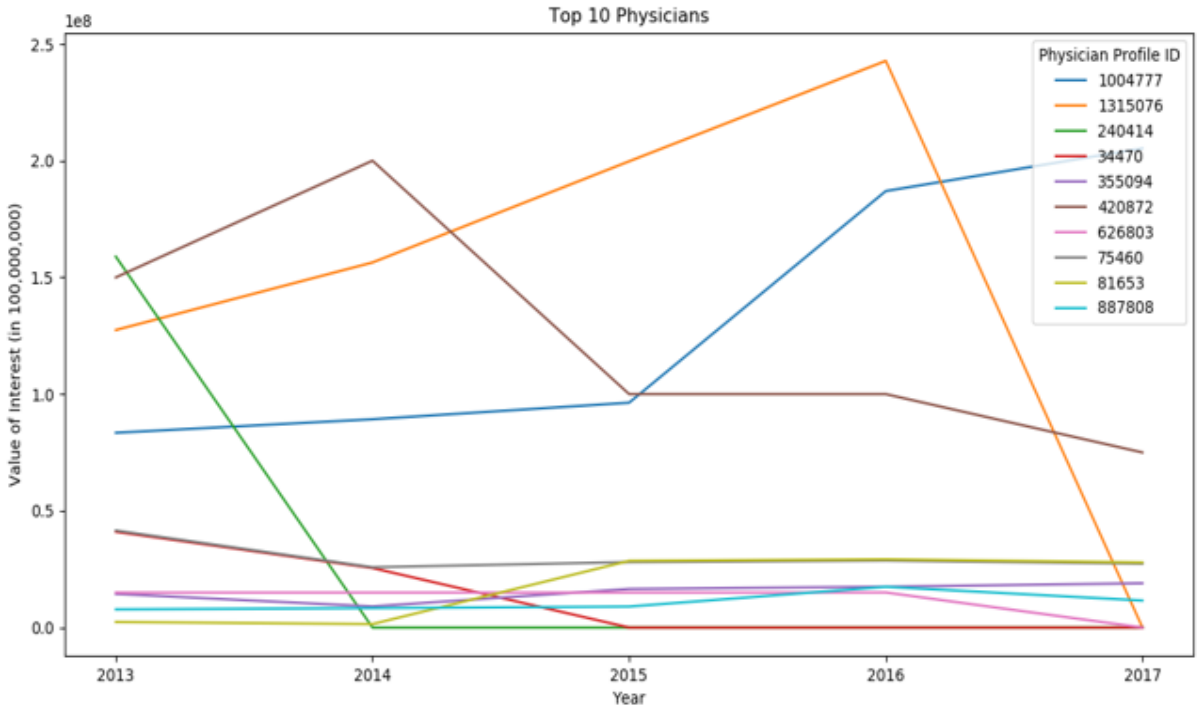


Fig 11: Dashboard of Trend Analysis

5.2 Reporting

Finally, a report is created using SSRS to be obtain investment and return details of the physicians.

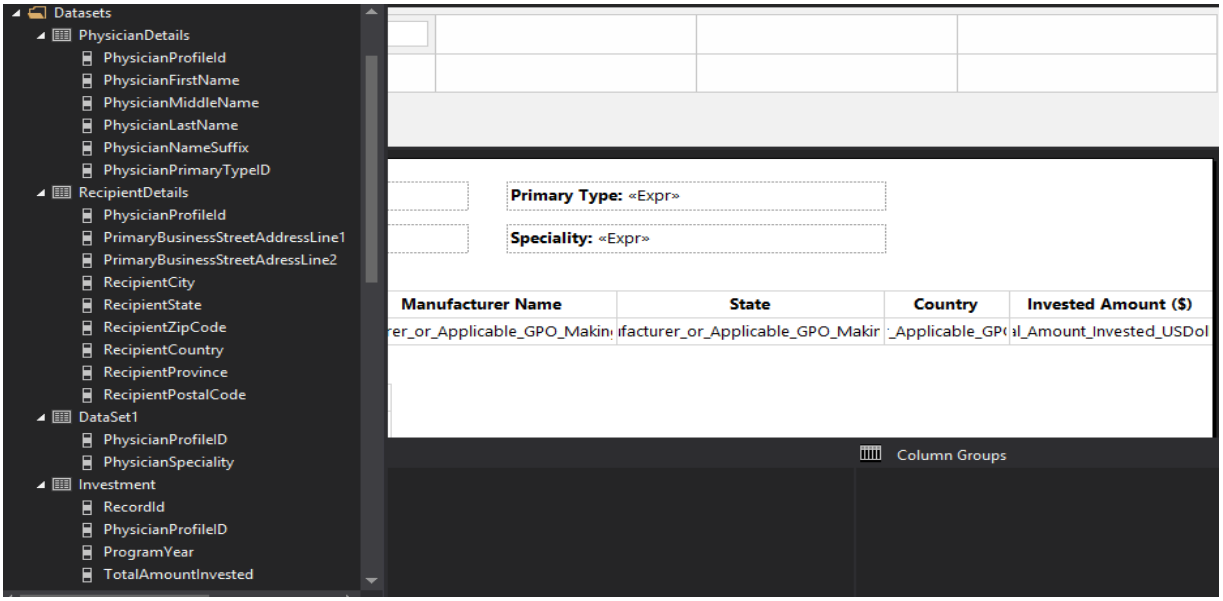


Fig 12: Datasets and design used to create the report

Robert Klamar

Primary Type: Medical Doctor

Address: 471 Marker Rd, Versailles, OH, 45380

Speciality: Allopathic & Osteopathic Physicians| Family Medicine

Investment Details

Program Year	Manufacturer Name	State	Country	Invested Amount ($)
2017	Midmark Corporation	OH	United States	57073.00
2015	Midmark Corporation	OH	United States	57073.00
2014	Midmark Corporation	OH	United States	57073.00
2016	Midmark Corporation	OH	United States	57073.00
2013	Midmark Corporation	OH	United States	57073.00

Return on Interest

Program Year	Value of Interest ($)
2017	205274459.00
2015	96321258.00
2014	89237990.00
2016	186993309.00
2013	83434288.00

Fig 13: Report for 'Robert Klamer'