Data Cleaning for Penda Health Center Dataset

Preparing and Transforming Penda Health Center Data for Analysis

This comprehensive guide outlines the sequential process I followed to perform data cleaning on the dataset. I have also uploaded a SQL File that has detailed steps on how I successfully integrated the three tables using Microsoft SQL Server and exported the combined results to a CSV file in MS Excel format. The SQL file also contains EDA that I did.

In [1]:
```
# Import Library and Dataset
import pandas as pd
import numpy as np

hospitals_df = pd.read_csv('D:/Data Analysis/Patient_Data/PendaHealth_Data.csv')
hospitals_df.head()
hospitals_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 48147 entries, 0 to 48146
Data columns (total 11 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   VisitCode      48147 non-null  object 
 1   PatientCode    48147 non-null  object 
 2   VisitDateTime  48147 non-null  object 
 3   Date           48147 non-null  object 
 4   Time           48147 non-null  object 
 5   MedicalCenter  48147 non-null  object 
 6   VisitCategory  48147 non-null  object 
 7   Payor          48147 non-null  object 
 8   NPS_Score      2022 non-null   float64
 9   Amount         48147 non-null  int64  
 10  Diagnosis      22119 non-null  object 
dtypes: float64(1), int64(1), object(9)
memory usage: 4.0+ MB
```

Column Name and Definition

1. PatientCode: Which is a unique ID assigned to each Patient.
2. VisitDateTime: Which is the date on which the visit occurred.
3. MedicalCenter: The medical centre at which the visit occurred.
4. Payor: The name of the payor for the visit, it can either be cash i.e., the person paid for themselves or it can be Insurance i.e., the bill was paid by an insurance company.
5. NPS_Score: Score provided by patient that rates their satisfaction level with the visit. Score range is 0-10.
6. Amount: The total amount invoiced for the visit.
7. Diagnosis: The nature of illness recorded for each visit.

Despite there being nun values in NPS_Score and Diagnosis, I won't bother to remove them because I won't be using NPS_Score in my Analysis. For the Null Diagnosis, I will Assume that there was non made that's why there are no entries.

First thing I will do is to check if there are duplicates. I will Use the 'PatientCode' Column becaus it is a Unique ID assigned to the patient.

In [42]:
```
# Check If There Are Duplicates Found in 'PatientCode' Column In The DataFrame named "hospitals_df"
# 'PatientCode' is  a Unique ID Assigned To Each Patient

duplicates = hospitals_df.duplicated(subset="PatientCode")
if duplicates.any():
    print("Duplicates found in the 'PatientCode' column.")
else:
    print("No duplicates found in the 'PatientCode' column.")
```

```
Duplicates found in the 'PatientCode' column.
```

- I will identify duplicate rows in the 'PatientCode' column in the DataFrame named & put it in a separate DataFrame.

In [ ]:
```
# Identify Duplicate Rows In The 'PatientCode' Column In The DataFrame Named "hospitals_df" & Put In A Separate DataFrame.
duplicate_rows2 = hospitals_df[hospitals_df.duplicated(subset="PatientCode", keep=False)]
duplicate_df2 = pd.DataFrame(duplicate_rows2)
duplicate_df2.head(20)
duplicate_df2.info()

# Save It As An Excel File
duplicate_df2.to_excel('D:/Data Analysis/Patient_Data/AccordingToPatientCode.xlsx')
```

Results

After further exploration in Excel, I have discovered the following;

- There are Patients who visited the medical center more than once.
- There were duplicate entries in the 'VisitCode' Column. This means that there are patients who visited the hospital twice in the same Date and Time! Indicating that that there might have been errors in data entry. or This Patients were diagnosed with more than one disease on the same Date and Time.

Remedy

I decided to divide the dataset into teo dataframes namely;

1. DuplicateDiagnoses_df This dataframe contains information about patients who were diagnosed with multiple diseases or the same type of disease on the same 'VisitDateTime'.
2. UniqueDiagnoses_df This dataframe contains information about patients who were NOT diagnosed with one or more diseases on same 'VisitDateTime'.

In [50]:
```
# Extract information from the "duplicate_df2" about patients who were diagnosed with multiple diseases or the same type of disease on the same day and t
duplicate_diagnoses = duplicate_df2[duplicate_df2.duplicated(subset=["PatientCode", "VisitDateTime", "Diagnosis"], keep=False)]
DuplicateDiagnoses_df = pd.DataFrame(duplicate_diagnoses)
DuplicateDiagnoses_df.head(10)
DuplicateDiagnoses_df.info()

# Save It As An Excel File
DuplicateDiagnoses_df.to_excel('D:/Data Analysis/Patient_Data/DuplicateDiagnoses_on_SamePatient.xlsx')
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1636 entries, 24 to 48123
Data columns (total 11 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   VisitCode      1636 non-null   object 
 1   PatientCode    1636 non-null   object 
 2   VisitDateTime  1636 non-null   object 
 3   Date           1636 non-null   object 
 4   Time           1636 non-null   object 
 5   MedicalCenter  1636 non-null   object 
 6   VisitCategory  1636 non-null   object 
 7   Payor          1636 non-null   object 
 8   NPS_Score      40 non-null     float64
 9   Amount         1636 non-null   int64  
 10  Diagnosis      561 non-null    object 
dtypes: float64(1), int64(1), object(9)
memory usage: 153.4+ KB
```

In [53]:
```
# Extract information from the "duplicate_df2" about patients who were not diagnosed with multiple diseases or the same type of disease on the same day a
unique_diagnoses = duplicate_df2.drop_duplicates(subset=["PatientCode", "VisitDateTime", "Diagnosis"], keep=False)
UniqueDiagnoses_df = pd.DataFrame(unique_diagnoses)
UniqueDiagnoses_df.head(10)
UniqueDiagnoses_df.info()

# Save It As An Excel File
UniqueDiagnoses_df.to_excel('D:/Data Analysis/Patient_Data/UniqueDiagnoses_on_SamePatient.xlsx')
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 27404 entries, 1 to 48143
Data columns (total 11 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   VisitCode      27404 non-null  object 
 1   PatientCode    27404 non-null  object 
 2   VisitDateTime  27404 non-null  object 
 3   Date           27404 non-null  object 
 4   Time           27404 non-null  object 
 5   MedicalCenter  27404 non-null  object 
 6   VisitCategory  27404 non-null  object 
 7   Payor          27404 non-null  object 
 8   NPS_Score      1008 non-null   float64
 9   Amount         27404 non-null  int64  
 10  Diagnosis      14584 non-null  object 
dtypes: float64(1), int64(1), object(9)
memory usage: 2.5+ MB
```

How I Fixed DataFrame 1 (DuplicateDiagnoses_df)

I will merge the duplicates based on the "VisitCode" column while summing the values in the "Amount" column only when they are not similar. I have written two chunks of code where;

- First one shows which rows in the 'Amount' column had different Values and which one has similar values.
- Second one produces the amount figures.

In [37]:
```
# To join the duplicate entries in the "DuplicateDiagnoses_on_SamePatient" DataFrame
# based on the "VisitCode" column while summing the values in the "Amount" column only when they are not similar,
# The following code will produce a dataframe that will show which row that was merged had two unique Diagnoses or similar Diagnosis

# Import the dataframe
duplicated_df = pd.read_excel('D:/Data Analysis/Patient_Data/DuplicateDiagnoses_on_SamePatient.xlsx')

# Joining duplicates based on VisitCode and summing Amount column based on condition
agg_funcs = {
    'PatientCode': 'first',
    'VisitDateTime': 'first',
    'Date': 'first',
    'Time': 'first',
    'MedicalCenter': 'first',
    'VisitCategory': 'first',
    'Payor': 'first',
    'NPS_Score': 'first',
    'Amount': lambda x: x.nunique(),
    'Diagnosis': 'first'
}

# Group by VisitCode and apply the aggregation functions
merged_df = duplicated_df.groupby('VisitCode').agg(agg_funcs).reset_index()

# Displaying the joined DataFrame
merged_df.head(50)
```

Out[37]:

	VisitCode	PatientCode	VisitDateTime	Date	Time	MedicalCenter	VisitCategory	Payor	NPS_Score	Amount	Diagnosis
0	XA-1060353	651e3c41-2437	01/03/2022 09:15	01/03/2022	09:15:39	Githurai 45	In-person Visit	Insurance Company A	NaN	1	NaN
1	XA-1060401	06a8c2a1-ea4b	01/03/2022 10:12	01/03/2022	10:12:41	Mathare North	In-person Visit	Cash	NaN	1	NaN
2	XA-1060415	06a8c2a1-ea4b	01/03/2022 10:18	01/03/2022	10:18:03	Mathare North	In-person Visit	Insurance Company A	NaN	2	acute nasopharyngitis
3	XA-1060563	e6fc5e5e-c79b	01/03/2022 11:46	01/03/2022	11:46:11	Tassia	In-person Visit	Cash	NaN	1	NaN
4	XA-1060599	291d22fd-efc0	01/03/2022 12:02	01/03/2022	12:02:38	Mathare North	In-person Visit	Cash	NaN	1	NaN
5	XA-1060607	85f3d491-5e7e	01/03/2022 12:07	01/03/2022	12:07:32	Tassia	In-person Visit	Cash	NaN	1	NaN
6	XA-1060680	8464d445-dabd	01/03/2022 12:44	01/03/2022	12:44:41	Mathare North	In-person Visit	Cash	NaN	1	NaN
7	XA-1060764	b86918d6-3b16	01/03/2022 13:48	01/03/2022	13:48:54	Tassia	In-person Visit	Cash	NaN	1	NaN
8	XA-1061118	37d1da03-29fc	01/03/2022 18:42	01/03/2022	18:42:09	Kimathi Street	In-person Visit	Cash	NaN	2	NaN
9	XA-1061228	737d0f26-ab71	01/03/2022 20:37	01/03/2022	20:37:08	Embakasi	In-person Visit	Insurance Company A	NaN	1	tonsillitis, acute bacterial
10	XA-1061264	97398728-3918	01/03/2022 22:11	01/03/2022	22:11:31	Tassia	In-person Visit	Cash	NaN	1	tonsillitis, acute bacterial
11	XA-1061306	f164698f-fe58	02/03/2022 07:27	02/03/2022	07:27:54	Tassia	In-person Visit	Cash	NaN	1	NaN
12	XA-1061667	6b8e1e93-75bc	02/03/2022 13:01	02/03/2022	13:01:42	Kimathi Street	In-person Visit	Cash	NaN	1	NaN
13	XA-1061681	844ee640-49dd	02/03/2022 13:11	02/03/2022	13:11:54	Kimathi Street	In-person Visit	Cash	NaN	1	diabetes mellitus type 2
14	XA-1061717	41e57083-ba5e	02/03/2022 13:51	02/03/2022	13:51:32	Mathare North	In-person Visit	Cash	NaN	1	NaN
15	XA-1061808	d5b221f2-33de	02/03/2022 15:29	02/03/2022	15:29:39	Githurai 45	In-person Visit	Cash	NaN	1	NaN
16	XA-1061926	09fbe587-8c77	02/03/2022 17:30	02/03/2022	17:30:43	Kimathi Street	In-person Visit	Cash	NaN	1	NaN
17	XA-1062063	10dfb72d-aa21	02/03/2022 19:39	02/03/2022	19:39:20	Tassia	In-person Visit	Cash	NaN	1	review
18	XA-1062295	1de9190c-83f1	03/03/2022 09:43	03/03/2022	09:43:21	Tassia	In-person Visit	Cash	NaN	2	NaN
19	XA-1062427	ed89f68e-e715	03/03/2022 11:23	03/03/2022	11:23:07	Call Centre	Telemedicine Visit	Cash	NaN	1	NaN
20	XA-1062531	7738eeb4-0279	03/03/2022 12:28	03/03/2022	12:28:40	Mathare North	In-person Visit	Cash	NaN	1	review
21	XA-1062533	1cb9412e-4a9f	03/03/2022 12:29	03/03/2022	12:29:09	Mathare North	In-person Visit	Cash	NaN	1	review
22	XA-1063160	6fbccf15-b76c	04/03/2022 09:40	04/03/2022	09:40:21	Githurai 45	In-person Visit	Cash	NaN	1	NaN
23	XA-1063672	b27dd15e-f584	04/03/2022 17:37	04/03/2022	17:37:39	Githurai 45	In-person Visit	Cash	NaN	1	review
24	XA-1063777	52623b5f-b9ee	04/03/2022 19:27	04/03/2022	19:27:12	Tassia	In-person Visit	Cash	NaN	1	NaN
25	XA-1064205	3eaf21bb-7223	05/03/2022 11:30	05/03/2022	11:30:57	Mathare North	In-person Visit	Cash	NaN	1	NaN
26	XA-1064253	4da9d8d2-3fc7	05/03/2022 11:53	05/03/2022	11:53:59	Kimathi Street	In-person Visit	Cash	NaN	1	NaN
27	XA-1064625	c0ee4271-0aa8	05/03/2022 17:15	05/03/2022	17:15:23	Embakasi	In-person Visit	Cash	NaN	2	normal pregnancy
28	XA-1064776	15aa154e-bbc3	05/03/2022 20:23	05/03/2022	20:23:09	Pipeline	In-person Visit	Insurance Company B	NaN	1	acute nasopharyngitis
29	XA-1064805	bc02f38c-c04b	05/03/2022 21:23	05/03/2022	21:23:09	Tassia	In-person Visit	Insurance Company B	NaN	2	NaN
30	XA-1065260	9c65d5a5-5040	06/03/2022 17:31	06/03/2022	17:31:23	Pipeline	In-person Visit	Cash	NaN	1	NaN
31	XA-1065346	419bc92e-f64e	06/03/2022 18:49	06/03/2022	18:49:23	Lucky Summer	In-person Visit	Cash	NaN	1	tonsillitis, acute bacterial
32	XA-1065612	7d475ba1-3726	07/03/2022 09:53	07/03/2022	09:53:26	Githurai 45	In-person Visit	Insurance Company A	NaN	1	review
33	XA-1065722	9a5b3974-1956	07/03/2022 11:08	07/03/2022	11:08:33	Lucky Summer	In-person Visit	Cash	NaN	1	NaN
34	XA-1065744	56b42576-ea0e	07/03/2022 11:16	07/03/2022	11:16:45	Githurai 45	In-person Visit	Insurance Company B	NaN	1	review
35	XA-1065776	c4fa55ed-8224	07/03/2022 11:28	07/03/2022	11:28:31	Lucky Summer	In-person Visit	Insurance Company B	NaN	1	first degree burn
36	XA-1065857	b900e786-158a	07/03/2022 12:07	07/03/2022	12:07:37	Githurai 45	In-person Visit	Cash	NaN	1	tonsillitis, acute bacterial
37	XA-1065886	4e7b17b6-4532	07/03/2022 12:25	07/03/2022	12:25:52	Kimathi Street	In-person Visit	Cash	NaN	1	NaN
38	XA-1066021	38ce8cf6-7caa	07/03/2022 13:51	07/03/2022	13:51:15	Mathare North	In-person Visit	Cash	NaN	1	NaN
39	XA-1066157	ba13e1bb-0ee1	07/03/2022 16:21	07/03/2022	16:21:30	Lucky Summer	In-person Visit	Insurance Company B	NaN	1	dermatitis
40	XA-1066164	61c0607f-101d	07/03/2022 16:25	07/03/2022	16:25:00	Tassia	In-person Visit	Insurance Company B	NaN	1	NaN
41	XA-1066485	88cfa2e1-933f	07/03/2022 22:15	07/03/2022	22:15:02	Call Centre	Telemedicine Visit	Cash	NaN	1	NaN
42	XA-1066564	479e207f-4ec0	08/03/2022 09:12	08/03/2022	09:12:12	Githurai 45	In-person Visit	Cash	NaN	1	NaN
43	XA-1066772	b554ec18-f786	08/03/2022 11:46	08/03/2022	11:46:46	Mathare North	In-person Visit	Cash	NaN	1	NaN
44	XA-1066862	b900e786-158a	08/03/2022 12:56	08/03/2022	12:56:43	Githurai 45	In-person Visit	Cash	NaN	1	NaN
45	XA-1067051	7d475ba1-3726	08/03/2022 16:17	08/03/2022	16:17:50	Githurai 45	In-person Visit	Insurance Company A	NaN	1	NaN
46	XA-1067422	479e207f-4ec0	09/03/2022 09:24	09/03/2022	09:24:04	Githurai 45	In-person Visit	Cash	NaN	1	review
47	XA-1067452	35728e2a-ca9a	09/03/2022 09:52	09/03/2022	09:52:11	Mathare North	In-person Visit	Cash	NaN	1	NaN
48	XA-1067833	b900e786-158a	09/03/2022 13:39	09/03/2022	13:39:48	Githurai 45	In-person Visit	Cash	NaN	1	tonsillitis, acute bacterial
49	XA-1067866	f34f0cf6-8fed	09/03/2022 14:16	09/03/2022	14:16:30	Tassia	In-person Visit	Cash	NaN	1	NaN

In [61]:
```
# To join the duplicate entries in the "DuplicateDiagnoses_on_SamePatient" DataFrame
# based on the "VisitCode" column while summing the values in the "Amount" column only when they are not similar,
# This code will produce a dataframe that will show the amount after the duplicated entries have been merged.

# Import the dataframe
duplicated_df = pd.read_excel('D:/Data Analysis/Patient_Data/DuplicateDiagnoses_on_SamePatient.xlsx')

# Group by VisitCode and sum the Amount column while keeping other columns
grouped_df = duplicated_df.groupby('VisitCode').agg(
    {'PatientCode': 'first', 'VisitDateTime': 'first', 'Date': 'first', 'Time': 'first', 'MedicalCenter': 'first', 'VisitCategory': 'first',
     'Payor': 'first', 'NPS_Score': 'first', 'Amount': 'sum', 'Diagnosis': 'first'}
)

# Create a new DataFrame to store the merged entries
merged_df = pd.DataFrame(columns=grouped_df.columns)

# Iterate over each group in the grouped DataFrame
for visit_code, group in grouped_df.groupby('VisitCode'):
    # Check if there are duplicate entries
    if len(group) > 1:
        # Concatenate the unique values of the non-Amount columns and sum the Amount column
        merged_entry = group.iloc[0].copy()
        merged_entry['Amount'] = group['Amount'].sum()
        merged_df = merged_df.append(merged_entry)
    else:
        merged_df = merged_df.append(group)

# Reset the index of the merged DataFrame
merged_df.reset_index(drop=True, inplace=True)

# Displaying the merged DataFrame
merged_df.head(50)

# Save It As An Excel File
merged_df.to_excel('D:/Data Analysis/Patient_Data/MergedDuplicateDiagnoses_on_SamePatient.xlsx')
```

Doing All The Above Using a Singular Code Block and Save Result In Excel

After employing the aforementioned code segments in a sequential manner to gain a comprehensive understanding of the data, I shall now proceed to consolidate the steps into a singular code block. The objective is to eliminate the non-unique duplicate entries within the dataset and preserve the resulting data in a distinct Excel spreadsheet file termed "Clean_Data." Subsequently, the contents of the "merged_df" DataFrame will be replicated and pasted into the aforementioned file

In [60]:
```
# Import the dataframe
hospitals_df = pd.read_csv('D:/Data Analysis/Patient_Data/PendaHealth_Data.csv')

# Identify the non-unique duplicate rows
duplicate_diagnoses = hospitals_df[hospitals_df.duplicated(subset=["PatientCode", "VisitDateTime", "Diagnosis"], keep=False)]
DuplicateDiagnoses_df = pd.DataFrame(duplicate_diagnoses)

# Remove non-unique duplicate rows from the original data
unique_data = hospitals_df.drop_duplicates(subset=["PatientCode", "VisitDateTime", "Diagnosis"], keep=False)
UniqueData_df = pd.DataFrame(unique_data)

# Save It As An Excel File
UniqueData_df.to_excel('D:/Data Analysis/Patient_Data/Clean_Data.xlsx')
```