ETL-Project-Hospital_Info

November 21, 2021

1 Extracting Data Sources

General Hospital Information: https://www.kaggle.com/cms/hospital-general-information

HVPB Efficiency Scores: https://catalog.data.gov/dataset/hospital-value-based-purchasing-hvbp-efficiency-scores-7b9f0

2010 Census: https://www.kaggle.com/census/us-population-by-zip-code

[1]:
```
import pandas as pd
from sqlalchemy import create_engine, Column, Integer, String
from sqlalchemy.ext.declarative import declarative_base
Base = declarative_base()
import psycopg2
```

[2]:
```
hosp = ".\HospInfo.csv"
hosp_df = pd.read_csv(hosp, encoding = "ISO-8859-1")
hosp_df.dtypes
```

[2]:
```
Facility ID                                    object
Facility Name                                  object
Address                                        object
City                                           object
State                                          object
ZIP Code                                        int64
County Name                                    object
Phone Number                                   object
Hospital Type                                  object
Hospital Ownership                             object
Emergency Services                               bool
Meets criteria for meaningful use of EHRs      object
Hospital overall rating                        object
Hospital overall rating footnote              float64
Mortality national comparison                  object
Mortality national comparison footnote        float64
Safety of care national comparison             object
Safety of care national comparison footnote    float64
Readmission national comparison                object
Readmission national comparison footnote      float64
```

```
Patient experience national comparison                          object
Patient experience national comparison footnote                float64
Effectiveness of care national comparison                       object
Effectiveness of care national comparison footnote             float64
Timeliness of care national comparison                          object
Timeliness of care national comparison footnote                float64
Efficient use of medical imaging national comparison            object
Efficient use of medical imaging national comparison footnote  float64
Location                                                        object
dtype: object
```

[3]:
```
hvbp = ".\Hospital_Value-Based_Purchasing__HVBP____Efficiency_Scores.csv"
hvbp_df = pd.read_csv(hvbp, encoding = "ISO-8859-1")
hvbp_df.dtypes
```

[3]:
```
Facility ID                      int64
Facility Name                   object
Address                         object
City                            object
State                           object
ZIP Code                         int64
County Name                     object
MSPB-1 Achievement Threshold   float64
MSPB-1 Benchmark               float64
MSPB-1 Baseline Rate            object
MSPB-1 Performance Rate        float64
MSPB-1 Achievement Points       object
MSPB-1 Improvement Points       object
MSPB-1 Measure Score            object
Location                        object
dtype: object
```

[4]:
```
census = ".\population_by_zip_2010.csv"
census_df = pd.read_csv(census, encoding = "ISO-8859-1")
census_df.dtypes
```

[4]:
```
population       int64
minimum_age    float64
maximum_age    float64
gender          object
zipcode          int64
geo_id          object
dtype: object
```

2 Transforming

2.1 Cleaning

Hospital Info Data

[5]:
```
# Drop unwanted columns from General Hospital Info
hosp_df_clean = hosp_df.iloc[:,0:10]
# hosp_df_clean.head()

# Remove non-US states from dataset
hospUS_df = hosp_df_clean[(hosp_df_clean['State'] != 'PR') &
                      (hosp_df_clean['State'] != 'MP') &
                      (hosp_df_clean['State'] != 'GU') &
                      (hosp_df_clean['State'] != 'VI') &
                      (hosp_df_clean['State'] != 'VT') &
                      (hosp_df_clean['State'] != 'DC')]
hospUS_df['State'].nunique()

# Rename ZIP Code to zipcode to match census data
hospUS_df = hospUS_df.rename(columns={'ZIP Code': 'zipcode'})

# Rename other columns to remove spaces
hospUS_df = hospUS_df.rename(columns={'Facility ID': 'facilityid', 'Facility␣
↪Name': 'Facility_Name', 'County Name': 'County', 'Phone Number': 'Phone',␣
↪'Hospital Type': 'Hospital_Type', 'Hospital Ownership':␣
↪'Hospital_Ownership'})

# Search for null fields in all columns
null = hospUS_df.isnull()
null.nunique()

hospUS_df.head()
```

[5]:

	facilityid	Facility_Name	Address	City	State	zipcode	County	Phone	Hospital_Type	Hospital_Ownership
0	31309	SAGE MEMORIAL HOSPITAL	STATE ROUTE 264 SOUTH 191	GANADO	AZ	86505	APACHE	(928) 755-4541	Critical Access Hospitals	Voluntary non-profit - Private
1	44021	WOODRIDGE BEHAVIORAL CENTER	600 NORTH 7TH STREET	WEST MEMPHIS	AR	72301	CRITTENDEN	(870) 394-4113	Psychiatric	Proprietary
2	100277	DOUGLAS GARDENS HOSPITAL	5200 NE 2ND AVE	MIAMI	FL	33137	MIAMI-DADE	(305) 751-8626		
3	104078	SUNCOAST BEHAVIORAL HEALTH CENTER	4480 51ST ST W	BRADENTON	FL	34210	MANATEE	(941) 792-2222		
4	130063	TREASURE VALLEY HOSPITAL	8800 WEST EMERALD STREET	BOISE	ID	83704	ADA	(208) 373-5000		

```
2       Acute Care Hospitals  Voluntary non-profit - Private
3                Psychiatric                     Proprietary
4       Acute Care Hospitals                     Proprietary
```

Hospital Efficiency Data

[6]:
```
# Rename ZIP Code to zipcode to match census data
hvbp_df = hvbp_df.rename(columns={'ZIP Code': 'zipcode'})

# Change '1 out of 10' columns to point integer only
points1 = hvbp_df['MSPB-1 Achievement Points']
for point in points1:
    if point != "10 out of 10":
        hvbp_df['Achievement Points'] = points1.astype(str).str[0]
    else:
        hvbp_df['Achievement Points'] = 10

points2 = hvbp_df['MSPB-1 Improvement Points']
for point in points2:
    if point != "10 out of 10":
        hvbp_df['Improvement Points'] = points2.astype(str).str[0]
    else:
        hvbp_df['Improvement Points'] = 10

scores = hvbp_df['MSPB-1 Measure Score']
for score in scores:
    if score != "10 out of 10":
        hvbp_df['Measure Score'] = scores.astype(str).str[0]
    else:
        hvbp_df['Measure Score'] = 10

hvbp_df['Facility ID'].astype(str)

# Drop some columns
hvbp_clean = hvbp_df.drop(columns=['Location', 'MSPB-1 Achievement Points',␣
 ↪'MSPB-1 Improvement Points', 'MSPB-1 Measure Score']).head()

# Rename other columns to remove spaces
hvbp_clean = hvbp_clean.rename(columns={'Facility ID': 'facilityid', 'Facility␣
 ↪Name': 'FacilityName', 'County Name': 'County', 'MSPB-1 Achievement␣
 ↪Threshold': 'Achievement_Threshold', 'MSPB-1 Benchmark': 'Benchmark',␣
 ↪'MSPB-1 Baseline Rate': 'Baseline_Rate', 'MSPB-1 Performance Rate':␣
 ↪'Performance_Rate', 'Achievement Points' : 'Achievement_Points',␣
 ↪'Improvement Points': 'Improvement_Points', 'Measure Score': 'Measure_Score'␣
 ↪})

# Search for null fields in all columns
null = hvbp_clean.isnull()
```

```
null.nunique()

hvbp_clean.head()
```

[6]:

	facilityid	FacilityName	Address	City	State	zipcode	County	Achievement_Threshold	Benchmark	Baseline_Rate	Performance_Rate	Achievement_Points	Improvement_Points	Measure_Score
0	10131	CRESTWOOD MEDICAL CENTER	ONE HOSPITAL DR SE	HUNTSVILLE	AL	35801	Madison	0.987067	0.844147	0.981644	0.973034	1	0	1
1	40154	BAPTIST HEALTH MEDICAL CENTER-CONWAY	1555 EXCHANGE AVENUE	CONWAY	AR	72032	Faulkner	0.987067	0.844147	0.845727	0.918049	5	0	5
2	50488	EDEN MEDICAL CENTER	20103 LAKE CHABOT ROAD	CASTRO VALLEY	CA	94546	Alameda	0.987067	0.844147	1.044817	1.039994	0	0	0
3	150069	KING'S DAUGHTERS' HEALTH	1373 EAST SR 62	MADISON	IN	47250	Jefferson	0.987067	0.844147	0.917796	0.973444	1	0	1
4	180010	SAINT JOSEPH HOSPITAL	ONE SAINT JOSEPH DRIVE	LEXINGTON	KY	40504	Fayette	0.987067	0.844147	0.982229	1.030032	0	0	0

Census Data

[7]:
```
# Dropping columns
census_clean = census_df[['zipcode', 'population']].copy()

# Changing the zipcode to categorical to avoid problems with leading zeros --␣
 ↪decided not to do this
# census_clean['zipcode'] = census_clean['zipcode'].astype('category')
print(census_clean.dtypes)

# Dropping rows that have a population less than 100
census_clean = census_clean.loc[(census_clean["population"] > 100)]

# Dropping Nulls
census_clean = census_clean.dropna(how="any")
```

```
# View sorted DF
census_clean.sort_values('zipcode', ascending = False)

# Create an index column
census_clean['index'] = census_clean.index
census_clean.head()
```

```
zipcode       int64
population    int64
dtype: object
```

[7]:

	zipcode	population	index
2	95117	1389	2
3	74074	231	3
5	75241	524	5
10	1119	306	10
22	49230	238	22

3 Loading

3.1 SQL Schema

[8]:
```
# Creating SQL tables
class hospinfo(Base):
    __tablename__ = 'hospinfo'
    facilityid = Column(Integer, primary_key=True)
    HospitalName = Column(String)
    Address = Column(String)
    City = Column(String)
    State  = Column(String)
    zipcode  = Column(Integer)
    County = Column(String)
    Phone = Column(Integer)
    Hospital_Type = Column(String)
    Hospital_Ownership = Column(String)

class hvbp(Base):
    __tablename__ = 'hvbp'
    facilityid = Column(String, primary_key=True)
    FacilityName = Column(String)
    Address = Column(String)
    City = Column(String)
    State = Column(String)
    zipcode = Column(Integer)
    County  = Column(String)
    Achievement_Threshold = Column(Integer)
    Benchmark = Column(Integer)
```

```
    Baseline_Rate = Column(String)
    Performance_Rate = Column(Integer)
    Improvement_Points = Column(String)
    Measure_Score = Column(String)
    Achievement_Points = Column(Integer)
    Improvement_Points = Column(Integer)
    Measure_Score = Column(Integer)

class census(Base):
    __tablename__ = 'census'
    index = Column(Integer, primary_key=True)
    zipcode = Column(Integer)
    population = Column(Integer)

Base.metadata.tables
```

[8]:
```
immutabledict({'hospinfo': Table('hospinfo', MetaData(bind=None),
Column('facilityid', Integer(), table=<hospinfo>, primary_key=True,
nullable=False), Column('HospitalName', String(), table=<hospinfo>),
Column('Address', String(), table=<hospinfo>), Column('City', String(),
table=<hospinfo>), Column('State', String(), table=<hospinfo>),
Column('zipcode', Integer(), table=<hospinfo>), Column('County', String(),
table=<hospinfo>), Column('Phone', Integer(), table=<hospinfo>),
Column('Hospital_Type', String(), table=<hospinfo>),
Column('Hospital_Ownership', String(), table=<hospinfo>), schema=None), 'hvbp':
Table('hvbp', MetaData(bind=None), Column('facilityid', String(), table=<hvbp>,
primary_key=True, nullable=False), Column('FacilityName', String(),
table=<hvbp>), Column('Address', String(), table=<hvbp>), Column('City',
String(), table=<hvbp>), Column('State', String(), table=<hvbp>),
Column('zipcode', Integer(), table=<hvbp>), Column('County', String(),
table=<hvbp>), Column('Achievement_Threshold', Integer(), table=<hvbp>),
Column('Benchmark', Integer(), table=<hvbp>), Column('Baseline_Rate', String(),
table=<hvbp>), Column('Performance_Rate', Integer(), table=<hvbp>),
Column('Achievement_Points', Integer(), table=<hvbp>),
Column('Improvement_Points', Integer(), table=<hvbp>), Column('Measure_Score',
Integer(), table=<hvbp>), schema=None), 'census': Table('census',
MetaData(bind=None), Column('index', Integer(), table=<census>,
primary_key=True, nullable=False), Column('zipcode', Integer(), table=<census>),
Column('population', Integer(), table=<census>), schema=None)})
```

[9]:
```
conn = "postgres:postgres@localhost:5432/hospdb"
engine = create_engine(f'postgresql://{conn}')
Base.metadata.create_all(engine)
print(engine.table_names())

#Load it into database
census_clean.to_sql(name='census', con=engine, if_exists='replace', index=False)
```

```
hospUS_df.to_sql(name='hospinfo', con=engine, if_exists='replace', index=False)
hvbp_clean.to_sql(name='hvbp', con=engine, if_exists='replace', index=False)
```

```
['hospinfo', 'hvbp', 'census']
```

3.2 SQL Queries

[12]:
```
# Confirm data has been added by querying the tables
pd.read_sql_table("census", con=engine).head()
pd.read_sql_table("hvbp", con=engine).head()
pd.read_sql_table("hospinfo", con=engine).head()
```

[12]:
```
  facilityid                     Facility_Name                     Address  \
0      31309             SAGE MEMORIAL HOSPITAL  STATE ROUTE 264 SOUTH 191
1      44021        WOODRIDGE BEHAVIORAL CENTER       600 NORTH 7TH STREET
2     100277           DOUGLAS GARDENS HOSPITAL            5200 NE 2ND AVE
3     104078  SUNCOAST BEHAVIORAL HEALTH CENTER             4480 51ST ST W
4     130063           TREASURE VALLEY HOSPITAL   8800 WEST EMERALD STREET

           City State  zipcode      County           Phone  \
0        GANADO    AZ    86505      APACHE  (928) 755-4541
1  WEST MEMPHIS    AR    72301  CRITTENDEN  (870) 394-4113
2         MIAMI    FL    33137  MIAMI-DADE  (305) 751-8626
3     BRADENTON    FL    34210     MANATEE  (941) 792-2222
4         BOISE    ID    83704         ADA  (208) 373-5000

               Hospital_Type              Hospital_Ownership
0  Critical Access Hospitals  Voluntary non-profit - Private
1               Psychiatric                     Proprietary
2      Acute Care Hospitals  Voluntary non-profit - Private
3               Psychiatric                     Proprietary
4      Acute Care Hospitals                     Proprietary
```

[ ]:
```
# Run SQL query to join tables
sql_query = "SELECT * FROM hospinfo \
    INNER JOIN hvbp ON hvbp.facilityid = hospinfo.facilityid;"
#     LEFT JOIN census ON hospinfo.zipcode = census.zipcode"

from sqlalchemy.orm import Session
session = Session(bind=engine)

result = session.execute(sql_query)
for row in result:
    print(row)
```

[ ]:
```

```