Exploratory Data Analysis

Columns to analyze

After reading the data dictionary, I decided to select the following columns to explore:

- Rndrng_NPI
- Rndrng_Prvdr_Last_Org_Name
- Rndrng_Prvdr_Ent_Cd
- Rndrng_Prvdr_St1
- Rndrng_Prvdr_St2
- Rndrng_Prvdr_City
- Rndrng_Prvdr_State_Abrvtn
- Rndrng_Prvdr_Zip5
- Rndrng_Prvdr_Cntry
- Rndrng_Prvdr_Type
- Rndrng_Prvdr_Mdcr_Prtcptg_Ind
- HCPCS_Cd
- HCPCS_Desc
- Place_Of_Srvc
- Tot_Benes
- Tot_Srvcs
- Tot_Bene_Day_Srvcs
- Avg_Sbmtd_Chrg
- Avg_Mdcr_Alowd_Amt
- Avg_Mdcr_Pymt_Amt
- Avg_Mdcr_Stdzd_Amt

Summary

- Size (csv): 3.206.087.491 bytes (2.98GB)
- 10,140,228 rows (rows are unique by each tuple of (Rndrng_NPI, HCPCS_Cd, Place_Of_Srvc))
- 1,093,367 distinct providers
  - Rndrng_Prvdr_Ent_Cd (individual or org): I 1,033,965 (94.56%) / O 59,402
  - Rndrng_Prvdr_Mdcr_Prtcptg_Ind (medicare participation): Y 1,092,104 (99.88%) / N 1,263
  - Rndrng_Prvdr_Cntry (country): 1,093,294 providers are from US (99.99%)
- 6,138 distinct services/procedures (HCPCS_Cd)
- 5,568 distinct services/prodedures descriptions (HPCS_Desc)
- 99 distinct service specialties/types (Rndrng_Prvdr_Type)
- Place_Of_Srvc (facility or not): F 3,887,551 (38.33%) / O 6,252,677

Useful columns for Map visual dashboard

Final list of all columns to store in Postgres database with column name in table science.med_provider_services

- Rndrng_NPI -> prvdr_id
- Rndrng_Prvdr_Last_Org_Name -> prvdr_name
- Rndrng_Prvdr_Ent_Cd -> prvdr_ent
- Rndrng_Prvdr_St1 -> prvdr_st1
- Rndrng_Prvdr_St2 -> prvdr_st2 (can be empty/null)
- Rndrng_Prvdr_City -> prvdr_city
- Rndrng_Prvdr_State_Abrvtn -> prvdr_state
- Latitude (extracted from Rndrng_Prvdr_Zip5) -> prvdr_lat
- Longitude (extracted from Rndrng_Prvdr_Zip5) -> prvdr_long
- Rndrng_Prvdr_Type -> prvdr_spec
- HCPCS_Cd -> srvc_code
- HCPCS_Desc -> srvc_desc
- Place_Of_Srvc -> place_of_svc
- Tot_Benes -> tot_users
- Tot_Srvcs -> tot_srvcs
- Avg_Sbmtd_Chrg -> avg_chrg_amt
- Avg_Prcnt_Mdcr_Amt = (Avg_Mdcr_Pymt_Amt / Avg_Sbmtd_Chrg) * 100 (avg % medicare paid amount out of charged amount) -> avg_prcnt_mdcr_amt ## Table key
- (Rndrng_NPI, HCPCS_Cd, Place_Of_Srvc) -> (prvdr_id, srvc_code, place_of_svc) ## Processor transformations
- Rename columns
- For geo localization: Rndrng_Prvdr_Zip5 (obtain latitude and longitude coordinates)
- Rndrng_Prvdr_Cntry: select only US to restrict map visual range, considering 99.99% of the providers are US based
- Avg_Prcnt_Mdcr_Amt = (Avg_Mdcr_Pymt_Amt / Avg_Sbmtd_Chrg) * 100: avg % Medicare paid amount out of Avg_Sbmtd_Chrg ## Map visual controls
- prvdr_spec
- srvc_desc ## Provider/service details
- prvdr_name
- prvdr_st1
- prvdr_st2
- prvdr_city
- prvdr_state?
- prvdr_ent? ## Map visual metrics
- tot_users (data point size)
- tot_srvcs (data point size)
- avg_chrg_amt (data point color scale)
- avg_prcnt_mdcr_amt (data point color scale)

Read data

In [7]:
```
import pandas as pd
pd.set_option('display.max_columns', None)

cols = ['Rndrng_NPI', 'Rndrng_Prvdr_Last_Org_Name', 'Rndrng_Prvdr_Ent_Cd', 'Rndrng_
        'Rndrng_Prvdr_State_Abrvtn', 'Rndrng_Prvdr_Zip5', 'Rndrng_Prvdr_Cntry','Rn
        'HCPCS_Desc', 'Place_Of_Srvc', 'Tot_Benes', 'Tot_Srvcs', 'Tot_Bene_Day_Srv

id_col = 'Rndrng_NPI' # 'HCPCS_Cd'

num_cols = ['Tot_Benes', 'Tot_Srvcs', 'Tot_Bene_Day_Srvcs', 'Avg_Sbmtd_Chrg', 'Avg_

cat_cols = ['Rndrng_Prvdr_Last_Org_Name', 'Rndrng_Prvdr_Ent_Cd', 'Rndrng_Prvdr_St1
            'Rndrng_Prvdr_State_Abrvtn', 'Rndrng_Prvdr_Zip5', 'Rndrng_Prvdr_Cntry'
            'HCPCS_Cd', 'HCPCS_Desc', 'Place_Of_Srvc']

selected_cols = ['Rndrng_NPI', 'Rndrng_Prvdr_Last_Org_Name', 'Rndrng_Prvdr_Ent_Cd'
                 'Rndrng_Prvdr_St2', 'Rndrng_Prvdr_City', 'Rndrng_Prvdr_State_Abrv
                 'Rndrng_Prvdr_Cntry', 'Rndrng_Prvdr_Type', 'HCPCS_Cd', 'HCPCS_Des
                 'Tot_Benes', 'Tot_Srvcs', 'Avg_Sbmtd_Chrg', 'Avg_Mdcr_Pymt_Amt',
```

In [ ]:
```
#df = pd.read_csv('sample.csv', usecols=columns)
df = pd.read_csv('producer/data/unprocessed/medicare_dataset.csv', usecols=selecte
```

Data spot-checks

In [11]:
```
df['Rndrng_NPI'].value_counts()
```

Out[11]:
```
1538144910    647
1891731626    613
1134277494    604
1932145778    595
1366479099    586
             ...
1801347661      1
1174858187      1
1801349501      1
1174858021      1
1780971119      1
Name: Rndrng_NPI, Length: 1093367, dtype: int64
```

In [12]:
```
df['HCPCS_Cd'].value_counts()
```

Out[12]:
```
99213    471670
99214    460412
99204    189211
99203    182297
99232    179981
          ...
68750         1
C9399         1
27632         1
44384         1
21400         1
Name: HCPCS_Cd, Length: 6138, dtype: int64
```

In [13]:
```
df['HCPCS_Desc'].value_counts()
```

Out[13]:
```
Established patient office or other outpatient visit, typically 15 minutes
471670
Established patient office or other outpatient, visit typically 25 minutes
460412
New patient office or other outpatient visit, typically 45 minutes
189211
New patient office or other outpatient visit, typically 30 minutes
182297
Subsequent hospital inpatient care, typically 25 minutes per day
179981

...
Creation of a tear drainage tract to the nasal cavity with insertion of tube or st
ent                     1
Unclassified drugs or biologicals
1
Removal (3 centimeters or greater) tissue growth beneath the skin of leg or ankle
1
Placement of stent in small bowel using an endoscope which is inserted through abd
ominal opening         1
Closed treatment of broken eye socket bone
1
Name: HCPCS_Desc, Length: 5568, dtype: int64
```

In [14]:
```
df.drop_duplicates(subset='Rndrng_NPI')['Rndrng_Prvdr_Cntry'].value_counts()
```

Out[14]:
```
US    1093294
CA         14
DE         13
JP          7
GB          6
SA          5
LB          3
IT          3
AA          2
IN          2
NL          2
IL          2
TR          2
CM          1
JO          1
CR          1
PK          1
KR          1
AR          1
BS          1
CN          1
UY          1
TH          1
TT          1
NO          1
Name: Rndrng_Prvdr_Cntry, dtype: int64
```

In [15]:
```
df.drop_duplicates(subset='Rndrng_NPI')['Rndrng_Prvdr_Ent_Cd'].value_counts()
```

Out[15]:
```
I    1033965
O      59402
Name: Rndrng_Prvdr_Ent_Cd, dtype: int64
```

In [16]:
```
df.drop_duplicates(subset='Rndrng_NPI')['Rndrng_Prvdr_Mdcr_Prtcptg_Ind'].value_cou
```

Out[16]:
```
Y    1092104
N       1263
Name: Rndrng_Prvdr_Mdcr_Prtcptg_Ind, dtype: int64
```

In [17]:
```
#df.drop_duplicates(subset='Rndrng_NPI')['Rndrng_Prvdr_Type'].value_counts()
df['Rndrng_Prvdr_Type'].value_counts()
```

Out[17]:
```
Diagnostic Radiology                        1255124
Internal Medicine                           1021134
Family Practice                              936587
Nurse Practitioner                           707036
Physician Assistant                          445857
                                             ...
Medical Genetics and Genomics                   48
Medicare Diabetes Preventive Program            46
Unknown Supplier/Provider Specialty             41
Slide Preparation Facility                      30
Intensive Cardiac Rehabilitation                 7
Name: Rndrng_Prvdr_Type, Length: 99, dtype: int64
```

In [3]:
```
df['Rndrng_Prvdr_RUCA_Desc'].value_counts()
```

Out[3]:
```
Metropolitan area core: primary flow within an urbanized area of 50,000 and greate
r                      8605521
Micropolitan area core: primary flow within an urban cluster of 10,000 to 49,999
693768
Metropolitan area high commuting: primary flow 30% or more to a urbanized area of
50,000 and greater     239317
Small town core: primary flow within an urban cluster of 2,500 to 9,999
217587
Secondary flow 30% to <50% to a larger urbanized area of 50,000 and greater
152061
Rural areas: primary flow to a tract outside a urbanized area of 50,000 and greate
r or UC                 74107
Secondary flow 30% to <50% to a urbanized area of 50,000 and greater
53551
Micropolitan high commuting: primary flow 30% or more to a urban cluster of 10,000
to 49,999              44691
Metropolitan area low commuting: primary flow 10% to <30% to a urbanized area of 5
0,000 and greater       12447
Unknown
11861
Small town high commuting: primary flow 30% or more to a urban cluster of 2,500 to
9,999                  10328
Secondary flow 30% to <50% to a urban cluster of 10,000 to 49,999
10176
Small town low commuting: primary flow 10% to <30% to a urban cluster of 2,500 to
9,999                   4085
Micropolitan low commuting: primary flow 10% to <30% to a urban cluster of 10,000
to 49,999               3799
Secondary flow 30% to <50% to a urban cluster of 2,500 to 9,999
825
Name: Rndrng_Prvdr_RUCA_Desc, dtype: int64
```

In [18]:
```
df['Place_Of_Srvc'].value_counts()
```

Out[18]:
```
O    6252677
F    3887551
Name: Place_Of_Srvc, dtype: int64
```

In [19]:
```
df['Rndrng_NPI'].value_counts()
```

Out[19]:
```
1538144910    647
1891731626    613
1134277494    604
1932145778    595
1366479099    586
              ...
1801347661      1
1174858187      1
1801349501      1
1174858021      1
1780971119      1
Name: Rndrng_NPI, Length: 1093367, dtype: int64
```

In [20]:
```
df.groupby(['Rndrng_NPI', 'HCPCS_Cd', 'Place_Of_Srvc']).count()
```

Out[20]:

			Rndrng_Prvdr_Last_Org_Name	Rndrng_Prvdr_Ent_Cd	Rndr
Rndrng_NPI	**HCPCS_Cd**	**Place_Of_Srvc**			
1003000126	**99217**	**F**	1	1	
	99220	**F**	1	1	
	99221	**F**	1	1	
	99223	**F**	1	1	
	99232	**F**	1	1	
...	**...**	**...**	...	...	
1992999825	**99214**	**F**	1	1	
		O	1	1	
	99215	**O**	1	1	
1992999874	**99223**	**F**	1	1	
	99233	**F**	1	1	

10140228 rows × 18 columns

In [21]:
```
# Check an example of provider with multiple services
# Notice that regardless of Place_Of_Srvc being 'O' (office, non-facility), procedu
df[df['Rndrng_NPI']==1538144910].sort_values(by='HCPCS_Cd')[cols]
```

Out[21]:

	Rndrng_NPI	Rndrng_Prvdr_Last_Org_Name	Rndrng_Prvdr_Ent_Cd	Rndrng_Prvdr_St1	Rn
5408542	1538144910	Laboratory Corporation Of America Holdings	O	1447 York Ct	
5408543	1538144910	Laboratory Corporation Of America Holdings	O	1447 York Ct	
5408544	1538144910	Laboratory Corporation Of America Holdings	O	1447 York Ct	
5408545	1538144910	Laboratory Corporation Of America Holdings	O	1447 York Ct	
5408546	1538144910	Laboratory Corporation Of America Holdings	O	1447 York Ct	
...	...	...	...	...	
5409184	1538144910	Laboratory Corporation Of America Holdings	O	1447 York Ct	
5409185	1538144910	Laboratory Corporation Of America Holdings	O	1447 York Ct	
5409186	1538144910	Laboratory Corporation Of America Holdings	O	1447 York Ct	
5409187	1538144910	Laboratory Corporation Of America Holdings	O	1447 York Ct	
5409188	1538144910	Laboratory Corporation Of America Holdings	O	1447 York Ct	

647 rows × 21 columns

In [22]:
```
# Check an example of provider with multiple services
df[df['Rndrng_NPI']==1538144910].sort_values(by='HCPCS_Cd')[num_cols]
```

Out[22]:

	Tot_Benes	Tot_Srvcs	Tot_Bene_Day_Srvcs	Avg_Sbmtd_Chrg	Avg_Mdcr_Alowd_Amt	Avg
5408542	3305	3455.0	3455	731.931644	354.288964	
5408543	19	19.0	19	810.000000	269.840000	
5408544	917841	1846259.0	1846237	23.524481	2.938138	
5408545	39	39.0	39	125.873590	13.460000	
5408546	202566	297597.0	297570	40.039348	9.199015	
...	...	...	...	...	...	
5409184	5393	7262.0	7262	275.164240	192.885084	
5409185	56500	86489.0	86489	339.383158	234.286759	
5409186	37	37.0	37	43.466216	14.690000	
5409187	27	324.0	37	1.083179	1.006173	
5409188	21	23.0	23	5.464783	5.355217	

647 rows × 7 columns

In [23]:
```
df[df['Rndrng_Prvdr_Mdcr_Prtcptg_Ind']=='N']['Rndrng_NPI'].value_counts()
```

Out[23]:
```
1730209289    21
1386822641    18
1669463030    14
1558415414    13
1467561191    12
              ..
1396846770     1
1396831590     1
1396813911     1
1396765228     1
1992925572     1
Name: Rndrng_NPI, Length: 1782, dtype: int64
```

In [24]:
```
# Check an example of provider with Rndrng_Prvdr_Mdcr_Prtcptg_Ind='N'
df[df['Rndrng_NPI']==1730209289].sort_values(by='HCPCS_Cd')[cols]
```

Out[24]:

	Rndrng_NPI	Rndrng_Prvdr_Last_Org_Name	Rndrng_Prvdr_Ent_Cd	Rndrng_Prvdr_St1	Rn
7459614	1730209289	Flowers	I	299 Highway 51	
7459615	1730209289	Flowers	I	299 Highway 51	
7459616	1730209289	Flowers	I	299 Highway 51	
7459617	1730209289	Flowers	I	299 Highway 51	
7459618	1730209289	Flowers	I	299 Highway 51	
7459619	1730209289	Flowers	I	299 Highway 51	
7459620	1730209289	Flowers	I	299 Highway 51	
7459621	1730209289	Flowers	I	299 Highway 51	
7459622	1730209289	Flowers	I	299 Highway 51	
7459623	1730209289	Flowers	I	299 Highway 51	
7459624	1730209289	Flowers	I	299 Highway 51	

	Rndrng_NPI	Rndrng_Prvdr_Last_Org_Name	Rndrng_Prvdr_Ent_Cd	Rndrng_Prvdr_St1	Rn
7459625	1730209289	Flowers	I	299 Highway 51	
7459626	1730209289	Flowers	I	299 Highway 51	
7459627	1730209289	Flowers	I	299 Highway 51	
7459628	1730209289	Flowers	I	299 Highway 51	
7459629	1730209289	Flowers	I	299 Highway 51	
7459630	1730209289	Flowers	I	299 Highway 51	
7459631	1730209289	Flowers	I	299 Highway 51	
7459632	1730209289	Flowers	I	299 Highway 51	
7459633	1730209289	Flowers	I	299 Highway 51	
7459634	1730209289	Flowers	I	299 Highway 51	
7459635	1730209289	Flowers	I	299 Highway 51	
7459636	1730209289	Flowers	I	299 Highway 51	
7459637	1730209289	Flowers	I	299 Highway 51	

	Rndrng_NPI	Rndrng_Prvdr_Last_Org_Name	Rndrng_Prvdr_Ent_Cd	Rndrng_Prvdr_St1	Rn
7459638	1730209289	Flowers	I	299 Highway 51	
7459639	1730209289	Flowers	I	299 Highway 51	
7459640	1730209289	Flowers	I	299 Highway 51	

In [25]:
```
# Check an example of provider with Rndrng_Prvdr_Mdcr_Prtcptg_Ind='N'
df[df['Rndrng_NPI']==1730209289].sort_values(by='HCPCS_Cd')[num_cols]
```

Out[25]:

	Tot_Benes	Tot_Srvcs	Tot_Bene_Day_Srvcs	Avg_Sbmtd_Chrg	Avg_Mdcr_Alowd_Amt	Avg
7459614	427	524.0	524	89.421145	76.841698	
7459615	115	171.0	128	52.638187	45.106199	
7459616	13	13.0	13	117.942308	101.072308	
7459617	59	65.0	60	73.397077	62.298615	
7459618	66	72.0	68	110.467917	93.405972	
7459619	30	33.0	32	130.156364	111.629091	
7459620	13	13.0	13	116.278462	99.629231	
7459621	38	45.0	41	93.796889	80.353333	
7459622	26	26.0	26	132.316538	113.210385	
7459623	13	14.0	14	157.750000	135.338571	
7459624	687	906.0	906	50.239978	43.050751	
7459625	502	2547.0	649	5.536549	4.701559	
7459626	21	24.0	24	150.900000	129.463333	
7459627	43	45.0	45	108.031778	89.207333	
7459628	56	66.0	59	122.061061	104.464242	
7459629	103	123.0	112	156.653008	133.764390	
7459630	26	27.0	27	152.268519	129.631481	
7459631	30	31.0	31	177.409677	152.121935	
7459632	40	45.0	44	158.488889	135.730667	
7459633	28	32.0	31	201.639375	168.697188	
7459634	21	23.0	22	95.427826	79.900435	
7459635	28	29.0	29	77.938966	65.871379	
7459636	76	76.0	76	110.448158	93.479474	
7459637	14	14.0	14	166.900000	142.935000	
7459638	470	598.0	598	45.220184	38.720753	
7459639	804	951.0	951	74.991293	64.347497	
7459640	87	89.0	89	110.489775	94.702921	

In [9]:

```
# Compute summary stats on all numerical columns
# Verify consistency (is it a valid percentage? Are there divisions by zero?) of av
df['avg_prcnt_mdcr_amt'] = (df['Avg_Mdcr_Pymt_Amt'] / df['Avg_Sbmtd_Chrg']) * 100
df[num_cols+['avg_prcnt_mdcr_amt']].describe()
```

Out[9]:

	Tot_Benes	Tot_Srvcs	Tot_Bene_Day_Srvcs	Avg_Sbmtd_Chrg	Avg_Mdcr_Alowd_Amt
count	1.014023e+07	1.014023e+07	1.014023e+07	1.014023e+07	1.014023e+07
mean	8.669064e+01	2.556841e+02	1.383125e+02	3.753479e+02	1.028363e+02
std	1.215888e+03	7.274250e+03	2.058068e+03	1.172499e+03	3.029378e+02
min	1.100000e+01	2.900000e+00	1.100000e+01	4.906820e-05	4.906820e-05
25%	1.700000e+01	2.000000e+01	2.000000e+01	6.100000e+01	2.277086e+01
50%	3.200000e+01	4.300000e+01	4.000000e+01	1.520000e+02	6.578000e+01
75%	7.400000e+01	1.170000e+02	1.050000e+02	3.160000e+02	1.130800e+02
max	9.186660e+05	1.129063e+07	1.846237e+06	9.999999e+04	4.461226e+04

Check missing data

In [27]:
```
# Check all relevant columns for NULLs
selected_cols = ['Rndrng_NPI', 'Rndrng_Prvdr_Last_Org_Name', 'Rndrng_Prvdr_Ent_Cd',
                 'Rndrng_Prvdr_St2', 'Rndrng_Prvdr_City', 'Rndrng_Prvdr_State_Abrv
                 'Rndrng_Prvdr_Cntry', 'Rndrng_Prvdr_Type', 'HCPCS_Cd', 'HCPCS_Des
                 'Tot_Benes', 'Tot_Srvcs', 'Avg_Sbmtd_Chrg', 'Avg_Mdcr_Pymt_Amt']
for col in selected_cols:
    if df[col].isnull().values.any():
        print(f'[{col}] has null values!')
```

```
[Rndrng_Prvdr_St2] has null values!
[Rndrng_Prvdr_Zip5] has null values!
```

In [28]:
```
# Only 2 rows have ZIP code null, but since country is 'CR' those rows are going t
df[df['Rndrng_Prvdr_Zip5'].isnull()]
```

Out[28]:

	Rndrng_NPI	Rndrng_Prvdr_Last_Org_Name	Rndrng_Prvdr_Ent_Cd	Rndrng_Prvdr_St1	Rn
5982984	1588870349	Wixson	I	Rotunda 8, Casa # 163	
5982985	1588870349	Wixson	I	Rotunda 8, Casa # 163	

In [29]:
```
# ~6.7M out of ~10M rows have Street2 NULL
df[df['Rndrng_Prvdr_St2'].isnull()]
```

Out[29]:

	Rndrng_NPI	Rndrng_Prvdr_Last_Org_Name	Rndrng_Prvdr_Ent_Cd	Rndrng_Prvdr_St1	R
0	1003000126	Enkeshafi	I	900 Seton Dr	
1	1003000126	Enkeshafi	I	900 Seton Dr	
2	1003000126	Enkeshafi	I	900 Seton Dr	
3	1003000126	Enkeshafi	I	900 Seton Dr	
4	1003000126	Enkeshafi	I	900 Seton Dr	
...	...	...	...	...	
10140205	1992999759	Soriano	I	171 Clover Point Cir	
10140206	1992999759	Soriano	I	171 Clover Point Cir	
10140207	1992999759	Soriano	I	171 Clover Point Cir	
10140226	1992999874	Joffe	I	5620 Brook Rd	
10140227	1992999874	Joffe	I	5620 Brook Rd	

6691815 rows × 22 columns

Explore geo localization from ZIP code

In [30]:
```
df.head()
```

Out[30]:

	Rndrng_NPI	Rndrng_Prvdr_Last_Org_Name	Rndrng_Prvdr_Ent_Cd	Rndrng_Prvdr_St1	Rndrng_Pr
0	1003000126	Enkeshafi	I	900 Seton Dr	
1	1003000126	Enkeshafi	I	900 Seton Dr	
2	1003000126	Enkeshafi	I	900 Seton Dr	
3	1003000126	Enkeshafi	I	900 Seton Dr	
4	1003000126	Enkeshafi	I	900 Seton Dr	

In [31]:
```
import pgeocode

nomi = pgeocode.Nominatim('us')
```

In [32]:
```
# Result makes sense when compared to Rndrng_Prvdr_City and Rndrng_Prvdr_State_Abr
nomi.query_postal_code('21502')
```

Out[32]:
```
postal_code            21502
country_code              US
place_name        Cumberland
state_name          Maryland
state_code                MD
county_name         Allegany
county_code              1.0
community_name           NaN
community_code           NaN
latitude             39.5992
longitude           -78.8444
accuracy                 4.0
Name: 0, dtype: object
```

TODO Interesting visuals

- Heatmap procedures/specialties x regions (US states) / RUCA desc
- Heatmap providers x procedures/specialty

Aggregate by count Rndrng_NPI, Avg_Sbmtd_Chrg, Tot_Benes or Tot_Srvcs, % Medicare paid

Careful with very high number of providers and procedures

Test some visuals

In [1]:
```
import pandas as pd
pd.set_option('display.max_columns', None)
selected_cols = ['Rndrng_Prvdr_State_Abrvtn', 'Rndrng_Prvdr_Cntry', 'Rndrng_Prvdr_
                 'Tot_Benes', 'Tot_Srvcs', 'Avg_Sbmtd_Chrg', 'Avg_Mdcr_Pymt_Amt']
df = pd.read_csv('producer/data/unprocessed/medicare_dataset.csv', usecols=selecte
df['avg_prcnt_mdcr_amt'] = (df['Avg_Mdcr_Pymt_Amt'] / df['Avg_Sbmtd_Chrg']) * 100
df[df['Rndrng_Prvdr_Cntry'] == 'US']
```

Out[1]:

	Rndrng_Prvdr_State_Abrvtn	Rndrng_Prvdr_Cntry	Rndrng_Prvdr_Type	HCPCS_Desc	Tot
0	MD	US	Internal Medicine	Hospital observation care on day of discharge	
1	MD	US	Internal Medicine	Hospital observation care, typically 70 minutes	
2	MD	US	Internal Medicine	Initial hospital inpatient care, typically 30 ...	
3	MD	US	Internal Medicine	Initial hospital inpatient care, typically 70 ...	
4	MD	US	Internal Medicine	Subsequent hospital inpatient care, typically ...	
...	...	...	...	...	
10140223	WA	US	Otolaryngology	Established patient office or other outpatient...	
10140224	WA	US	Otolaryngology	Established patient office or other outpatient...	
10140225	WA	US	Otolaryngology	Established patient office or other outpatient...	
10140226	VA	US	Internal Medicine	Initial hospital inpatient care, typically 70 ...	
10140227	VA	US	Internal Medicine	Subsequent hospital inpatient care, typically ...	

10139650 rows × 9 columns

In [2]:
```
import plotly.express as px
```

In [3]:
```
#ax = sns.heatmap(df.pivot("Rndrng_Prvdr_Type", "Rndrng_Prvdr_State_Abrvtn", "Avg_
fig = px.density_heatmap(df, x="Rndrng_Prvdr_Type", y="Rndrng_Prvdr_State_Abrvtn",
```

In [4]:
```
fig.show()
```

Pandas profiling

In [4]:
```
from pandas_profiling import ProfileReport
```

In [5]:
```
profile = ProfileReport(df, title="Pandas Profiling Report")
```

In [6]:
```
profile
```

```
Summarize dataset:   0%|          | 0/5 [00:00<?, ?it/s]
Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]
Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]
```

Overview

Dataset statistics

Number of variables	18
Number of observations	10140228
Missing cells	6691817
Missing cells (%)	3.7%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	1.4 GiB
Average record size in memory	144.0 B

Variable types

Numeric	6
Categorical	11
Unsupported	1

Alerts

`Rndrng_Prvdr_Last_Org_Name` has a high cardinality: 267873 distinct values	**High cardinality**
`Rndrng_Prvdr_St1` has a high cardinality: 387128 distinct	**High cardinality**

Out[6]:

In [ ]: