msmdesign.nyc

data analytics & insights information design

exploratorydataanalysis

What is Exploratory Data Analysis?

Exploratory data analysis (EDA) is a technique used by data scientists to inspect, characterize and briefly summarize the contents of a dataset. EDA is often the first step when encountering a new or unfamiliar dataset. EDA helps the data scientist become acquainted with a dataset and test some basic assumptions about the data. By the end of the EDA process, some initial insights can be drawn from the dataset and a framework for further analysis or modeling is established.

DCA Fines and Fees

Dataset Analyzed: *DCA Fines and Fees*

About This Dataset: All fees charged by DCA for services and, all fines issued by an administrative judge resulting from violations. Each row is a fine or fee. Data provided by the Department of Consumer and Worker Protection (DCWP), the City of New York: https://data.cityofnewyork.us/Business/DCA-Fines-and-Fees/2k3g-r445

Acknowledgements: NYC Open Data https://opendata.cityofnewyork.us/

EDA Catalogue Number: INS-006

EDA Publication Date: Saturday, January 7, 2023

Language: Python

Libraries Used: NumPy, pandas, matplotlib, seaborn

EDA Author: David White

Contact: david@msmdesign.nyc | msmdesign.nyc

---

0. Prepare the workspace

0.1 Import Python libraries, packages and functions

In [3]:
```
# import libraries for data wrangling, aggregate functions and basic descriptive stati
import numpy as np
import pandas as pd

# import data visualization packages
import matplotlib.pyplot as plt
import seaborn as sns
```

0.2 Adjust display options to make plots easier to read and understand

In [92]:
```
# specify seaborn styling options
sns.set_theme(
    context='talk',
    style='whitegrid',
    palette='viridis',
    font='Courier New',
    font_scale=1.15)

# allow plots to display inline within the notebook
%matplotlib inline
```

0.3 Set Markdown tables to align-left within notebook cells

In [1]:
```
%%html
<style>
table {float:left}
</style>
```

0.4 Display all rows of output by default

In [75]:
```
pd.set_option('display.max_rows', None)

# to reset:
# pd.reset_option('display.max_rows')
```

0.5 Format large numbers and display floating point values to two decimal places

In [172…
```
pd.set_option('display.float_format',  '{:,.2f}'.format)

# to reset:
# pd.reset_option('display.float_format')
```

0.6 Load the raw data file into the notebook and visually confirm that it has been read in as expected

In [135…
```
# load the data from a csv file (stored locally) into a new DataFrame object

csv = r"F:\Creative Cloud Files\MSM Client 001 - Mister Shepherd Media LLC\MSM Design\

fees_temp = pd.read_csv(csv, encoding='utf-8')
```

In [136…
```
# glimpse the first three rows

fees_temp.head(3)
```

Out[136]:

	RECORD ID	RECORD TYPE	BUSINESS NAME	BUSINESS NAME2	INDUSTRY	FEE SEQUENCE ID	FEE TYPE	FEE DESCRIPTION
0	0829164-AMPP	Application	DACOSTA, WINSTON	NaN	Motion Picture Projectionist - 123	1370511	PLANREVIEW	Plan Review Fee
1	0828326-ATTD	Application	MAZZIO, MICHAEL R	NaN	Tow Truck Driver - 125	1436584	PLANREVIEW	Plan Review Fee
2	00777032-1-ENFO	Enforcement	JAY AMBAY INC.	PATEL BROTHERS	Stoop Line Stand - 033	240179	CNV_SI	SI - Certificate of Inspection fee (scales)

In [137…
```
# glimpse the last three rows

fees_temp.tail(3)
```

Out[137]:

	RECORD ID	RECORD TYPE	BUSINESS NAME	BUSINESS NAME2	INDUSTRY	FEE SEQUENCE ID	FEE TYPE	DESCRIPT
2565578	7398-2021-AGAR	Application	MRK GARAGE LLC	NaN	Garage - 049	3398901	LICENSE	Garag Parking License
2565579	7399-2021-AGAR	Application	WEST 12TH STREET GARAGE LLC	NaN	Garage - 049	3398903	LICENSE	Garag Parking License
2565580	7400-2021-AEHD	Application	GIFTED TECHNOLOGIES REPAIR HUB LLC	NaN	Electronic & Home Appliance Service Dealer - 115	3398919	LICENSE	Electron H Appli Service De

In [138…

```
# glimpse ten randomly selected rows

fees_temp.sample(10, random_state=72)
```

Out[138]:

	RECORD ID	RECORD TYPE	BUSINESS NAME	BUSINESS NAME2	INDUSTRY	FEE SEQUENCE ID	FEE TYPE
1564258	4479-2020-ADJC	Adjudication	DUANE READE ETAL PTRS	NaN	Drug Store Retail - 810	3197797	OL VIO
1673805	1191552-RTTD	Renewal	ANAZAGASTY, CHELIN	NaN	Tow Truck Driver - 125	736472	RENEWAL
1257701	4491-2022-ASTF	Application	BROWN, JOEDE	NaN	Temporary Street Fair Vendor Permit - 111	3457665	LICENSE
2275294	398-2021-MBGO	Amendment	PELHAM PARKWAY BABE RUTH LADIES AUXILIARY	NaN	Bingo Game Operator - 089	3391235	BC-7 FEE
1444684	1469760-AHIC	Application	HIBISCUS FLOWERS INC.	NaN	Home Improvement Contractor - 100	1251260	FINGERPRINT
534364	0952131-RLKS	Renewal	CARPIO, LUIS V	NaN	Locksmith - 062	1418003	RENEWAL
1149774	CL000214272-ADJC	Adjudication	CHILGOK PUNGMI SUNSIK INC	NATURAL HEALTH FOOD	Misc Non-Food Retail - 817	142870	CL VIO
1800023	1261849-RHIS	Renewal	NORIEGA, KENWYN	NaN	Home Improvement Salesperson - 101	841394	RENEWAL
2375619	0842694-RHIS	Renewal	COLMONE, GASPARE	NaN	Home Improvement Salesperson - 101	1304116	RENEWAL
1193700	09104986-1-ENFO	Enforcement	COLON DELI GROCERY CORP.	NaN	Grocery-Retail - 808	326935	CNV_SI

The data has been loaded and has been read in as expected.

0.7. Check the data type of each column

In [139…

```
# display a listing of each of the DataFrame's columns and its data type

fees_temp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2565581 entries, 0 to 2565580
Data columns (total 11 columns):
 #   Column           Dtype
---  ------           -----
 0   RECORD ID        object
 1   RECORD TYPE      object
 2   BUSINESS NAME    object
 3   BUSINESS NAME2   object
 4   INDUSTRY         object
 5   FEE SEQUENCE ID  int64
 6   FEE TYPE         object
 7   FEE DESCRIPTION  object
 8   FEE AMOUNT       float64
 9   FEE DATE         object
 10  FEE STATUS       object
dtypes: float64(1), int64(1), object(9)
memory usage: 215.3+ MB
```

Nearly all of the columns have been read in as *object* data type. We'll need to change the data type of some columns to a something more appropriate.

0.8 Refer to the data dictionary and make sure that our DataFrame's data types match the source data. Reassign data types where needed.

In [140…
```
# cast column(s) containing dates to datetime data type

fees_temp['FEE DATE'] = pd.to_datetime(fines_temp['FEE DATE'], errors='coerce')
```

In [141…
```
# cast column(s) containing categorical varibles to categorical data type

fees_temp['RECORD TYPE'] = fees_temp['RECORD TYPE'].astype('category')
fees_temp['INDUSTRY'] = fees_temp['INDUSTRY'].astype('category')
fees_temp['FEE TYPE'] = fees_temp['FEE TYPE'].astype('category')
fees_temp['FEE STATUS'] = fees_temp['FEE STATUS'].astype('category')
```

In [142…
```
# display the DataFrame info once again to confirm that the data type changes have bee

fees_temp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2565581 entries, 0 to 2565580
Data columns (total 11 columns):
 #   Column           Dtype
---  ------           -----
 0   RECORD ID        object
 1   RECORD TYPE      category
 2   BUSINESS NAME    object
 3   BUSINESS NAME2   object
 4   INDUSTRY         category
 5   FEE SEQUENCE ID  int64
 6   FEE TYPE         category
 7   FEE DESCRIPTION  object
 8   FEE AMOUNT       float64
 9   FEE DATE         datetime64[ns]
 10  FEE STATUS       category
dtypes: category(4), datetime64[ns](1), float64(1), int64(1), object(4)
memory usage: 151.7+ MB
```

---

1. Describe the characteristics of the dataset

1.1 How many rows and how many columns are in our dataset?

In [143…

```
# display the number of rows and columns in the DataFrame

rows = fees_temp.shape[0]
columns = fees_temp.shape[1]

print(f'There are {rows} rows and {columns} columns in the dataset.')
```

```
There are 2565581 rows and 11 columns in the dataset.
```

1.2 Identify the index of our DataFrame

In [144…

```
# display the index of the DataFrame

fees_temp.index
```

Out[144]:

```
RangeIndex(start=0, stop=2565581, step=1)
```

Our DataFrame has an interger index. We know from the data dictionary that each row is an individual fine or fee.

1.3 What are the column headings in our dataset?

In [145…

```
# display a list of the DataFrame's columns

list(fees_temp.columns)
```

Out[145]:
```
['RECORD ID',
 'RECORD TYPE',
 'BUSINESS NAME',
 'BUSINESS NAME2',
 'INDUSTRY',
 'FEE SEQUENCE ID',
 'FEE TYPE',
 'FEE DESCRIPTION',
 'FEE AMOUNT',
 'FEE DATE',
 'FEE STATUS']
```

1.4 What are the data types of each column?

In [146…
```
# display the data type of each column in the DataFrame

fees_temp.dtypes
```

Out[146]:
```
RECORD ID                 object
RECORD TYPE             category
BUSINESS NAME             object
BUSINESS NAME2            object
INDUSTRY                category
FEE SEQUENCE ID            int64
FEE TYPE                category
FEE DESCRIPTION           object
FEE AMOUNT               float64
FEE DATE          datetime64[ns]
FEE STATUS              category
dtype: object
```

1.5 How many null values are in each column?

In [147…
```
# display the number of missing values in each column of the DataFrame

fees_temp.isna().sum()
```

Out[147]:
```
RECORD ID               0
RECORD TYPE             0
BUSINESS NAME        6068
BUSINESS NAME2    1908343
INDUSTRY            48507
FEE SEQUENCE ID         0
FEE TYPE                0
FEE DESCRIPTION         0
FEE AMOUNT              0
FEE DATE                0
FEE STATUS              0
dtype: int64
```

1.6 How many unique values are there in each column?

In [148…
```
# display the count of unique elements in each column

fees_temp.nunique(axis=0, dropna=True)
```

Out[148]:
```
RECORD ID          1295259
RECORD TYPE             25
BUSINESS NAME       321251
BUSINESS NAME2       64023
INDUSTRY               130
FEE SEQUENCE ID    2565579
FEE TYPE               167
FEE DESCRIPTION        341
FEE AMOUNT           37486
FEE DATE              9281
FEE STATUS               4
dtype: int64
```

---

2. Briefly summarize the contents of the dataset

2.1 Summarize the columns containing numerical variables

In [149…
```
# describe numeric columns only

num_cols = ['FEE AMOUNT']

fees_temp[num_cols].describe(include=[np.number])
```

Out[149]:

	FEE AMOUNT
count	2,565,581.00
mean	797.19
std	335,758.92
min	-456.00
25%	40.00
50%	100.00
75%	300.00
max	525,000,000.00

2.2 Summarize the columns containing datetime variables

In [150…
```
# summarize the data contained in columns with the 'datetime' data type only

date_cols = ['FEE DATE']

fees_temp[date_cols].describe(datetime_is_numeric=True)
```

Out[150]:

	FEE DATE
count	2565581
mean	2011-06-29 11:42:25.064218112
min	1985-03-06 00:00:00
25%	2006-06-02 00:00:00
50%	2012-06-29 00:00:00
75%	2017-02-17 00:00:00
max	2022-11-21 00:00:00

2.3 Summarize the columns containing categorical variables

In [151…

```
# summarize the data contained in columns with the 'category' data type only

fees_temp.describe(include=['category'])
```

Out[151]:

	RECORD TYPE	INDUSTRY	FEE TYPE	FEE STATUS
count	2565581	2517074	2565581	2565581
unique	25	130	167	4
top	Application	Home Improvement Contractor - 100	RENEWAL	INVOICED
freq	876592	377394	703695	2347359

---

3. Select a subset of data for closer examination

3.1 Select a subset of columns

In [152…

```
# display all columns

list(fees_temp.columns)
```

Out[152]:

```
['RECORD ID',
 'RECORD TYPE',
 'BUSINESS NAME',
 'BUSINESS NAME2',
 'INDUSTRY',
 'FEE SEQUENCE ID',
 'FEE TYPE',
 'FEE DESCRIPTION',
 'FEE AMOUNT',
 'FEE DATE',
 'FEE STATUS']
```

In [153…

```
# select a subset of columns to examine
selected_cols = ['RECORD ID',
                 'RECORD TYPE',
```

```
                  'INDUSTRY',
                  'FEE TYPE',
                  'FEE AMOUNT',
                  'FEE DATE',
                  'FEE STATUS']

fees = fees_temp[selected_cols]
```

3.2 Display the shape of the data subset

In [154…

```
rows = fees.shape[0]
columns = fees.shape[1]

print(f'There are {rows} rows and {columns} columns in the subset.')
```

```
There are 2565581 rows and 7 columns in the subset.
```

---

4. Examine the individual variables in the dataset

4.1 What is the distribution of fees by industry type?

In [155…

```
fees['INDUSTRY'].value_counts(normalize=True)
```

Out[155]:

Home Improvement Contractor - 100	0.15
Bingo Game Operator - 089	0.08
Home Improvement Salesperson - 101	0.08
Tobacco Retail Dealer	0.07
Grocery-Retail - 808	0.07
Cigarette Retail Dealer - 127	0.05
Secondhand Dealer [General] - 006	0.04
Sidewalk Cafe - 013	0.03
General Vendor - 094	0.03
Electronic Store - 001	0.03
Stoop Line Stand - 033	0.02
Tow Truck Driver - 125	0.02
Temporary Street Fair Vendor Permit - 111	0.02
Supermarket - 819	0.02
Laundry - 064	0.02
Sightseeing Guide - 021	0.02
Locksmith - 062	0.02
Electronic & Home Appliance Service Dealer - 115	0.02
Process Server (Individual) - 110	0.01
Gas Station-Retail - 815	0.01
Garage - 049	0.01
Tow Truck Company - 124	0.01
Misc Non-Food Retail - 817	0.01
Electronic Cigarette Dealer	0.01
Laundries	0.01
Secondhand Dealer Auto - 005	0.01
Laundry Jobber - 066	0.01
Debt Collection Agency - 122	0.01
Fuel Oil Dealer - 814	0.01
Drug Store Retail - 810	0.01
Salons And Barbershop - 841	0.01
Pedicab Driver - 131	0.01
Parking Lot - 050	0.01
Ticket Seller	0.00
Pawnbroker - 080	0.00
Employment Agency - 034	0.00
Newsstand - 024	0.00
Amusement Device (Portable) - 018	0.00
Restaurant - 818	0.00
Dealer In Products For The Disabled - 119	0.00
Pedicab Business - 130	0.00
Auctioneer - 036	0.00
Tax Preparers - 891	0.00
Motion Picture Projectionist - 123	0.00
Horse Drawn Cab Driver - 086	0.00
Amusement Device (Temporary) - 090	0.00
Dealer in Products for the Disabled - 119	0.00
Jewelry Store-Retail - 823	0.00
Cabaret - 073	0.00
Wearing Apparel - 450	0.00
Special Sale - 102	0.00
Amusement Device (Permanent) - 016	0.00
Catering Establishment - 075	0.00
Process Server (Organization) - 109	0.00
Garage & Parking Lot - 098	0.00
Scrap Metal Processor - 118	0.00
Collections Case	0.00
Mobile Food Vendor - 881	0.00
Scale Dealer/Repairer - 107	0.00
Car Wash	0.00

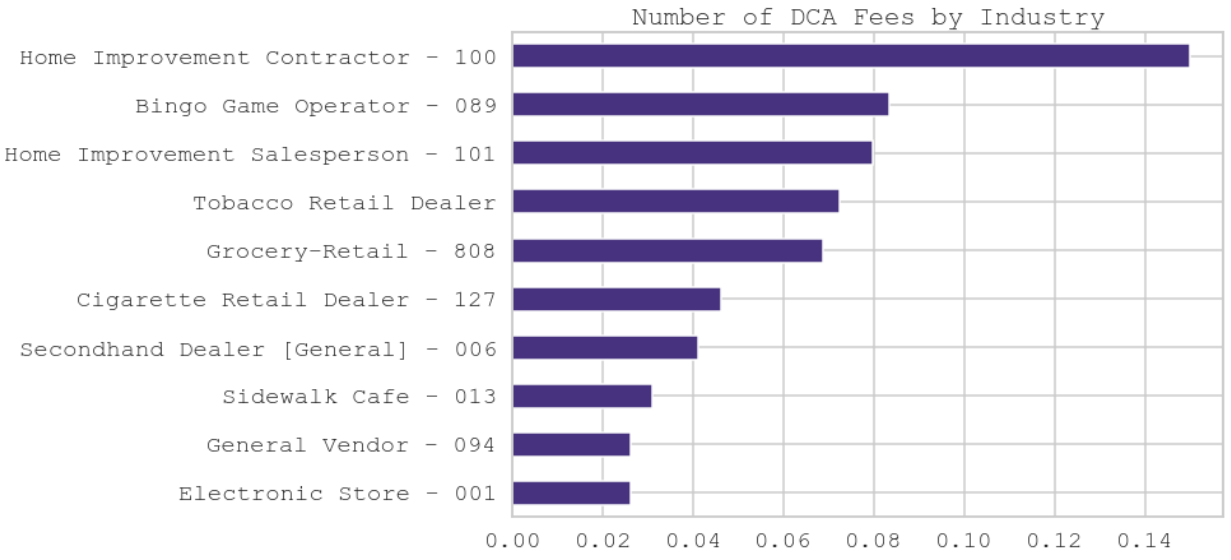
Furniture Sales - 242	0.00
Retail Store - 820	0.00
Games Of Chance - 088	0.00
Hardware-Retail - 811	0.00
Horse Drawn Cab Owner - 087	0.00
Storage Warehouse - 120	0.00
Other	0.00
Tobacco Prod'T Sales - 890	0.00
Retail Laundry	0.00
Supermarket - 819	0.00
Pool Or Billiard Room - 046	0.00
Gift Certificate - 895	0.00
Misc Archived	0.00
Sightseeing Bus - 078	0.00
Air Condtioning Law - 899	0.00
Amusement Arcade - 014	0.00
Gasoline Truck-Retail - 822	0.00
Gaming Cafe - 129	0.00
Dry Cleaners - 230	0.00
Immigration Svc Prv - 893	0.00
Auction House - 128	0.00
Motion Picture Operator - 123	0.00
Funeral Homes - 888	0.00
Commercial Lessor (Bingo/Games Of Chance) - 091	0.00
Locksmith Apprentice - 063	0.00
Tenant Screening - 480	0.00
Travel Agency - 440	0.00
Megastore - 821	0.00
Imitation Gun - 836	0.00
General Vendor Distribution - 097	0.00
Box Cutter - 831	0.00
Pool or Billiard Room - 046	0.00
Wholesale Food Market - 718	0.00
Gov'T Agency Retail - 824	0.00
Booting Company - 126	0.00
Laser Pointer Sales - 834	0.00
Floor Coverings - 241	0.00
Ticket Seller Business	0.00
Health Spa - 839	0.00
Auto Rental - 213	0.00
Construction Labor Provider	0.00
Third Party Food Delivery	0.00
Secondhand Dealer - Firearm - 006A	0.00
Mini-Storage Company - 830	0.00
Spray Paint Sls Mnor - 832	0.00
Appliances - 244	0.00
009	0.00
Bail Bonds	0.00
H25	0.00
Mailorder Misc - 319	0.00
H15	0.00
H05	0.00
Photography Services - 415	0.00
H70	0.00
Auto Dealership - 212	0.00
010	0.00
003	0.00
Industrial Laundry	0.00
Hotel/Motel - 460	0.00
Distress Prop Consultants - 247	0.00

```
Pregnancy Service Center (PSC)                    0.00
H06                                               0.00
Auto Leasing - 211                                0.00
I &                                               0.00
Tickets-Live Perf - 260                           0.00
H92                                               0.00
H10                                               0.00
037                                               0.00
Debt Settlement - 248                             0.00
H03                                               0.00
Name: INDUSTRY, dtype: float64
```

In [156…]
```
# show only those industries which represent at least 3% of all values

fees['INDUSTRY'].value_counts(normalize=True).head(10)
```

Out[156]:
```
Home Improvement Contractor - 100    0.15
Bingo Game Operator - 089            0.08
Home Improvement Salesperson - 101   0.08
Tobacco Retail Dealer                0.07
Grocery-Retail - 808                 0.07
Cigarette Retail Dealer - 127        0.05
Secondhand Dealer [General] - 006    0.04
Sidewalk Cafe - 013                  0.03
General Vendor - 094                 0.03
Electronic Store - 001               0.03
Name: INDUSTRY, dtype: float64
```

In [158…]
```
fees['INDUSTRY'].value_counts(normalize=True).head(10).sort_values().plot(kind='barh',
                                                                          figsize=(10
                                                                          title='Numb
```



4.2 Does this trend hold when looking at the amount of the fees (rather than just the number of fees assessed)?

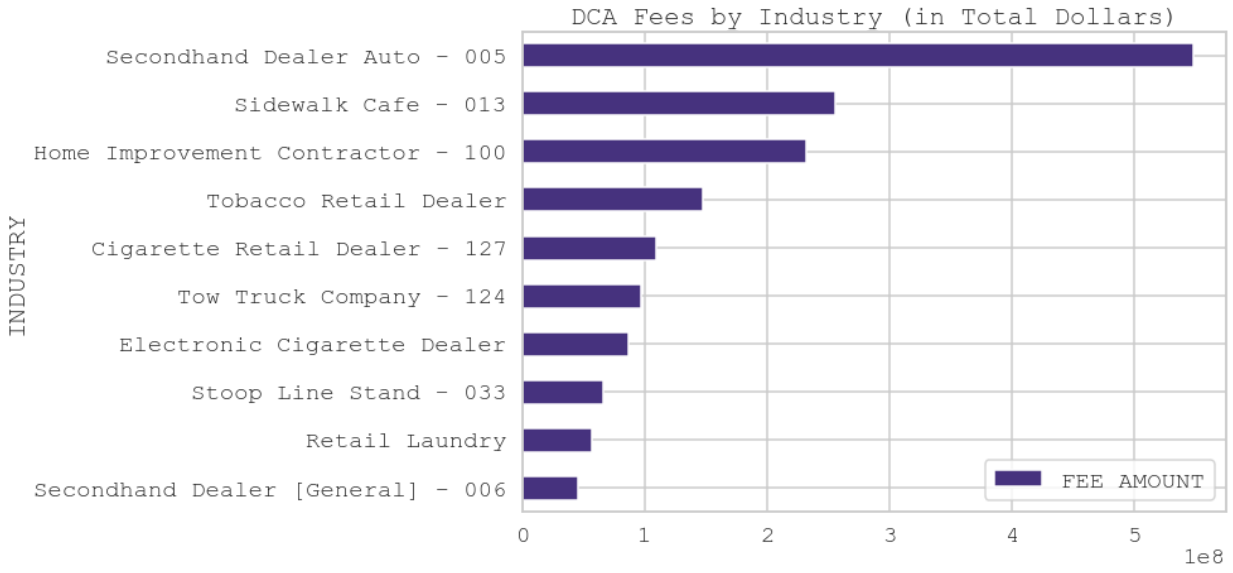
In [183…]
```
fees_pivot = pd.pivot_table(data=fees,
                            values='FEE AMOUNT',
                            index='INDUSTRY',
                            aggfunc='sum')

fees_pivot.sort_values(by='FEE AMOUNT', ascending=False).head(10)
```

Out[183]:

INDUSTRY	FEE AMOUNT
Secondhand Dealer Auto - 005	548,078,017.44
Sidewalk Cafe - 013	255,730,903.99
Home Improvement Contractor - 100	232,438,515.32
Tobacco Retail Dealer	147,384,082.01
Cigarette Retail Dealer - 127	109,378,512.85
Tow Truck Company - 124	97,302,277.00
Electronic Cigarette Dealer	86,498,369.73
Stoop Line Stand - 033	65,956,563.08
Retail Laundry	56,316,460.00
Secondhand Dealer [General] - 006	45,535,273.45

In [184…

```
fees_pivot.sort_values(by='FEE AMOUNT', ascending=False).head(10).sort_values(by='FEE
```



4.2 What is the distribution of fees by year?

In [162…

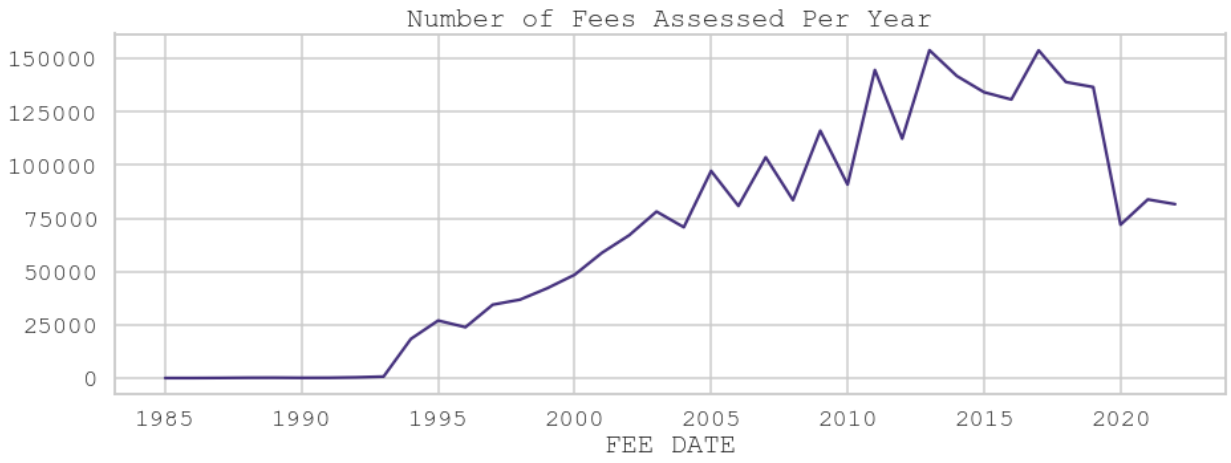
```
# total number of fees by year
fees['FEE DATE'].groupby(fees['FEE DATE'].dt.year).count()
```

Out[162]:

```
FEE DATE
1985         3
1986         4
1987        55
1988       151
1989       178
1990       102
1991       132
1992       326
1993       683
1994     18375
1995     27017
1996     23882
1997     34464
1998     36829
1999     42202
2000     48521
2001     58820
2002     67083
2003     78200
2004     70843
2005     97270
2006     80834
2007    103694
2008     83562
2009    116174
2010     90874
2011    144685
2012    112445
2013    153981
2014    141902
2015    134253
2016    130834
2017    153911
2018    139028
2019    136671
2020     72024
2021     83895
2022     81674
Name: FEE DATE, dtype: int64
```

In [165…

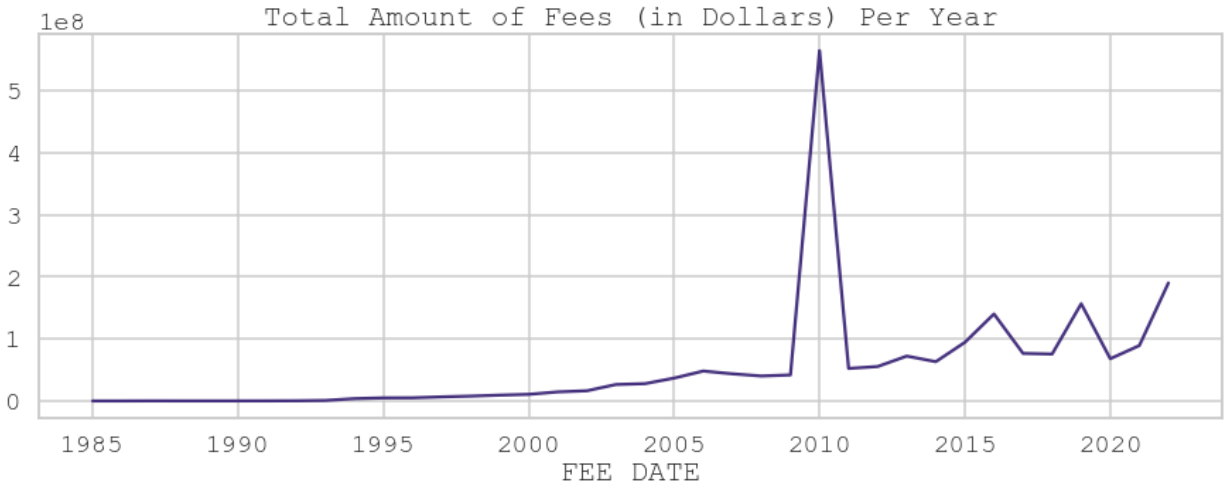
```
fees['FEE DATE'].groupby(fees['FEE DATE'].dt.year).count().plot(figsize=(15,5),
                                                                title='Number of Fees
```



In [167…
```
# total amount of fees by year
fees['FEE AMOUNT'].groupby(fees['FEE DATE'].dt.year).sum()
```

Out[167]:
```
FEE DATE
1985         2,366.00
1986         2,660.00
1987       114,333.97
1988        61,300.00
1989        77,583.74
1990        71,781.41
1991       122,060.51
1992       259,774.51
1993       814,163.61
1994     3,732,506.96
1995     4,919,217.02
1996     5,046,538.31
1997     6,581,001.31
1998     7,798,266.49
1999     9,344,627.67
2000    10,657,689.91
2001    14,582,677.59
2002    16,400,507.25
2003    26,490,691.04
2004    27,888,604.73
2005    36,830,648.14
2006    48,169,008.14
2007    43,733,685.31
2008    40,229,409.97
2009    41,888,960.73
2010   564,729,883.21
2011    52,373,146.57
2012    55,487,626.90
2013    72,312,548.96
2014    63,286,383.80
2015    94,512,698.90
2016   140,233,122.42
2017    76,644,288.83
2018    75,632,526.46
2019   156,692,655.42
2020    68,047,204.49
2021    89,258,051.88
2022   190,231,361.41
Name: FEE AMOUNT, dtype: float64
```

In [168…
```
fees['FEE AMOUNT'].groupby(fees['FEE DATE'].dt.year).sum().plot(figsize=(15,5),
                                                                title="Total Amount
```



4.3 What is the distribution of fees by 'FEE TYPE'

In [171…

```
fees['FEE TYPE'].value_counts()
```

Out[171]:

RENEWAL	703695
LICENSE	525278
TRUSTFUNDHIC	148476
CNV_SI	127132
FINGERPRINT	114445
LL VIO	87326
CL VIO	81369
OL VIO	72866
CNV_TFEE	59266
PL VIO	58976
SCALE-01	46594
TP VIO	44464
SS VIO	38288
TS VIO	35311
WH VIO	33164
DCA-SUS	26510
SWC-CON	21000
EXAMHIC	20650
WM VIO	19390
BLUEDOT	19015
BC-7 FEE	18023
PROCESSING	17655
TO VIO	16485
TAXCLEARREN	15116
CNV_MS	13487
INTEREST	10919
SWC-CON-ONL	10742
CNV_TC	10192
SWC-CIN-INT	9335
CNV_EX	9302
SCALE02	9234
APPEAL	8362
LICENSE REPL	6526
PETROL-19	6457
LATE	6249
CNV_PC	5846
PETROL-22	5131
WS VIO	5088
CNV_FS	4970
LICENSEDOC15	4728
EXAMPSI	4668
TTCINSPECT	4623
PLANREVIEW	4574
RENEWALVET	4531
EXAMSSG	3920
DCA-MFAL	3906
LICENSEDOC10	3192
PETROL-32	2958
NGC	2851
CNV_IP	2618
LICENSEDOC0	2577
CD VIO	2419
CNV_IC	2358
CNV_LF	2329
DOBINSPECT	2306
ADDROOMREN	2106
CLATE	1955
PETROL-21	1848
SV VIO	1835
DCA-PP-LF01	1802

DARP ENROLL	1659
TRUSTFUNDTTC	1480
SWC-CONADJ	1363
SEC-DEP-UN	1316
TIME-35	1257
DCA-PP-DEF01	1187
SL VIO	1136
CT	1096
APPLICATION	1059
PLAN-FEE-EN	1031
DD VIO	948
TTCQUALINSP	794
PETROL-17	759
SWC-CON-LATE	756
LICENSE0	661
BLUEDOT0	581
RENEWAL_PH	563
FINGERPRINTE	558
ADDTLROOM	543
TO	542
RENEWAL0	521
PETROL-80	498
RENEWAL2	497
LASVEGAS	483
TRUCK-72	413
ADDLPED	389
FRU FINE	362
LE	360
LICENSE2	314
TAXCLEARANCE	285
CNV_GL	260
PETROL-85	251
ZERORAFFLE	238
REMAININGLIC	228
DCATTIDREPL	211
RAFFLE	202
LIQUID-47	199
INSPECT	187
RENEWLCKGOV	182
TTCREINSPECT	177
FE	175
BELLJAR	158
PDCREINSPECT	158
GC-7FEE	151
SWC-CON-MOD	135
NWSREINSPECT	125
CT-REST	124
COMPT-70	121
FOIL	119
REINSPECT	117
SEC-DEP-EN	110
GC-7RFEE	107
EXAMMPO	104
LP VIO	94
DCAPLATEREPL	89
CT-FINE	89
SVE	82
COIATT	79
NRRF	79
SPEC PURP	78

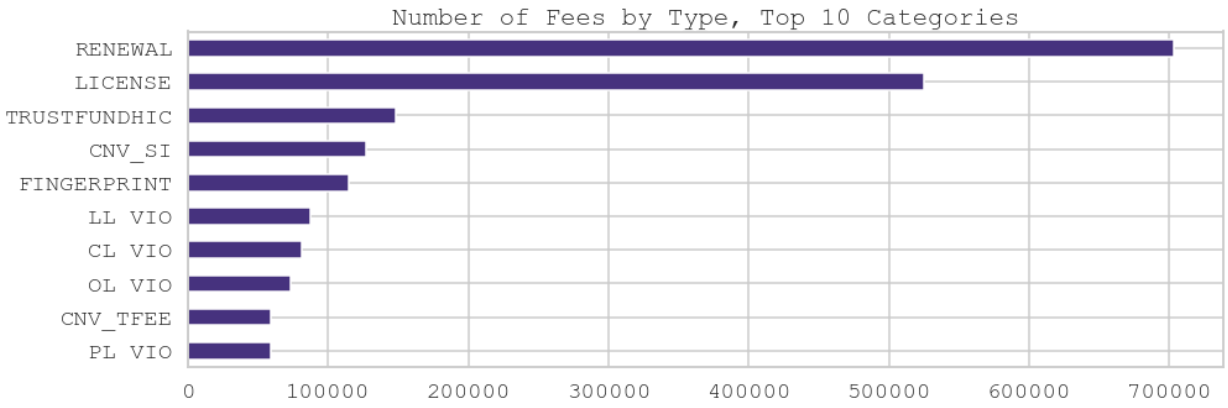
```
SFF                77
CRC                77
GL VIO             77
SVRF               76
TOW FEE            76
SUBP FEE           76
IP VIO             76
PS                 76
PSFF               76
SCALE-60           71
TO-REST            56
TO-FINE            55
DCADECALREPL       49
PLANREVIEW2        46
BLUEDOT2           39
SCALE-03           37
TRUCK-90           37
WEIGHT-45          30
SCALE-05           23
RENEWALBLIND       22
CNV_SC             22
TRUSTFUNDPSI       16
LATEFEE            15
DCASETREPL         12
SCALE-04           10
TTCRENEWINSP        9
APPFEEBLIND         8
LICREPLACE          7
ADDTLROOMBD         7
PROVER-50           7
WTS-41              6
PETROL-18           6
RENEWPSIGOV         4
WTS-42              4
TAPES-56            3
LICDOC              2
PROVER-49           2
ODOMETER-101        2
DCA-ECB             1
TIME-100            1
ENFLOCKSMITH        1
ENFPADLOCK          1
SCALE-06            1
WEIGHT-09           1
WEIGHT-46           1
ADTINVEST           1
LIQUID-15           1
Name: FEE TYPE, dtype: int64
```

In [173…

```
# display the top ten categories only

fees['FEE TYPE'].value_counts().head(10)
```

Out[173]:
```
RENEWAL          703695
LICENSE          525278
TRUSTFUNDHIC     148476
CNV_SI           127132
FINGERPRINT      114445
LL VIO            87326
CL VIO            81369
OL VIO            72866
CNV_TFEE          59266
PL VIO            58976
Name: FEE TYPE, dtype: int64
```

In [177…
```
fees['FEE TYPE'].value_counts().head(10).sort_values(ascending=True).plot(kind='barh',
                                                    figsize=(15,5),
                                                    title='Number of Fees by Type, Top 10 Ca
```

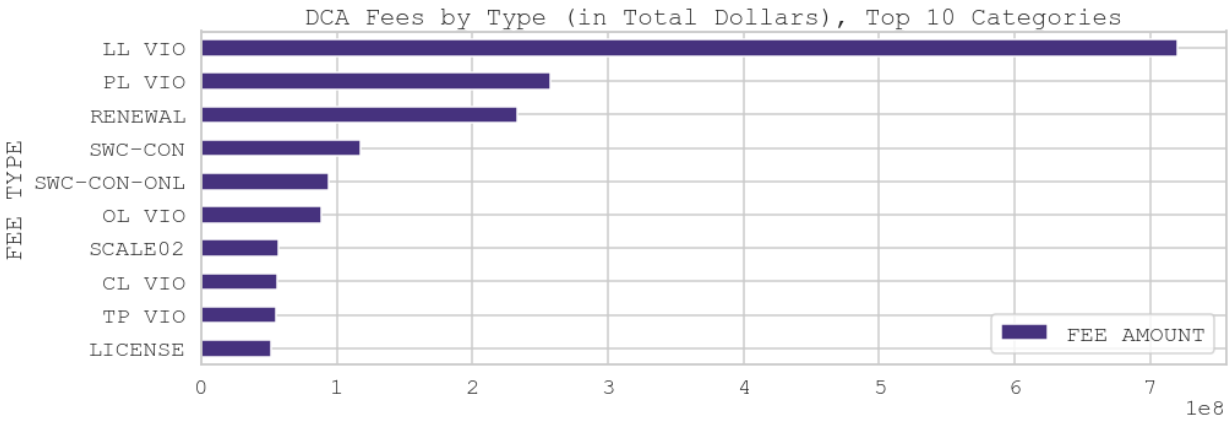


In [179…
```
# as before, compare this to the amount (rather than the count) of fees assessed

types_pivot = pd.pivot_table(data=fees,
                             values='FEE AMOUNT',
                             index='FEE TYPE',
                             aggfunc='sum')

types_pivot.sort_values(by='FEE AMOUNT', ascending=False).head(10)
```

Out[179]:

FEE TYPE	FEE AMOUNT
LL VIO	719,961,331.26
PL VIO	257,875,770.95
RENEWAL	233,561,940.56
SWC-CON	117,552,593.77
SWC-CON-ONL	93,935,977.20
OL VIO	88,771,593.09
SCALE02	56,790,400.00
CL VIO	56,160,978.86
TP VIO	54,971,031.85
LICENSE	51,447,626.39

In [182…
```
types_pivot.sort_values(by='FEE AMOUNT', ascending=False).head(10).sort_values(by='FEE
```



Next steps

In [185…
```
# export data for data graphic creation

fees_by_indsustry_top10 = fees_pivot.sort_values(by='FEE AMOUNT', ascending=False).hea
```

In [186…
```
fees_by_indsustry_top10.to_csv('fees_by_indsustry_top10.csv')
```
