DataAppendix

Adam Chow, Tori Stoner, Aniyah McWilliams

10/22/24

Variable: DR_NO

Definition: Division of Records Number: Official File number made up of 2 digit year, area ID, and 5 digits

Creation

Provided by original Data.gov Dataset

Missing value: NaN

Variable: Date Rptd

Definition: The date that the crime was reported to the police

Creation

Provided by original Data.gov Dataset

Missing value: NaN

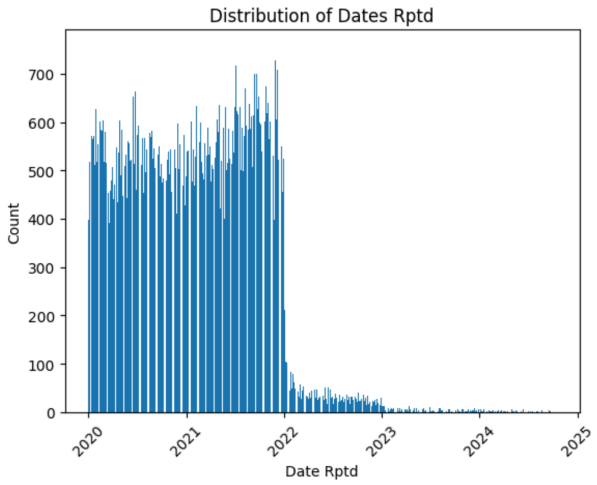
```
df["Date Rptd"].dt.date.describe()
```

	Date Rptd
count	417509
unique	1676
top	2021-11-01
freq	753

dtype: object

```
df['Date Only'] = df['Date Rptd'].dt.date

rptd_counts = df["Date Only"].value_counts().sort_index()
plt.bar(rptd_counts.index, rptd_counts.values)
plt.xlabel("Date Rptd")
plt.ylabel("Count")
plt.title("Distribution of Dates Rptd")
plt.xticks(rotation=45)
plt.show()
```



Variable: DATE OCC

Definition: The date on which the crime occurred

Creation

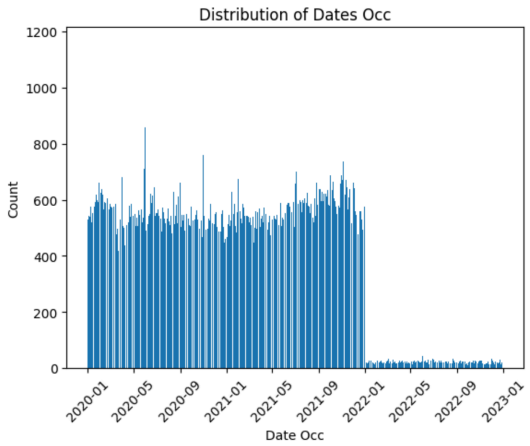
Provided by original Data.gov Dataset

Missing value: NaN

```
df["DATE OCC"].dt.date.describe()
```

	DATE OCC
count	417509
unique	1096
top	2020-01-01
freq	1157

```
df['Date Occ Only'] = df['DATE OCC'].dt.date
occ_counts = df["Date Occ Only"].value_counts().sort_index()
plt.bar(occ_counts.index, occ_counts.values)
plt.xlabel("Date Occ")
plt.ylabel("Count")
plt.title("Distribution of Dates Occ")
plt.xticks(rotation=45)
plt.show()
```



Variable: Time OCC

Definition: The time in which the crime occurred

Creation

Provided by original Data.gov Dataset

Missing value: NaN

```
def format_time(time_str):
    """Formats time string to HHMM format."""
    try:
        # Attempt to convert to integer, assuming it's already in HHMM
format or close to it
        time_int = int(time_str)
        # Pad with leading zeros if necessary
        return str(time_int).zfill(4)
    except ValueError:
        # If conversion to integer fails, assume it's an invalid format
        return None  # Or handle invalid formats differently, e.g., raise
an error

# Apply the formatting function to the 'TIME OCC' column
df['TIME OCC'] = df['TIME OCC'].astype(str).apply(format_time)
```

```
# Convert to datetime, handling errors
df['TIME OCC'] = pd.to_datetime(df['TIME OCC'], format='%H%M',
errors='coerce').dt.time
```

```
df["TIME OCC"].describe()
```

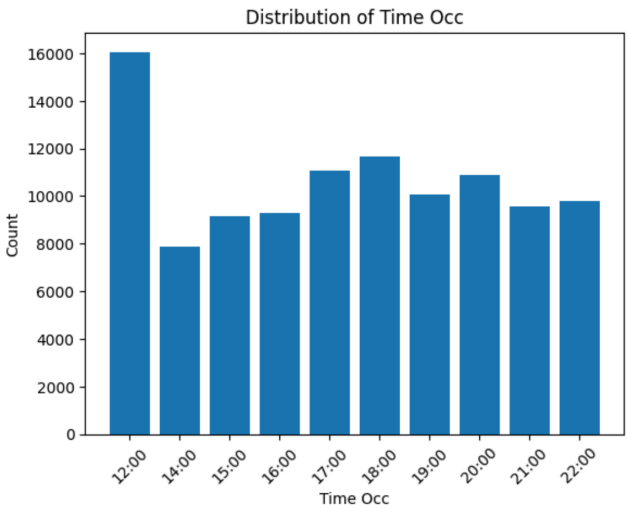
	TIME OCC
count	417509
unique	1439
top	12:00:00
freq	16062

dtype: object

```
  time_counts = df["TIME OCC"].value_counts().nlargest(10).sort_index()

        # Convert datetime.time objects to strings for plotting
 # The strftime method is used to format the time as desired like "HH:MM"
       plt.bar([t.strftime("%H:%M") for t in time_counts.index],
                           time_counts.values)

                          plt.xlabel("Time Occ")
                           plt.ylabel("Count")
                 plt.title("Distribution of Time Occ")
                        plt.xticks(rotation=45)
                              plt.show()
```



Variable: AREA

Definition: The number corresponding to the 21 LAPD police stations

Creation

Provided by original Data.gov Dataset

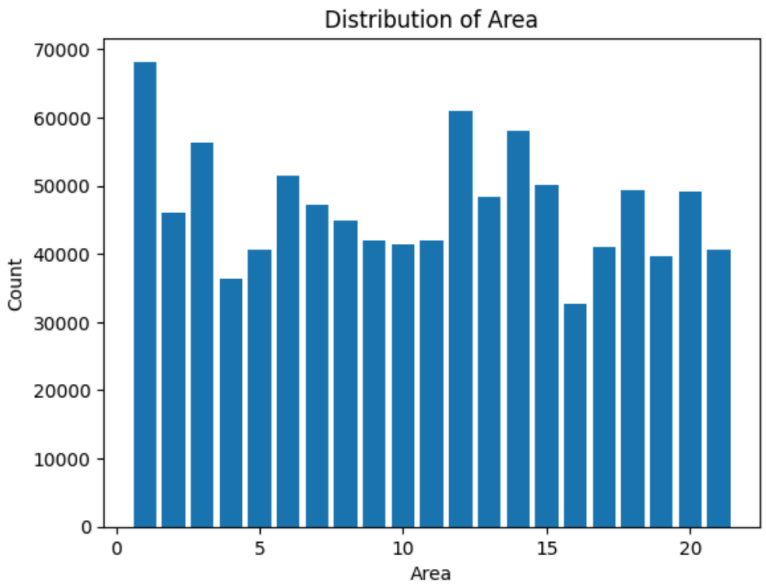
Missing value: NaN

```
df["AREA"].describe()
```

	AREA
count	986500.000000
mean	10.698086
std	6.108873
min	1.000000
25%	5.000000
50%	11.000000
75%	16.000000
max	21.000000

dtype: float64

```
Area_counts = df["AREA"].value_counts().sort_index()
plt.bar(Area_counts.index, Area_counts.values)
plt.xlabel("Area")
plt.ylabel("Count")
plt.title("Distribution of Area")
plt.show()
```



Variable: AREA NAME

Definition: The name corresponding to the 21 LAPD police stations

Creation

Provided by original Data.gov Dataset

Missing value: Nan

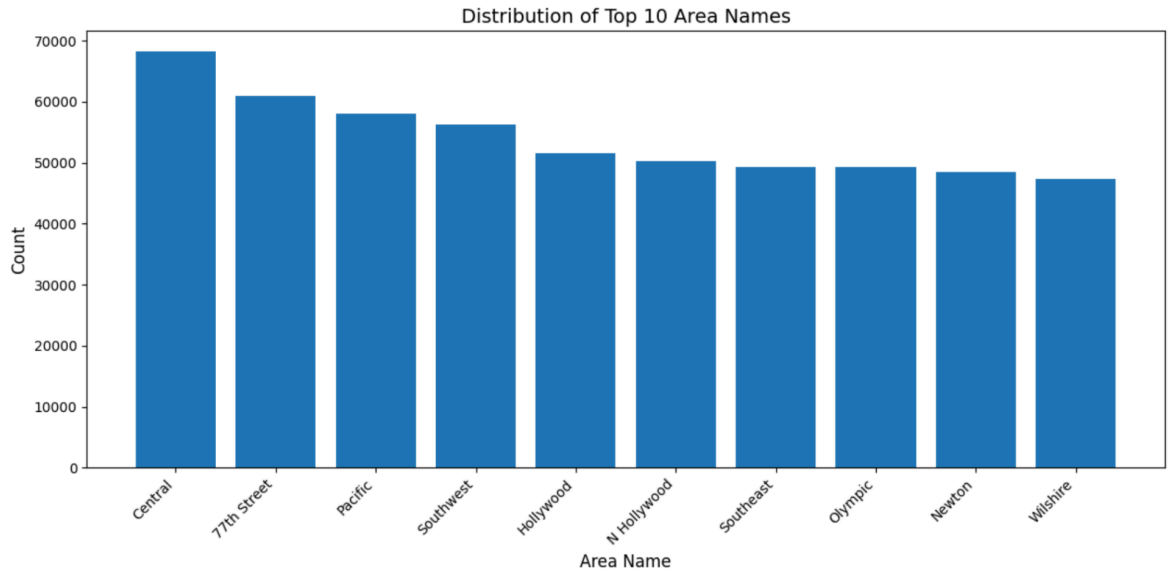
```
df["AREA NAME"].describe()
```

	AREA NAME
count	986500
unique	21
top	Central
freq	68166

dtype: object

```
name_counts = df["AREA NAME"].value_counts().nlargest(10)

plt.figure(figsize=(12, 6))  # Adjust figure size for better readability
plt.bar(name_counts.index, name_counts.values)
plt.xlabel("Area Name", fontsize=12)
plt.ylabel("Count", fontsize=12)
plt.title("Distribution of Top 10 Area Names", fontsize=14)
plt.xticks(rotation=45, ha="right", fontsize=10)  # Rotate x-axis labels for readability
plt.tight_layout()  # Adjust layout to prevent overlapping labels
plt.show()
```



Variable: Crm Cd Desc

Definition: Describes the committed crime

Creation

Provided by original Data.gov Dataset

Missing value: NaN

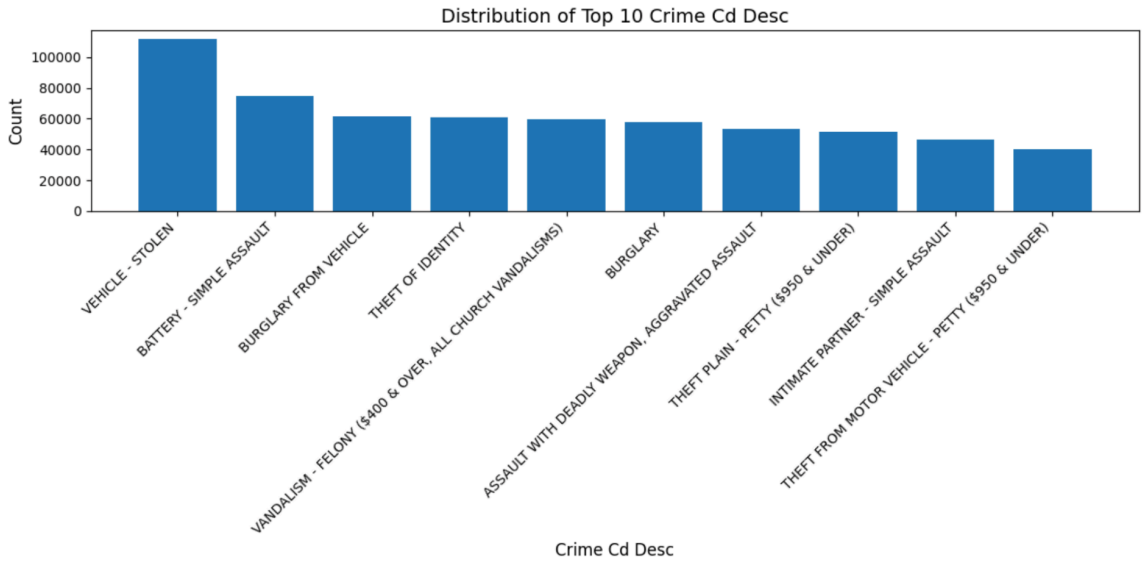
```
df["Crm Cd Desc"].describe()
```

	Crm Cd Desc
count	986500
unique	140
top	VEHICLE - STOLEN
freq	111632

dtype: object

```
crimecode_counts = df["Crm Cd Desc"].value_counts().nlargest(10)

plt.figure(figsize=(12, 6))  # Adjust figure size for better readability
plt.bar(crimecode_counts.index, crimecode_counts.values)
plt.xlabel("Crime Cd Desc", fontsize=12)
plt.ylabel("Count", fontsize=12)
plt.title("Distribution of Top 10 Crime Cd Desc", fontsize=14)
plt.xticks(rotation=45, ha="right", fontsize=10)  # Rotate x-axis labels for readability
plt.tight_layout()  # Adjust layout to prevent overlapping labels
plt.show()
```



Variable: Mocodes

Definition: Modus Operandi: Activities associated with the suspect in commission of the crime

Creation

Provided by original Data.gov Dataset

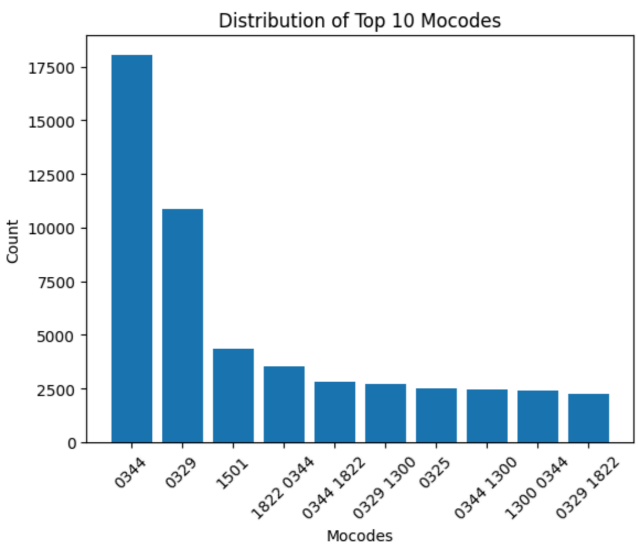
Missing value: NaN

```
df["Mocodes"].describe()
```

	Mocodes
count	840065
unique	309567
top	0344
freq	41144

dtype: object

```
mocodes_counts = df["Mocodes"].value_counts().nlargest(10)
plt.bar(mocodes_counts.index, mocodes_counts.values)
plt.xlabel("Mocodes")
plt.ylabel("Count")
plt.title("Distribution of Top 10 Mocodes")
plt.xticks(rotation=45)
plt.show()
```



Variable: Vict Age

Definition: Age of Victim

Creation

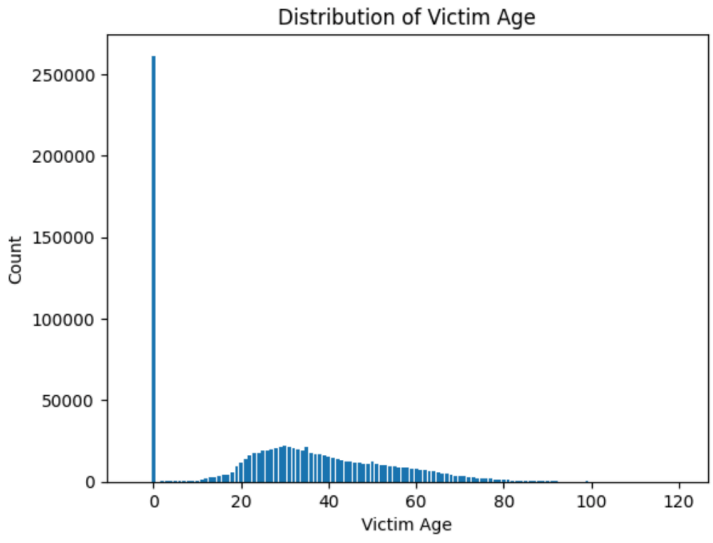
Provided by original Data.gov Dataset

Missing value: NaN

```
df["Vict Age"].describe()
```

	Vict Age
count	986500.000000
mean	29.045177
std	21.976666
min	-4.000000
25%	0.000000
50%	30.000000
75%	44.000000
max	120.000000

```
age_counts = df["Vict Age"].value_counts().sort_index()
plt.bar(age_counts.index, age_counts.values)
plt.xlabel("Victim Age")
plt.ylabel("Count")
plt.title("Distribution of Victim Age")
plt.show()
```



Variable: Vict Sex

Definition: Description of the victim's sex

Creation

Provided by original Data.gov Dataset

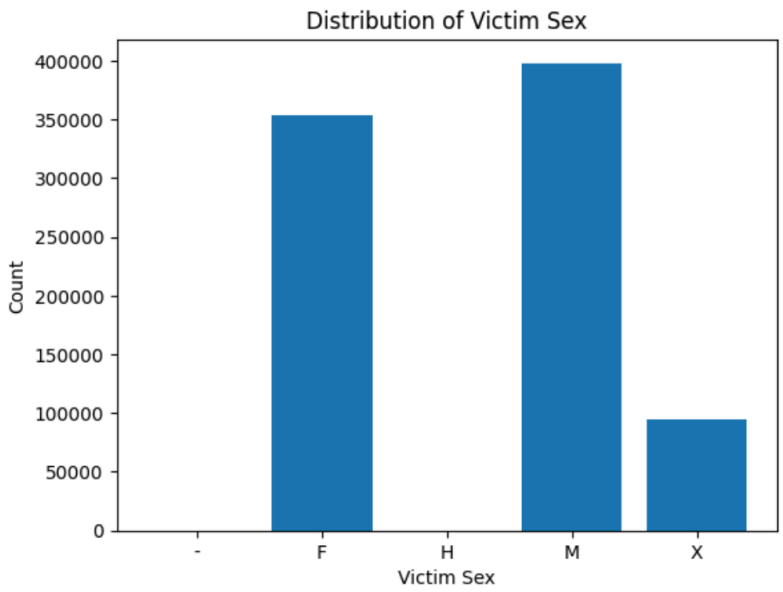
Missing value: NaN

```
df["Vict Sex"].describe()
```

	Vict Sex
count	846925
unique	5
top	M
freq	397948

dtype: object

```
sex_counts = df["Vict Sex"].value_counts().sort_index()
plt.bar(sex_counts.index, sex_counts.values)
plt.xlabel("Victim Sex")
plt.ylabel("Count")
plt.title("Distribution of Victim Sex")
plt.show()
```



Variable: Vict Descent

Definition: Description of the victim's ethnic background

Creation

Provided by original Data.gov Dataset

Missing value: NaN

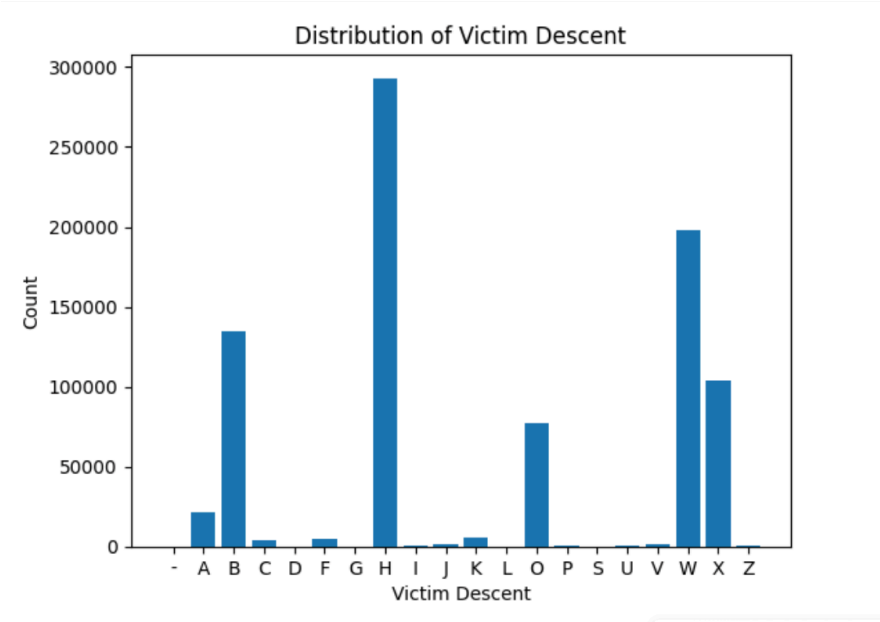
```
df["Vict Descent"].describe()
```

	Vict Descent
count	846914
unique	20
top	H
freq	293088

dtype: object

```
descent_counts = df["Vict Descent"].value_counts().sort_index()
```

```
plt.bar(descent_counts.index, descent_counts.values)
plt.xlabel("Victim Descent")
plt.ylabel("Count")
plt.title("Distribution of Victim Descent")
plt.show()
```



Variable: Premis Desc

Definition: Defines the type of structure, vehicle, or location where the crime took place

Creation

Provided by original Data.gov Dataset

Missing value: NaN

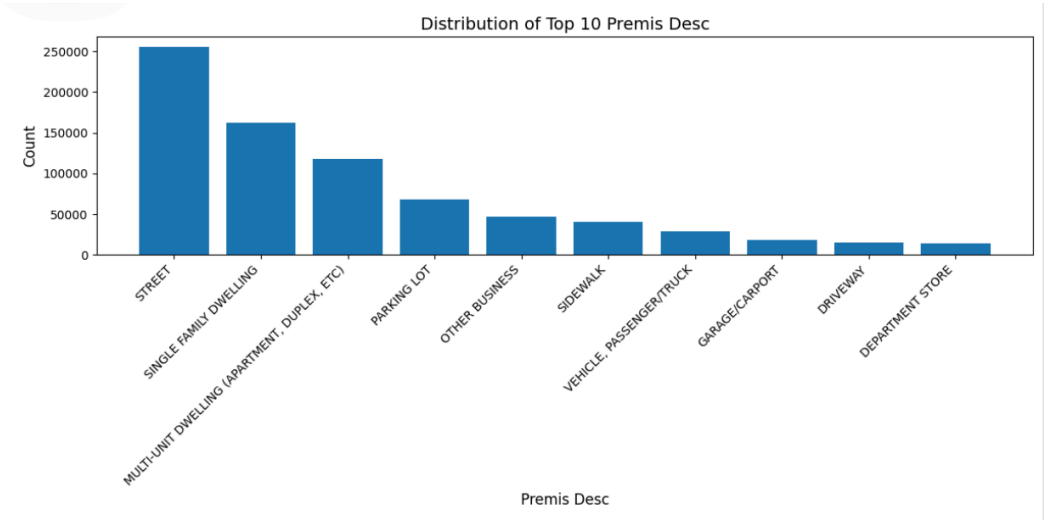
```
df["Premis Desc"].describe()
```

	Premis Desc
count	985915
unique	306
top	STREET
freq	254978

dtype: object

```
premis_counts = df["Premis Desc"].value_counts().nlargest(10)

plt.figure(figsize=(12, 6))  # Adjust figure size for better readability
plt.bar(premis_counts.index, premis_counts.values)
plt.xlabel("Premis Desc", fontsize=12)
plt.ylabel("Count", fontsize=12)
plt.title("Distribution of Top 10 Premis Desc", fontsize=14)
plt.xticks(rotation=45, ha="right", fontsize=10)  # Rotate x-axis labels
for readability
plt.tight_layout()  # Adjust layout to prevent overlapping labels
plt.show()
```



Variable: Weapon Desc

Definition: Type of weapon used in the crime

Creation

Provided by original Data.gov Dataset

Missing value: NaN

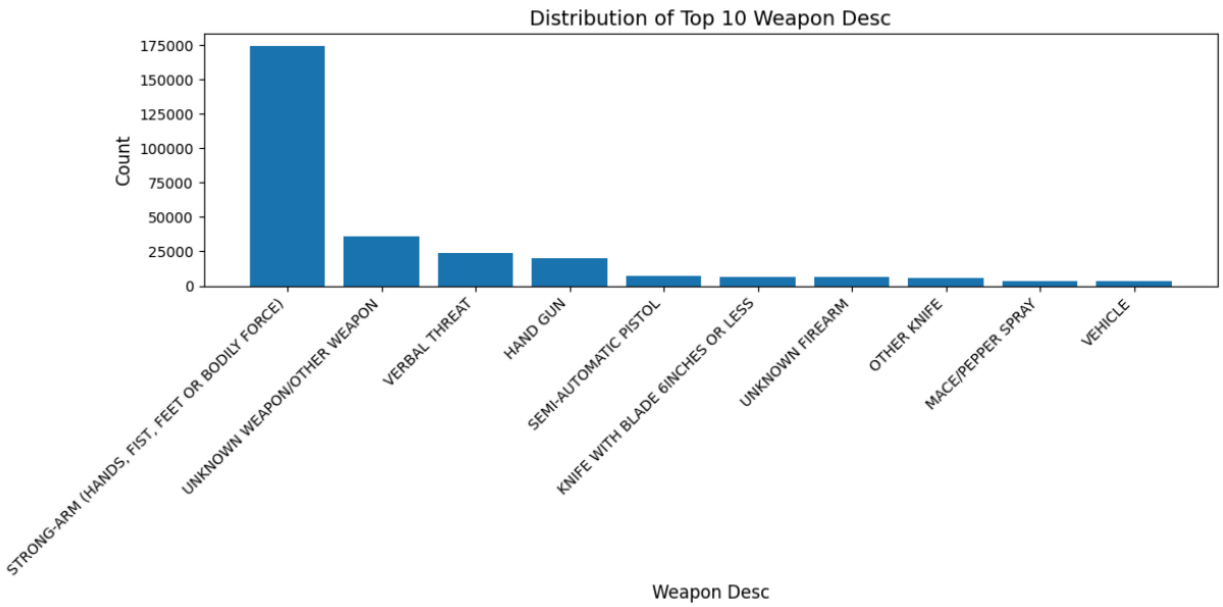
```
df["Weapon Desc"].describe()
```

	Weapon Desc
count	326368
unique	79
top	STRONG-ARM (HANDS, FIST, FEET OR BODILY FORCE)
freq	174484

dtype: object

```
weapon_counts = df["Weapon Desc"].value_counts().nlargest(10)

plt.figure(figsize=(12, 6))  # Adjust figure size for better readability
plt.bar(weapon_counts.index, weapon_counts.values)
plt.xlabel("Weapon Desc", fontsize=12)
plt.ylabel("Count", fontsize=12)
plt.title("Distribution of Top 10 Weapon Desc", fontsize=14)
plt.xticks(rotation=45, ha="right", fontsize=10)  # Rotate x-axis labels
for readability
plt.tight_layout()  # Adjust layout to prevent overlapping labels
plt.show()
```



Variable: Status Desc

Definition: Defines the status of the case

Creation

Provided by original Data.gov Dataset

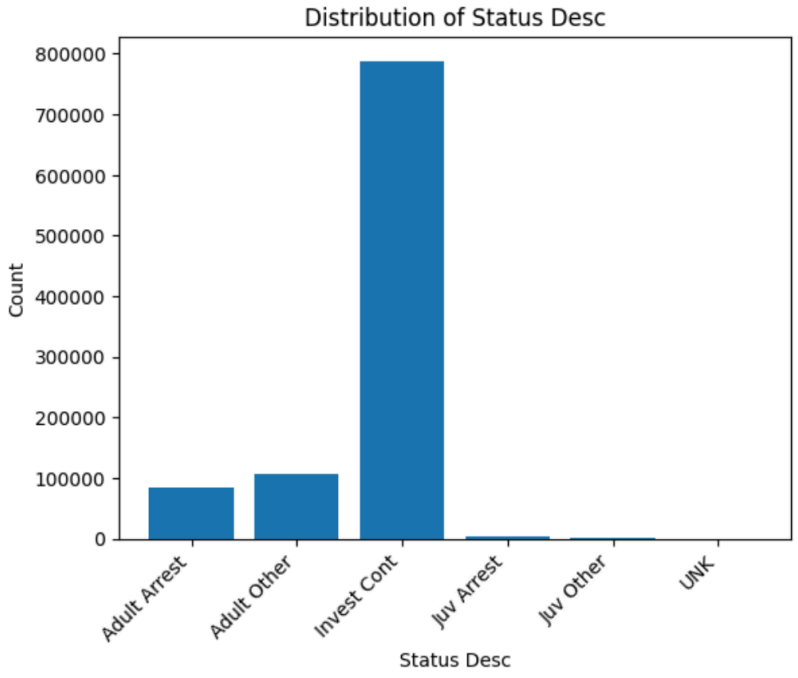
Missing value: NaN

```
df["Status Desc"].describe()
```

	Status Desc
count	986500
unique	6
top	Invest Cont
freq	788335

dtype: object

```
status_counts = df["Status Desc"].value_counts().sort_index()
plt.bar(status_counts.index, status_counts.values)
plt.xlabel("Status Desc")
plt.ylabel("Count")
plt.title("Distribution of Status Desc")
plt.xticks(rotation=45, ha="right")
plt.show()
```



Variable: Crm Cd 1

Definition: Indicates the crime committed. Crime Code 1 is the primary and most serious one.

Creation

Provided by original Data.gov Dataset

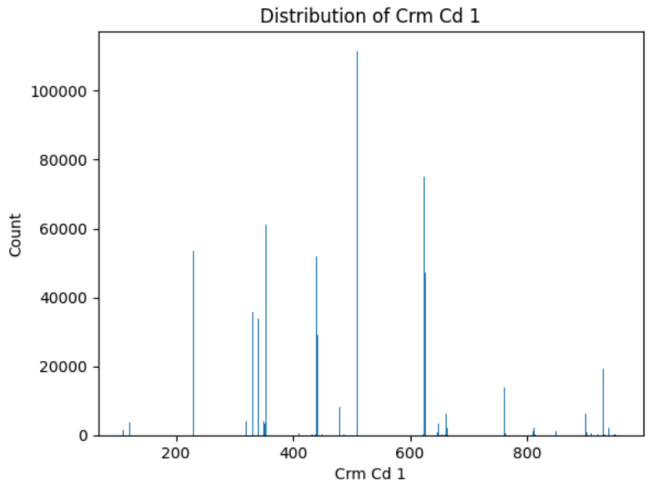
Missing value: NaN

```
df["Crm Cd 1"].describe()
```

	Crm Cd 1
count	986489.000000
mean	500.538333
std	205.891829
min	110.000000
25%	331.000000
50%	442.000000
75%	626.000000
max	956.000000

dtype: float64

```
crimeCode_counts = df["Crm Cd 1"].value_counts().sort_index()
plt.bar(crimeCode_counts.index, crimeCode_counts.values)
plt.xlabel("Crm Cd 1")
plt.ylabel("Count")
plt.title("Distribution of Crm Cd 1")
plt.show()
```



Variable: Crm Cd 2

Definition: Code 2, 3, and 4 are respectively less serious offenses.

Creation

Provided by original Data.gov Dataset

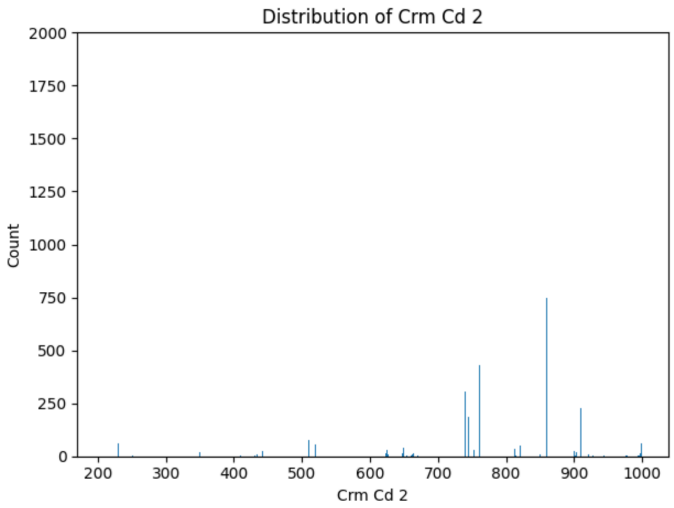
Missing value: NaN

```
df["Crm Cd 2"].describe()
```

	Crm Cd 2
count	68912.000000
mean	958.162091
std	110.250287
min	210.000000
25%	998.000000
50%	998.000000
75%	998.000000
max	999.000000

dtype: float64

```
crimeCode2_counts = df["Crm Cd 2"].value_counts().sort_index()
plt.figure(figsize=(12,6))
plt.bar(crimeCode2_counts.index, crimeCode2_counts.values)
plt.xlabel("Crm Cd 2")
plt.ylabel("Count")
plt.title("Distribution of Crm Cd 2")
plt.xticks(rotation=90, ha="right", fontsize=10)  # Rotate labels by 90 degrees
plt.tight_layout()
plt.show()
```



Variable: LOCATION

Definition: Street Address of the crime incident rounded to the nearest hundredth block to maintain anonymity

Creation

Provided by original Data.gov Dataset

Missing Values:

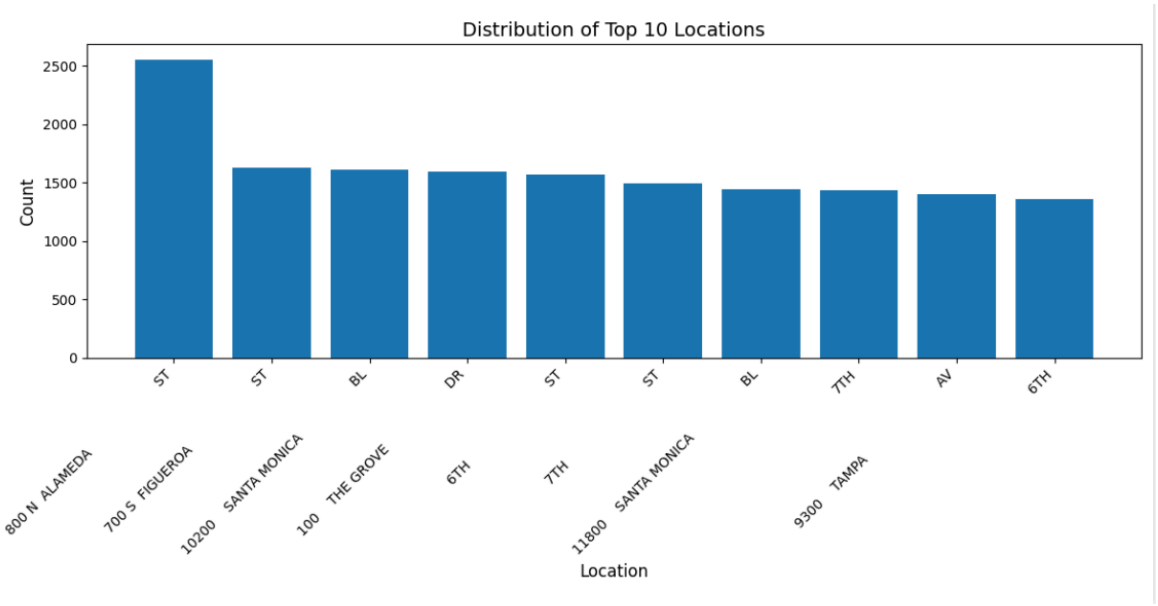
```
df["LOCATION"].describe()
```

	LOCATION
count	986500
unique	66322
top	800 N ALAMEDA ST
freq	2556

dtype: object

```
location_counts = df["LOCATION"].value_counts().nlargest(10)

plt.figure(figsize=(12, 6))  # Adjust figure size for better readability
plt.bar(location_counts.index, location_counts.values)
plt.xlabel("Location", fontsize=12)
plt.ylabel("Count", fontsize=12)
plt.title("Distribution of Top 10 Locations", fontsize=14)
plt.xticks(rotation=45, ha="right", fontsize=10)  # Rotate x-axis labels
for readability
plt.tight_layout()
plt.show()
```



Variable: LAT

Definition: Latitude of the Crime

Creation

Provided by original Data.gov Dataset

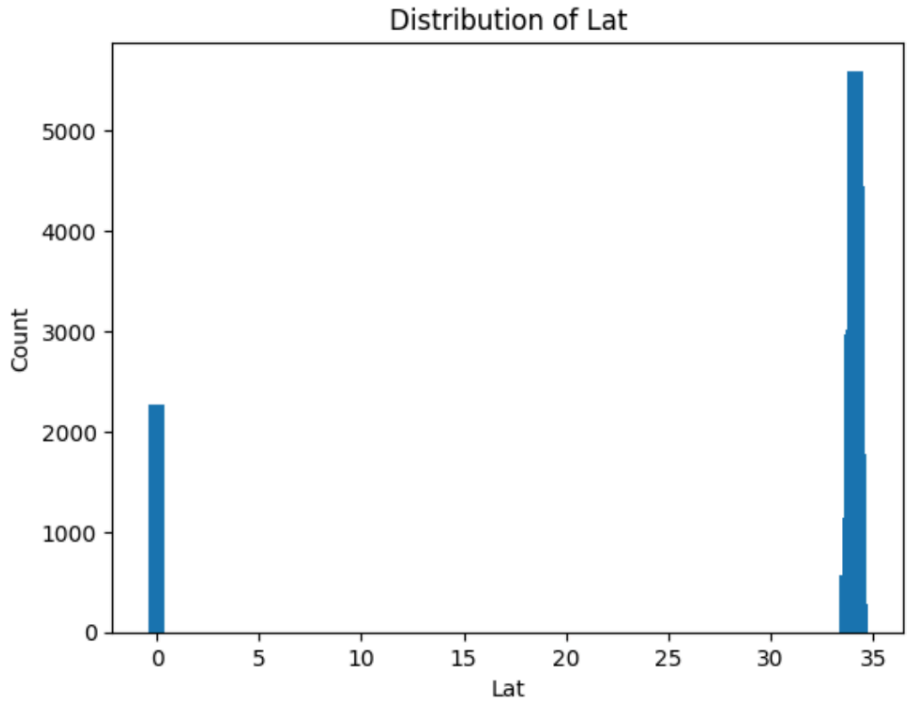
Missing Values: NaN

```
df["LAT"].describe()
```

	LAT
count	986500.000000
mean	33.996033
std	1.633543
min	0.000000
25%	34.014600
50%	34.058900
75%	34.164900
max	34.334300

dtype: float64

```
lat_counts = df["LAT"].value_counts().sort_index()
plt.bar(lat_counts.index, lat_counts.values)
plt.xlabel("Lat")
plt.ylabel("Count")
plt.title("Distribution of Lat")
plt.show()
```



Variable: LON

Definition: Longitude

Creation

Provided by original Data.gov Dataset

Missing Values: NaN

```
df["LON"].describe()
```

	LON
count	986500.000000
mean	-118.083281
std	5.661853
min	-118.667600
25%	-118.430500
50%	-118.322500
75%	-118.273900
max	0.000000

dtype: float64

```
lon_counts = df["LON"].value_counts().sort_index()
plt.bar(lon_counts.index, lon_counts.values)
plt.xlabel("Lon")
plt.ylabel("Count")
plt.title("Distribution of Lon")
plt.show()
```