```
In [1]: import pandas as pd
        import numpy as np
        import seaborn as sns
        import matplotlib.pyplot as plt
```

```
In [2]: import warnings
        warnings.filterwarnings("ignore")
```

```
In [3]: Data = pd.read_csv("C:\\Users\\DELL\\Desktop\\Data Analytics\\DataSets\\Crime_Dataset_India.csv")
```

1. NUMERICAL ANALYSIS

```
In [4]: Data.shape
```

```
Out[4]: (40160, 14)
```

```
In [5]: Data.columns.values
```

```
Out[5]: array(['Report Number', 'Date Reported', 'Date of Occurrence',
               'Time of Occurrence', 'City', 'Crime Code', 'Crime Description',
               'Victim Age', 'Victim Gender', 'Weapon Used', 'Crime Domain',
               'Police Deployed', 'Case Closed', 'Date Case Closed'], dtype=object)
```

```
In [6]: Data.head()
```

Out[6]:

	Report Number	Date Reported	Date of Occurrence	Time of Occurrence	City	Crime Code	Crime Description	Victim Age	Victim Gender	Weapon Used	Crime Domain	Police Deployed	Case Closed
0	1	02-01-2020 00:00	01-01-2020 00:00	01-01-2020 01:11	Ahmedabad	576	IDENTITY THEFT	16	M	Blunt Object	Violent Crime	13	No
1	2	01-01-2020 19:00	01-01-2020 01:00	01-01-2020 06:26	Chennai	128	HOMICIDE	37	M	Poison	Other Crime	9	No
2	3	02-01-2020 05:00	01-01-2020 02:00	01-01-2020 14:30	Ludhiana	271	KIDNAPPING	48	F	Blunt Object	Other Crime	15	No
3	4	01-01-2020 05:00	01-01-2020 03:00	01-01-2020 14:46	Pune	170	BURGLARY	49	F	Firearm	Other Crime	1	Yes
4	5	01-01-2020 21:00	01-01-2020 04:00	01-01-2020 16:51	Pune	421	VANDALISM	30	F	Other	Other Crime	18	Yes

In [7]:

```
Data.describe()
```

Out[7]:

	Report Number	Crime Code	Victim Age	Police Deployed
count	40160.000000	40160.000000	40160.00000	40160.000000
mean	20080.500000	349.360259	44.49126	10.006250
std	11593.337742	144.169205	20.22555	5.467951
min	1.000000	100.000000	10.00000	1.000000
25%	10040.750000	225.000000	27.00000	5.000000
50%	20080.500000	349.000000	44.00000	10.000000
75%	30120.250000	474.000000	62.00000	15.000000
max	40160.000000	599.000000	79.00000	19.000000

In [8]:

```
Data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 40160 entries, 0 to 40159
Data columns (total 14 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   Report Number       40160 non-null  int64 
 1   Date Reported       40160 non-null  object
 2   Date of Occurrence  40160 non-null  object
 3   Time of Occurrence  40160 non-null  object
 4   City                40160 non-null  object
 5   Crime Code          40160 non-null  int64 
 6   Crime Description   40160 non-null  object
 7   Victim Age          40160 non-null  int64 
 8   Victim Gender       40160 non-null  object
 9   Weapon Used         34370 non-null  object
 10  Crime Domain        40160 non-null  object
 11  Police Deployed     40160 non-null  int64 
 12  Case Closed         40160 non-null  object
 13  Date Case Closed    20062 non-null  object
dtypes: int64(4), object(10)
memory usage: 4.3+ MB
```

In [9]:

```
Data.isnull().sum()
```

Out[9]:

```
Report Number             0
Date Reported             0
Date of Occurrence        0
Time of Occurrence        0
City                      0
Crime Code                0
Crime Description         0
Victim Age                0
Victim Gender             0
Weapon Used            5790
Crime Domain              0
Police Deployed           0
Case Closed               0
Date Case Closed      20098
dtype: int64
```

In [10]:

```
Data['Weapon Used'].fillna("Unknown")
```

Out[10]:

```
0        Blunt Object
1              Poison
2        Blunt Object
3             Firearm
4               Other
             ...
40155         Firearm
40156         Unknown
40157           Other
40158    Blunt Object
40159          Poison
Name: Weapon Used, Length: 40160, dtype: object
```

2. UNIVARIATE ANALYSIS

In [11]:

```
Data['Date of Occurrence'] = pd.to_datetime(Data['Date of Occurrence'], errors='coerce')
Data['Year of Occurrence'] = Data['Date of Occurrence'].dt.year
year = Data['Year of Occurrence'].value_counts().sort_index()
```

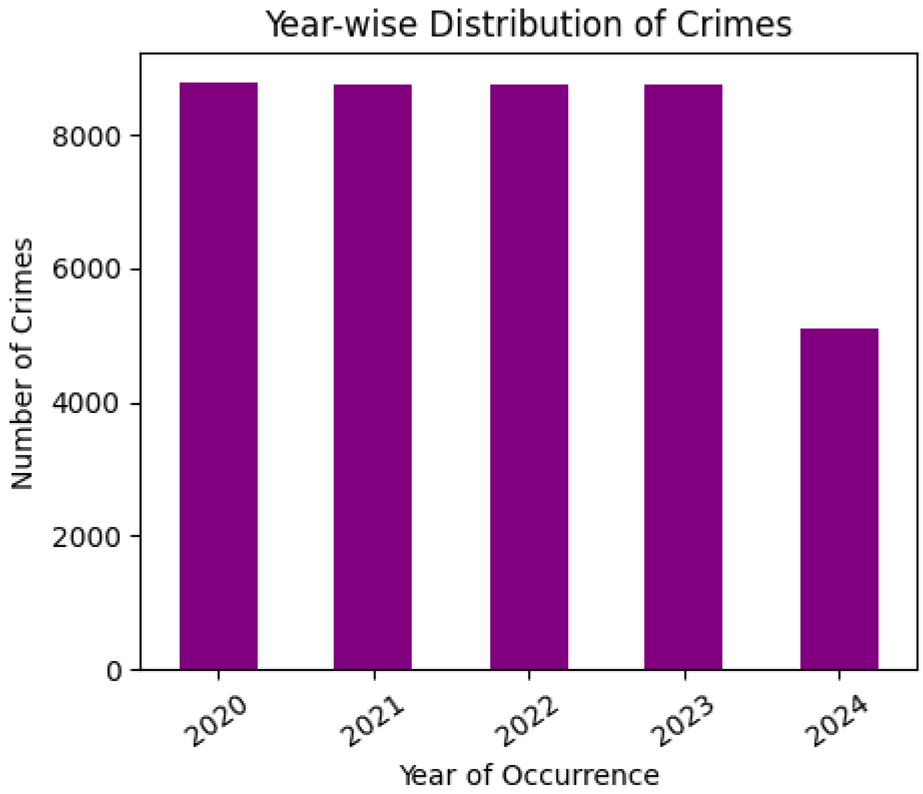
In [12]:

```
print(year)
```

```
Year of Occurrence
2020    8784
2021    8760
2022    8760
2023    8760
2024    5096
Name: count, dtype: int64
```

In [13]:

```
plt.figure(figsize=(5,4))
year.plot(kind='bar', color='purple')

plt.title('Year-wise Distribution of Crimes')
plt.xlabel('Year of Occurrence')
plt.ylabel('Number of Crimes')

plt.xticks(rotation=35)
plt.show()
```



In [14]:

```
print(Data['Time of Occurrence'].dtype)
```

```
object
```

In [15]:

```
print(Data['Time of Occurrence'].head(10))
```

```
0    01-01-2020 01:11
1    01-01-2020 06:26
2    01-01-2020 14:30
3    01-01-2020 14:46
4    01-01-2020 16:51
5    01-01-2020 17:09
6    01-01-2020 14:08
7    02-01-2020 06:33
8    02-01-2020 06:34
9    01-01-2020 17:50
Name: Time of Occurrence, dtype: object
```

In [16]:

```
Data['Time of Occurrence'] = pd.to_datetime(Data['Time of Occurrence'], format='%d-%m-%Y %H:%M')
Data['Hour'] = Data['Time of Occurrence'].dt.hour
```

In [17]:

```
Data['Hour'].describe()
```

Out[17]:

```
count    40160.000000
mean        11.496564
std          6.901710
min          0.000000
25%          5.000000
50%         11.000000
75%         17.000000
max         23.000000
Name: Hour, dtype: float64
```

In [18]:

```
Hours = Data['Hour'].value_counts().sort_index()
```

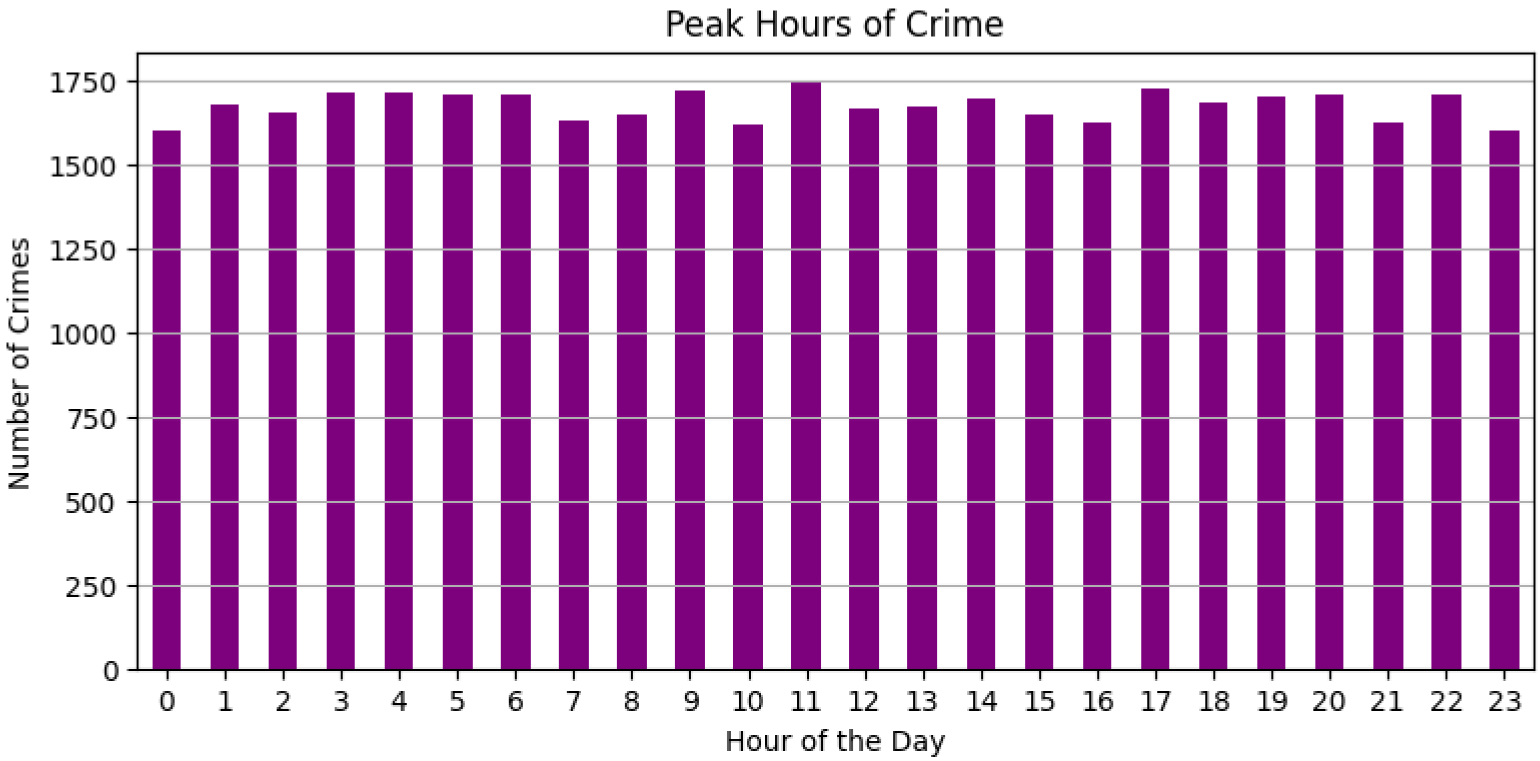
In [19]:

```
print(Hours)
```

```
Hour
0     1599
1     1674
2     1650
3     1712
4     1713
5     1707
6     1708
7     1626
8     1644
9     1719
10    1619
11    1745
12    1666
13    1671
14    1693
15    1649
16    1622
17    1721
18    1684
19    1701
20    1707
21    1624
22    1707
23    1599
Name: count, dtype: int64
```

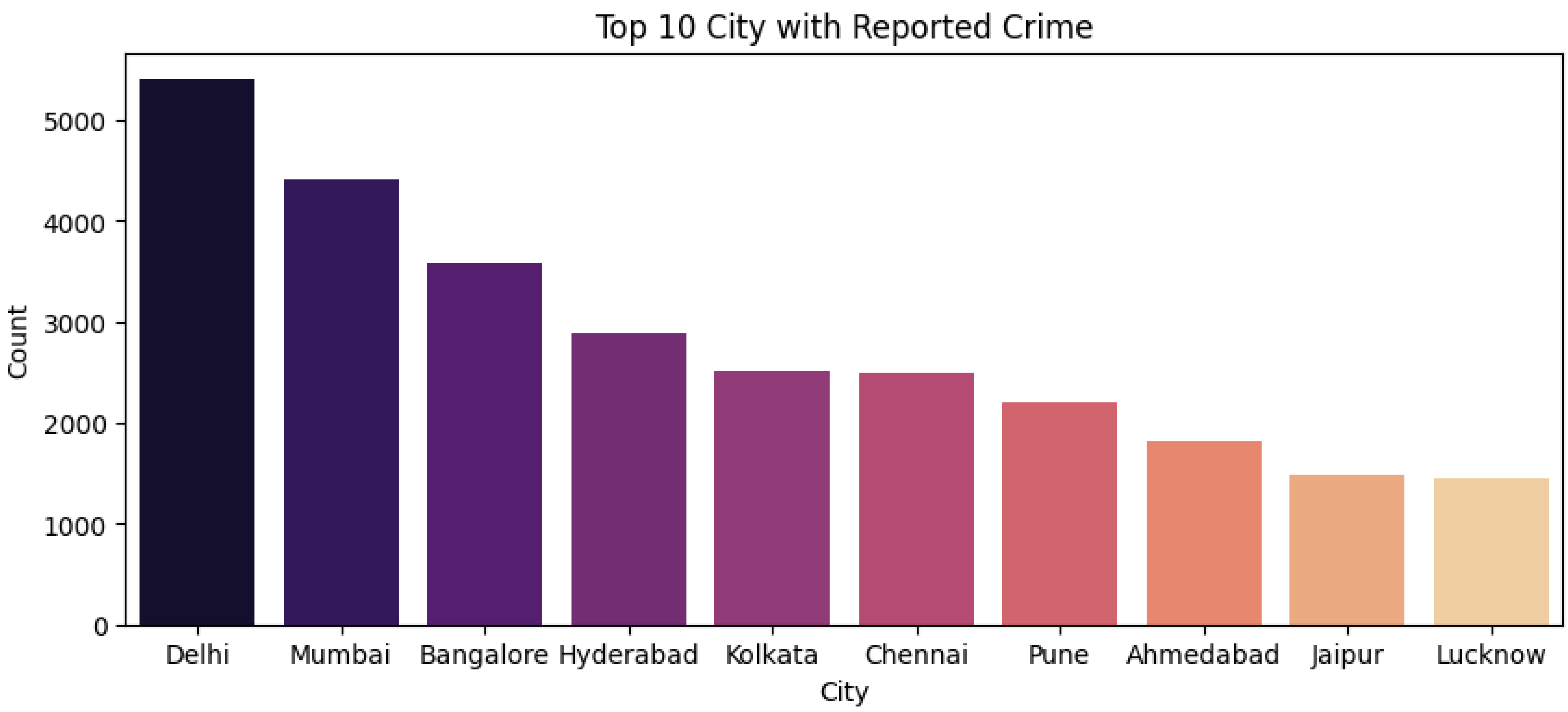
In [20]:

```
plt.figure(figsize=(9, 4))
Hours.plot(kind='bar', color='purple')
plt.title('Peak Hours of Crime')
plt.xlabel('Hour of the Day')
plt.ylabel('Number of Crimes')
plt.xticks(rotation=0)
plt.grid(axis='y')
plt.show()
```



```
In [21]: top_10_cities = Data['City'].value_counts().head(10).reset_index()

         plt.figure(figsize=(10,4))
         sns.barplot(top_10_cities, x='City',y='count', palette='magma')
         plt.title('Top 10 City with Reported Crime')
         plt.xlabel('City')
         plt.ylabel('Count')
```

```
Out[21]: Text(0, 0.5, 'Count')
```



In [22]:

```
#Top - 5 Cities with highest crime rates.
CrimeRate = Data['City'].value_counts()
Cities = CrimeRate.head(5)
```

In [23]:

```
print(Cities)
```

```
City
Delhi        5400
Mumbai       4415
Bangalore    3588
Hyderabad    2881
Kolkata      2518
Name: count, dtype: int64
```

In [24]:

```
Data['Victim Age'].max()
```

Out[24]:

```
np.int64(79)
```

In [25]:

```
Data['Victim Age'].min()
```

Out[25]:

```
np.int64(10)
```

In [26]:

```
labels = ['10-20', '20-30', '30-40', '40-50', '50-60', '60-70', '70-80']
bins = [10, 20, 30, 40, 50, 60, 70, 80,]
Data['Age Groups'] = pd.cut(Data['Victim Age'], bins, labels = labels, include_lowest = True)
Data[['Victim Age', 'Age Groups']].head(5)
```

Out[26]:

	Victim Age	Age Groups
0	16	10-20
1	37	30-40
2	48	40-50
3	49	40-50
4	30	20-30

In [27]:

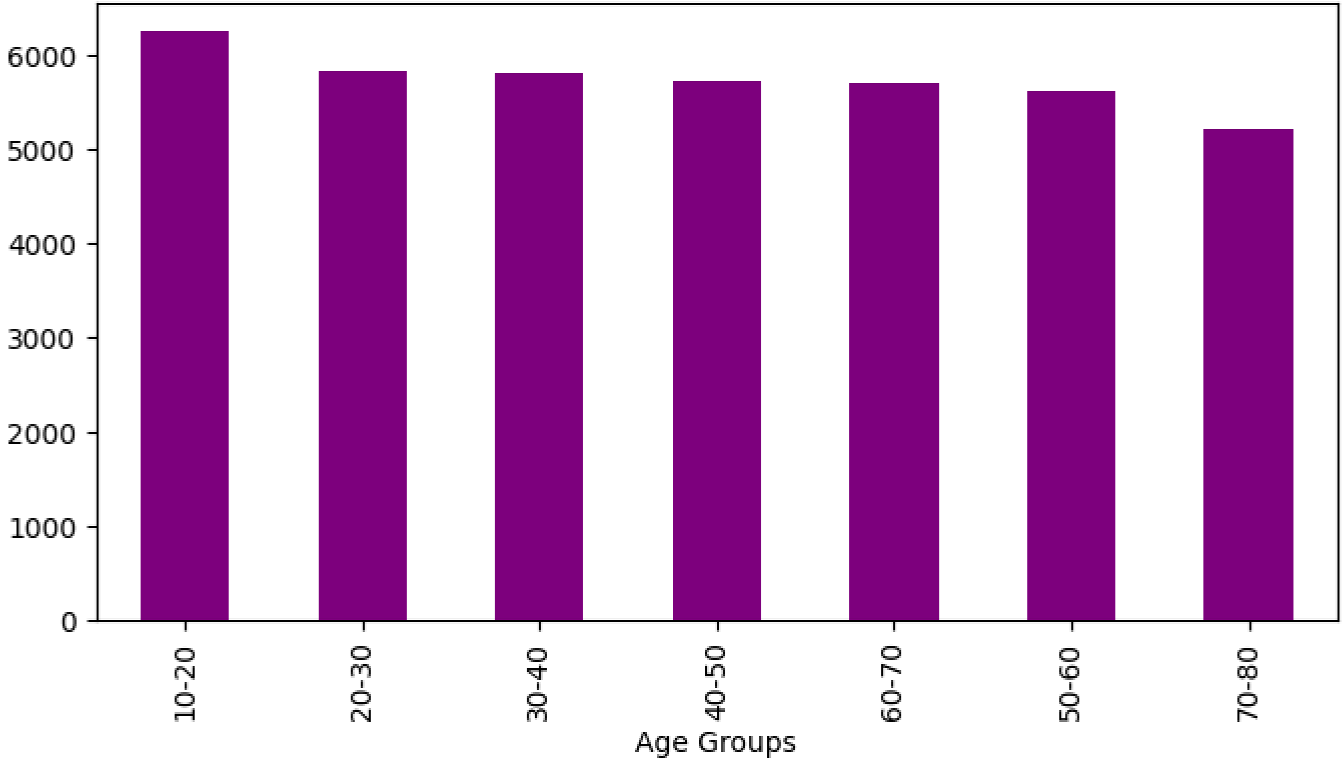
```
def add_labels(x,y):
    for i in range(len(x)):
        plt.text(i, y[i], y[i], ha='center')

Data['Age Groups'].value_counts()
```

Out[27]:

```
Age Groups
10-20    6246
20-30    5836
30-40    5807
40-50    5727
60-70    5709
50-60    5620
70-80    5215
Name: count, dtype: int64
```

```
In [28]: Data['Age Groups'].value_counts().plot(kind='bar', figsize=(8,4), color = 'purple')
```

```
Out[28]: <Axes: xlabel='Age Groups'>
```



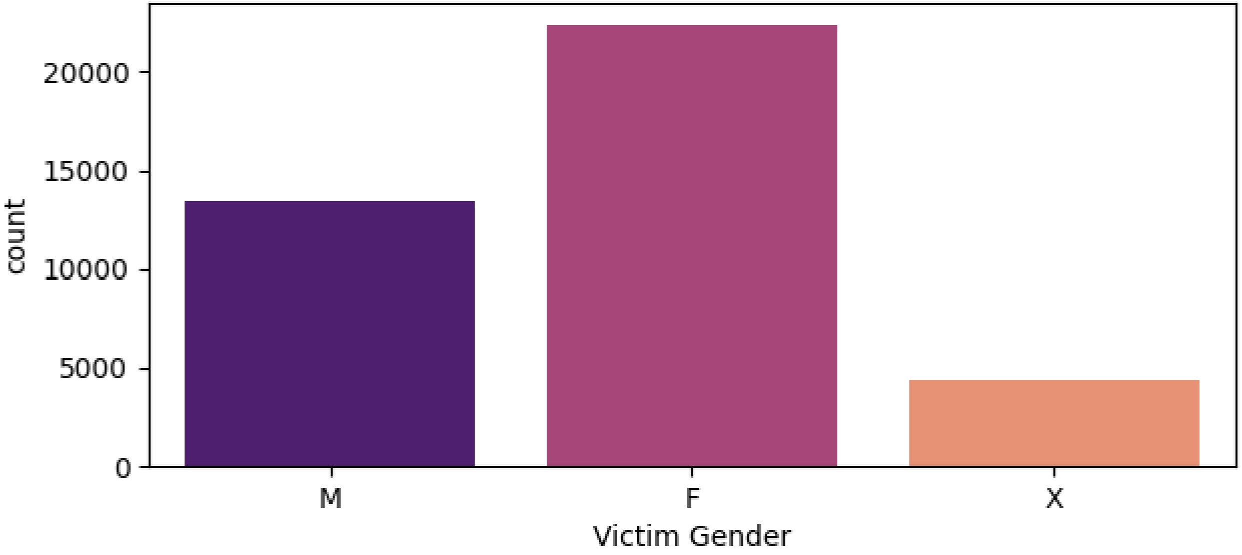
```
In [29]: Data['Victim Gender'].value_counts()
```

```
Out[29]: Victim Gender
         F    22423
         M    13405
         X     4332
         Name: count, dtype: int64
```

The **letter X** is used as a gender identity option to indicate that someone's gender is not exclusively male or female. It's an umbrella term that can include people who identify as nonbinary, agender, genderqueer, gender fluid, or bigender

In [30]:

```
plt.figure(figsize = (7,3))
sns.countplot(data= Data, x = 'Victim Gender', palette = 'magma')
plt.show()
```

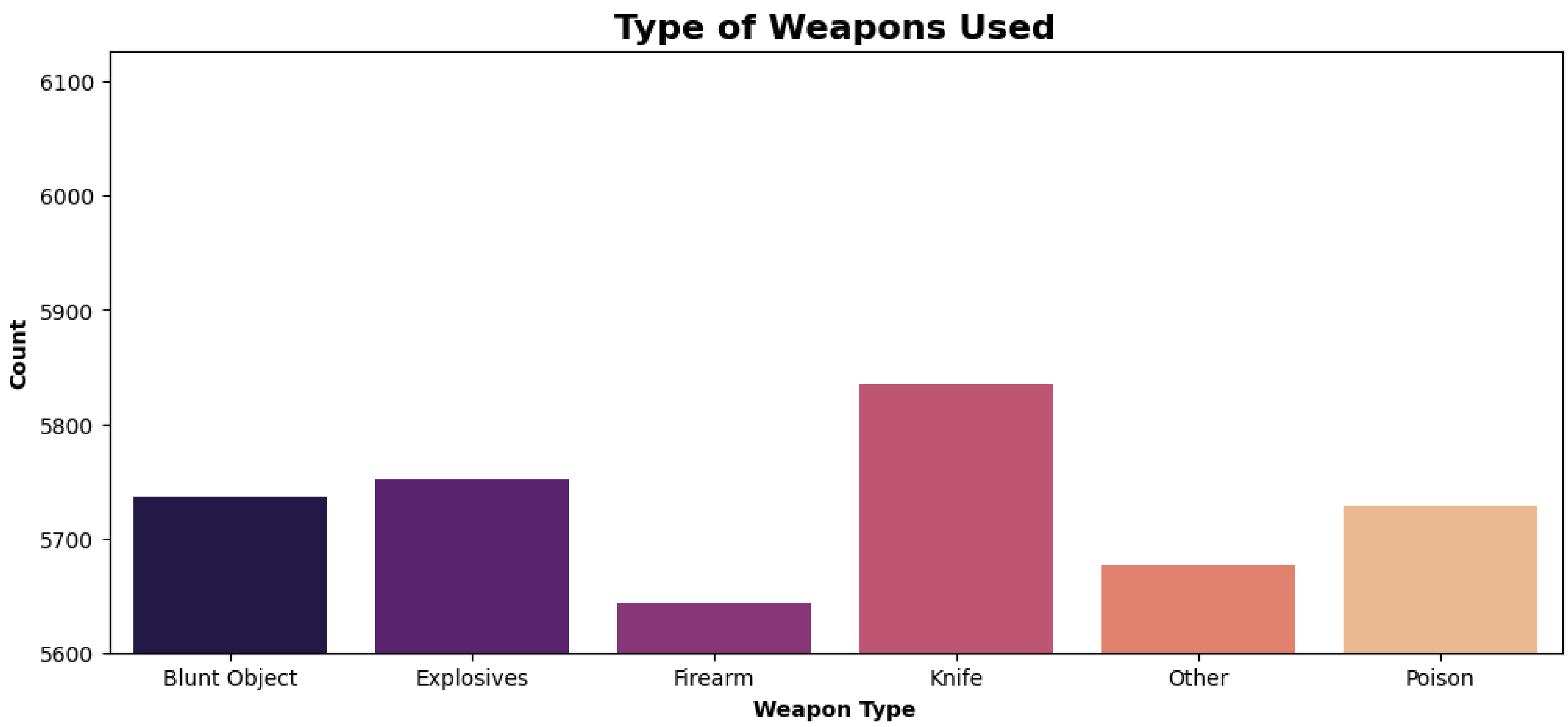


In [31]:

```
# count of type of weapons used
type_of_weapons_used = Data['Weapon Used'].value_counts().reset_index().sort_values(by='Weapon Used')

plt.figure(figsize=(12,5))
sns.barplot(type_of_weapons_used, x='Weapon Used', y='count', palette='magma').set_ylim(5600)
plt.title("Type of Weapons Used",fontsize=16, fontweight='bold')
plt.xlabel('Weapon Type',fontweight='bold')
plt.ylabel('Count',fontweight='bold')
```

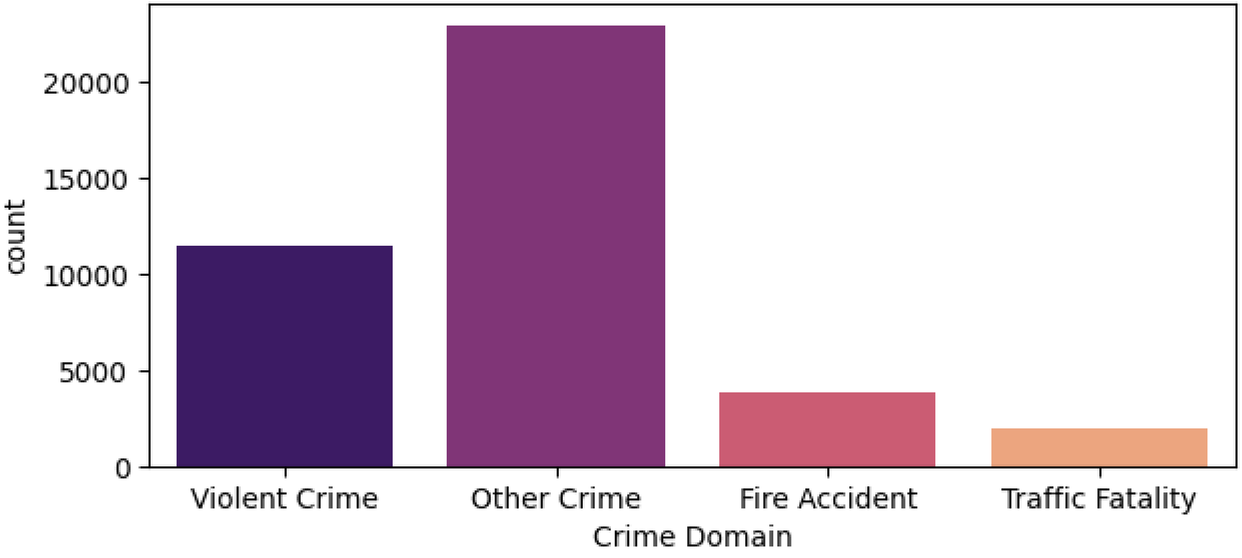
Out[31]:

```
Text(0, 0.5, 'Count')
```



In [32]:

```
plt.figure(figsize = (7,3))
sns.countplot(data= Data, x = 'Crime Domain', palette = 'magma')
plt.show()
```



In [33]:

```
Data['Crime Description'].unique()
```

Out[33]:

```
array(['IDENTITY THEFT', 'HOMICIDE', 'KIDNAPPING', 'BURGLARY',
       'VANDALISM', 'ASSAULT', 'VEHICLE - STOLEN', 'COUNTERFEITING',
       'EXTORTION', 'PUBLIC INTOXICATION', 'FRAUD', 'SEXUAL ASSAULT',
       'DRUG OFFENSE', 'ARSON', 'CYBERCRIME', 'TRAFFIC VIOLATION',
       'SHOPLIFTING', 'ILLEGAL POSSESSION', 'FIREARM OFFENSE', 'ROBBERY',
       'DOMESTIC VIOLENCE'], dtype=object)
```

In [34]:

```
Data['Case Closed'].value_counts()
```

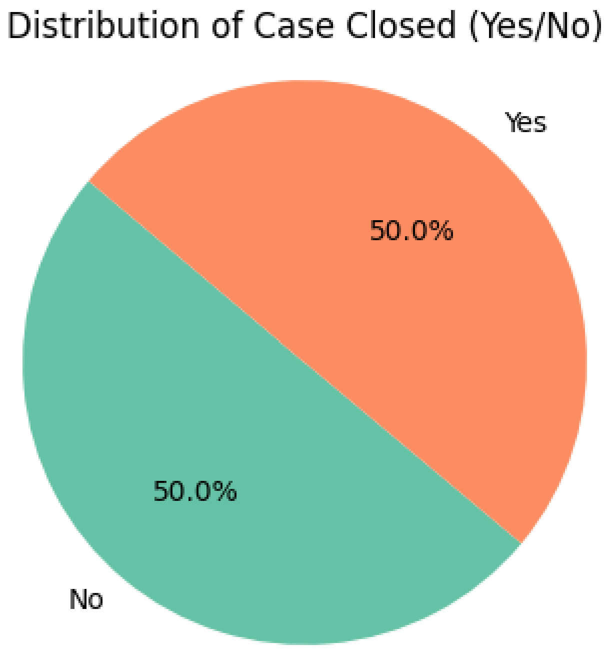
Out[34]:

```
Case Closed
No     20098
Yes    20062
Name: count, dtype: int64
```

In [35]:

```
case_closed_counts = Data['Case Closed'].value_counts()

# Create the pie chart
labels = case_closed_counts.index
sizes = case_closed_counts.values
```

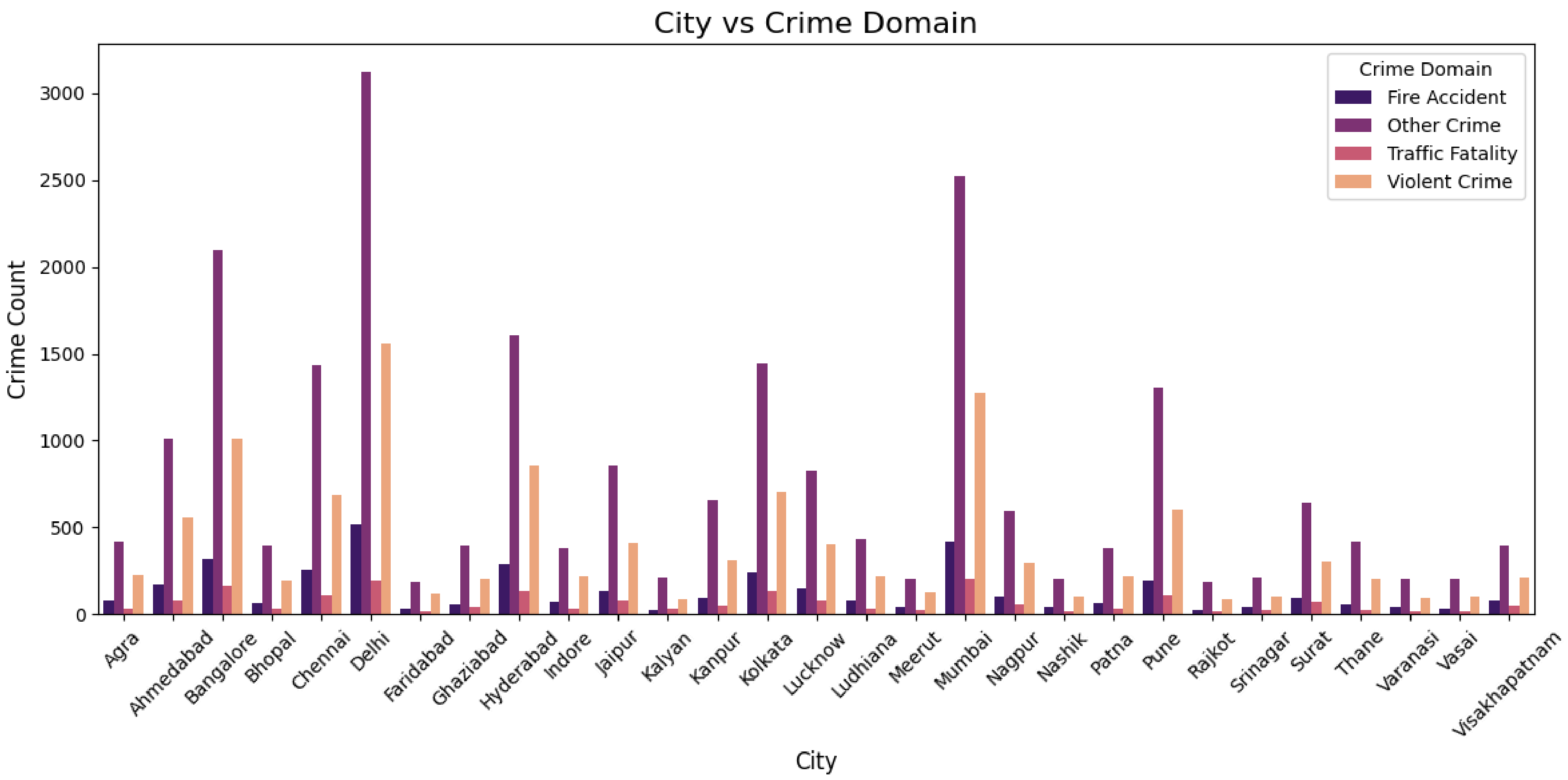
```
plt.figure(figsize=(4, 4))
plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=140, colors=['#66c2a5', '#fc8d62'])
plt.title('Distribution of Case Closed (Yes/No)')
plt.axis('equal')  # Equal aspect ratio ensures that pie chart is circular.
plt.show()
```



3. BIVARIATE ANALYSIS

In [47]:

```
# City VS. Crime Domain
crime_by_city_domain = Data.groupby(['City', 'Crime Domain']).size().reset_index(name='Crime Count')
plt.figure(figsize=(12, 6))
sns.barplot(x='City', y='Crime Count', hue='Crime Domain', data=crime_by_city_domain, palette = 'magma')
plt.title('City vs Crime Domain', fontsize=16)
plt.xlabel('City', fontsize=12)
plt.ylabel('Crime Count', fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



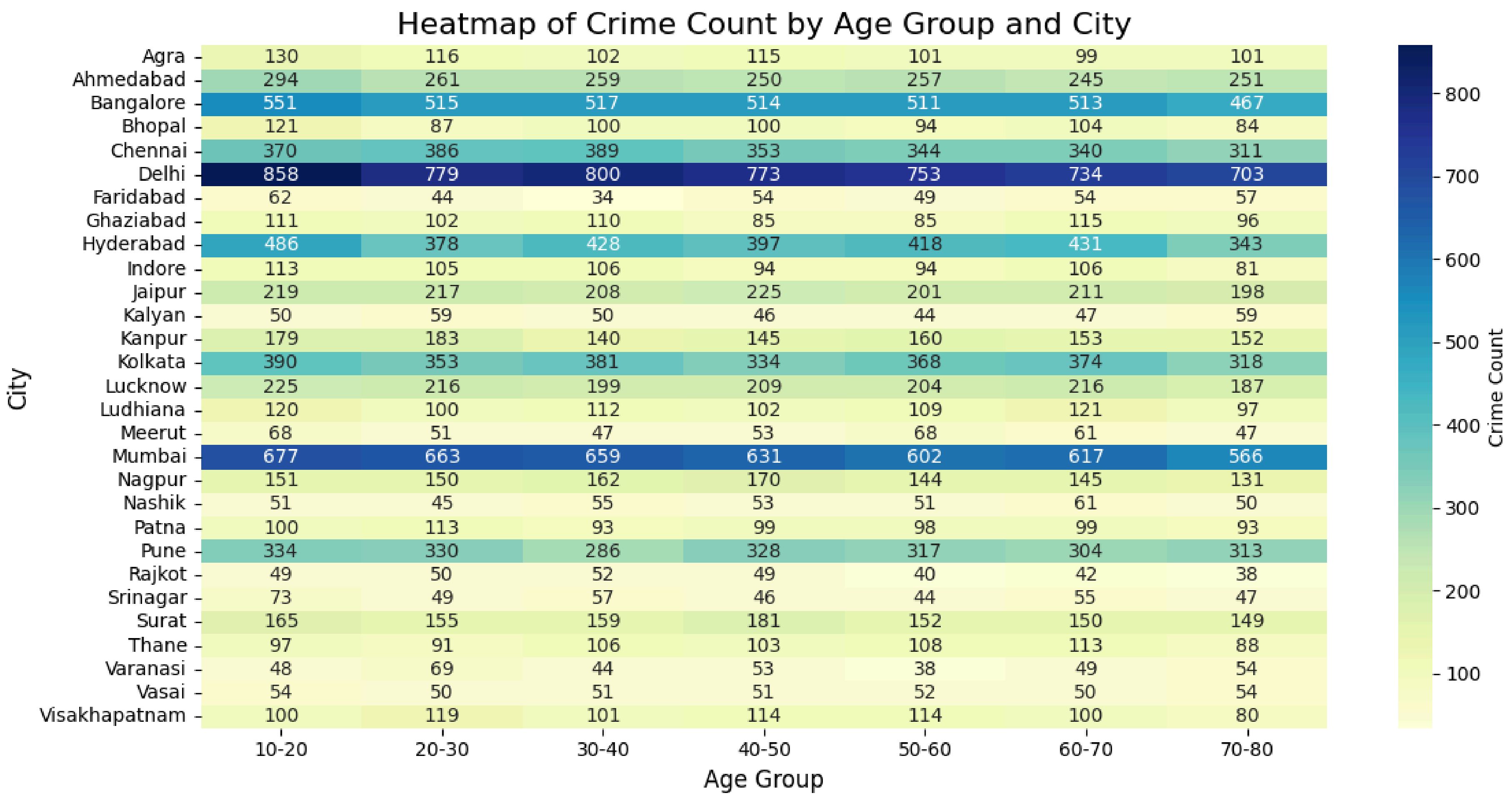
In [50]:

```
age_group_counts = Data.groupby(['City', 'Age Groups']).size().reset_index(name='Crime Count')
print(age_group_counts)
```

```
              City Age Groups  Crime Count
0             Agra      10-20          130
1             Agra      20-30          116
2             Agra      30-40          102
3             Agra      40-50          115
4             Agra      50-60          101
..             ...        ...          ...
198  Visakhapatnam      30-40          101
199  Visakhapatnam      40-50          114
200  Visakhapatnam      50-60          114
201  Visakhapatnam      60-70          100
202  Visakhapatnam      70-80           80

[203 rows x 3 columns]
```

In [54]:

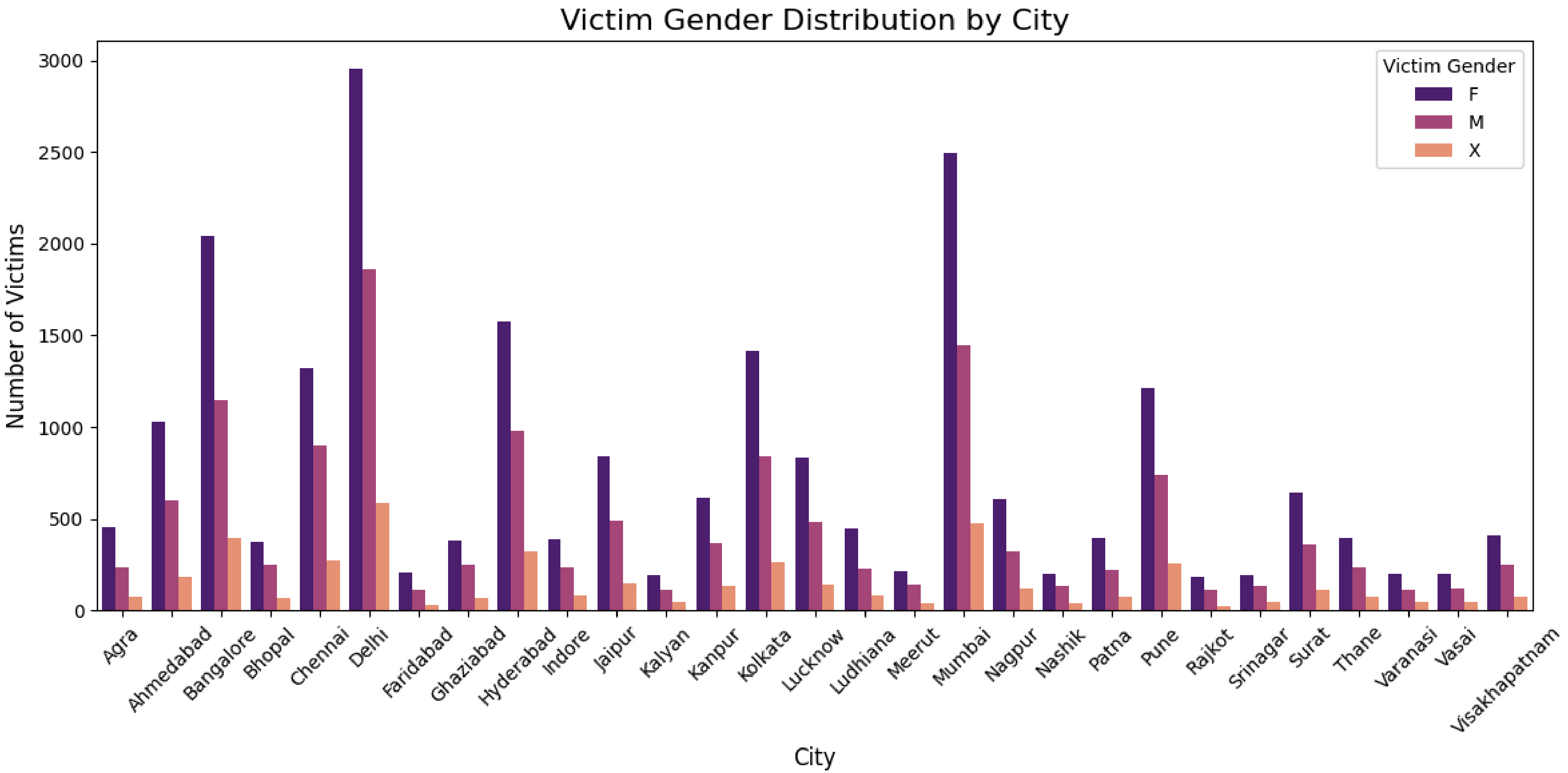
```
heatmap_data = age_group_counts.pivot(index='City', columns='Age Groups', values='Crime Count').fillna(0)

plt.figure(figsize=(12, 6))
sns.heatmap(heatmap_data, annot=True, fmt='g', cmap='YlGnBu', cbar_kws={'label': 'Crime Count'})
plt.title('Heatmap of Crime Count by Age Group and City', fontsize=16)
plt.xlabel('Age Group', fontsize=12)
plt.ylabel('City', fontsize=12)
plt.tight_layout()
plt.show()
```



City	10-20	20-30	30-40	40-50	50-60	60-70	70-80
Agra	130	116	102	115	101	99	101
Ahmedabad	294	261	259	250	257	245	251
Bangalore	551	515	517	514	511	513	467
Bhopal	121	87	100	100	94	104	84
Chennai	370	386	389	353	344	340	311
Delhi	858	779	800	773	753	734	703
Faridabad	62	44	34	54	49	54	57
Ghaziabad	111	102	110	85	85	115	96
Hyderabad	486	378	428	397	418	431	343
Indore	113	105	106	94	94	106	81
Jaipur	219	217	208	225	201	211	198
Kalyan	50	59	50	46	44	47	59
Kanpur	179	183	140	145	160	153	152
Kolkata	390	353	381	334	368	374	318
Lucknow	225	216	199	209	204	216	187
Ludhiana	120	100	112	102	109	121	97
Meerut	68	51	47	53	68	61	47
Mumbai	677	663	659	631	602	617	566
Nagpur	151	150	162	170	144	145	131
Nashik	51	45	55	53	51	61	50
Patna	100	113	93	99	98	99	93
Pune	334	330	286	328	317	304	313
Rajkot	49	50	52	49	40	42	38
Srinagar	73	49	57	46	44	55	47
Surat	165	155	159	181	152	150	149
Thane	97	91	106	103	108	113	88
Varanasi	48	69	44	53	38	49	54
Vasai	54	50	51	51	52	50	54
Visakhapatnam	100	119	101	114	114	100	80

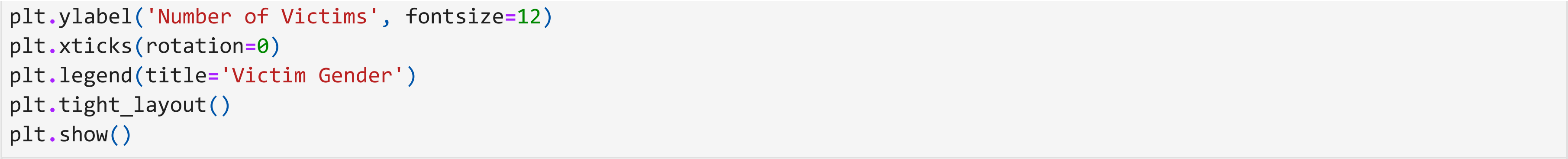
In [57]:

```
gender_city_counts = Data.groupby(['City', 'Victim Gender']).size().reset_index(name='Victim Count')

# Set the figure size
plt.figure(figsize=(12, 6))

# Create a bar plot
sns.barplot(x='City', y='Victim Count', hue='Victim Gender', data=gender_city_counts, palette='magma')

# Customize the plot
plt.title('Victim Gender Distribution by City', fontsize=16)
plt.xlabel('City', fontsize=12)
plt.ylabel('Number of Victims', fontsize=12)
plt.xticks(rotation=45)  # Rotate city names for better visibility
plt.legend(title='Victim Gender')
plt.tight_layout()  # Adjust layout to fit everything nicely
plt.show()
```

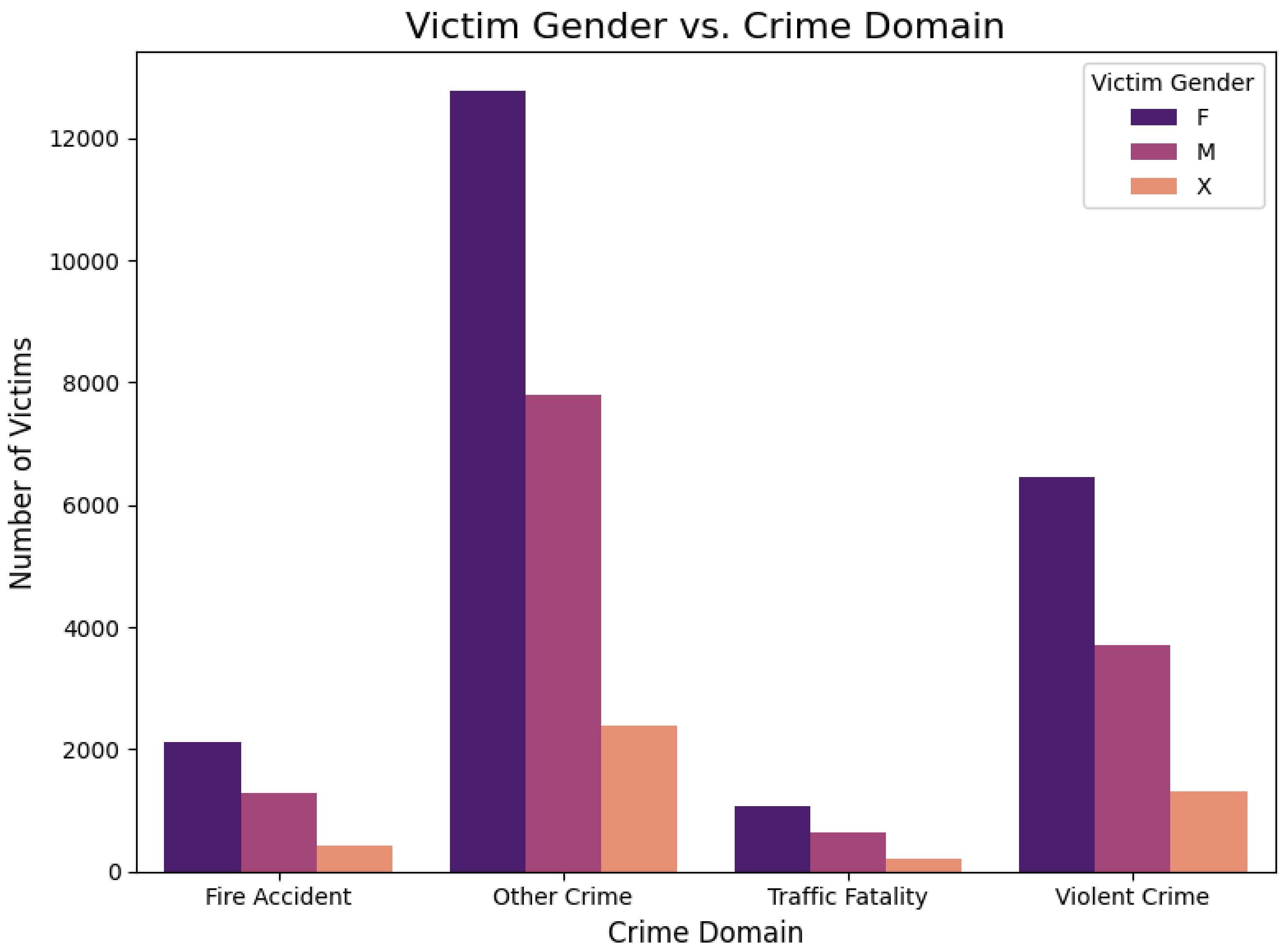


In [61]:

```
crime_gender_counts = Data.groupby(['Victim Gender', 'Crime Domain']).size().reset_index(name='Count')

# Create a bar plot
plt.figure(figsize=(8, 6))
sns.barplot(x='Crime Domain', y='Count', hue='Victim Gender', data=crime_gender_counts, palette='magma')
plt.title('Victim Gender vs. Crime Domain', fontsize=16)
plt.xlabel('Crime Domain', fontsize=12)
```

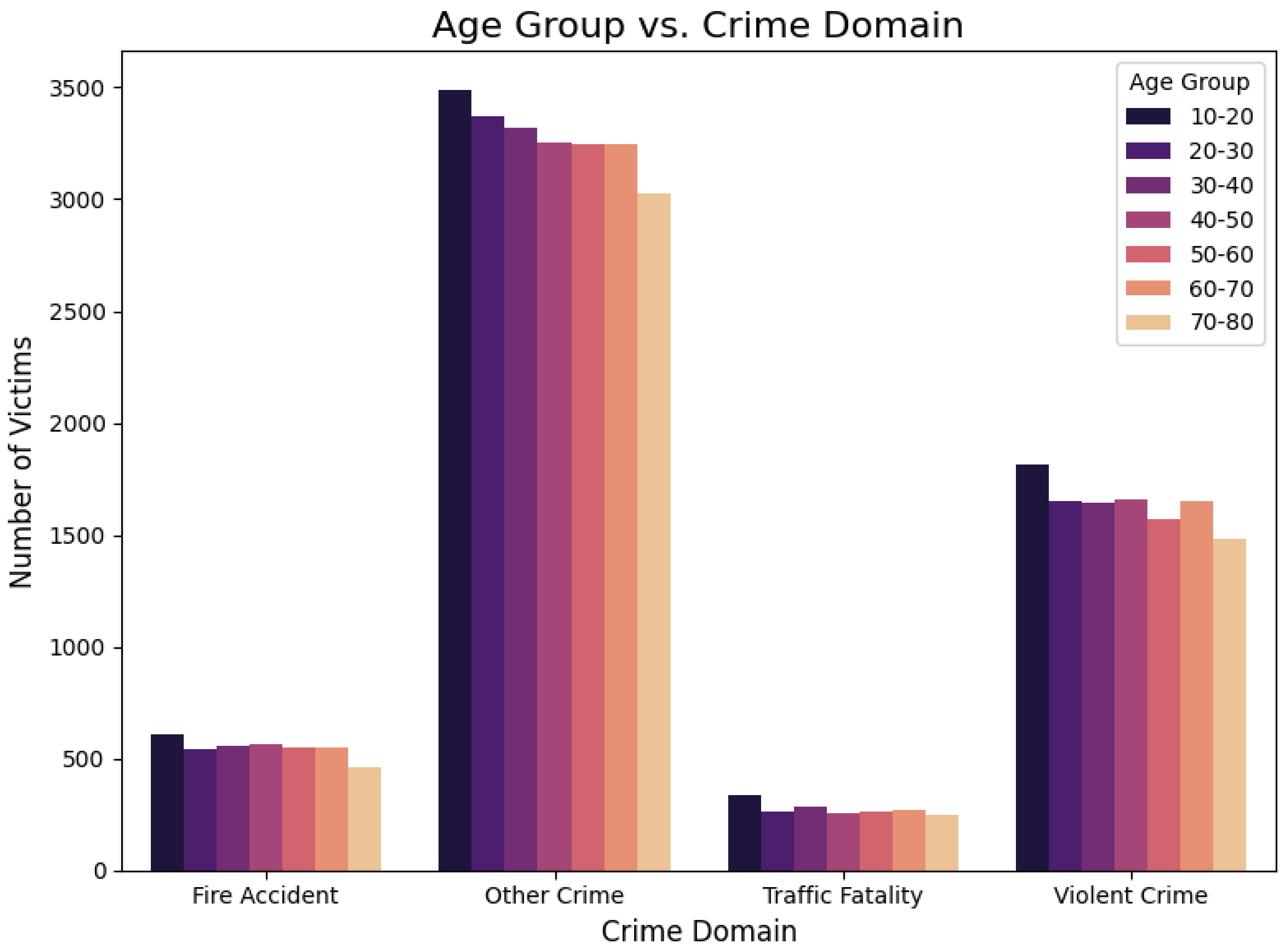


```
plt.ylabel('Number of Victims', fontsize=12)
plt.xticks(rotation=0)
plt.legend(title='Victim Gender')
plt.tight_layout()
plt.show()
```



In [65]:

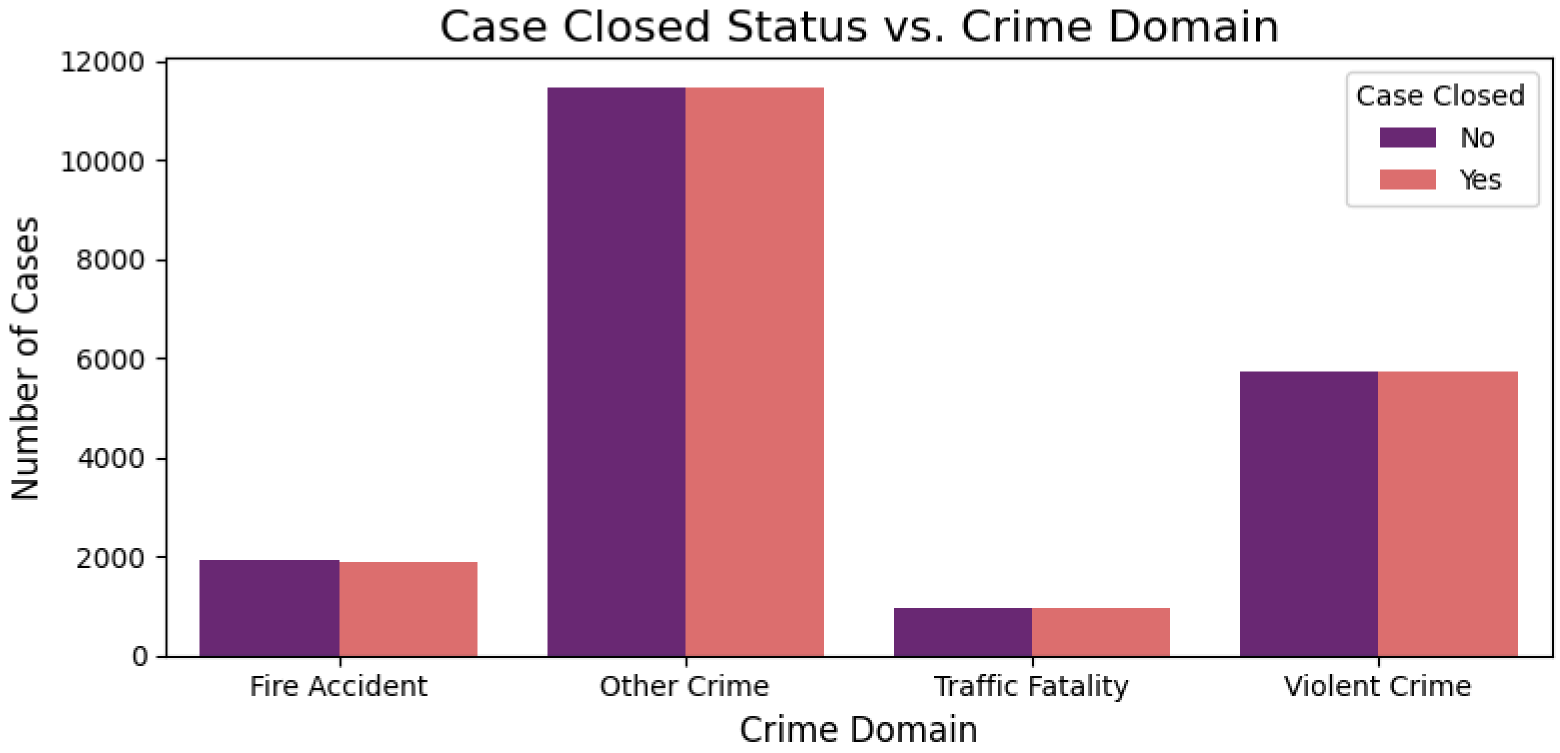
```
crime_age_counts = Data.groupby(['Age Groups', 'Crime Domain']).size().reset_index(name='Count')

# Create a bar plot
plt.figure(figsize=(8, 6))
sns.barplot(x='Crime Domain', y='Count', hue='Age Groups', data=crime_age_counts, palette='magma')
plt.title('Age Group vs. Crime Domain', fontsize=16)
plt.xlabel('Crime Domain', fontsize=12)
plt.ylabel('Number of Victims', fontsize=12)
plt.xticks(rotation=0)
plt.legend(title='Age Group')
plt.tight_layout()
plt.show()
```



In [68]:

```
case_closed_counts = Data.groupby(['Crime Domain', 'Case Closed']).size().reset_index(name='Count')

# Create a bar plot
plt.figure(figsize=(8, 4))
sns.barplot(x='Crime Domain', y='Count', hue='Case Closed', data=case_closed_counts, palette='magma')
```

```
plt.title('Case Closed Status vs. Crime Domain', fontsize=16)
plt.xlabel('Crime Domain', fontsize=12)
plt.ylabel('Number of Cases', fontsize=12)
plt.xticks(rotation=0)
plt.legend(title='Case Closed')
plt.tight_layout()
plt.show()
```



```
In [ ]:
```