Analyzing PUBG Data with Python

PlayerUnknown's Battlegrounds (PUBG) has taken the gaming world by storm, offering an immersive battle royale experience. Beyond its gaming aspect, PUBG also provides a wealth of data that can be mined and analyzed to gain insights into player behavior, strategies, and performance.

In this Jupyter Notebook project, we will delve into the exciting world of PUBG data analysis using Python. We will be working with a dataset containing a treasure trove of information, including player statistics, match details, and in-game events. Our goal is to harness the power of Python libraries such as Pandas, NumPy, and Matplotlib to extract meaningful insights from this dataset.

Import Library ¶

```
In [2]: import pandas as pd
```

```
In [29]: import pandas as pd
         import seaborn as sns
         import matplotlib.pyplot as plt
         import seaborn as sns
         import plotly.express as px
```

Uploading Csv fle

```
In [30]: df = pd.read_csv(r"C:\Users\Syed Arif\Desktop\Pubg_Stats.csv")
```

Data Preprocessing

.head()

head is used show to the By default = 5 rows in the dataset

```
In [31]: df.head()
Out[31]:
```

_Name	Matches_Played	Kills	Deaths	Assists	Damage_Dealt	Headshots	Wins	Top_10s	Revives
ıMaster	250	587	143	98	15243	234	32	145	67
perLion	312	823	218	112	18975	312	42	189	95
ıGamer	186	492	84	56	11786	156	28	97	48
ɜrStrike	409	923	267	134	21037	288	55	258	128
Demon	143	368	68	42	9865	123	20	72	36

.shape

It show the total no of rows & Column in the dataset

```
In [33]: df.shape
Out[33]: (221, 15)
```

.tail()

tail is used to show rows by Descending order

```
In [32]: df.tail()
Out[32]:
```

	Unnamed: 0	Player_Name	Matches_Played	Kills	Deaths	Assists	Damage_Dealt	Headshots
216	216	CrimsonRider	294	743	187	132	17567	242
217	217	BlazingSorcerer	203	521	109	72	13123	193
218	218	FrozenFlare	206	553	117	76	13756	196
219	219	AbyssGuardian	220	597	144	98	14967	212
220	220	SpectralPhantom	225	624	149	100	15345	219

.Columns

It show the no of each Column

```
In [34]: df.columns
Out[34]: Index(['Unnamed: 0', 'Player_Name', 'Matches_Played', 'Kills', 'Deaths',
                'Assists', 'Damage_Dealt', 'Headshots', 'Wins', 'Top_10s', 'Revives',
                'Distance_Traveled', 'Weapons_Used', 'Time_Survived', 'Rank'],
               dtype='object')
```

.dtypes

Demon	143	368	68	42	9865	123	20	72	36

shape

It show the total no of rows & Column in the dataset

In [33]:
```
df.shape
```

Out[33]:
```
(221, 15)
```

.tail()

tail is used to show rows by Descending order

In [32]:
```
df.tail()
```

Out[32]:

	Unnamed: 0	Player_Name	Matches_Played	Kills	Deaths	Assists	Damage_Dealt	Headshots
216	216	CrimsonRider	294	743	187	132	17567	242
217	217	BlazingSorcerer	203	521	109	72	13123	193
218	218	FrozenFlare	206	553	117	76	13756	196
219	219	AbyssGuardian	220	597	144	98	14967	212
220	220	SpectralPhantom	225	624	149	100	15345	219

.Columns

It show the no of each Column

In [34]:
```
df.columns
```

Out[34]:
```
Index(['Unnamed: 0', 'Player_Name', 'Matches_Played', 'Kills', 'Deaths',
       'Assists', 'Damage_Dealt', 'Headshots', 'Wins', 'Top_10s', 'Revives',
       'Distance_Traveled', 'Weapons_Used', 'Time_Survived', 'Rank'],
      dtype='object')
```

.dtypes

This Attribute show the data type of each column

In [35]:
```
df.dtypes
```

Out[35]:
```
Unnamed: 0            int64
Player_Name          object
Matches_Played        int64
Kills                 int64
Deaths                int64
Assists               int64
Damage_Dealt          int64
Headshots             int64
Wins                  int64
Top_10s               int64
Revives               int64
Distance_Traveled     int64
Weapons_Used          int64
Time_Survived         int64
Rank                 object
dtype: object
```

.unique()

In a column, It show the unique value of specific column.

In [36]:
```
df["Player_Name"].unique()
```

Out[36]:
```
array(['StealthMaster', 'SniperLion', 'NinjaGamer', 'ThunderStrike',
       'SpeedDemon', 'BlazeFury', 'RapidShadow', 'Frostbite',
       'SavageQueen', 'SwiftStriker', 'VenomousViper', 'PhoenixFury',
       'SteelStorm', 'BlazingBlade', 'StormChaser', 'Nightmare',
       'CrimsonTide', 'SilentShadow', 'VengefulViper', 'SolarFlare',
       'SkyDancer', 'RogueWraith', 'LethalLynx', 'FrostFang',
       'ScarletStrider', 'RagingRaptor', 'ShadowWisp', 'VenomStrike',
       'FireFury', 'BlazingSun', 'ShadowStrike', 'SteelGuardian',
       'WickedWitch', 'RuthlessRaptor', 'FrostyFox', 'ViperVenom',
       'CrimsonReaper', 'PhantomGhost', 'StormStrider', 'StormBreaker',
       'SapphireSword', 'ShadowReign', 'DragonSlayer', 'SilverShadow',
       'EagleEye', 'BlazingStorm', 'MidnightSage', 'RapidBlaze',
       'FrostFire', 'ScarletWitch', 'RagingTiger', 'SpectralRogue',
       'BlazingRaptor', 'EternalShadow', 'WickedStrider', 'CrimsonStorm',
       'RuthlessReaper', 'FrostFury', 'ShadowBlade', 'RapidPhantom',
       'ViperStrike', 'EternalBlaze', 'Vengeance', 'LunarShadow',
       'Deathstrike', 'AzureBlade', 'RavenHeart', 'SerpentFury',
       'CrimsonRogue', 'VoidSeeker', 'AstralSword', 'FrozenFlame',
       'TwilightWarden', 'ShadowPhenix', 'PhantomStrider', 'EternalFire',
       'NebulaBlade', 'SilverHawk', 'SolarSword', 'EclipseShadow',
       'StarBlade', 'LethalWraith', 'RadiantBlaze', 'FrostGuardian',
       'MysticSerpent', 'InfernoStorm', 'BlazeRanger', 'RagingFire',
       'ShadowDancer', 'PhoenixWings', 'IceStorm', 'MoonlitSorcerer',
       'DarkReaper', 'CosmicGhost', 'StormRider', 'FlareRogue',
       'RadiantBlade', 'TempestPhantom', 'SapphireViper', 'EternalFlame',
       'StarlightBlade', 'CrimsonRider', 'BlazingSorcerer', 'FrozenFlare',
```

In [36]:
```
df["Player_Name"].unique()
```

Out[36]:
```
array(['StealthMaster', 'SniperLion', 'NinjaGamer', 'ThunderStrike',
       'SpeedDemon', 'BlazeFury', 'RapidShadow', 'Frostbite',
       'SavageQueen', 'SwiftStriker', 'VenomousViper', 'PhoenixFury',
       'SteelStorm', 'BlazingBlade', 'StormChaser', 'Nightmare',
       'CrimsonTide', 'SilentShadow', 'VengefulViper', 'SolarFlare',
       'SkyDancer', 'RogueWraith', 'LethalLynx', 'FrostFang',
       'ScarletStrider', 'RagingRaptor', 'ShadowWisp', 'VenomStrike',
       'FireFury', 'BlazingSun', 'ShadowStrike', 'SteelGuardian',
       'WickedWitch', 'RuthlessRaptor', 'FrostyFox', 'ViperVenom',
       'CrimsonReaper', 'PhantomGhost', 'StormStrider', 'StormBreaker',
       'SapphireSword', 'ShadowReign', 'DragonSlayer', 'SilverShadow',
       'EagleEye', 'BlazingStorm', 'MidnightSage', 'RapidBlaze',
       'FrostFire', 'ScarletWitch', 'RagingTiger', 'SpectralRogue',
       'BlazingRaptor', 'EternalShadow', 'WickedStrider', 'CrimsonStorm',
       'RuthlessReaper', 'FrostFury', 'ShadowBlade', 'RapidPhantom',
       'ViperStrike', 'EternalBlaze', 'Vengeance', 'LunarShadow',
       'Deathstrike', 'AzureBlade', 'RavenHeart', 'SerpentFury',
       'CrimsonRogue', 'VoidSeeker', 'AstralSword', 'FrozenFlame',
       'TwilightWarden', 'ShadowPhoenix', 'PhantomStrider', 'EternalFire',
       'NebulaBlade', 'SilverHawk', 'SolarSword', 'EclipseShadow',
       'StarBlade', 'LethalWraith', 'RadiantBlaze', 'FrostGuardian',
       'MysticSerpent', 'InfernoStorm', 'BlazeRanger', 'RagingFire',
       'ShadowDancer', 'PhoenixWings', 'IceStorm', 'MoonlitSorcerer',
       'DarkReaper', 'CosmicGhost', 'StormRider', 'FlareRogue',
       'RadiantBlade', 'TempestPhantom', 'SapphireViper', 'EternalFlame',
       'StarlightBlade', 'CrimsonRider', 'BlazingSorcerer', 'FrozenFlare',
       'AbyssGuardian', 'SpectralPhantom'], dtype=object)
```

.nuique()

It will show the total no of unque value from whole data frame

In [37]:
```
df.nunique()
```

Out[37]:
```
Unnamed: 0           221
Player_Name          106
Matches_Played        70
Kills                 90
Deaths                82
Assists               65
Damage_Dealt         102
Headshots             70
Wins                  28
Top_10s               59
Revives               39
Distance_Traveled    105
Weapons_Used           9
Time_Survived        107
Rank                   4
dtype: int64
```

.describe()

It show the Count, mean, median etc

In [38]:
```
df.describe()
```

Out[38]:

	Unnamed: 0	Matches_Played	Kills	Deaths	Assists	Damage_Dealt	Headshots
count	221.000000	221.000000	221.000000	221.000000	221.000000	221.000000	221.000000
mean	110.000000	234.624434	612.674208	142.579186	92.615385	14801.004525	207.361991
std	63.941379	37.178429	89.311216	32.882564	21.423045	1902.947975	29.775909
min	0.000000	143.000000	368.000000	68.000000	42.000000	9865.000000	123.000000
25%	55.000000	206.000000	543.000000	117.000000	76.000000	13589.000000	193.000000
50%	110.000000	224.000000	604.000000	138.000000	92.000000	14894.000000	210.000000
75%	165.000000	257.000000	674.000000	167.000000	111.000000	15987.000000	226.000000
max	220.000000	409.000000	923.000000	267.000000	139.000000	21037.000000	312.000000

.isnull()

It shows the how many null values

In [40]:
```
df.isnull()
```

Out[40]:

Unnamed:	Player_Name	Matches_Played	Kills	Deaths	Assists	Damage_Dealt	Headshots

.value_counts

It Shows all the unique values with their count

In [39]:
```
df["Player_Name"].value_counts()
```

Out[39]:
```
VengefulViper      7
Frostbite          5
VenomousViper      5
LethalLynx         4
Nightmare          4
                  ..
MidnightSage       1
BlazingStorm       1
EagleEye           1
SilverShadow       1
SpectralPhantom    1
Name: Player_Name, Length: 106, dtype: int64
```

.describe()

It show the Count, mean, median etc

In [38]:
```
df.describe()
```

Out[38]:

	Unnamed: 0	Matches_Played	Kills	Deaths	Assists	Damage_Dealt	Headshots
count	221.000000	221.000000	221.000000	221.000000	221.000000	221.000000	221.000000
mean	110.000000	234.624434	612.674208	142.579186	92.615385	14801.004525	207.361991
std	63.941379	37.178429	89.311216	32.882564	21.423045	1902.947975	29.775909
min	0.000000	143.000000	368.000000	68.000000	42.000000	9865.000000	123.000000
25%	55.000000	206.000000	543.000000	117.000000	76.000000	13589.000000	193.000000
50%	110.000000	224.000000	604.000000	138.000000	92.000000	14894.000000	210.000000
75%	165.000000	257.000000	674.000000	167.000000	111.000000	15987.000000	226.000000
max	220.000000	409.000000	923.000000	267.000000	139.000000	21037.000000	312.000000

.value_counts

It Shows all the unique values with their count

In [39]:
```
df["Player_Name"].value_counts()
```

Out[39]:
```
VengefulViper      7
Frostbite          5
VenomousViper      5
LethalLynx         4
Nightmare          4
                  ..
MidnightSage       1
BlazingStorm       1
EagleEye           1
SilverShadow       1
SpectralPhantom    1
Name: Player_Name, Length: 106, dtype: int64
```

.isnull()

It shows the how many null values

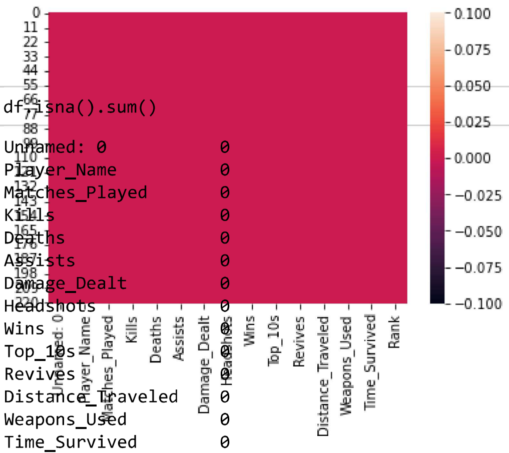
In [40]:
```
df.isnull()
```

Out[40]:

	Unnamed: 0	Player_Name	Matches_Played	Kills	Deaths	Assists	Damage_Dealt	Headshots
0	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...
216	False	False	False	False	False	False	False	False
217	False	False	False	False	False	False	False	False
218	False	False	False	False	False	False	False	False
219	False	False	False	False	False	False	False	False
220	False	False	False	False	False	False	False	False

221 rows × 15 columns

In [41]:
```
sns.heatmap(df.isnull())
plt.show()
```



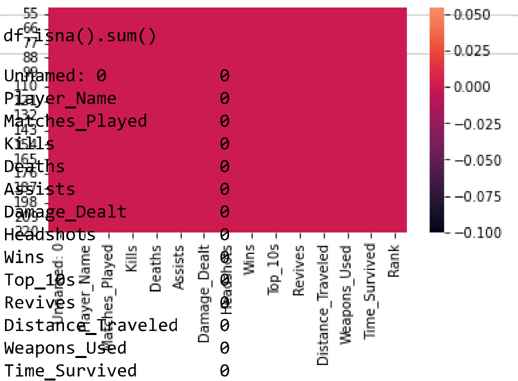
In [42]:
```
df.isna().sum()
```

Out[42]:
```
Unnamed: 0           0
Player_Name          0
Matches_Played       0
Kills                0
Deaths               0
Assists              0
Damage_Dealt         0
Headshots            0
Wins                 0
Top_10s              0
Revives              0
Distance_Traveled    0
Weapons_Used         0
Time_Survived        0
Rank                 0
dtype: int64
```

Drop the Unnamed Column

In [43]:
```
df.drop(['Unnamed: 0'],axis=1,inplace=True)
```

Show the Rank in Barplot



In [42]:
```
df.isna().sum()
```

Out[42]:
```
Unnamed: 0           0
Player_Name          0
Matches_Played       0
Kills                0
Deaths               0
Assists              0
Damage_Dealt         0
Headshots            0
Wins                 0
Top_10s              0
Revives              0
Distance_Traveled    0
Weapons_Used         0
Time_Survived        0
Rank                 0
dtype: int64
```

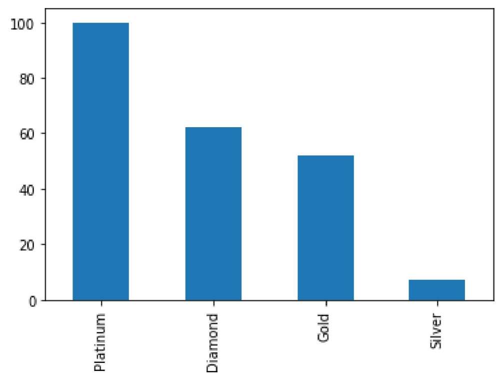
Drop the Unnamed Column

In [43]:
```
df.drop(['Unnamed: 0'],axis=1,inplace=True)
```

Show the Rank in Barplot

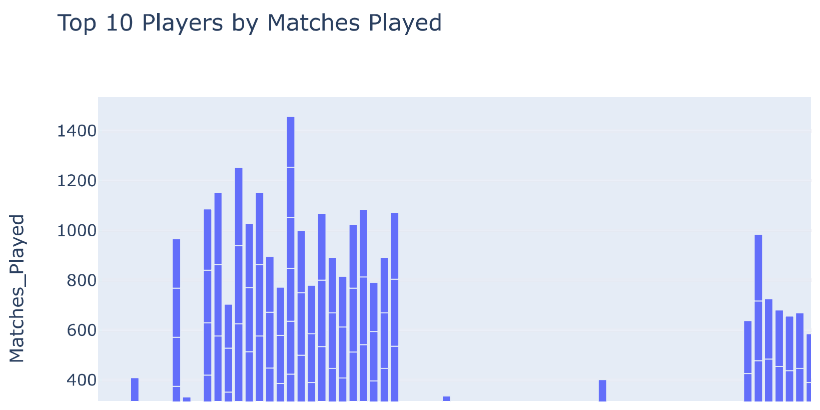
In [44]:
```
df.Rank.value_counts().plot(kind = "bar")
```

Out[44]:
```
<AxesSubplot:>
```

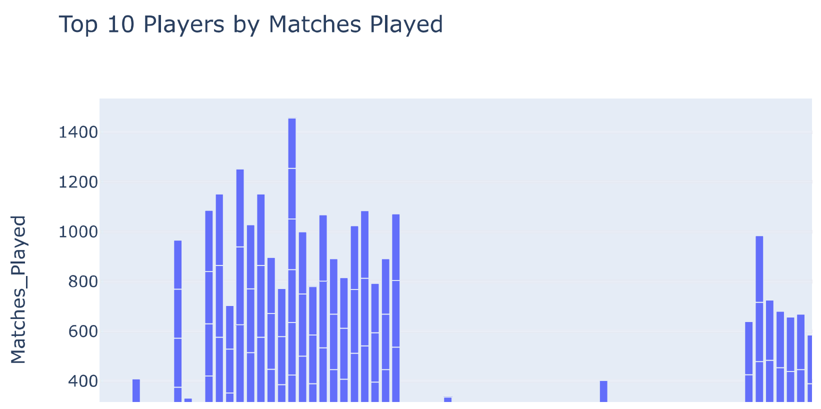


Top 10 players By Matches Played

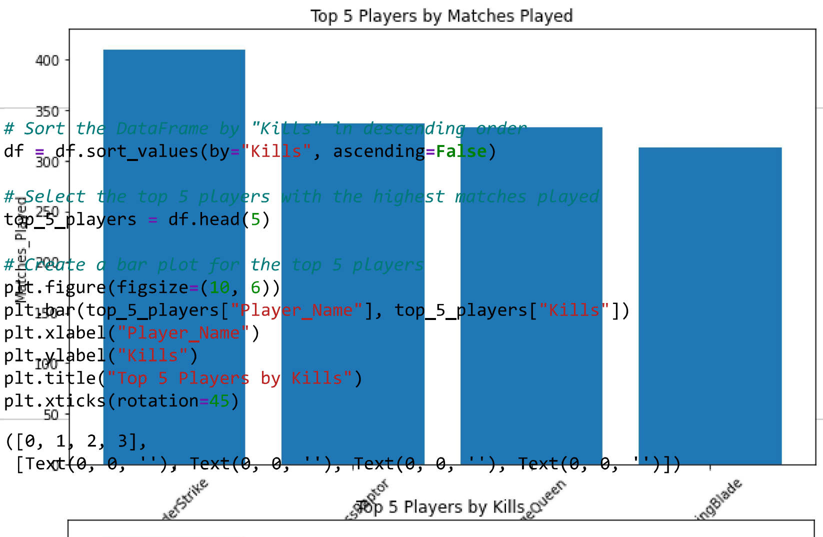
In [48]:
```
bar_top_10_players = px.bar(df, x="Player_Name", y="Matches_Played", title="Top 1
bar_top_10_players.show()
```



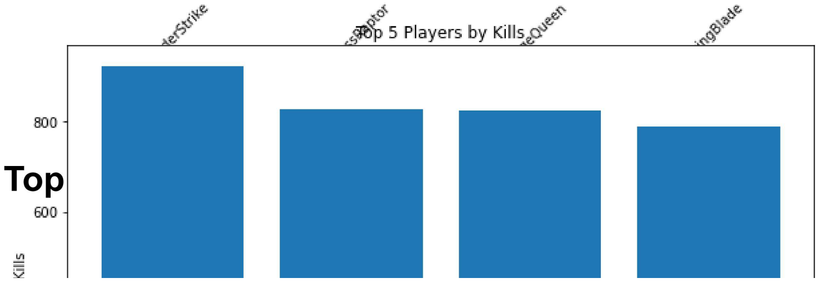
```
In [48]: bar_top_10_players = px.bar(df, x="Player_Name", y="Matches_Played", title="Top 1
         bar_top_10_players.show()
```



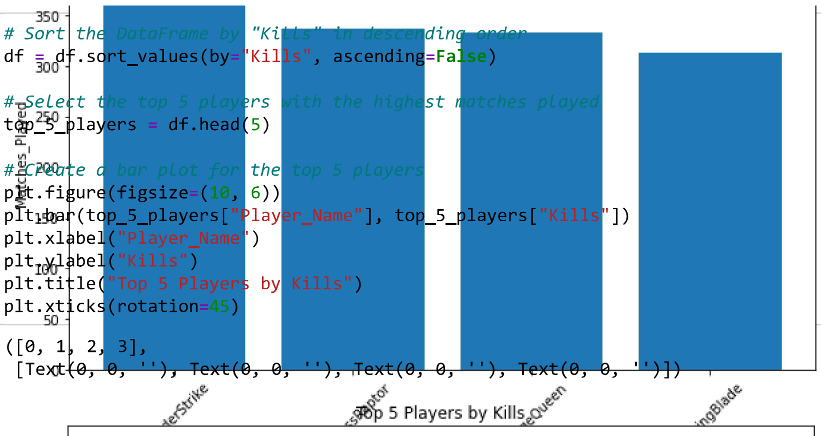
```
In [56]: # Sort the DataFrame by "Matches Played" in descending order
         df = df.sort_values(by="Matches_Played", ascending=False)

         # Select the top 5 players with the highest matches played
         top_5_players = df.head(5)

         # Create a bar plot for the top 5 players
         plt.figure(figsize=(10, 6))
         plt.bar(top_5_players["Player_Name"], top_5_players["Matches_Played"])
         plt.xlabel("Player_Name")
         plt.ylabel("Matches_Played")
         plt.title("Top 5 Players by Matches Played")
         plt.xticks(rotation=45)

Out[56]: ([0, 1, 2, 3],
          [Text(0, 0, ''), Text(0, 0, ''), Text(0, 0, ''), Text(0, 0, '')])
```



```
In [59]: # Sort the DataFrame by "Kills" in descending order
         df = df.sort_values(by="Kills", ascending=False)

         # Select the top 5 players with the highest matches played
         top_5_players = df.head(5)

         # Create a bar plot for the top 5 players
         plt.figure(figsize=(10, 6))
         plt.bar(top_5_players["Player_Name"], top_5_players["Kills"])
         plt.xlabel("Player_Name")
         plt.ylabel("Kills")
         plt.title("Top 5 Players by Kills")
         plt.xticks(rotation=45)

Out[59]: ([0, 1, 2, 3],
          [Text(0, 0, ''), Text(0, 0, ''), Text(0, 0, ''), Text(0, 0, '')])
```



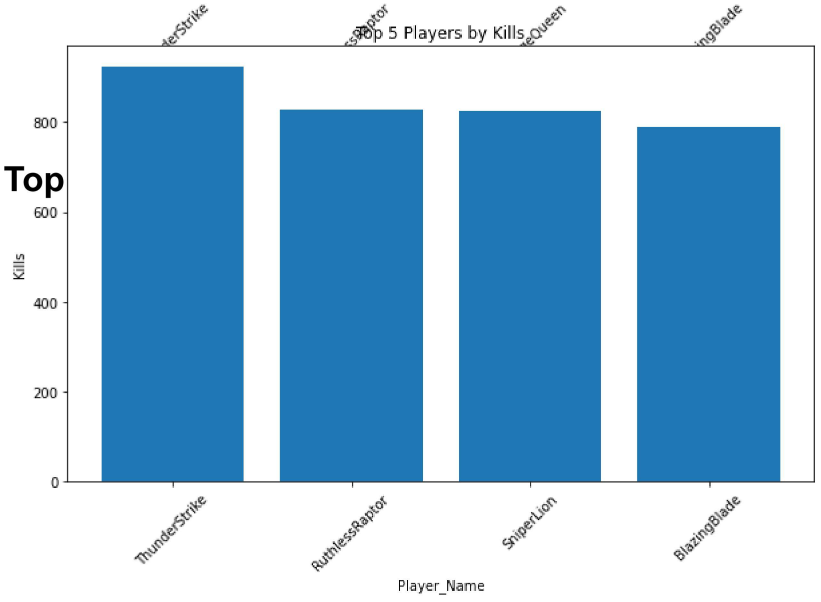
Top



In [59]:
```
# Sort the DataFrame by "Kills" in descending order
df = df.sort_values(by="Kills", ascending=False)

# Select the top 5 players with the highest matches played
top_5_players = df.head(5)

# Create a bar plot for the top 5 players
plt.figure(figsize=(10, 6))
plt.bar(top_5_players["Player_Name"], top_5_players["Kills"])
plt.xlabel("Player_Name")
plt.ylabel("Kills")
plt.title("Top 5 Players by Kills")
plt.xticks(rotation=45)
```

Out[59]:
```
([0, 1, 2, 3],
 [Text(0, 0, ''), Text(0, 0, ''), Text(0, 0, ''), Text(0, 0, '')])
```

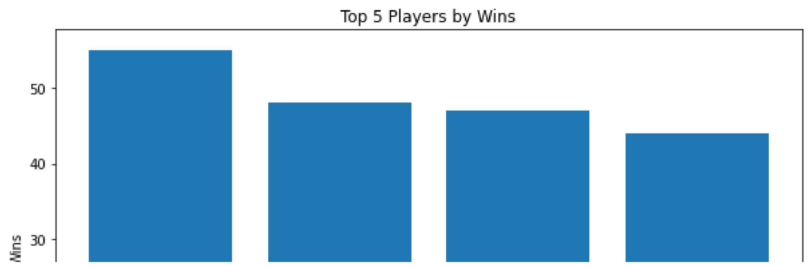
Top



Top 5 players By Wins

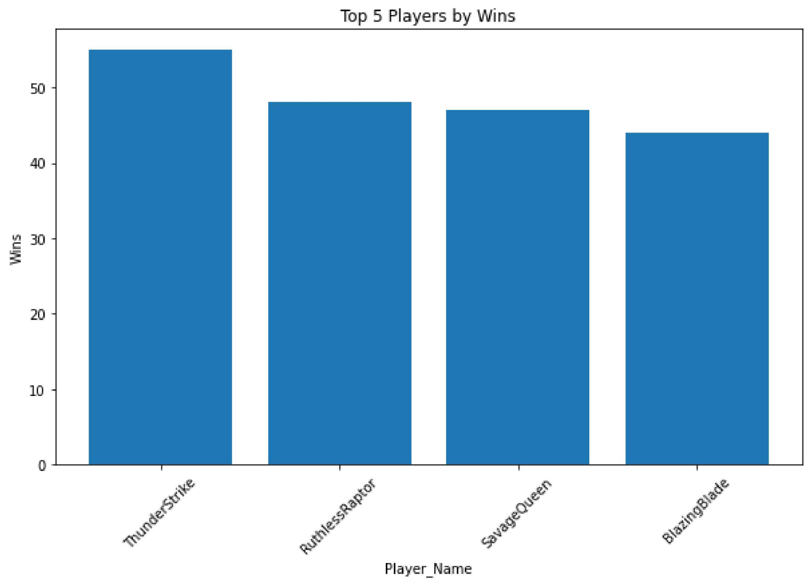
In [60]:
```
# Sort the DataFrame by "Wins" in descending order
df = df.sort_values(by="Wins", ascending=False)

# Select the top 5 players with the highest matches played
top_5_players = df.head(5)

# Create a bar plot for the top 5 players
plt.figure(figsize=(10, 6))
plt.bar(top_5_players["Player_Name"], top_5_players["Wins"])
plt.xlabel("Player_Name")
plt.ylabel("Wins")
plt.title("Top 5 Players by Wins")
plt.xticks(rotation=45)
```

Out[60]:
```
([0, 1, 2, 3],
 [Text(0, 0, ''), Text(0, 0, ''), Text(0, 0, ''), Text(0, 0, '')])
```

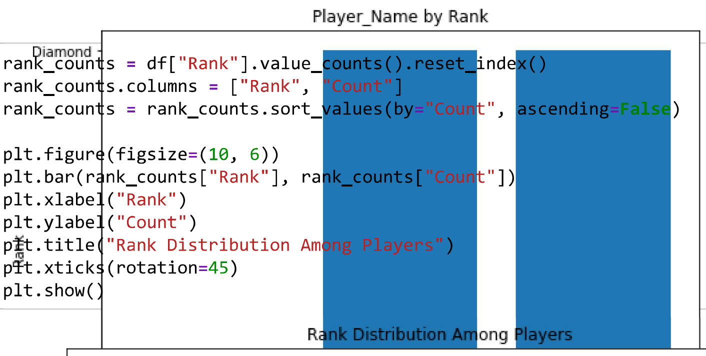


```
In [60]: # Sort the DataFrame by "Wins" in descending order
         df = df.sort_values(by="Wins", ascending=False)

         # Select the top 5 players with the highest matches played
         top_5_players = df.head(5)

         # Create a bar plot for the top 5 players
         plt.figure(figsize=(10, 6))
         plt.bar(top_5_players["Player_Name"], top_5_players["Wins"])
         plt.xlabel("Player_Name")
         plt.ylabel("Wins")
         plt.title("Top 5 Players by Wins")
         plt.xticks(rotation=45)
```

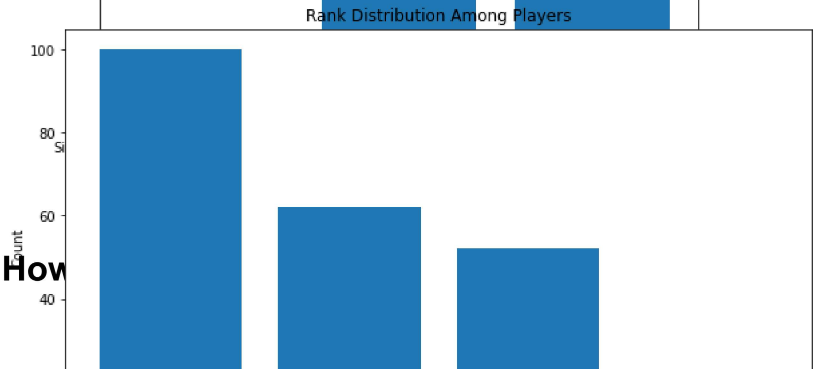
```
Out[60]: ([0, 1, 2, 3],
          [Text(0, 0, ''), Text(0, 0, ''), Text(0, 0, ''), Text(0, 0, '')])
```



```
In [67]: # Sort the DataFrame by "Matches Played" in descending order
         df = df.sort_values(by="Player_Name", ascending=False)

         # Select the top 5 players with the highest matches played
         top_5_players = df.head(5)
         plt.figure(figsize=(8, 6))
         plt.bar(top_5_players["Player_Name"], top_5_players["Rank"])
         plt.xlabel('Player_Name')
         plt.ylabel('Rank')
         plt.title('Player_Name by Rank')
         plt.show()
```

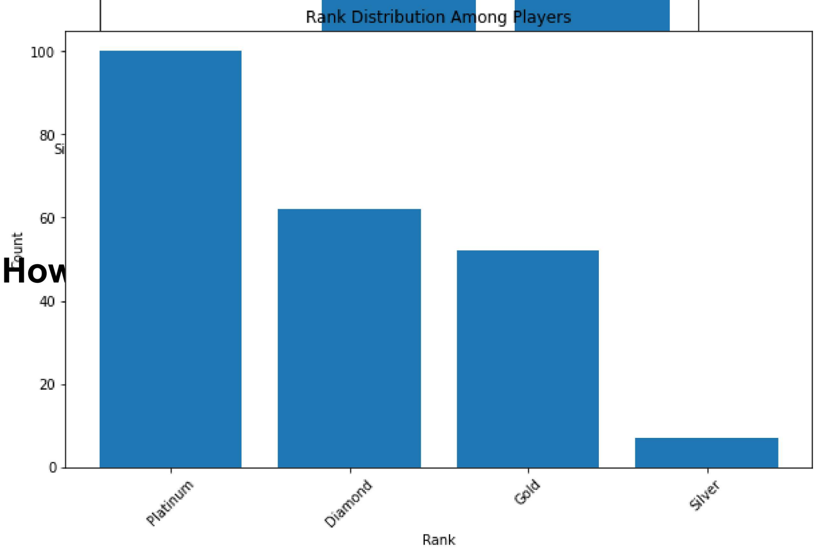


```
In [69]: rank_counts = df["Rank"].value_counts().reset_index()
         rank_counts.columns = ["Rank", "Count"]
         rank_counts = rank_counts.sort_values(by="Count", ascending=False)

         plt.figure(figsize=(10, 6))
         plt.bar(rank_counts["Rank"], rank_counts["Count"])
         plt.xlabel("Rank")
         plt.ylabel("Count")
         plt.title("Rank Distribution Among Players")
         plt.xticks(rotation=45)
         plt.show()
```



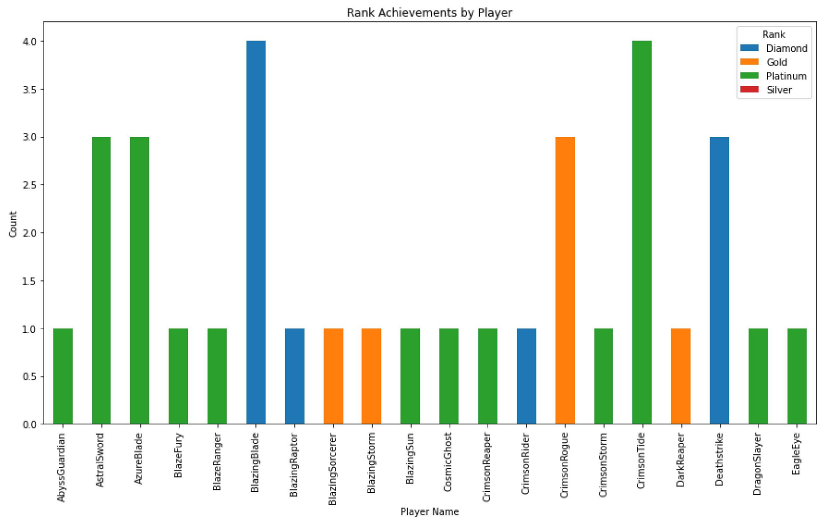
How

In [69]:
```
rank_counts = df["Rank"].value_counts().reset_index()
rank_counts.columns = ["Rank", "Count"]
rank_counts = rank_counts.sort_values(by="Count", ascending=False)

plt.figure(figsize=(10, 6))
plt.bar(rank_counts["Rank"], rank_counts["Count"])
plt.xlabel("Rank")
plt.ylabel("Count")
plt.title("Rank Distribution Among Players")
plt.xticks(rotation=45)
plt.show()
```

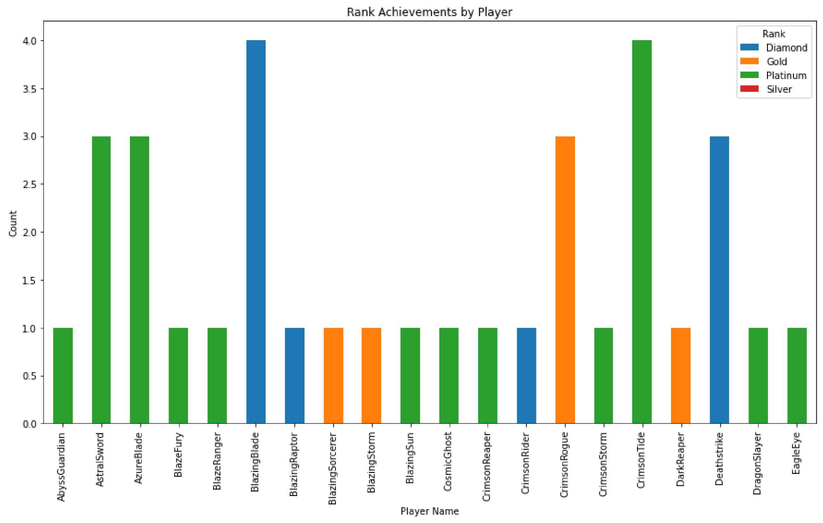


How many Times Rank Published by Player Name

In [74]:
```
# Create a cross-tabulation to count how many times each rank was achieved by eac
cross_tab = pd.crosstab(df["Player_Name"], df["Rank"]).head(20)

# Plot the bar chart
cross_tab.plot(kind="bar", stacked=True, figsize=(15, 8))

# Customize the plot
plt.xlabel("Player Name")
plt.ylabel("Count")
plt.title("Rank Achievements by Player")

# Show the plot
plt.show()
```



In [ ]:


Rank Achievements by Player
Rank
Diamond
Gold
Platinum
Silver
Count
Player Name
AbyssGuardian
AstralSword
AzureBlade
BlazeFury
BlazeRanger
BlazingBlade
BlazingRaptor
BlazingSorcerer
BlazingStorm
BlazingSun
CosmicGhost
CrimsonReaper
CrimsonRider
CrimsonRogue
CrimsonStorm
CrimsonTide
DarkReaper
Deathstrike
DragonSlayer
EagleEye

```
In [ ]:
```