In [ ]:
```
🏏Virat Kohli Performance Analysis🏏
Mohammad Areeb
Linkedin: www.linkedin.com/in/mohammadareeb2544
Github: https://github.com/areeb399
```

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

In [2]:
```
df = pd.read_csv("Virat.csv")
df
```

Out[2]:

	index	runs	opponent	ground	date	match	Match_No	total
0	0	12	SriLanka	Dambulla	18-08-2008	ODI	1	12
1	1	37	SriLanka	Dambulla	20-08-2008	ODI	2	49
2	2	25	SriLanka	Colombo(RPS)	24-08-2008	ODI	3	74
3	3	54	SriLanka	Colombo(RPS)	27-08-2008	ODI	4	128
4	4	31	SriLanka	Colombo(RPS)	29-08-2008	ODI	5	159
...	...	...	...	...	...	...	...	...
511	535	11	England	Birmingham	01-07-2022	Test	512	23661
512	536	20	England	Birmingham	01-07-2022	Test	513	23681
513	537	1	England	Birmingham	09-07-2022	T20	514	23682
514	538	11	England	Nottingham	10-07-2022	T20	515	23693
515	539	16	England	Lord's	14-07-2022	ODI	516	23709

516 rows × 8 columns

In [3]:
```
df.head(10)
```

Out[3]:

	index	runs	opponent	ground	date	match	Match_No	total
0	0	12	SriLanka	Dambulla	18-08-2008	ODI	1	12
1	1	37	SriLanka	Dambulla	20-08-2008	ODI	2	49
2	2	25	SriLanka	Colombo(RPS)	24-08-2008	ODI	3	74
3	3	54	SriLanka	Colombo(RPS)	27-08-2008	ODI	4	128
4	4	31	SriLanka	Colombo(RPS)	29-08-2008	ODI	5	159
5	5	2	SriLanka	Colombo(RPS)	14-09-2009	ODI	6	161
6	6	16	Pakistan	Centurion	26-09-2009	ODI	7	177
7	8	79	WestIndies	Johannesburg	30-09-2009	ODI	8	256
8	9	30	Australia	Vadodara	25-10-2009	ODI	9	286
9	10	10	Australia	Mohali	02-11-2009	ODI	10	296

In [4]:
```
df.tail()
```

Out[4]:

	index	runs	opponent	ground	date	match	Match_No	total
511	535	11	England	Birmingham	01-07-2022	Test	512	23661
512	536	20	England	Birmingham	01-07-2022	Test	513	23681
513	537	1	England	Birmingham	09-07-2022	T20	514	23682
514	538	11	England	Nottingham	10-07-2022	T20	515	23693
515	539	16	England	Lord's	14-07-2022	ODI	516	23709

In [5]:
```
df.shape
```

Out[5]:
```
(516, 8)
```

In [6]:
```
df.dropna(inplace = True)
```

In [8]:
```
df.isna()
```

Out[8]:

	index	runs	opponent	ground	date	match	Match_No	total
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
...	...	...	...	...	...	...	...	...
511	False	False	False	False	False	False	False	False
512	False	False	False	False	False	False	False	False
513	False	False	False	False	False	False	False	False
514	False	False	False	False	False	False	False	False
515	False	False	False	False	False	False	False	False

516 rows × 8 columns

In [10]:
```
df.isna().sum()
```

Out[10]:
```
index       0
runs        0
opponent    0
ground      0
date        0
match       0
Match_No    0
total       0
dtype: int64
```

In [56]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 516 entries, 0 to 515
Data columns (total 8 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   index     516 non-null    int64 
 1   runs      516 non-null    int64 
 2   opponent  516 non-null    object
 3   ground    516 non-null    object
 4   date      516 non-null    object
 5   match     516 non-null    object
 6   Match_No  516 non-null    int64 
 7   total     516 non-null    int64 
dtypes: int64(4), object(4)
memory usage: 32.4+ KB
```

In [58]:
```
df.describe()
```

Out[58]:

	index	runs	Match_No	total
count	516.000000	516.000000	516.000000	516.000000
mean	270.118217	45.947674	258.500000	11681.726744
std	155.219618	44.584372	149.100637	7301.114849
min	0.000000	0.000000	1.000000	12.000000
25%	134.750000	11.000000	129.750000	5328.250000
50%	270.500000	32.500000	258.500000	10886.500000
75%	403.250000	70.250000	387.250000	18535.750000
max	539.000000	254.000000	516.000000	23709.000000

In [81]:
```
# Changing the data type

df.runs = df.runs.astype(int)
df['date'] = pd.to_datetime(df['date'])

# Add a new column for year
df['year'] = df['date'].dt.year

# Add a new column for month
df['month'] = df['date'].dt.month
```

In [82]:
```
df.head()
```

Out[82]:

	index	runs	opponent	ground	date	match	Match_No	total	year	month
0	0	12	SriLanka	Dambulla	2008-08-18	ODI	1	12	2008	8
1	1	37	SriLanka	Dambulla	2008-08-20	ODI	2	49	2008	8
2	2	25	SriLanka	Colombo(RPS)	2008-08-24	ODI	3	74	2008	8
3	3	54	SriLanka	Colombo(RPS)	2008-08-27	ODI	4	128	2008	8
4	4	31	SriLanka	Colombo(RPS)	2008-08-29	ODI	5	159	2008	8

In [129…
```
#Total Runs Scored

total_runs =df['runs'].sum()

print("Total Runs Scored : ", total_runs)
```

```
Total Runs Scored :  23709
```

In [ ]:

In [127…
```
# Calculate the average runs scored
average_runs = df['runs'].mean()

print("Average Runs Scored:", average_runs)
```

```
Average Runs Scored: 45.94767441860465
```

In [ ]:

In [128…
```python
# Find the maximum runs scored
max_runs = df['runs'].max()

print("Maximum Runs Scored:", max_runs)
```

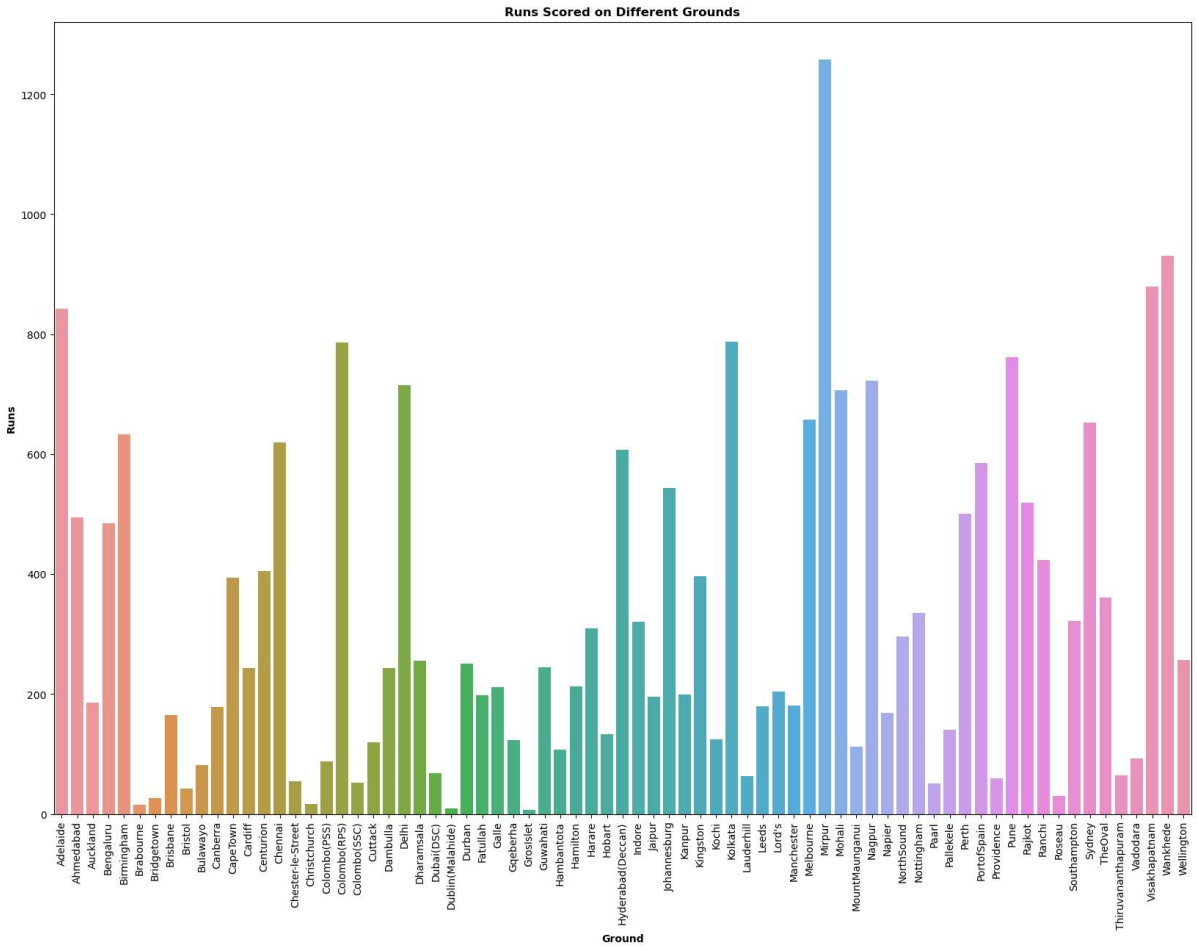
```
Maximum Runs Scored: 254
```

In [ ]:

In [64]:
```python
#Grouping the data by the 'ground' column and then calculating the sum of runs for

Ground = df.groupby('ground')['runs'].sum().reset_index()

print(Ground)
```

```
                ground  runs
0             Adelaide   843
1            Ahmedabad   494
2             Auckland   186
3            Bengaluru   485
4           Birmingham   633
..                 ...   ...
68  Thiruvananthapuram    65
69            Vadodara    93
70       Visakhapatnam   879
71            Wankhede   931
72          Wellington   257

[73 rows x 2 columns]
```

In [65]:
```python
#Creating a bar plot using seaborn (sns) for grounds based on the runs scored

plt.figure(figsize = (20,14))

sns.barplot(x = Ground['ground'], y = Ground['runs'])

plt.xticks(rotation = 90)

plt.xlabel('Ground', fontweight = 'bold')
plt.ylabel('Runs', fontweight = 'bold')
plt.title('Runs Scored on Different Grounds', fontweight = 'bold')

plt.show()
```

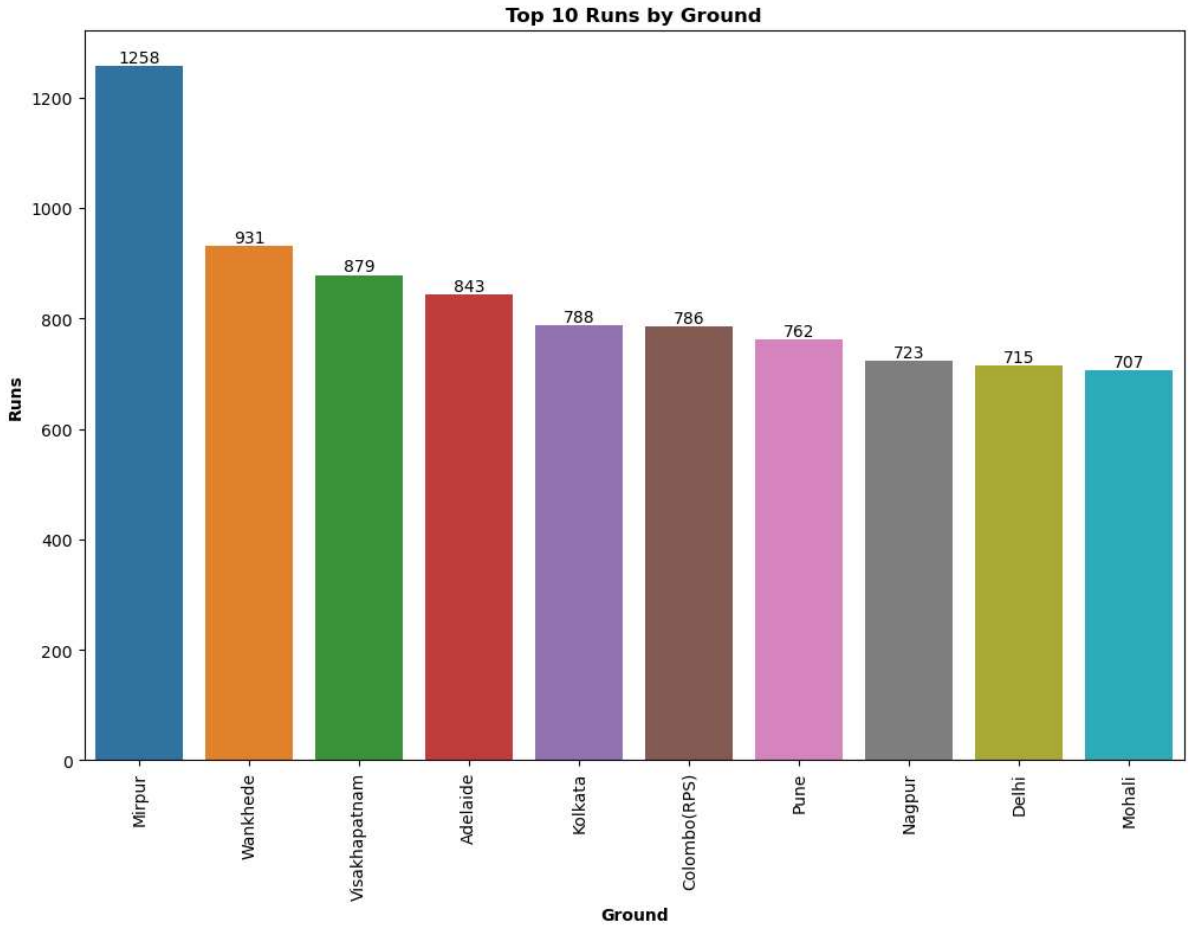


In [66]:
```
# Sort the DataFrame in descending order based on the 'runs' column
Ground_sorted = Ground.sort_values(by='runs', ascending=False)

# Get the top 5 rows
top_10_values = Ground_sorted.head(10)

# Print the result
print(top_10_values)
```

```
          ground  runs
49         Mirpur  1258
71       Wankhede   931
70  Visakhapatnam   879
0        Adelaide   843
43        Kolkata   788
18   Colombo(RPS)   786
61           Pune   762
52         Nagpur   723
22          Delhi   715
50         Mohali   707
```

In [67]:
```
#Creating a bar plot using seaborn (sns) for the top 10 grounds based on the runs s

plt.figure(figsize=(12, 8))

fig = sns.barplot(x = 'ground' , y = 'runs' , data = top_10_values)
```

```
for bars in fig.containers:
    fig.bar_label(bars)

plt.xticks(rotation = 90)

plt.xlabel('Ground', fontweight = 'bold')
plt.ylabel('Runs', fontweight = 'bold')
plt.title('Top 10 Runs by Ground', fontweight = 'bold')

plt.show()
```



In [ ]:
```
Conclusion:
    The maximum runs scored in Mirpur Ground is 1258
```

In [ ]:
```

```

In [68]:
```
df.head()
```

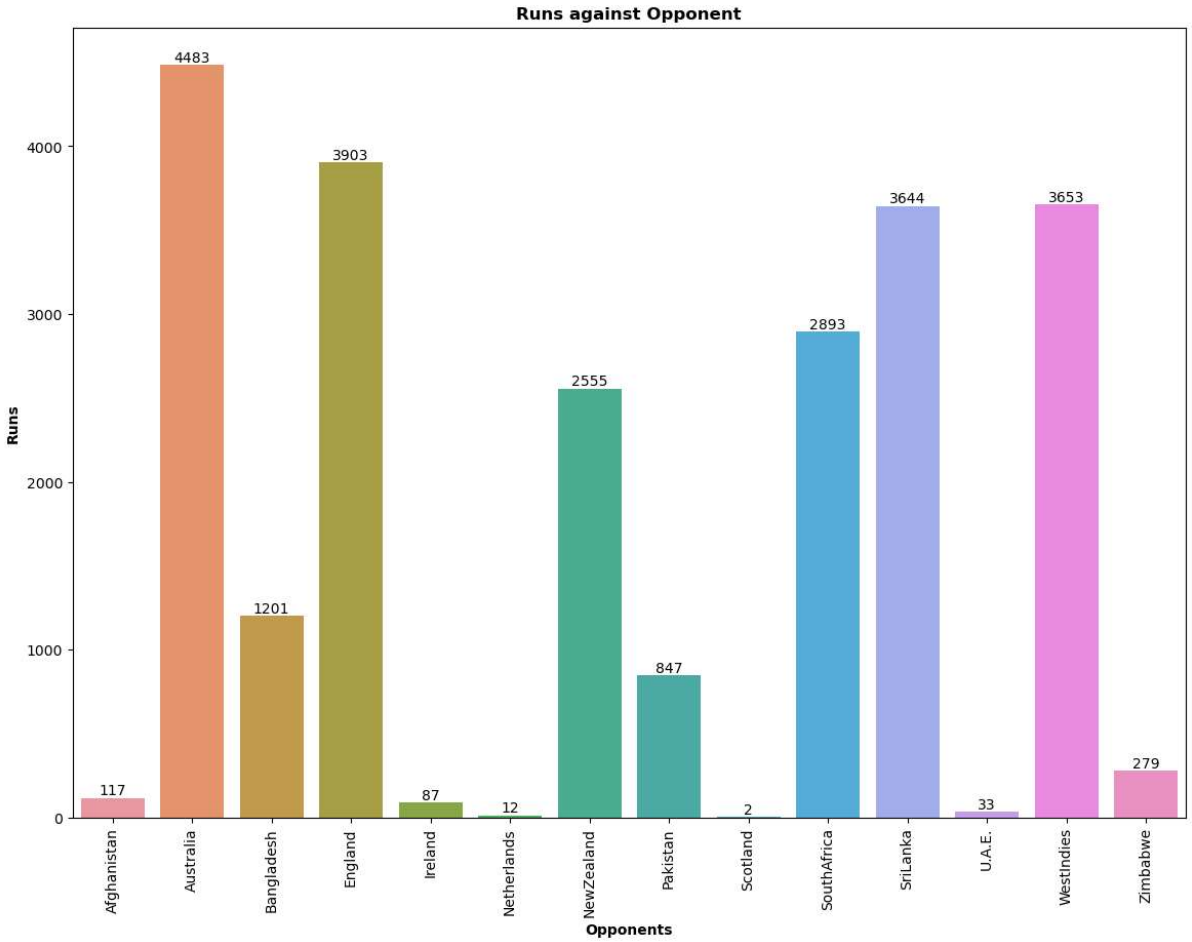
Out[68]:

	index	runs	opponent	ground	date	match	Match_No	total
0	0	12	SriLanka	Dambulla	18-08-2008	ODI	1	12
1	1	37	SriLanka	Dambulla	20-08-2008	ODI	2	49
2	2	25	SriLanka	Colombo(RPS)	24-08-2008	ODI	3	74
3	3	54	SriLanka	Colombo(RPS)	27-08-2008	ODI	4	128
4	4	31	SriLanka	Colombo(RPS)	29-08-2008	ODI	5	159

In [69]:
```
#Grouping the data by the 'opponent' column and calculates the sum of runs scored a

Opponent = df.groupby('opponent')['runs'].sum().reset_index()

print(Opponent)
```

```
       opponent  runs
0   Afghanistan   117
1     Australia  4483
2    Bangladesh  1201
3       England  3903
4       Ireland    87
5   Netherlands    12
6    NewZealand  2555
7      Pakistan   847
8      Scotland     2
9   SouthAfrica  2893
10     SriLanka  3644
11       U.A.E.    33
12   WestIndies  3653
13     Zimbabwe   279
```

In [70]:
```
#Creating a bar plot using seaborn (sns) for the opponents and their corresponding

plt.figure(figsize = (14,10))

fig = sns.barplot(x = Opponent['opponent'], y = Opponent['runs'])

for bars in fig.containers:
    fig.bar_label(bars)

plt.xticks(rotation = 90)

plt.xlabel('Opponents', fontweight = 'bold')
plt.ylabel('Runs', fontweight = 'bold')
plt.title('Runs against Opponent', fontweight = 'bold')

plt.show()
```

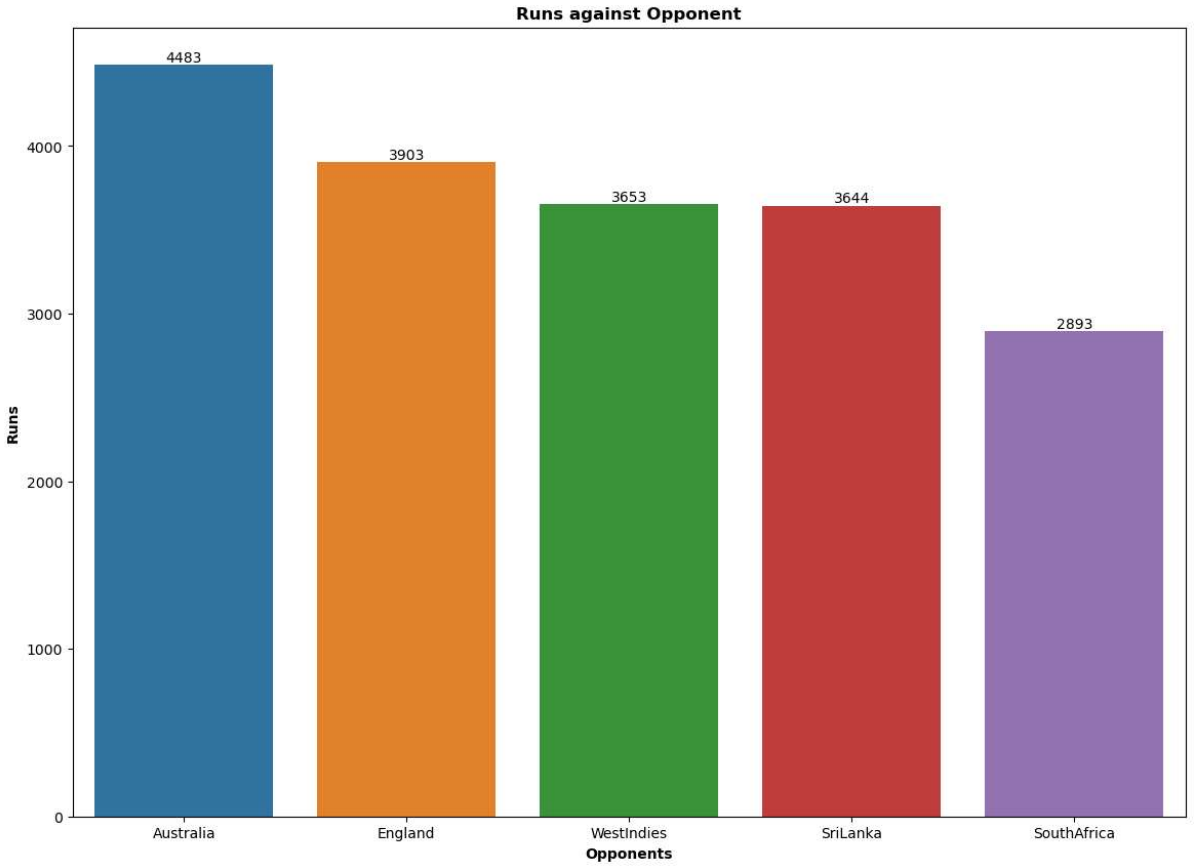


In [71]:
```
# Sort the DataFrame in descending order based on the 'runs' column
opponent_sorted = Opponent.sort_values(by='runs', ascending=False)

# Get the top 5 rows
top_5_values = opponent_sorted.head(5)

# Print the result
print(top_5_values)
```

```
       opponent  runs
1     Australia  4483
3       England  3903
12   WestIndies  3653
10     SriLanka  3644
9   SouthAfrica  2893
```

In [72]:
```
#Creating a bar plot using seaborn (sns) for the top 5 opponents based on the runs

plt.figure(figsize = (14,10))

fig = sns.barplot(x = top_5_values['opponent'], y = top_5_values['runs'])

for bars in fig.containers:
    fig.bar_label(bars)


plt.xlabel('Opponents', fontweight = 'bold')
```

```
plt.ylabel('Runs', fontweight = 'bold')
plt.title('Runs against Opponent', fontweight = 'bold')

plt.show()
```



In [ ]:
```
Conclusion :
    The sum of runs scored against Australia are 4483, followed by 3903 against Eng
```

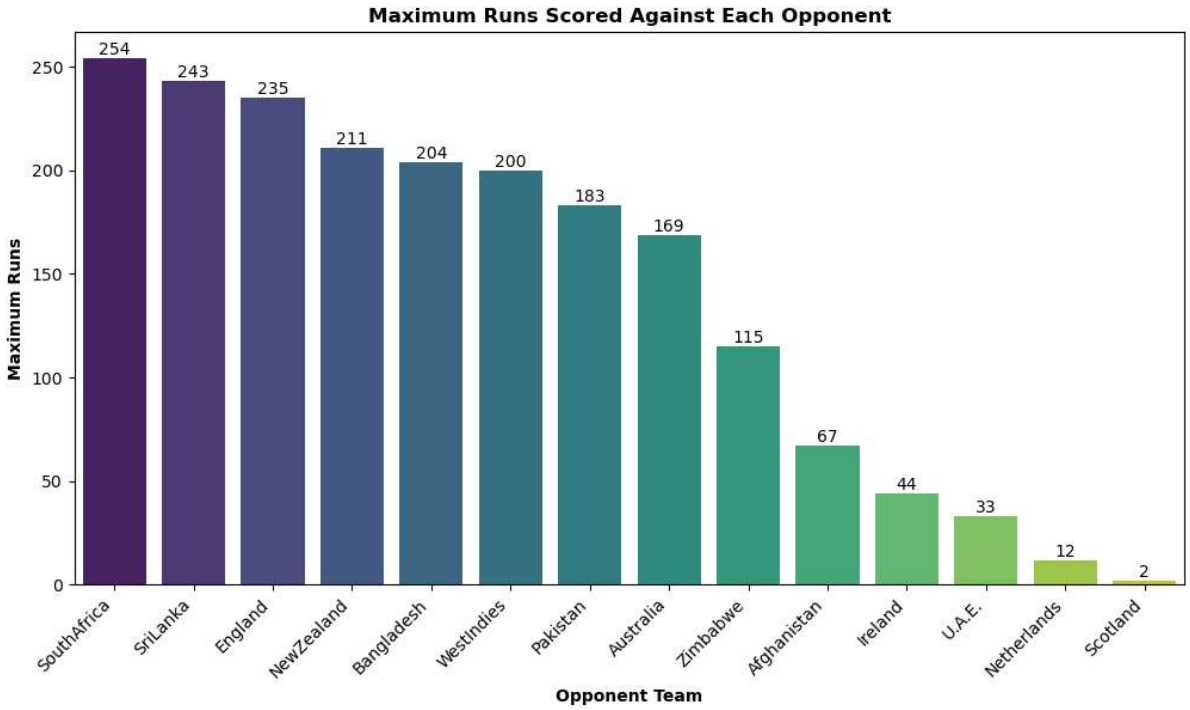
In [ ]:
```

```

In [133…
```
# Calculate the maximum runs scored against each opponent
max_runs = df.groupby('opponent')['runs'].max().reset_index()

# Sort the DataFrame by maximum runs in descending order
max_runs = max_runs.sort_values(by='runs', ascending=False)

print(max_runs)
```

	opponent	runs
9	SouthAfrica	254
10	SriLanka	243
3	England	235
6	NewZealand	211
2	Bangladesh	204
12	WestIndies	200
7	Pakistan	183
1	Australia	169
13	Zimbabwe	115
0	Afghanistan	67
4	Ireland	44
11	U.A.E.	33
5	Netherlands	12
8	Scotland	2

In [141…

```
#Creating a bar plot in seaborn (sns) that shows the maximum runs scored against ea

plt.figure(figsize=(12, 6))

fig = sns.barplot(data=max_runs, x='opponent', y='runs', palette='viridis')

for bars in fig.containers:
    fig.bar_label(bars)


plt.title('Maximum Runs Scored Against Each Opponent', fontweight = 'bold')
plt.xlabel('Opponent Team' , fontweight = 'bold')
plt.ylabel('Maximum Runs' , fontweight = 'bold')


plt.xticks(rotation=45, ha = 'right')


plt.show()
```



In [ ]:
```
Conclusion:
    The maximum runs scored against South Africa are 254, followed by 243 against S
```

In [ ]:
```
```

In [84]:
```
df.head()
```

Out[84]:

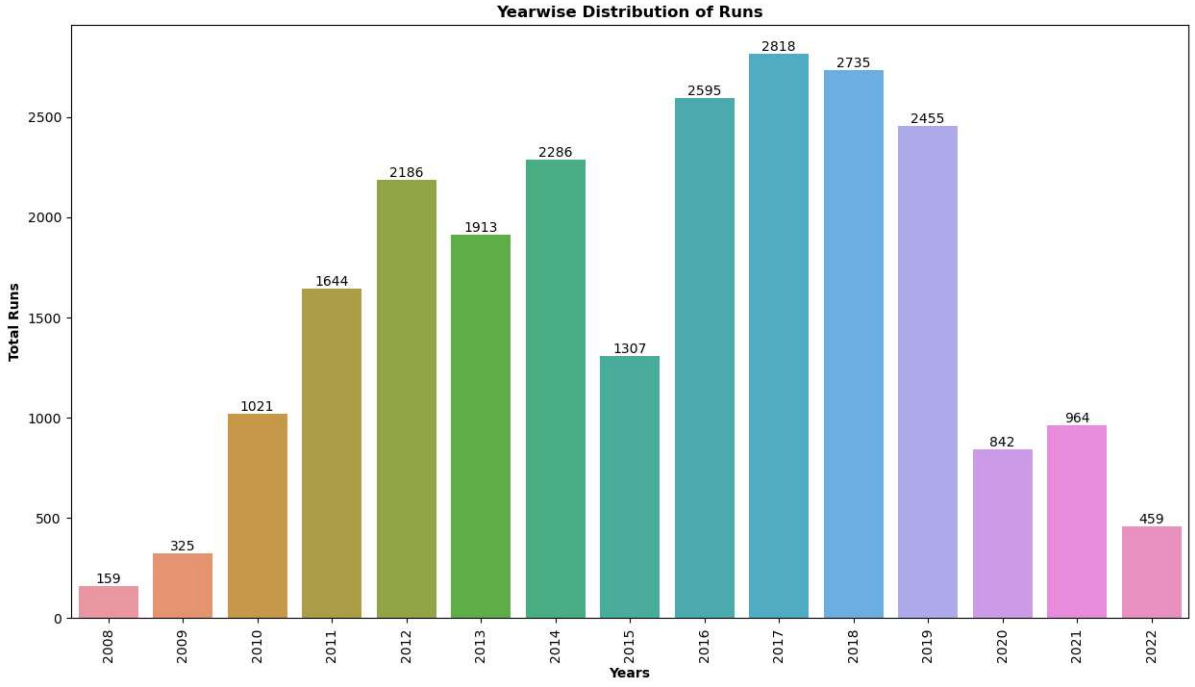
	index	runs	opponent	ground	date	match	Match_No	total	year	month
0	0	12	SriLanka	Dambulla	2008-08-18	ODI	1	12	2008	8
1	1	37	SriLanka	Dambulla	2008-08-20	ODI	2	49	2008	8
2	2	25	SriLanka	Colombo(RPS)	2008-08-24	ODI	3	74	2008	8
3	3	54	SriLanka	Colombo(RPS)	2008-08-27	ODI	4	128	2008	8
4	4	31	SriLanka	Colombo(RPS)	2008-08-29	ODI	5	159	2008	8

In [87]:
```
#Grouping the data by the 'year' column and calculates the sum of runs for each yea

yearwise = df.groupby('year')['runs'].sum()

print(yearwise)
```

```
year
2008     159
2009     325
2010    1021
2011    1644
2012    2186
2013    1913
2014    2286
2015    1307
2016    2595
2017    2818
2018    2735
2019    2455
2020     842
2021     964
2022     459
Name: runs, dtype: int64
```

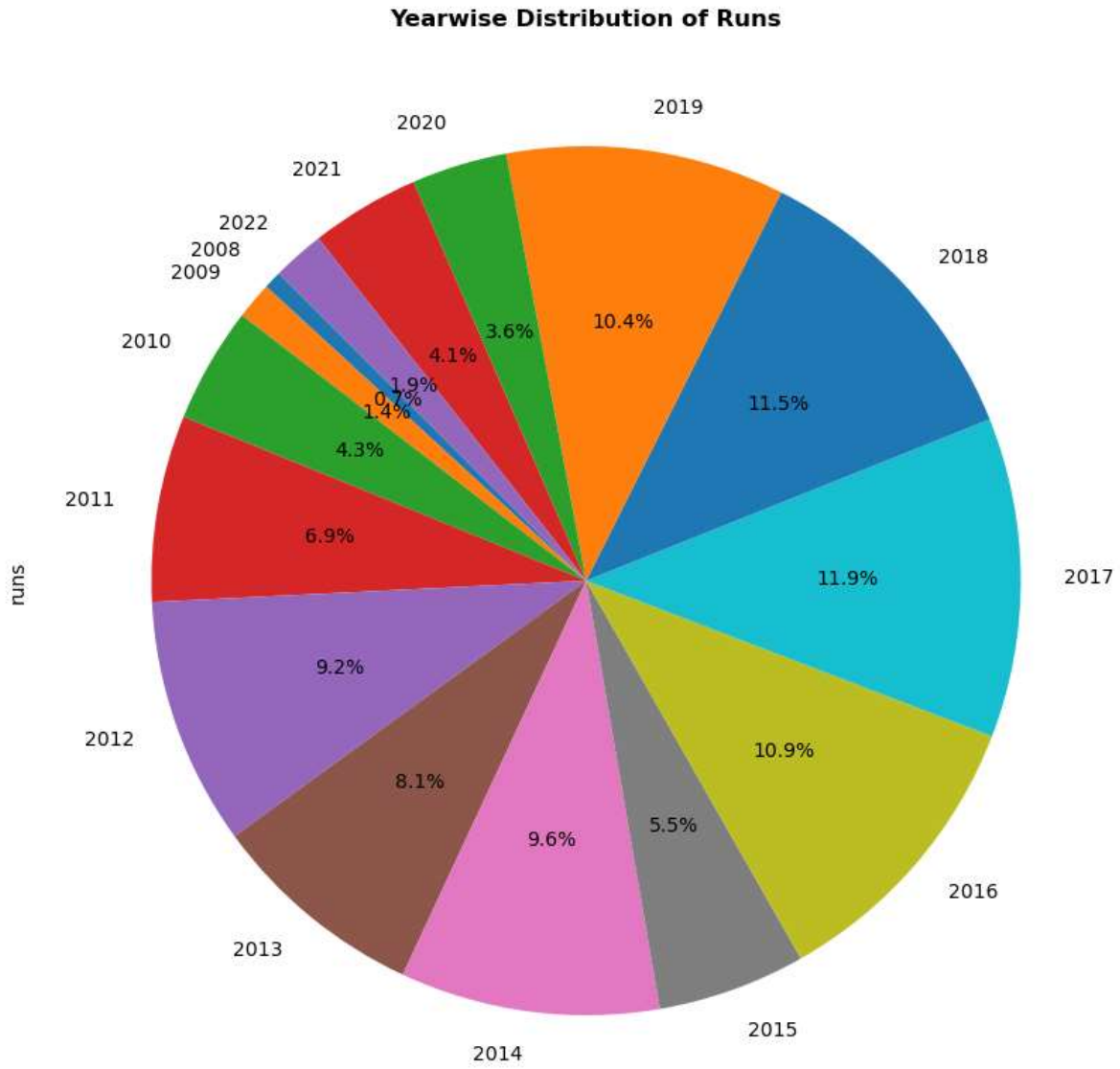
In [106…

```
#Creating a bar plot using seaborn (sns) for the yearwise distribution of runs

plt.figure(figsize = (15,8))

fig = sns.barplot(x = yearwise.index, y = yearwise.values)


for bars in fig.containers:
    fig.bar_label(bars)

plt.xticks(rotation = 90)
plt.xlabel('Years', fontweight = 'bold')
plt.ylabel('Total Runs', fontweight = 'bold')
plt.title('Yearwise Distribution of Runs', fontweight = 'bold')

plt.show()
```



In [ ]:
```
Conclusion:
    The highest runs scored in the year 2017 was 2818.
```

In [108…
```
#Creating a pie chart for the yearwise distribution of runs

plt.figure(figsize=(18, 10))


yearwise.plot(kind='pie', autopct='%0.1f%%', startangle=135)


plt.title("Yearwise Distribution of Runs", fontweight='bold')

plt.show()
```



In [ ]:
```

```

In [109…
```
#unique years present in the 'year' column

```

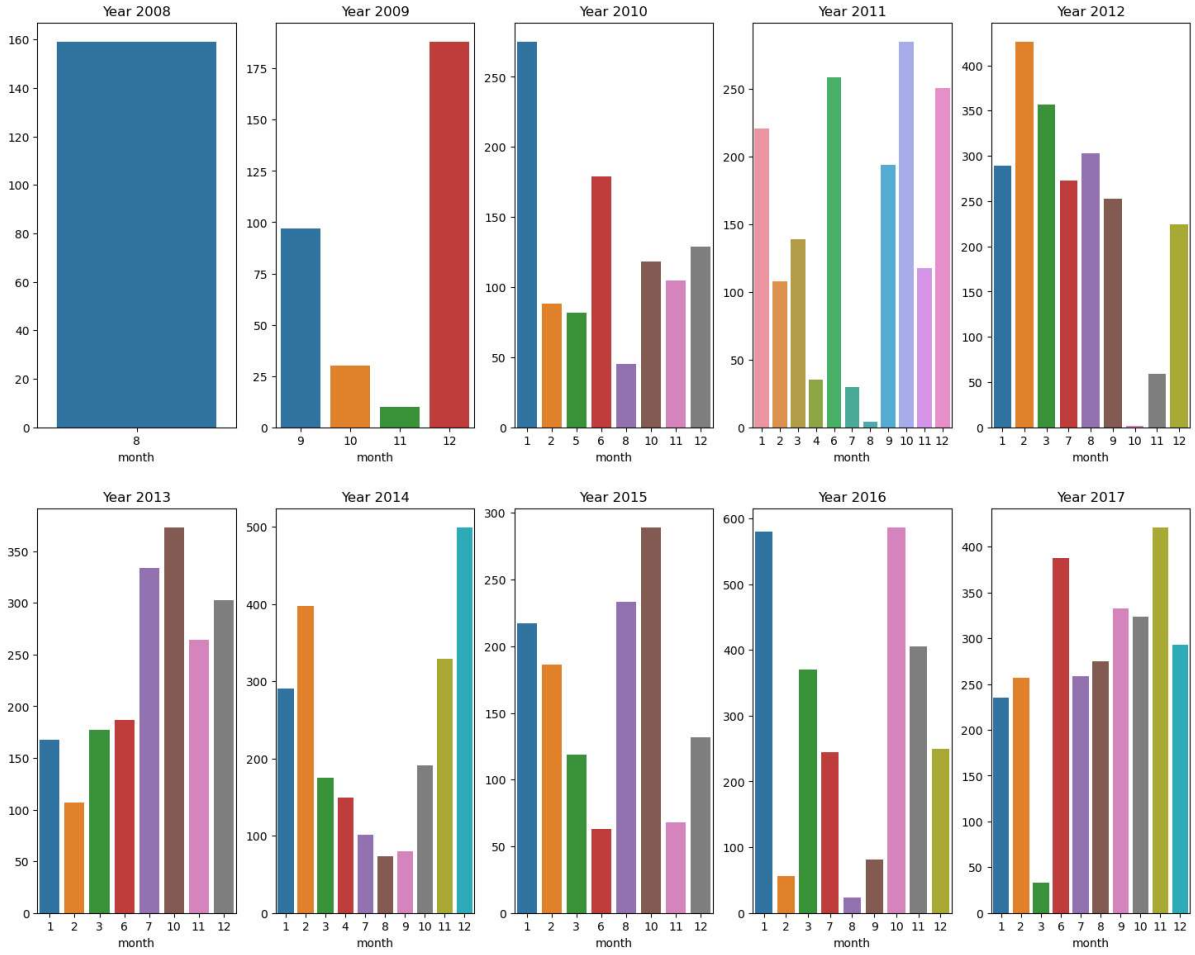
```
df.year.value_counts().index.sort_values(ascending=True)
```

Out[109]:
```
Int64Index([2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018,
            2019, 2020, 2021, 2022],
           dtype='int64')
```

In [114…
```
#Grouping the data by "year" and "month" columns, then calculates the sum of runs f

runs = df.groupby(["year","month"])["runs"].sum().reset_index()

print(runs)
```

```
     year  month  runs
0    2008      8   159
1    2009      9    97
2    2009     10    30
3    2009     11    10
4    2009     12   188
..    ...    ...   ...
108  2021     12    89
109  2022      1   224
110  2022      2    95
111  2022      3    81
112  2022      7    59

[113 rows x 3 columns]
```

In [142…
```
#Plotting bar charts for each year

plt.figure(figsize=(18,14))

l = [2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017]
i = 1
for x in l:
    df_temp = df[df.year==x]
    plt.subplot(2,5,i)
    plt.title("Year "+ str(x))

    x = df_temp.groupby("month")["runs"].sum()

    sns.barplot(data=df_temp, x=x.index, y=x.values)
    i = i+1
```


Year 2008
Year 2009
Year 2010
Year 2011
Year 2012
Year 2013
Year 2014
Year 2015
Year 2016
Year 2017
month

In [ ]: