# T20 World Cup 2024 Match Analysis

*Now, let's get started with T20 World Cup 2024 match analysis of India vs USA.*

In [2]:
```
import pandas as pd
data = pd.read_csv("D:\\india-usa_innings_data.csv")
```

*Let's see what we have got in the Innings dataset*

In [5]:
```
#checking values in the dataset
print(data.head())
```

```
            batter          bowler non_striker  runs_batter  runs_extras  \
0  Shayan Jahangir  Arshdeep Singh   SR Taylor            0            0
1         AGS Gous  Arshdeep Singh   SR Taylor            0            0
2         AGS Gous  Arshdeep Singh   SR Taylor            0            0
3         AGS Gous  Arshdeep Singh   SR Taylor            0            1
4         AGS Gous  Arshdeep Singh   SR Taylor            2            0

   runs_total wickets_0_player_out wickets_0_kind                      team  \
0           0      Shayan Jahangir            lbw  United States of America
1           0                  NaN            NaN  United States of America
2           0                  NaN            NaN  United States of America
3           1                  NaN            NaN  United States of America
4           2                  NaN            NaN  United States of America

   over  ...  wickets_0_fielders_0_name review_by review_umpire review_batter  \
0     0  ...                       NaN       NaN           NaN           NaN
1     0  ...                       NaN       NaN           NaN           NaN
2     0  ...                       NaN       NaN           NaN           NaN
3     0  ...                       NaN       NaN           NaN           NaN
4     0  ...                       NaN       NaN           NaN           NaN

  review_decision review_type extras_legbyes  wickets_0_fielders_1_name  \
0             NaN         NaN            NaN                       NaN
1             NaN         NaN            NaN                       NaN
2             NaN         NaN            NaN                       NaN
3             NaN         NaN            NaN                       NaN
4             NaN         NaN            NaN                       NaN

  extras_noballs  extras_penalty
0            NaN             NaN
1            NaN             NaN
2            NaN             NaN
3            NaN             NaN
4            NaN             NaN

[5 rows x 21 columns]
```

*Let's have a look at the missing values and data types in Dataset.*

In [8]:
```
#checking for missing values in the dataset
missing_values = data.isnull().sum()

#checking data types columns
```

```
data_types = data.dtypes

print("Missing Values:")
print(missing_values)
print("\nData Types:")
print(data_types)
```

```
Missing Values:
batter                        0
bowler                        0
non_striker                   0
runs_batter                   0
runs_extras                   0
runs_total                    0
wickets_0_player_out        225
wickets_0_kind              225
team                          0
over                          0
extras_wides                231
wickets_0_fielders_0_name   228
review_by                   235
review_umpire               235
review_batter               235
review_decision             235
review_type                 235
extras_legbyes              234
wickets_0_fielders_1_name   235
extras_noballs              235
extras_penalty              235
dtype: int64

Data Types:
batter                       object
bowler                       object
non_striker                  object
runs_batter                   int64
runs_extras                   int64
runs_total                    int64
wickets_0_player_out         object
wickets_0_kind               object
team                         object
over                          int64
extras_wides                float64
wickets_0_fielders_0_name    object
review_by                    object
review_umpire                object
review_batter                object
review_decision              object
review_type                  object
extras_legbyes              float64
wickets_0_fielders_1_name    object
extras_noballs              float64
extras_penalty              float64
dtype: object
```

*The Data has null values in various columns. But in such datasets, even null values have a meaning, so we will leave them as it is and move forward. Because changing null values can affect in player and teams statistics.*

# Let's group the data for analysis

In [12]:
```
# total runs scored by each team
total_runs = data.groupby("team")["runs_total"].sum()
print(total_runs)
```

```
team
India                        111
United States of America     110
Name: runs_total, dtype: int64
```

In [14]:
```
# total wickets taken by each team
total_wickets = data["wickets_0_player_out"].notna().groupby(data["team"]).sum()
print(total_wickets)
```

```
team
India                        3
United States of America     8
Name: wickets_0_player_out, dtype: int64
```

In [16]:
```
# total extras
total_extras = data[["team","runs_extras","extras_noballs","extras_legbyes","ext
print(total_extras)
```

```
                          runs_extras  extras_noballs  extras_legbyes  \
team
India                               9             1.0             1.0
United States of America            8             0.0             1.0

                          extras_penalty
team
India                                5.0
United States of America             0.0
```

In [18]:
```
# ball faced by each batter
balls_faced = data.groupby("batter").size()
print(balls_faced)
```

```
batter
AGS Gous              6
Aaron Jones          22
CJ Anderson          12
Harmeet Singh        10
Jasdeep Singh         7
NR Kumar             24
RG Sharma             6
RR Pant              20
S Dube               37
SA Yadav             49
SC van Schalkwyk     10
SR Taylor            31
Shayan Jahangir       1
V Kohli               1
dtype: int64
```

In [20]:
```
# runs scored by each batter
batter_runs = data.groupby("batter")["runs_batter"].sum()
print(batter_runs)
```

```
batter
AGS Gous              2
Aaron Jones          11
CJ Anderson          15
Harmeet Singh        10
Jasdeep Singh         2
NR Kumar             27
RG Sharma             3
RR Pant              18
S Dube               31
SA Yadav             50
SC van Schalkwyk     11
SR Taylor            24
Shayan Jahangir       0
V Kohli               0
Name: runs_batter, dtype: int64
```

In [22]:
```
# strike rate of each batter
strike_rate = (batter_runs/balls_faced)*100
print(strike_rate)
```

```
batter
AGS Gous              33.333333
Aaron Jones           50.000000
CJ Anderson          125.000000
Harmeet Singh        100.000000
Jasdeep Singh         28.571429
NR Kumar             112.500000
RG Sharma             50.000000
RR Pant               90.000000
S Dube                83.783784
SA Yadav             102.040816
SC van Schalkwyk     110.000000
SR Taylor             77.419355
Shayan Jahangir        0.000000
V Kohli                0.000000
dtype: float64
```

In [24]:
```
# boundaries hit by each batter
boundaries_hit = data[(data["runs_batter"]==4)|(data["runs_batter"]==6)].groupby
print(boundaries_hit)
```

```
runs_batter       4  6
batter
Aaron Jones       0  1
CJ Anderson       1  1
Harmeet Singh     0  1
NR Kumar          2  1
RR Pant           1  1
S Dube            1  1
SA Yadav          2  2
SC van Schalkwyk  1  0
SR Taylor         0  2
```

In [26]:
```
# wickets taken by each bowler
wickets_taken = data["wickets_0_player_out"].notna().groupby(data["bowler"]).sum
print(wickets_taken)
```

```
bowler
AR Patel            1
Ali Khan            1
Arshdeep Singh      4
CJ Anderson         0
HH Pandya           2
JJ Bumrah           0
Jasdeep Singh       0
Mohammed Siraj      1
S Dube              0
SC van Schalkwyk    0
SN Netravalkar      2
Name: wickets_0_player_out, dtype: int64
```

In [27]:
```python
# runs conceded by each bowler
runs_conceded = data.groupby("bowler")["runs_total"].sum()
print(runs_conceded)
```

```
bowler
AR Patel            25
Ali Khan            22
Arshdeep Singh       9
CJ Anderson         22
HH Pandya           15
JJ Bumrah           25
Jasdeep Singh       24
Mohammed Siraj      25
S Dube              11
SC van Schalkwyk    25
SN Netravalkar      18
Name: runs_total, dtype: int64
```

In [28]:
```python
# balls bowled by each bowler
balls_bowled = data.groupby("bowler").size()
print(balls_bowled)
```

```
bowler
AR Patel            19
Ali Khan            21
Arshdeep Singh      25
CJ Anderson         19
HH Pandya           24
JJ Bumrah           25
Jasdeep Singh       25
Mohammed Siraj      24
S Dube               6
SC van Schalkwyk    24
SN Netravalkar      24
dtype: int64
```

In [29]:
```python
# economy rate of each bowler
economy_rate = runs_conceded / (balls_bowled/6)
print(economy_rate)
```

```
bowler
AR Patel             7.894737
Ali Khan             6.285714
Arshdeep Singh       2.160000
CJ Anderson          6.947368
HH Pandya            3.750000
JJ Bumrah            6.000000
Jasdeep Singh        5.760000
Mohammed Siraj       6.250000
S Dube              11.000000
SC van Schalkwyk     6.250000
SN Netravalkar       4.500000
dtype: float64
```

In [30]:
```
# dot balls bowled by each bowler
dot_balls = data[data["runs_total"]==0].groupby("bowler").size()
print(dot_balls)
```

```
bowler
AR Patel             5
Ali Khan             7
Arshdeep Singh      17
CJ Anderson          8
HH Pandya           18
JJ Bumrah           14
Jasdeep Singh       11
Mohammed Siraj      11
S Dube               3
SC van Schalkwyk     8
SN Netravalkar      13
dtype: int64
```

*Combining all these statistics into Single dataframe for batters and bowlers.*

In [37]:
```
batter_stats = pd.DataFrame({
    "Runs" : batter_runs,
    "Balls Faced" : balls_faced,
    "Strike Rate" : strike_rate,
    "Boundaries Hit" : boundaries_hit.sum(axis=1)
    }).join(boundaries_hit)

print(batter_stats)
```

```
                Runs  Balls Faced  Strike Rate  Boundaries Hit    4    6
batter
AGS Gous           2            6    33.333333             NaN  NaN  NaN
Aaron Jones       11           22    50.000000             1.0  0.0  1.0
CJ Anderson       15           12   125.000000             2.0  1.0  1.0
Harmeet Singh     10           10   100.000000             1.0  0.0  1.0
Jasdeep Singh      2            7    28.571429             NaN  NaN  NaN
NR Kumar          27           24   112.500000             3.0  2.0  1.0
RG Sharma          3            6    50.000000             NaN  NaN  NaN
RR Pant           18           20    90.000000             2.0  1.0  1.0
S Dube            31           37    83.783784             2.0  1.0  1.0
SA Yadav          50           49   102.040816             4.0  2.0  2.0
SC van Schalkwyk  11           10   110.000000             1.0  1.0  0.0
SR Taylor         24           31    77.419355             2.0  0.0  2.0
Shayan Jahangir    0            1     0.000000             NaN  NaN  NaN
V Kohli            0            1     0.000000             NaN  NaN  NaN
```

In [38]:
```
bowler_stats = pd.DataFrame({
    "Wickets" : wickets_taken,
    "Runs Conceded" : runs_conceded,
    "Balls Bowled" : balls_bowled,
    "Economy Rate" : economy_rate,
    "Dot Balls" : dot_balls
})
print(bowler_stats)
```

```
                 Wickets  Runs Conceded  Balls Bowled  Economy Rate  \
bowler
AR Patel               1             25            19      7.894737
Ali Khan               1             22            21      6.285714
Arshdeep Singh         4              9            25      2.160000
CJ Anderson            0             22            19      6.947368
HH Pandya              2             15            24      3.750000
JJ Bumrah              0             25            25      6.000000
Jasdeep Singh          0             24            25      5.760000
Mohammed Siraj         1             25            24      6.250000
S Dube                 0             11             6     11.000000
SC van Schalkwyk       0             25            24      6.250000
SN Netravalkar         2             18            24      4.500000

                 Dot Balls
bowler
AR Patel                 5
Ali Khan                 7
Arshdeep Singh          17
CJ Anderson              8
HH Pandya               18
JJ Bumrah               14
Jasdeep Singh           11
Mohammed Siraj          11
S Dube                   3
SC van Schalkwyk         8
SN Netravalkar          13
```

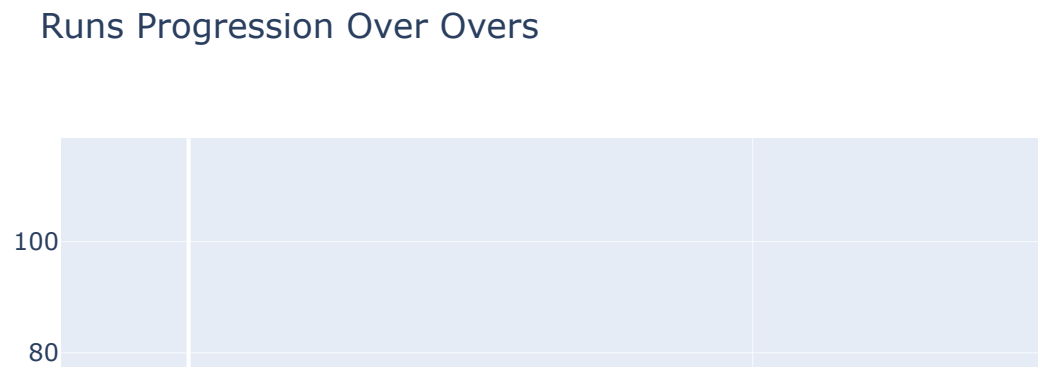
*I have grouped the data that we needed to analyze this data properly.*

# Team Performance

**Now, let's have a look at the progression of the run over overs :**

In [44]:
```
import plotly.graph_objects as go

# run progression of both the teams over the overs
india_runs_progression = data[data["team"]=="India"].groupby("over")["runs_total
usa_runs_progression = data[data["team"]=="United States of America"].groupby("o

fig = go.Figure()

# to add a trace for India's run progression in a line chart with markers
fig.add_trace(go.Scatter(
    x = india_runs_progression.index,
    y = india_runs_progression.values,
    mode = "lines+markers",
    name = "India"
```

```
))

# to add a trace for USA run progression in a line chart with markers
fig.add_trace(go.Scatter(
    x = usa_runs_progression.index,
    y = usa_runs_progression.values,
    mode = "lines+markers",
    name = "USA"
))

# layout for plotting figure for comparison of both teams run
fig.update_layout(
    title = "Runs Progression Over Overs",
    xaxis_title = "Overs",
    yaxis_title = "Runs",
    legend_title = "Teams"
)

fig.show()
```




The graph shows the progression of the cumulative run over the overs for the India and the USA in their T20 World Cup Match. Initially both teams had a steady run rate, with India slightly ahead in the early overs. As the innings progressed, USA gained momentum and took the lead briefly around the middle overs. However, India accelerated their scoring in the later overs, surpassing the USA and maintaining the lead

until the end. The key takeaway is India's strong finish, which enabled them to secure the win by consistently increasing their run rate in the final overs.

**Now, let's have a look at the wickets timeline:**

In [47]:
```
# wickets of both the team over the Overs
india_wickets = data[(data["team"]=="India") & data["wickets_0_player_out"].notn
usa_wickets = data[(data["team"]=="United States of America") & data["wickets_0_

fig = go.Figure()

# tracing wickets of team India by overs with the help of Bar chart
fig.add_trace(go.Bar(
    x=india_wickets.index,
    y=india_wickets.values,
    name = "India",
    marker_color = "blue",
    opacity= 0.6
))

# tracing wickets of team USA by overs with the help of Bar chart
fig.add_trace(go.Bar(
    x=usa_wickets.index,
    y = usa_wickets.values,
    name ="USA",
    marker_color = "red",
    opacity= 0.6
))

# plotting bar chart figure for wicket Timeine
fig.update_layout(
    title = "Wickets Timeline",
    xaxis_title = "Overs",
    yaxis_title = "Wickets",
    barmode ="group",
    legend_title = "Teams"
)

fig.show()
```

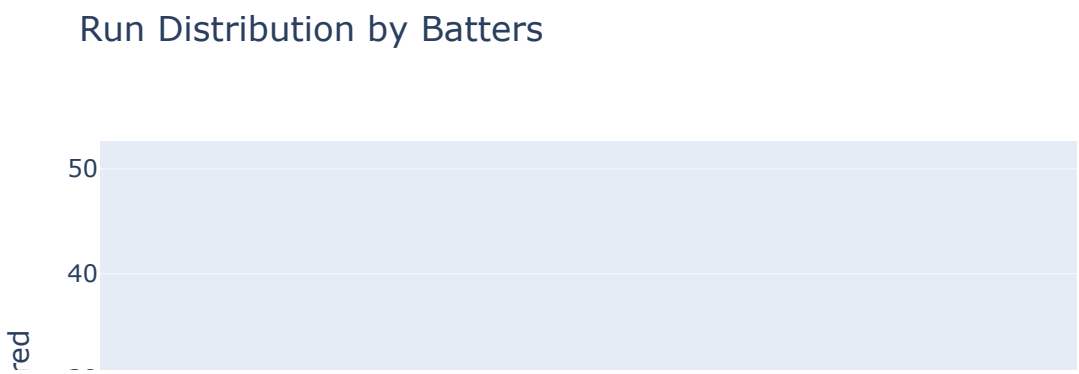

The wickets timeline graph illustrates the distribution of wickets taken over the overs for both India and the USA. The USA lost wickets more frequently, especially in the early overs, with two wickets falling in the first over, followed by consistent wicket losses throughout their innings. In contrast, India experienced their wickets losses more evenly spread across their innings, with a couple of early wickets but maintaining longer partnerships in the middle overs. the frequent loss of wickets by the USA disrupted their momentum, while India's ability to avoid clusters of wickets falling in succession helped them maintain a steady scoring rate ultimately secure the win.

# Players Performance

**Now, let's have a look at the run distribution by batters:**

In [53]:
```
import plotly.express as px

# creating bar chart for run distribution of each batter
fig = px.bar(
    batter_stats,
    x = batter_stats.index,
    y = "Runs",
    title= "Run Distribution by Batters",
    labels = {"x":"Batter","Runs":"Runs Scored"},
)
```

```
# plotting the bar chart for run distribution
fig.update_layout(
    xaxis_title = "Batter",
    yaxis_title = "Runs Scored",
    xaxis = dict(tickangle=90)
)
fig.show()
```




Notably, S.A. Yadav emerged as the highest scorer with a significant contribution, followed by NR Kumar and S.Dube. These three players were pivotal in their team's innings, providing the bulk of the runs,

**Now, let's have a look at the bowling performance:**

In [56]:
```
fig = go.Figure()

# scattering bowling performance of each bowler with economy and wickets
fig.add_trace(go.Scatter(
    x=bowler_stats['Economy Rate'],
    y=bowler_stats['Wickets'],
    mode='markers+text',
    text=bowler_stats.index,
    textposition='top center',
    textfont=dict(
        family="sans serif",
        size=12,
```

```
        color="black"
    ),
    marker=dict(color='red', size=10),
    name='Bowlers'
))

fig.update_layout(
    title = "Bowling Performance",
    xaxis_title = "Economy Rate",
    yaxis_title = "Wickets Taken",
    width = 800,
    height = 600
)

fig.show()
```


The bowling performance graph compares the economy rate and wickets taken by various bowlers in the match between India and USA. Arshdeep Singh stands out as the most effective bowler, taking the highest number of wickets (4) with a commendable economy rate. Other notable performances include HH Pandya and SN Netravalkar, both

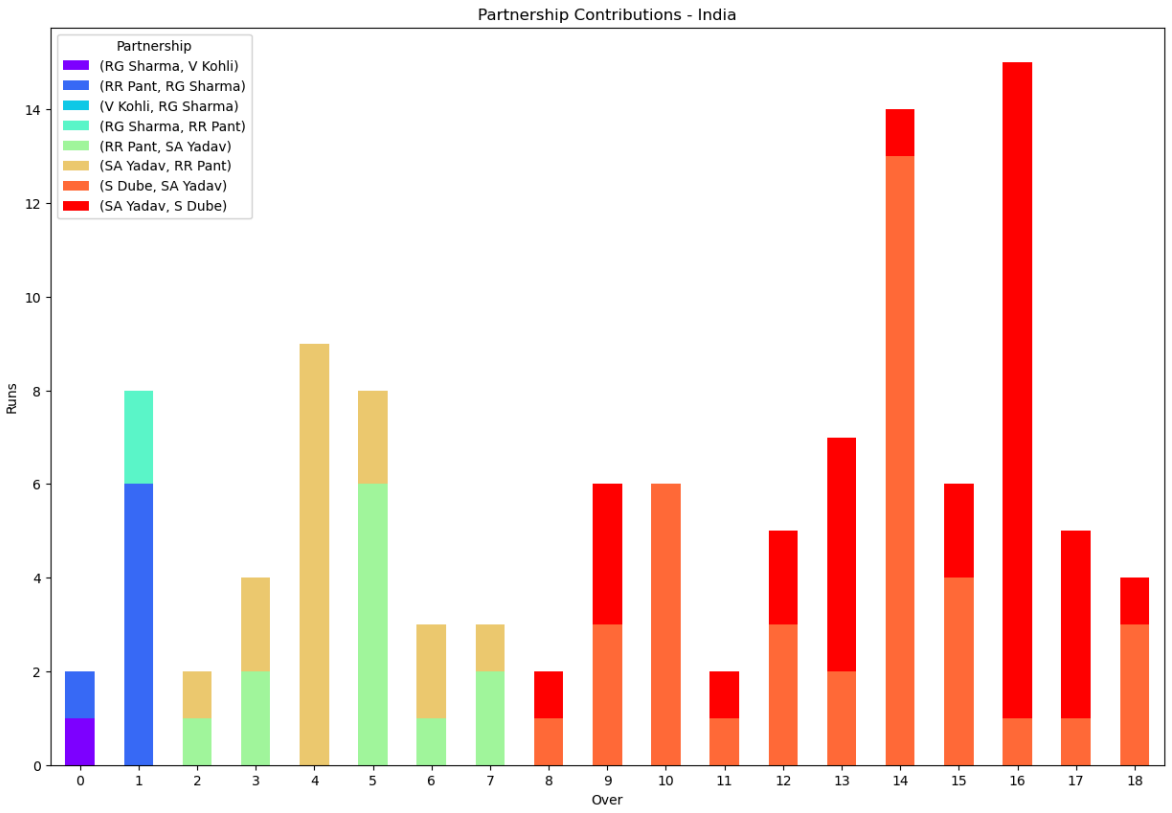
taking (2) wickets each with moderate economy rates. Bowlers like S Dube, having a higher economy rate, contributed less in terms of wickets.

# Contributions of Partnership

**Now, let's have a look at the partnership contributions in the India's innings:**

In [60]:
```
# separate data for India and USA
india_partnership_data = data[data['team'] == 'India'].groupby(['over', 'batter'
usa_partnership_data = data[data['team'] == 'United States of America'].groupby(

# create pivot tables for better visualization
india_partnership_pivot = india_partnership_data.pivot(index='over', columns=['b
usa_partnership_pivot = usa_partnership_data.pivot(index='over', columns=['batte
```

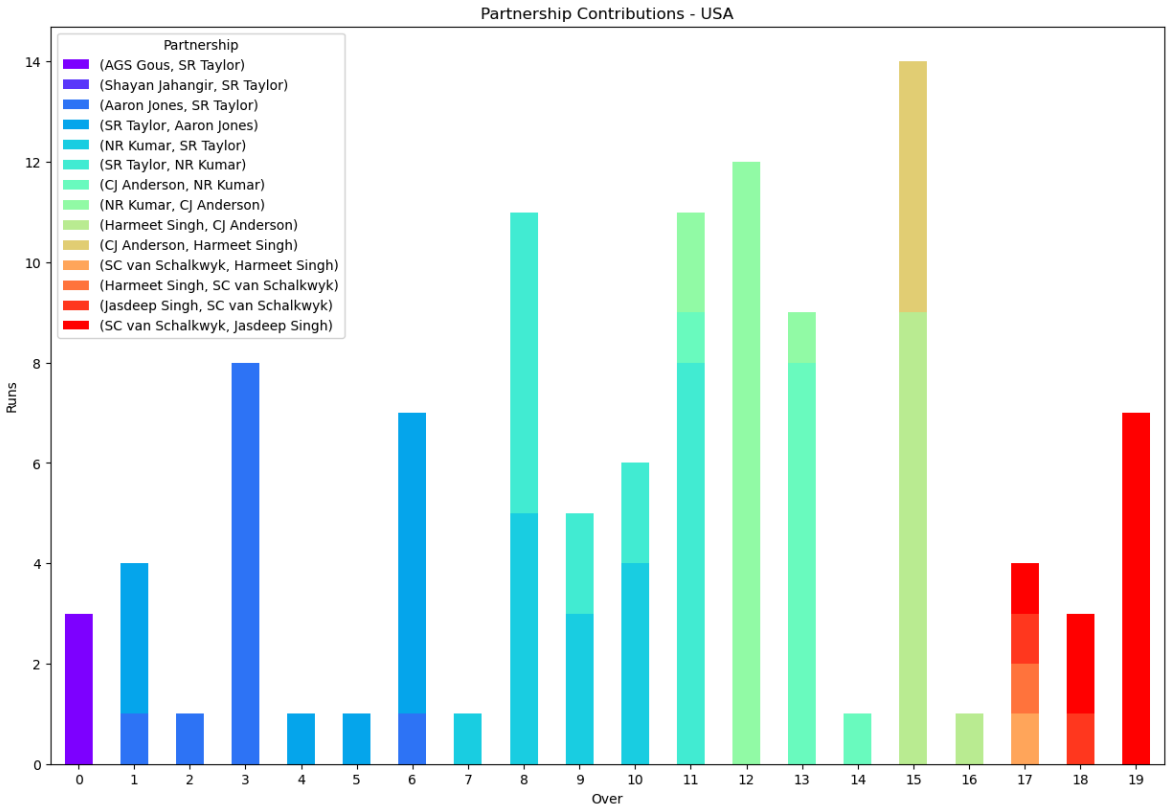
In [61]:
```
import matplotlib.pyplot as plt

# plot grouped bar chart for partnership contribution india
india_partnership_pivot.plot(kind='bar', stacked=True, figsize=(15,10), colormap

plt.title("Partnership Contributions - India")
plt.xlabel("Over")
plt.ylabel("Runs")
plt.legend(title="Partnership")
plt.xticks(rotation=0)
plt.show()
```




The partnership contributions graph for india shows the runs scored by various batting partnerships over each over. Notably, the partnerships of RG Sharma & RR Pant and SA Yadav & S Dube were particularly productive, especially in the middle and the death overs, contributing significantly to the team's total.

**Now, let's have a look at the partnership contribution in the USA's Innings:**

In [64]:
```
# plotting bar chart for contributions of partnership from USA
usa_partnership_pivot.plot(kind="bar", stacked=True, figsize=(15,10),colormap="r

plt.title("Partnership Contributions - USA")
plt.xlabel("Over")
plt.ylabel("Runs")
plt.legend(title="Partnership")
plt.xticks(rotation=0)
plt.show()
```



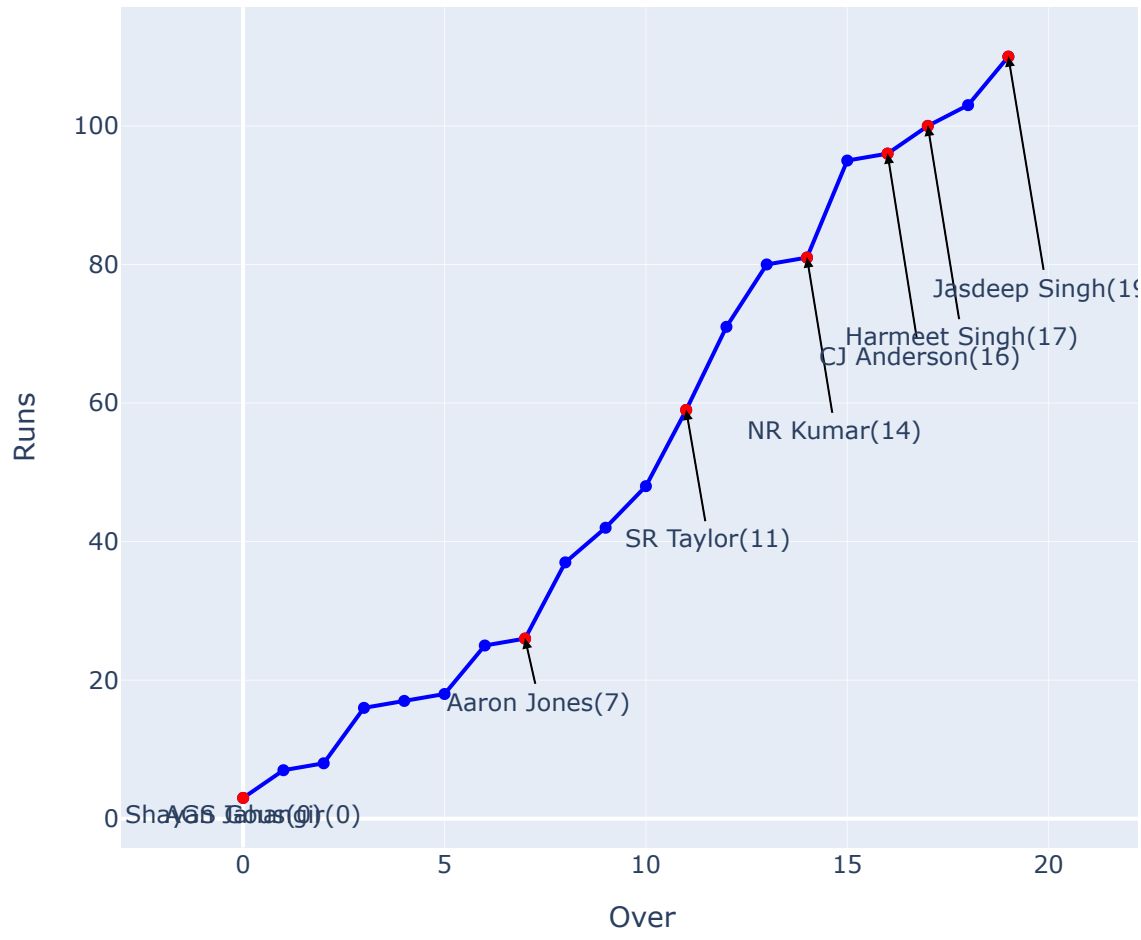

The partnership contributions graph for the USA highlights the runs scored by different batting pairs over each over. Key partnerships such as SR Taylor & Aaron Jones and NR Kumar & SR Taylor significantly boosted the scoring, particularly in the middle and late overs. However, the contributions are more sporadic compared to India, with several Partnerships contributing only marginally.

# Key Moments in Match

**Now, let's have a look at the key moments for the USA in the innings which resulted in a low target:**

In [70]:
```
# cumulative runs for team over the overs
usa_cumulative_runs = data[data["team"]=="United States of America"].groupby("ov

# extract key moments where wickets fall or significant runs were scored
usa_key_moments = data[(data["team"]=="United States of America")& data["wickets
```

```python
# significant runs scored by USA
usa_significant_runs = data[(data["team"]=="United States of America")& data["ru

usa_wickets_fall = data[(data["team"]=="United States of America") & data["wicke

fig = go.Figure()

# plotting figure for cumulative runs
fig.add_trace(go.Scatter(
    x=usa_cumulative_runs.index,
    y = usa_cumulative_runs.values,
    mode = "lines+markers",
    name = "USA Cumulative Runs",
    line = dict(color= "blue")
))

#plotting figure for wickets fall of USA
fig.add_trace(go.Scatter(
    x= usa_wickets_fall.index,
    y = usa_cumulative_runs.loc[usa_wickets_fall.index],
    mode = "markers",
    name = "USA Wickets Fall",
    marker = dict(color= "red")
))

# add annotations for key moments
for _, row in usa_key_moments.iterrows():
  fig.add_annotation(
      x=row["over"],
      y= usa_cumulative_runs.loc[row["over"]],
      text=f"{row['batter']}({row['over']})",
      showarrow= True,
      arrowhead=2,
      ax=row["over"],
      ay = usa_cumulative_runs.loc[row['over']]+5,
      arrowcolor="black"
  )

fig.update_layout(
    title = "USA Key Moments in Innings",
    xaxis_title = "Over",
    yaxis_title = "Runs",
    legend_title = "USA Innings",
    width = 800,
    height = 600,
    autosize=False
)

fig.show()
```

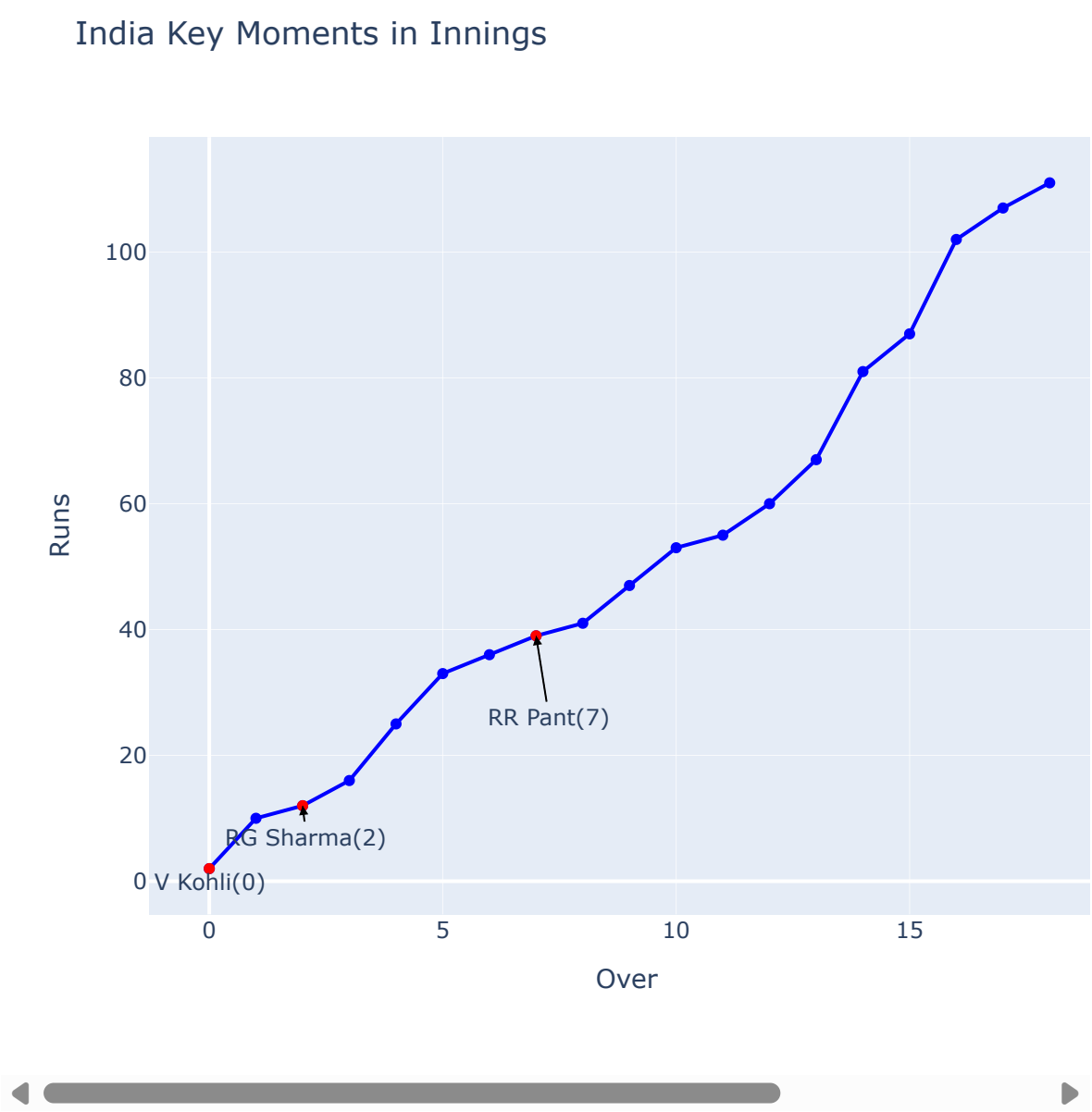



The graph highlights the key moments in the USA's innings, showing the progression of the cumulative run with wickets marked. Early wickets, such as those of Shayan Jahangir and AGS Gous in the first over, set back the USA's momentum. Despite recoveries led by partnerships involving SR Taylor and NR Kumar, regular wickets in the middle and late overs, particularly around the 14th to 19th overs, hindered their progress. This dismissals of key players like Aaron Jones, SR Taylor, and later batsmen such as Harmeet Singh and CJ Anderson, prevented the USA from building a substantial and uniterrupted run flow, ultimately impacting their total score.

**Now, similarly have look at the key moments for India:**

In [74]:
```
# cumulative runs for team over the overs
india_cumulative_runs = data[data["team"]=="India"].groupby("over")["runs_total"

# extract key moments where wickets fall or significant runs were scored
india_key_moments = data[(data["team"]=="India")& data["wickets_0_player_out"].n

# significant runs scored by India
india_significant_runs = data[(data["team"]=="India")& data["runs_total"]>=4]
```

```
india_wickets_fall = data[(data["team"]=="India") & data["wickets_0_player_out"]

fig = go.Figure()

# plotting figure for cumulative runs of India
fig.add_trace(go.Scatter(
    x=india_cumulative_runs.index,
    y = india_cumulative_runs.values,
    mode = "lines+markers",
    name = "India Cumulative Runs",
    line = dict(color= "blue")
))

#plotting figure for wickets fall of India
fig.add_trace(go.Scatter(
    x= india_wickets_fall.index,
    y = india_cumulative_runs.loc[india_wickets_fall.index],
    mode = "markers",
    name = "India Wickets Fall",
    marker = dict(color= "red")
))

# add annotations for key moments
for _, row in india_key_moments.iterrows():
  fig.add_annotation(
      x=row["over"],
      y= india_cumulative_runs.loc[row["over"]],
      text=f"{row['batter']}({row['over']})",
      showarrow= True,
      arrowhead=2,
      ax=row["over"],
      ay = india_cumulative_runs.loc[row['over']]+5,
      arrowcolor="black"
  )

fig.update_layout(
    title = "India Key Moments in Innings",
    xaxis_title = "Over",
    yaxis_title = "Runs",
    legend_title = "India Innings",
    width = 800,
    height = 600,
    autosize=False
)

fig.show()
```




Despite an early setback with the dismissals of V. Kohli and RG Sharma in the first two overs, India Managed to maintain a steady run rate. The wicket of RR Pant in the 7th over was another crucial moment, but subsequent partnerships helped stablize the innings.

# Run Rate Comparison

**Now, let's finish a comparison with run rate per over of both the team:**

In [78]:
```
india_run_rate_per_over = data[data["team"]=="India"].groupby("over")["runs_tota
usa_run_rate_per_over = data[data["team"]=="United States of America"].groupby("

fig = go.Figure()

fig.add_trace(go.Scatter(
    x= india_run_rate_per_over.index,
    y = india_run_rate_per_over.values,
    mode = "lines+markers",
    name = "India Run Rate",
    line = dict(color="blue")
```

```
))

fig.add_trace(go.Scatter(
    x= usa_run_rate_per_over.index,
    y = usa_run_rate_per_over.values,
    mode = "lines+markers",
    line = dict(color="red"),
    name = "USA Run Rate"
))

fig.update_layout(
    title = "Comparison of Run Rate Per Over ",
    legend_title = "Run Rate",
    xaxis_title = "Over",
    yaxis_title = "Runs",
    width = 800,
    height= 600,
    autosize=False
)

fig.show()
```

The USA experienced significant fluctuations in their run rate, with peaks in the 10th and 15th overs, but also several low scoring overs, indicating inconsistency. India's run rate was relatively more stable , with a notable increase towards the end of their innings. This stability in India's run rate, especially in the death overs, allowed them to maintain pressure and chase the target successfully. Tne graph highlights India's ability to keep a more consistent scoring pace, while the USA's variable run rate reflects periods of struggle to maintain momentum.

# Conclusion

In conclusion, India's strategy of consistent scoring, effective partnerships, and a balance bowling attack proved successful against the USA's inconsistent batting perfomance and less impactful bowling. So, this is how you can analyze a cricket match tell the story behind it as a Data Analyst.