Before we get Started:

let's first install the necessary libraries (if required)

In [1]:
```
#!pip install pandas
#!pip install numpy
#!pip install matplotlib
#!pip install seaborn
#!pip install wordcloud
#!pip install plotly
```

Step 1: Import the necessary libraries

First, we need to import the necessary libraries into our Python script.

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
import plotly.graph_objects as go
import plotly.express as px
import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)
```

Step 2: Load the data

Next, we need to load the data that we want to visualize. For this tutorial, we will be using the 2021-2022 NBA Player Stats - Regular dataset, which is openly available in Kaggle.

In [3]:
```
nba_stats = pd.read_csv("2021-2022 NBA Player Stats - Regular.csv",sep=';')

nba_stats.head()
```

Out[3]:

	Rk	Player	Pos	Age	Tm	G	GS	MP	FG	FGA	...	FT%	ORB	DRB	TRB	AST	STL	BLK	TOV	PF	PTS
0	1	Precious Achiuwa	C	22	TOR	73	28	23.6	3.6	8.3	...	0.595	2.0	4.5	6.5	1.1	0.5	0.6	1.2	2.1	9.1
1	2	Steven Adams	C	28	MEM	76	75	26.3	2.8	5.1	...	0.543	4.6	5.4	10.0	3.4	0.9	0.8	1.5	2.0	6.9
2	3	Bam Adebayo	C	24	MIA	56	56	32.6	7.3	13.0	...	0.753	2.4	7.6	10.1	3.4	1.4	0.8	2.6	3.1	19.1
3	4	Santi Aldama	PF	21	MEM	32	0	11.3	1.7	4.1	...	0.625	1.0	1.7	2.7	0.7	0.2	0.3	0.5	1.1	4.1
4	5	LaMarcus Aldridge	C	36	BRK	47	12	22.3	5.4	9.7	...	0.873	1.6	3.9	5.5	0.9	0.3	1.0	0.9	1.7	12.9

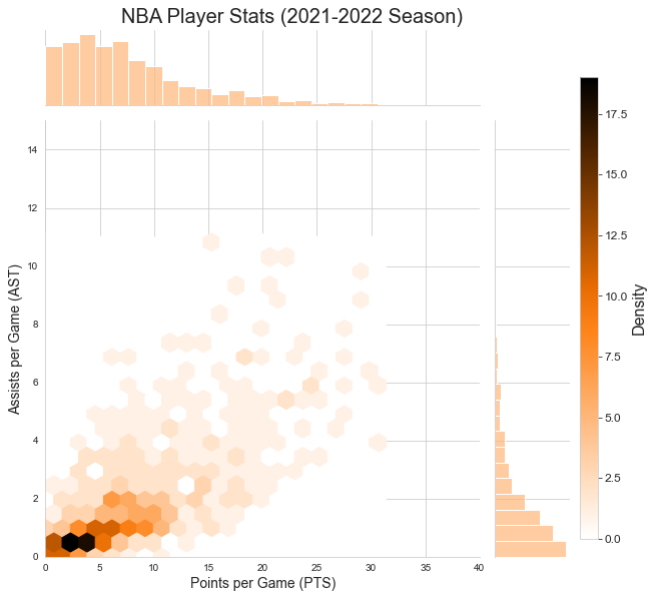
5 rows × 30 columns

Step 3: Creating a Joint Plot

In this code we are creating a joint plot using NBA player statistics for the 2021-2022 season, specifically looking at points per game and assists per game. It uses a hexagonal binning method to visualize the density of data points. The colorbar on the right shows the density of data points in the plot.

In [4]:
```
plt.figure(figsize=(10, 8)) # set the figure size
sns.set_style("whitegrid") # set the style of the plot
sns.set_palette("magma_r") # set the color palette of the plot

# create the jointplot
g = sns.jointplot(x="PTS", y="AST", data=nba_stats, kind="hex",
                  height=8, xlim=(0,40), ylim=(0,15), bins=20, gridsize=20,
                  marginal_kws=dict(bins=20))

# set the title and axis labels
g.fig.suptitle("NBA Player Stats (2021-2022 Season)", fontsize=20, y=1.02)
g.ax_joint.set_xlabel("Points per Game (PTS)", fontsize=14)
g.ax_joint.set_ylabel("Assists per Game (AST)", fontsize=14)

# customize the colorbar
cb_ax = g.fig.add_axes([1, 0.1, 0.03, 0.8])
cb_ax.tick_params(labelsize=12)
cb = plt.colorbar(cax=cb_ax)
cb.set_label('Density', fontsize=16)

plt.show()
```

```
<Figure size 720x576 with 0 Axes>
```



Step 4: Creating a WordCloud

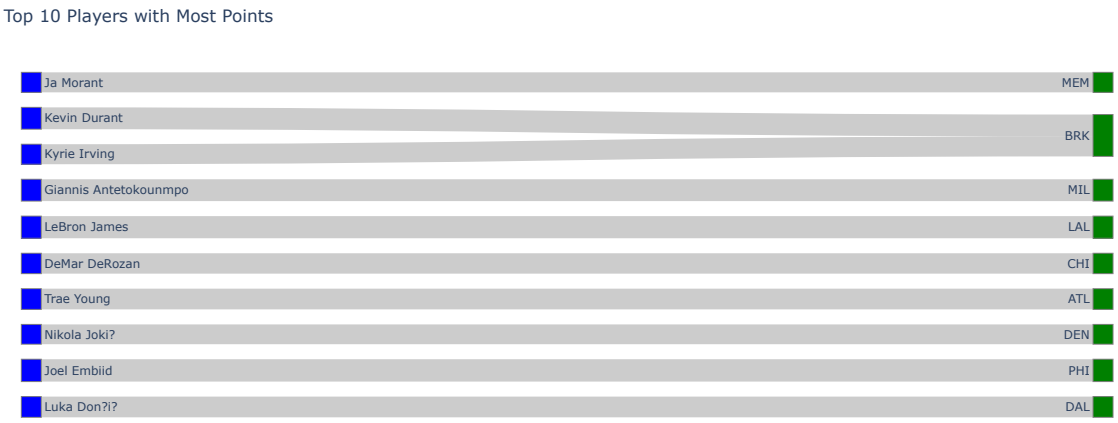
Here we are generating a word cloud from the names of NBA players and their teams. The text is processed to remove common words like "Player", "PTS", "FG", "FT", etc. and a WordCloud object is created using the remaining words. The word cloud is then plotted with a pastel color scheme and a steel blue contour. The resulting plot shows the most frequently occurring names in larger font sizes, creating a visually appealing representation of the most popular players and teams in the NBA.

In [5]:
```
players = nba_stats['Player']
teams = nba_stats['Tm']

# Concatenate the player and team names into a single string
text = " ".join(players + teams)

# Set stopwords
stopwords = set(STOPWORDS)
stopwords.update(["Player", "Tm", "PTS", "FG", "FGA", "FT", "FTA"])

# Generate a word cloud object
wordcloud = WordCloud(width=800, height=600, background_color='white', stopwords=stopwords,
                      contour_width=2, contour_color='steelblue', colormap='Pastel1', random_state=42).generate(text)

# Plot the word cloud
plt.figure(figsize=(10, 10))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.tight_layout(pad=0)

# Save the word cloud image
#plt.savefig("basketball_wordcloud.png", dpi=300)

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


Step 5: Creating a Sankey Diagram

In this plot, we are creating a Sankey diagram using plotly to visualize the top 10 players with the most points in the 2021-2022 NBA season and the teams they play for. It first selects the top 10 players with the most points and creates a list of all unique teams. It then creates nodes for each player and team, and links between the players and their respective teams based on the number of points the player scored. The resulting plot shows the flow of points from the top 10 players to their respective teams.

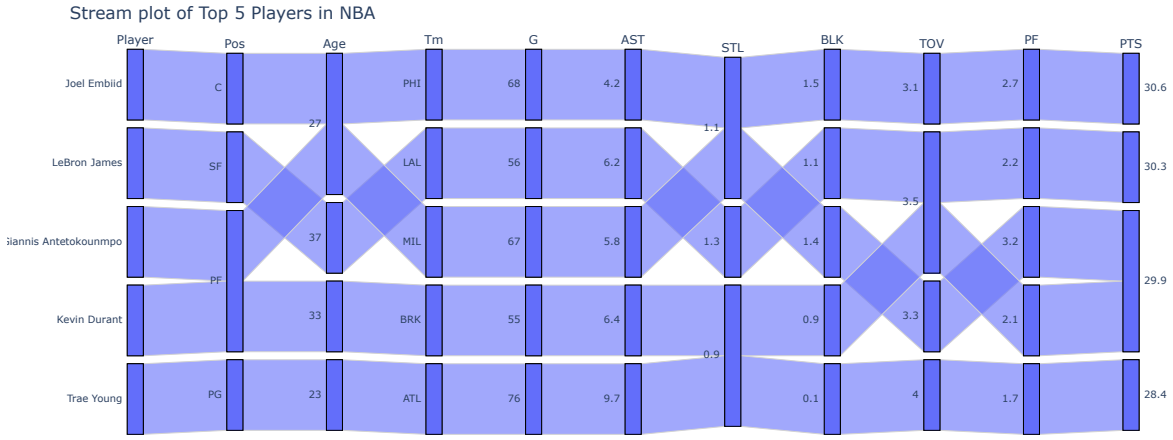
In [6]:
```
# Select the top 10 players with the most points
top_10_players = nba_stats.sort_values("PTS", ascending=False).head(10)

# Create a list of all unique teams
teams = nba_stats["Tm"].unique()

# Create nodes for players and teams
node_labels = list(top_10_players["Player"]) + list(teams)
node_values = [value for value in top_10_players["PTS"]] + [1]*len(teams)

# Create links between players and teams
link_sources = [node_labels.index(player) for player in top_10_players["Player"]]
link_targets = [len(top_10_players) + list(teams).index(team) for team in top_10_players["Tm"]]
link_values = list(top_10_players["PTS"])

# Create the Sankey diagram using plotly
fig = go.Figure(data=[go.Sankey(
    node = dict(
      pad = 15,
      thickness = 20,
      line = dict(color = "grey", width = 0.5),
      label = node_labels,
      color = ["blue"]*len(top_10_players) + ["green"]*len(teams)
    ),
    link = dict(
      source = link_sources, # indices correspond to labels, eg A1, A2, A1, B1, ...
      target = link_targets,
      value = link_values
  ))])

# Set layout properties for the diagram
fig.update_layout(
    title = "Top 10 Players with Most Points",
    font=dict(size=12)
)

# Display the Sankey diagram in the notebook
fig.show()
```



Step 6: Creating a Streamplot

Here we are creating a stream plot using the Plotly Express library. The data used for the plot is the top 5 players with the most points from the NBA statistics dataset. The stream plot shows how the players are distributed based on various categorical variables such as their position, team, and game statistics.

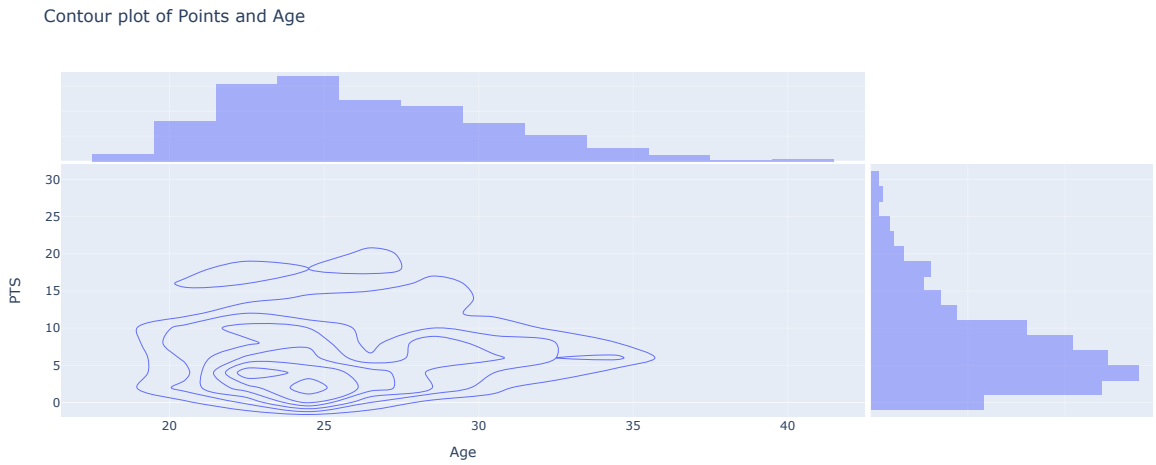
In [7]:
```
# Select the top 5 players with the most points
top_5_players = nba_stats.sort_values("PTS", ascending=False).head(5)
top_5_players = top_5_players[['Player', 'Pos', 'Age', 'Tm', 'G','AST', 'STL', 'BLK', 'TOV', 'PF', 'PTS']]

# Create a Stream plot
fig = px.parallel_categories(top_5_players)
fig.update_layout(title="Stream plot of Top 5 Players in NBA")

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



Step 7: Creating a Countour Plot

In this plot we are creating a contour plot using the plotly, with age on the x-axis, points on the y-axis, and steals on the z-axis. The plot shows the density of points and age data with contour lines, and the marginal histograms on the sides.

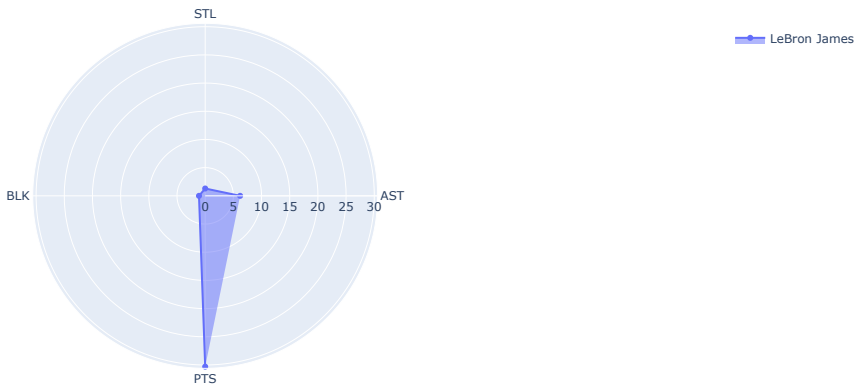
In [8]:
```
# Select a subset of the data
subset = nba_stats[['Age', 'PTS', 'STL']]

# Create a contour plot
fig = px.density_contour(subset, x="Age", y="PTS", z="STL", title="Contour plot of Points and Age",
                         marginal_x="histogram", marginal_y="histogram")

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



Step 8: Creating a Polar Plot

Finally we create polar plot of LeBron James' season stats. It selects all rows in the NBA stats dataset where the player is "LeBron James" and creates a scatterpolar trace for each row. Each trace represents one season and shows the number of assists, steals, blocks, and points LeBron James scored in that season. The polar plot has a radial axis that goes from 0 to the maximum number of points scored by LeBron James in a season.

In [9]:
```
import plotly.graph_objects as go

# Select the top 10 players with the most points
lebron_players = nba_stats[nba_stats['Player'] == "LeBron James"]

# Create a trace for each player
traces = []
for i, player in lebron_players.iterrows():
    trace = go.Scatterpolar(
        r = [player["AST"], player["STL"], player["BLK"], player["PTS"]],
        theta = ["AST", "STL", "BLK", "PTS"],
        fill = "toself",
        name = player["Player"]
    )
    traces.append(trace)

# Create the layout for the Polar plot
layout = go.Layout(
    polar=dict(
        radialaxis=dict(
            visible=True,
            range=[0, top_10_players["PTS"].max()]
        )
    ),
    showlegend=True,
    title="Lebron James Season Stats"
)

# Create the figure for the Polar plot
fig = go.Figure(data=traces, layout=layout)

# Display the Polar plot in the notebook
fig.show()
```



Conclusion

- In this tutorial, we covered the advance data visualization techniques in Python using the Matplotlib, Wordcloud and Plotly libraries.
- We showed how to create Joint Plot, WordCloud, Sankey Diagram, Stream Plot, Contour Plot and Polar Plot using Python code.
- By following these steps, you should now have a good understanding of how to create various types of plots in Python for data analysis and visualization.
- **Follow Muhammad Bilal Alam for more tutorials like this**