```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

import plotly.express as px
import plotly.io as pio
pio.renderers.default = 'notebook'
```

```
pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
```

```
df=pd.read_csv('flight_data.csv')
```

```
df.head()
```

	Date of Booking	Date of Journey	Airline-Class	Departure Time	Arrival Time	Duration	Total Stops	Price
0	15/01/2023	16/01/2023	SpiceJet \nSG-8169\nECONOMY	20:00\nDelhi	22:05\nMumbai	02h 05m	non-stop	5,335
1	15/01/2023	16/01/2023	Indigo \n6E-2519\nECONOMY	23:00\nDelhi	01:20\nMumbai	02h 20m	non-stop	5,899
2	15/01/2023	16/01/2023	GO FIRST \nG8- 354\nECONOMY	22:30\nDelhi	00:40\nMumbai	02h 10m	non-stop	5,801
3	15/01/2023	16/01/2023	SpiceJet \nSG-8709\nECONOMY	18:50\nDelhi	20:55\nMumbai	02h 05m	non-stop	5,794
4	15/01/2023	16/01/2023	Air India \nAI-805\nECONOMY	20:00\nDelhi	22:10\nMumbai	02h 10m	non-stop	5,955

```
df1=df.copy()
```

Pre-processing Data

```
def clean_flight_data(df):

    #Airline Class
    df['Airline-Name']=df['Airline-Class'].str.split('\n').str[0].str.strip()

    df['flight_code']=df['Airline-Class'].str.split('\n').str[1].str.strip()

    df['Class'] = df['Airline-Class'].str.split('\n').str[-1].str.strip()

    #journey date
    df['Date of Journey']=pd.to_datetime(df['Date of Journey'],format='%d/%m/%Y')

    #date of booking
    df['Date of Booking']=pd.to_datetime(df['Date of Booking'],format='%d/%m/%Y')

    #days before flight
    df['days_before_flight']=(df['Date of Journey']-df['Date of Booking']).dt.days

    # day of booking
    df['journey_day_name']=df['Date of Journey'].dt.day_name()



    # Departure City
    df['Departure City']=df['Departure Time'].str.split('\n').str[1].str.strip()

    # Arrival City
    df['Arrival City']=df['Arrival Time'].str.split('\n').str[1].str.strip()


    #TotalStops Cleanup
    df['Total Stops'] = df['Total Stops'].str.replace(r'\n\s*\t*', '', regex=True)
    df['Total Stops'] = df['Total Stops'].str.replace(r'(stop).*', r'\1', regex=True)

    #Departure Time
    df['Departure_Time']=df['Departure Time'].str.split('\n').str[0].str.strip()

    #Arrival Time
    df['Arrival_Time']=df['Arrival Time'].str.split('\n').str[0].str.strip()

    #Duration
    df['Duration'] = df['Duration'].str.extract(r'(\d+)h (\d+)m') \
                                  .astype(float) \
                                  .apply(lambda x: round(x[0] + x[1] / 60, 4), axis=1)
    # Extract hour from 'arrival' column
    df['arrival_time'] = pd.to_datetime(df['Arrival_Time'], format='%H:%M').dt.hour

    # Categorize arrivals based on the hour
    df['arrival_category'] = df['arrival_time'].apply(lambda x: 'Before 7pm' if x < 19 else 'After 7pm')

    #drop columns
    df.drop(['Date of Booking','Date of Journey','Airline-Class','Departure Time','Arrival Time','arrival_time'],axis=1,inplace=True)

    return df


df = clean_flight_data(df)


df.head()
```

	Duration	Total Stops	Price	Airline-Name	flight_code	Class	days_before_flight	journey_day_name	Departure City	Arrival City	Departure_Tim
0	2.0833	non-stop	5,335	SpiceJet	SG-8169	ECONOMY	1	Monday	Delhi	Mumbai	20:0
1	2.3333	non-stop	5,899	Indigo	6E-2519	ECONOMY	1	Monday	Delhi	Mumbai	23:0
2	2.1667	non-stop	5,801	GO FIRST	G8- 354	ECONOMY	1	Monday	Delhi	Mumbai	22:3

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 452088 entries, 0 to 452087
Data columns (total 13 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   Duration            452088 non-null  float64
 1   Total Stops         452088 non-null  object
 2   Price               452088 non-null  object
```

```
 3   Airline-Name        452088 non-null  object
 4   flight_code         452088 non-null  object
 5   Class               452088 non-null  object
 6   days_before_flight  452088 non-null  int64
 7   journey_day_name    452088 non-null  object
 8   Departure City      452088 non-null  object
 9   Arrival City        452088 non-null  object
 10  Departure_Time      452088 non-null  object
 11  Arrival_Time        452088 non-null  object
 12  arrival_category    452088 non-null  object
dtypes: float64(1), int64(1), object(11)
memory usage: 44.8+ MB
```

```
df['Price'] = df['Price'].replace(',', '', regex=True).astype(int)
```

EDA

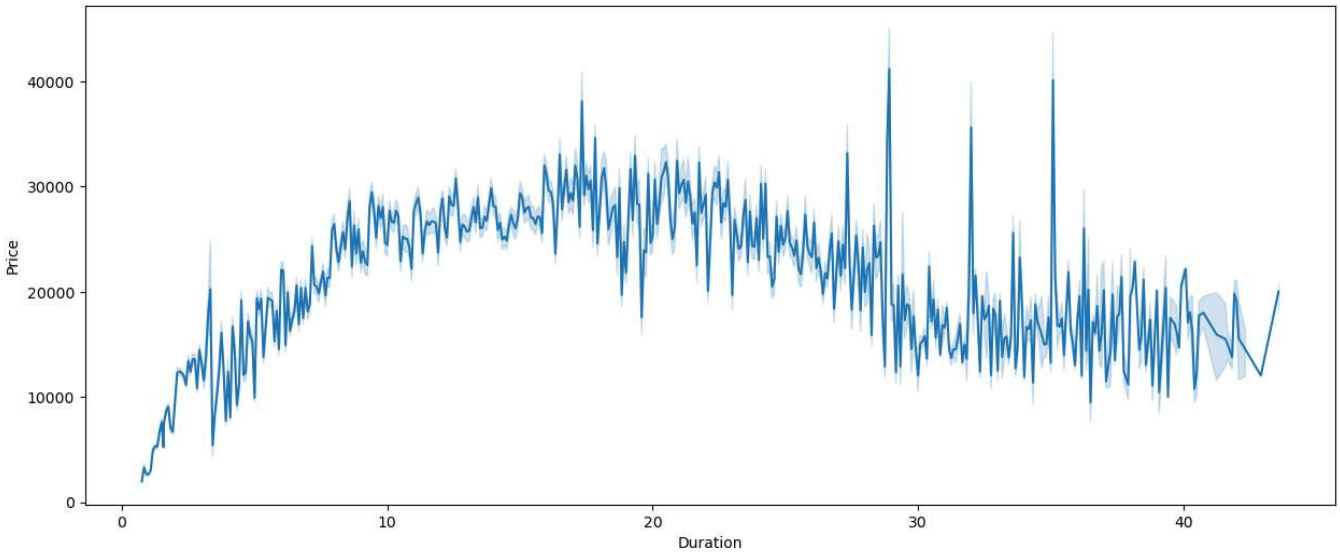
Duration

```
fig = px.box(df, x='Total Stops',y='Duration')

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

- non-stop flights - 50% of flights have a duration of +-2hrs
- 1-stop flights - 50% of flights have a duration of +-12hrs
- 2+-stop flights - 50% of flights have a duration of +-16hrs

```
plt.figure(figsize=(15,6))

sns.lineplot(data=df,x='Duration',y='Price')
```

```
<Axes: xlabel='Duration', ylabel='Price'>
```



- The lower the flight duration the lower the price.
- The higher the flight duration the higher the price

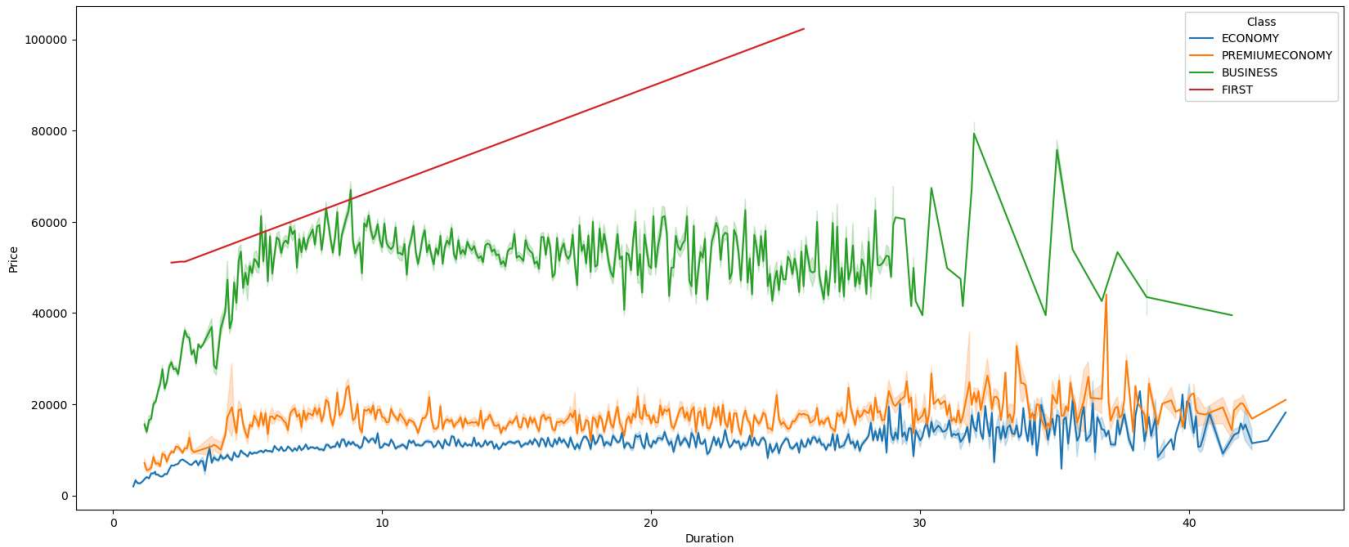
```
## Total Stops

fig = px.histogram(df, y='Total Stops')

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

```
fig = px.box(df, x='Total Stops',y='Price')

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

Price

```
plt.figure(figsize=(20,8))
sns.lineplot(data=df,x='Duration',y='Price',hue='Class')
```

```
<Axes: xlabel='Duration', ylabel='Price'>
```

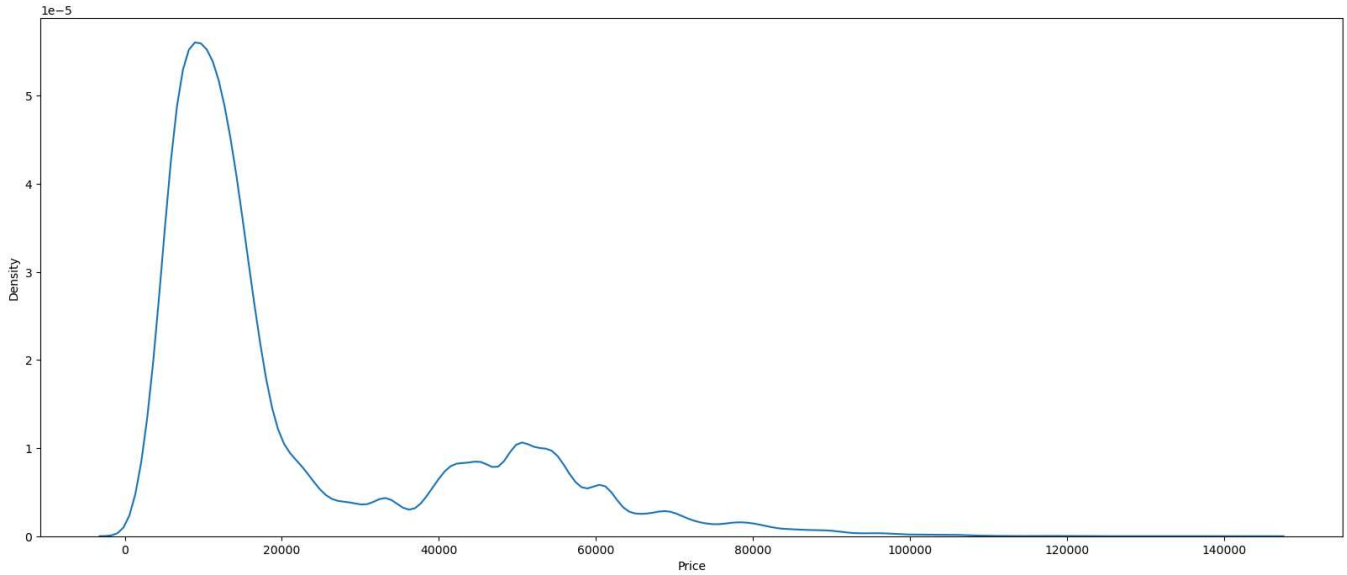



```
fig = px.box(df,x='Class',y='Price')
fig.show()
```

Start coding or generate with AI.

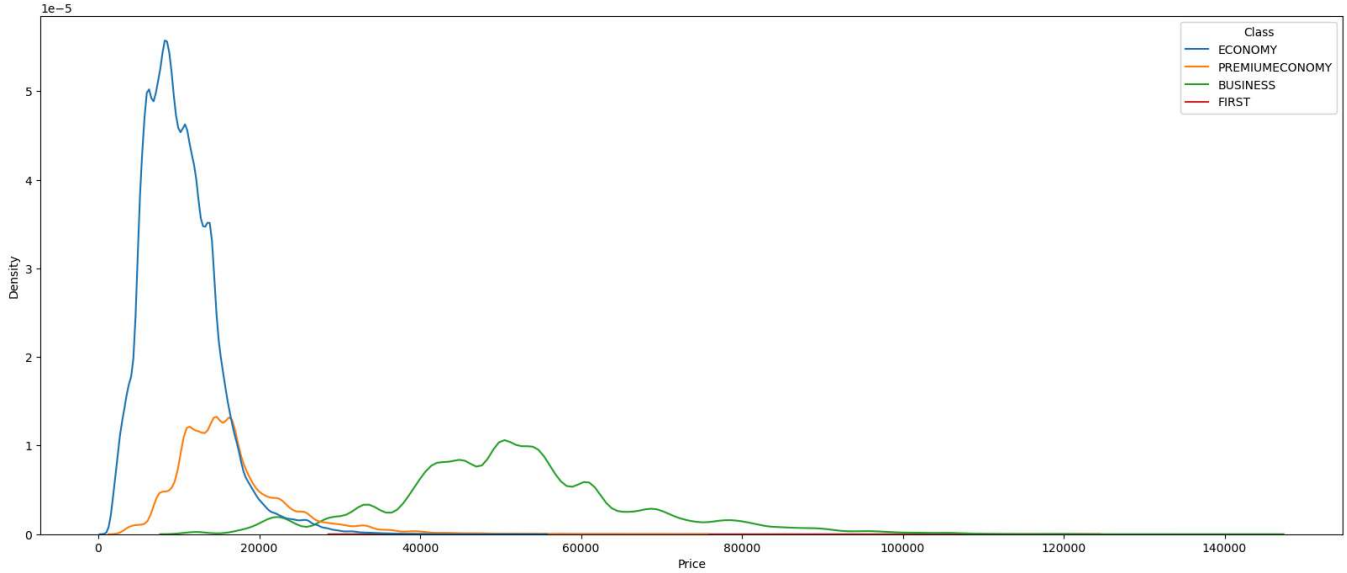
```
plt.figure(figsize=(20,8))
sns.kdeplot(data=df, x="Price")
```

```
<Axes: xlabel='Price', ylabel='Density'>
```




```
plt.figure(figsize=(20,8))
sns.kdeplot(data=df, x="Price",hue='Class')
```

```
<Axes: xlabel='Price', ylabel='Density'>
```

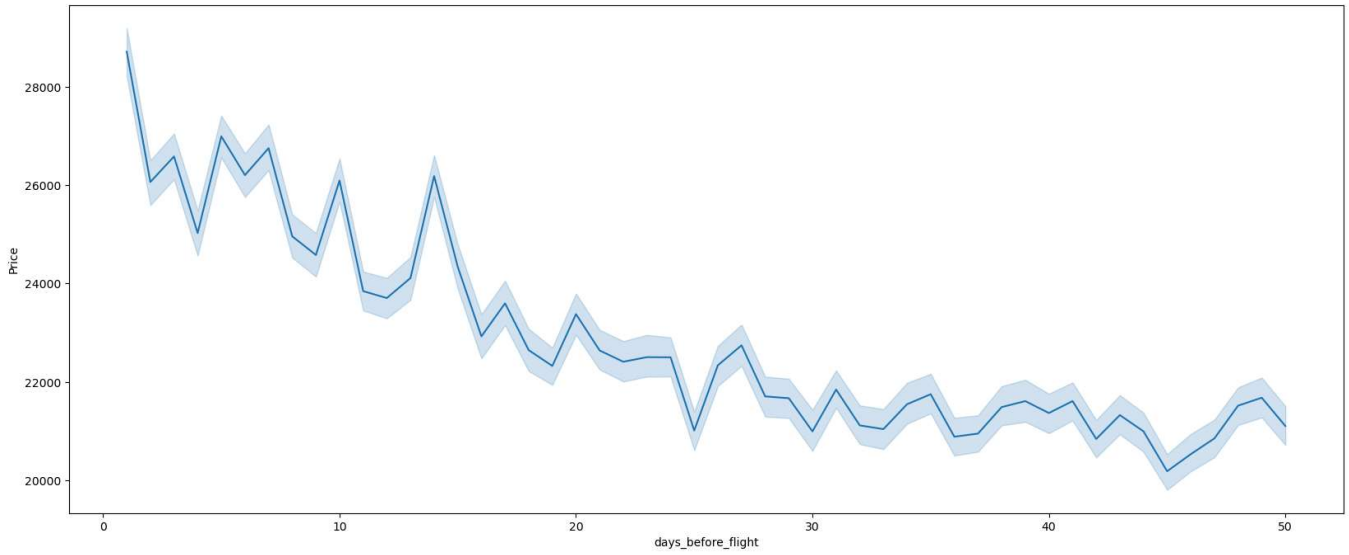


- Majority of airline price's are clustered to the left
- Much more larger values to the right

Days Before Flight

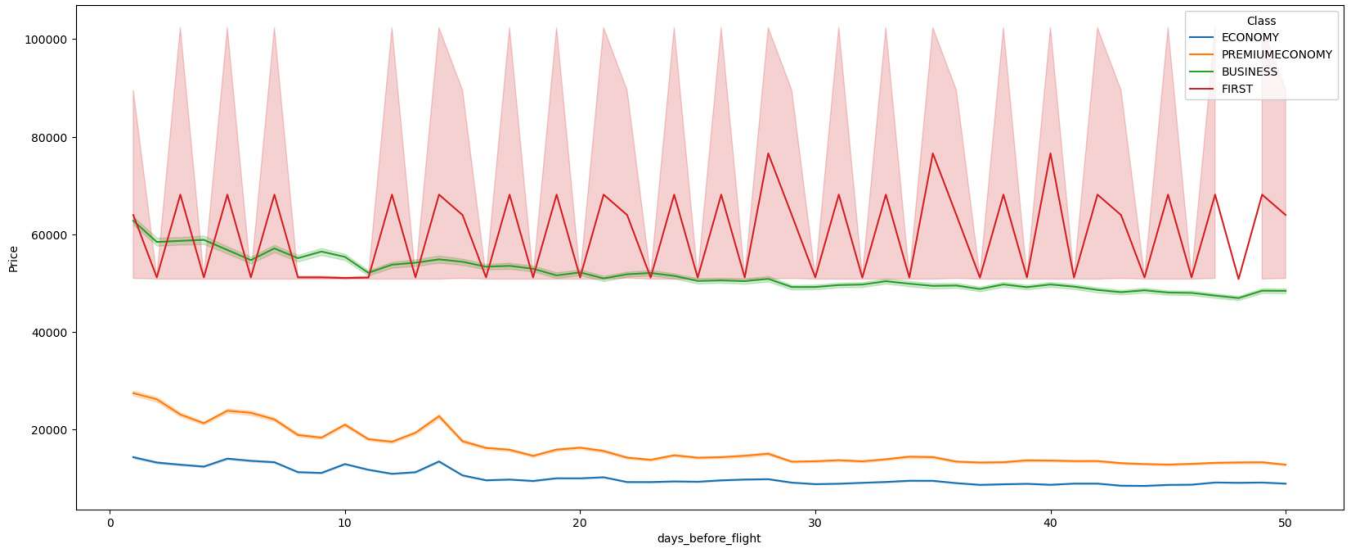
```
plt.figure(figsize=(20,8))
sns.lineplot(data=df,x='days_before_flight',y='Price')
```

```
<Axes: xlabel='days_before_flight', ylabel='Price'>
```




```
plt.figure(figsize=(20,8))
sns.lineplot(data=df,x='days_before_flight',y='Price',hue='Class')
```

```
<Axes: xlabel='days_before_flight', ylabel='Price'>
```



- The longer the days before flight the lower the price and vice-versa for Economy,Premium Econony and Business
- For First Class, they is variability of the price with days before flight

```
## Days before flight
fig = px.box(df, x='days_before_flight')
fig.show()
```

Journey Day Name

```
fig = px.histogram(df, x='journey_day_name')

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

- Monday is the day with the most flights

```
fig = px.box(df, x='journey_day_name',y='Price',color='Class')

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

Airline Class and Name

```
fig = px.box(df, x='Airline-Name',y='Price')

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

```
fig = px.box(df, x='Airline-Name',y='Price',color='Class')

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

```
fig = px.box(df, x='Airline-Name',y='Price',color='Total Stops')

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

```
fig = px.box(df, x='Class',y='Price')

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

Departure & Arrival City

```
fig = px.box(df, x='Departure City',y='Price')

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

```
fig = px.box(df, x='Arrival City',y='Price')

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

Arrival Category

```
fig = px.histogram(df, x='arrival_category')

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