A PROJECT UNDERSTANDING VISUALIZATIONS

In [1]:
```
# Import tools that we will be using.

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime
from pandas_profiling import ProfileReport # Importing pandas profiling so we can have a good idea of our data.

# This line will read the csv file.

df = pd.read_csv('SampleSuperstoreNew.csv', encoding='windows-1252')

# This part of the code is just to supress any unwanted warnings that may appear.

import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
profile = ProfileReport(df, title='Pandas Profiling Sampe Super Store', html={'style':{'full_width':True}})
profile
```

```
Summarize dataset:   0%|          | 0/5 [00:00<?, ?it/s]
Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]
Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]
```

Pandas Profiling Sampe Super Store

Overview | Variables | Interactions | Correlations | Missing values | Sample

Overview

Overview | Alerts 30 | Reproduction

Dataset statistics

Number of variables	21
Number of observations	9994
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	1.6 MiB
Average record size in memory	168.0 B

Variable types

Numeric	6
Categorical	15

Variables

Row ID
Real number (ℝ≥0)

Distinct	9994	**Minimum**	1
Distinct (%)	100.0%	**Maximum**	9994

Out[2]:

In [3]:
```
# And this line(df.head()) will show the first 5 rows of the DataFrame
df.head().style.set_properties(**{"background-color": "#ff6282","color": "black",
                                  "border-color": "black","font-size":"5pxpx0pt","width': 200})
```

Out[3]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	State	Postal Code	Region	Product ID
0	1	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky	42420	South	FUR-BO-10001798
1	2	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky	42420	South	FUR-CH-10000454
2	3	CA-2016-138688	6/12/2016	6/16/2016	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	California	90036	West	OFF-LA-10000240
3	4	US-2015-108966	10/11/2015	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Florida	33311	South	FUR-TA-10000577
4	5	US-2015-108966	10/11/2015	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Florida	33311	South	OFF-ST-10000760

In [4]:
```
df['Country'].unique() # Deleting the column of country because we are working a data frame that everything is
df.drop('Country', axis=1)
```

Out[4]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	City	State	Postal Code	Region	Product ID
0	1	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	Henderson	Kentucky	42420	South	FUR-BO-10001798
1	2	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	Henderson	Kentucky	42420	South	FUR-CH-10000454
2	3	CA-2016-138688	6/12/2016	6/16/2016	Second Class	DV-13045	Darrin Van Huff	Corporate	Los Angeles	California	90036	West	OFF-LA-10000240
3	4	US-2015-108966	10/11/2015	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	Fort Lauderdale	Florida	33311	South	FUR-TA-10000577
4	5	US-2015-108966	10/11/2015	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	Fort Lauderdale	Florida	33311	South	OFF-ST-10000760
...	...	...	...	...	...	...	...	...	...	...	...	...	...
9989	9990	CA-2014-110422	1/21/2014	1/23/2014	Second Class	TB-21400	Tom Boeckenhauer	Consumer	Miami	Florida	33180	South	FUR-FU-10001889
9990	9991	CA-2017-121258	2/26/2017	3/3/2017	Standard Class	DB-13060	Dave Brooks	Consumer	Costa Mesa	California	92627	West	FUR-FU-10000747
9991	9992	CA-2017-121258	2/26/2017	3/3/2017	Standard Class	DB-13060	Dave Brooks	Consumer	Costa Mesa	California	92627	West	TEC-PH-10003645
9992	9993	CA-2017-121258	2/26/2017	3/3/2017	Standard Class	DB-13060	Dave Brooks	Consumer	Costa Mesa	California	92627	West	OFF-PA-10004041
9993	9994	CA-2017-119914	5/4/2017	5/9/2017	Second Class	CC-12220	Chris Cortes	Consumer	Westminster	California	92683	West	OFF-AP-10002684

9994 rows × 20 columns

THE MAIN MOTIVATION

Before we begin, let me start by introducing myself...

About Myself: I am a Data Science Student currently studying at Integrify. And my day to class is to work with various types and sources of data and provide business insights on the problem that the stakeholders are trying to solve. This allows me to work with Python and SQL for the data pre-processing and handling part and further provide the necessary insights by visualizing.

Throughout this notebook, we will be tackling **various business problem** related scenarios to provide necessary insights.

COMPARING CATEGORICAL AND NUMERICAL VALUES

In [35]:
```
# First of all, we are going to take only the subset of data for our purpose.

df_bar = df[['Region','Sales']]

# Grouping the DataFrame based on Regions and taking mean of profit across each of them.

# The reason for sorting is to show the larger values at the beginning.

df_bar = df_bar.groupby('Region').mean().sort_values(by='Sales', ascending=False)

plt.figure(figsize=(15,8)) # Setting the figure size.

# Visualizing using Bar Chart.
plt.suptitle("Average Sales Across Different Regions", size=20)

# Plotting the BarChart.
plt.subplot(1,2,1)
plt.bar(x=df_bar.index, height='Sales', color=['#F05454','#30475E','#222831','#DDDDDD'], data=df_bar)

# Plotting the Horizontal BarChart (Use this if there are many unique values for a Categorical Feature)
plt.subplot(1,2,2)
plt.barh(y=df_bar.index, width='Sales', color=['#F05454','#30475E','#222831','#DDDDDD'], data=df_bar)
plt.gca().invert_yaxis() # Inverting the Y Axis.

plt.show()
```

CHECK TIME TREND

Consider The Following Business Problem Statement: Your Stakeholders would like to understand the trend for Average Profit and Sales across all these years.

In this case, the Profit and Sales are your Numerical features and your Order Date is your Date/Time Periods Feature. Therefore, a Line Chart is the best visualization to convey your insights! This is because, with Line Charts, you can show trend of both Sales and Profit over the Time Period.

In [11]:
```
# First of all, we are going to take only the subset of data for our purpose. (To keep things simple)
df_line = df[['Order Date','Sales','Profit']].sort_values('Order Date') # Chronological Ordering
df_line['Order Date'] = pd.to_datetime(df_line['Order Date']) # Converting into DateTime
df_line = df_line.groupby('Order Date').mean() # Groupby to get the average Sales and Profit on each day

# Visualizing the Line Chart
plt.figure(figsize=(15,8))
plt.plot(df_line.index, 'Sales', data=df_line, color='#F05454') # Avg Sales over Time
plt.plot(df_line.index, 'Profit', data=df_line, color='#30475E') # Avg Profit over Time
plt.title("Average Sales and Profit over Time Period(2014-2018)", size=20, pad=20)

plt.show()
```

CHECK DISTRIBUTION

In [12]:
```
# First of all, we are going to take only the subset of data for our purpose. (To keep things simple)
df_dist = df[['Sales','Profit']]

##########################################################
# Fixing the Outliers (It is okay if you do not understand this)
'''I am doing this so that we can see the visuals better. Do not
do this randomly'''

df_dist = df_dist[df_dist['Sales'] <= 2000]
df_dist = df_dist[df_dist['Profit'] <= 2000]
df_dist = df_dist[df_dist['Profit'] >= -200]

##########################################################

# Visualizing the histograms
plt.figure(figsize=(20,10))

plt.suptitle("Checking Distribution and Outliers for Sales and Profit", size=20)

# Histogram for Sales
plt.subplot(2,3,1)
plt.hist(df_dist['Sales'], bins=200, color='#F05454')
plt.xlim(0,1000)

# Boxplot of  for Sales
plt.subplot(2,3,2)
sns.boxplot(df_dist['Sales'], color='#F05454')

# KDEPlot for Sales
plt.subplot(2,3,3)
sns.kdeplot(x=df_dist['Sales'], color='#F05454')

##########################################################
# Histogram for Profit
plt.subplot(2,3,4)
plt.hist(df_dist['Profit'], bins=200, color='#30475E')
plt.xlim(-250,300)

# Boxplot of  for Profit
plt.subplot(2,3,5)
sns.boxplot(df_dist['Profit'], color='#30475E')

# KDEPlot for Sales
plt.subplot(2,3,6)
sns.kdeplot(x=df_dist['Profit'], color='#30475E')

plt.show()
```

COMPOSITION OF FEATURES

In [13]:
```
# Taking a subset of data (To keep things simple)
df_stackb = df[['Ship Mode','Sales','Profit']]

# taking a groupby on category and then ship mode...
df_stackb = df_stackb.groupby(['Ship Mode']).sum().reset_index()

# Visualizing the Stacked BarChart
plt.figure(figsize=(13,18))
plt.subplot(3,1,1)
plt.bar(x=df_stackb['Ship Mode'], height=df_stackb['Sales'], color='#F05454')
plt.bar(x=df_stackb['Ship Mode'], height=df_stackb['Profit'], bottom=df_stackb['Sales'], color='#30475E')
plt.title("Sales & Profit Across Ship Modes", size=20, pad=20)
plt.legend(['Sales','Profit'])

# Taking a subset of data (To keep things simple)
df_pie = df[['Category','Sales','Profit']]
# taking a groupby on category and then ship mode...
df_pie = df_pie.groupby(['Category']).sum().reset_index()

# Visualizing the Pie Chart (Profit)
plt.subplot(3,2,3)
plt.pie(df_pie['Profit'], labels=df_pie['Category'], colors=['#F05454','#30475E','#222831','#DDDDDD'])
plt.title("Composition of Category for Profit")

# Visualizing the Pie Chart (Sales)
plt.subplot(3,2,4)
plt.pie(df_pie['Sales'], labels=df_pie['Category'], colors=['#F05454','#30475E','#222831','#DDDDDD'])
plt.title("Composition of Category for Sales")

plt.show()
```