In [1]:

```
import pandas as pd
import plotly.express as px
from scipy.stats import f_oneway
from dython import nominal
```

In [2]:

```
df=pd.read_csv('data/Global_Superstore2.csv', encoding = "ISO-8859-1")
```

Entfernen aller Spalten, die in diesem Notebook nicht untersucht werden

In [3]:

```
def deleteColumns(pColumns):
    for i in range(0,len(pColumns)):
        del df[pColumns[i]]
```

In [4]:

```
deleteColumns(["Order ID", "Ship Date", "City", "State", "Country", "Market", "Region",
"Postal Code", "Product ID", "Product Name", "Category", "Sub-Category","Sales","Quanti
ty","Discount","Shipping Cost","Order Priority"])
```

In [5]:

```
df
```

Out[5]:

	Row ID	Order Date	Ship Mode	Customer ID	Customer Name	Segment	Profit
0	32298	31-07-2012	Same Day	RH-19495	Rick Hansen	Consumer	762.1845
1	26341	05-02-2013	Second Class	JR-16210	Justin Ritter	Corporate	-288.7650
2	25330	17-10-2013	First Class	CR-12730	Craig Reiter	Consumer	919.9710
3	13524	28-01-2013	First Class	KM-16375	Katherine Murray	Home Office	-96.5400
4	47221	05-11-2013	Same Day	RH-9495	Rick Hansen	Consumer	311.5200
...	...	...	...	...	...	...	...
51285	29002	19-06-2014	Same Day	KE-16420	Katrina Edelman	Corporate	4.5000
51286	35398	20-06-2014	Standard Class	ZC-21910	Zuschuss Carroll	Consumer	-1.1100
51287	40470	02-12-2013	Same Day	LB-16795	Laurel Beltran	Home Office	11.2308
51288	9596	18-02-2012	Standard Class	RB-19795	Ross Baird	Home Office	2.4000
51289	6147	22-05-2012	Second Class	MC-18100	Mick Crebagga	Consumer	1.8000

51290 rows × 7 columns

Definieren von Funktionen, die die "Clean Code Guidelines" erfüllen und im ganzen Dokument zur Analyse von Attributbeziehungen genutzt werden

In [6]:

```
def countColumn(pColumn, pColumnName, pYName):
    groups=df.groupby(pColumn)
    amount=groups.count()[["Row ID"]]
    dataset = pd.DataFrame({pColumnName: list(df.groupby(pColumn).groups.keys()), pYNam
e: amount["Row ID"]}, columns=[pColumnName, pYName])
    colour=amount["Row ID"]
    return dataset,colour
```

In [7]:

```
def dfOfAverageMeans(pColumn, pValue, pColumnName, pYName):
    means = []
    groups=df.groupby(pColumn)
    for index,group in groups:
        current = group[pValue]
        currentMean = current.mean()
        means.append(currentMean)
    dataset = pd.DataFrame({pColumnName: list(df.groupby(pColumn).groups.keys()), pYNam
e: means}, columns=[pColumnName, pYName])
    return dataset, means
```

In [8]:

```
def twoStepSunburst(pColumn1, pColumn2):
    dataframe = df.groupby(by=[pColumn1, pColumn2]).count()[["Row ID"]].rename(columns=
{"Row ID":"Anzahl"})
    dataframe = dataframe.reset_index()
    fig = px.sunburst(dataframe,path=[pColumn1, pColumn2], values="Anzahl")
    fig.show()
```

In [9]:

```
def numberOfTransactions(pColumn):
    count_r=df.groupby(by=pColumn).count()[["Row ID"]].rename(columns={"Row ID":"Number
of Transactions"})
    return count_r.sort_values(by="Number of Transactions")
```

In [10]:

```
def howOftenDoAmountsAppear(pColumn):
    counter=df.groupby(by=pColumn).count()[["Row ID"]].rename(columns={"Row ID":"Number
of Transactions"})
    counting_amounts=counter.groupby(['Number of Transactions']).size().reset_index(nam
e='counts')
    counting_amounts.sort_values(by="Number of Transactions")
    return counting_amounts
```

In [11]:

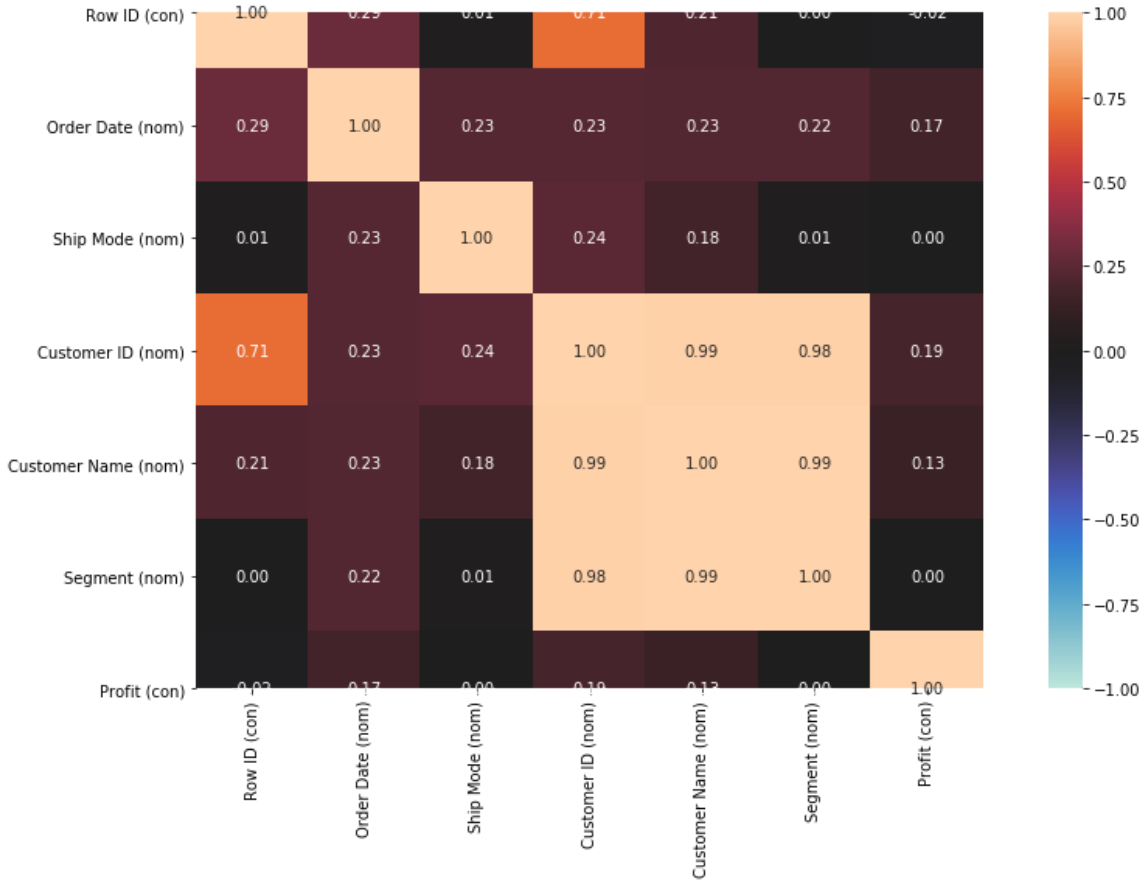
```
def calculateAnovaScore(pColumn1, pColumn2):
    CategoryGroupLists=df.groupby(pColumn1)[pColumn2].apply(list)
    AnovaResults = f_oneway(*CategoryGroupLists)
    print('P-Wert für Anova ist: ', AnovaResults[1])
    print('F Score für Anova ist: ', AnovaResults[0])
    if AnovaResults[1] < 0.05:
        print(f"Der Unterschied der {pColumn2}-Mittelwerte zwischen den verschiedenen G
ruppen von {pColumn1} ist signifikant, da der p-Wert unter 0.05 liegt.")
```

Berechnen und Erstellen der Korrelations Heatmap

Wenn im Folgenden von Werten aus der Korrelationstabelle/-heatmap gesprochen wird, ist diese gemeint

In [12]:

```
nominal.associations(df,figsize=(15,8),mark_columns=True)
```


Row ID (con)
Order Date (nom)
Ship Mode (nom)
Customer ID (nom)
Customer Name (nom)
Segment (nom)
Profit (con)
1.00
0.29 1.00 0.23 0.23 0.23 0.22 0.17
0.01 0.23 1.00 0.24 0.18 0.01 0.00
0.71 0.23 0.24 1.00 0.99 0.98 0.19
0.21 0.23 0.18 0.99 1.00 0.99 0.13
0.00 0.22 0.01 0.98 0.99 1.00 0.00
1.00
Row ID (con)
Order Date (nom)
Ship Mode (nom)
Customer ID (nom)
Customer Name (nom)
Segment (nom)
Profit (con)
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

Out[12]:

```
{'corr':                     Row ID (con)  Order Date (nom)  Ship Mode (n
om)  \
 Row ID (con)             1.000000          0.291618         0.014595
 Order Date (nom)         0.291618          1.000000         0.230390
 Ship Mode (nom)          0.014595          0.230390         1.000000
 Customer ID (nom)        0.709910          0.231772         0.244661
 Customer Name (nom)      0.213943          0.225683         0.176333
 Segment (nom)            0.004748          0.224403         0.010939
 Profit (con)            -0.019037          0.170611         0.002551

                      Customer ID (nom)  Customer Name (nom)  Segment (no
m)  \
 Row ID (con)                  0.709910             0.213943       0.00474
8
 Order Date (nom)              0.231772             0.225683       0.22440
3
 Ship Mode (nom)               0.244661             0.176333       0.01093
9
 Customer ID (nom)             1.000000             0.992097       0.98440
7
 Customer Name (nom)           0.992097             1.000000       0.99224
9
 Segment (nom)                 0.984407             0.992249       1.00000
0
 Profit (con)                  0.187327             0.133492       0.00293
7

                      Profit (con)
 Row ID (con)            -0.019037
 Order Date (nom)         0.170611
 Ship Mode (nom)          0.002551
 Customer ID (nom)        0.187327
 Customer Name (nom)      0.133492
 Segment (nom)            0.002937
 Profit (con)             1.000000  ,
 'ax': <matplotlib.axes._subplots.AxesSubplot at 0x115512cbc50>}
```

1. Analyse des Attributs Order Date

Es ergibt keinen Sinn sich den Profit einzelner Tage anzuschauen, da für jeden Tag nur einige Transaktionen vorliegen und somit keine validen Vorhersagen getroffen werden können. Daher werden im Folgenden die Daten einmal auf Monate und einmal auf Quartale aggregiert.

1.1 Aufteilen in Quartale

In [13]:

```
q1 = df[df['Order Date'].str.contains('-01-|-02-|-03-')]
q2 = df[df['Order Date'].str.contains('-04-|-05-|-06-')]
q3 = df[df['Order Date'].str.contains('-07-|-08-|-09-')]
q4 = df[df['Order Date'].str.contains('-10-|-11-|-12-')]
```

1.2 Aufteilen in Monate

In [14]:

```
months=["01","02","03","04","05","06","07","08","09","10","11","12"]
months2=["January","February","March","April","May","June","July","August","September",
"October","November","December"]
month_means=[]
for i in months:
    current = df[df["Order Date"].str.contains(f"-{i}-")]
    month_means.append(current.Profit.mean())
```

1.3 Korrelation zwischen Order Date und Profit

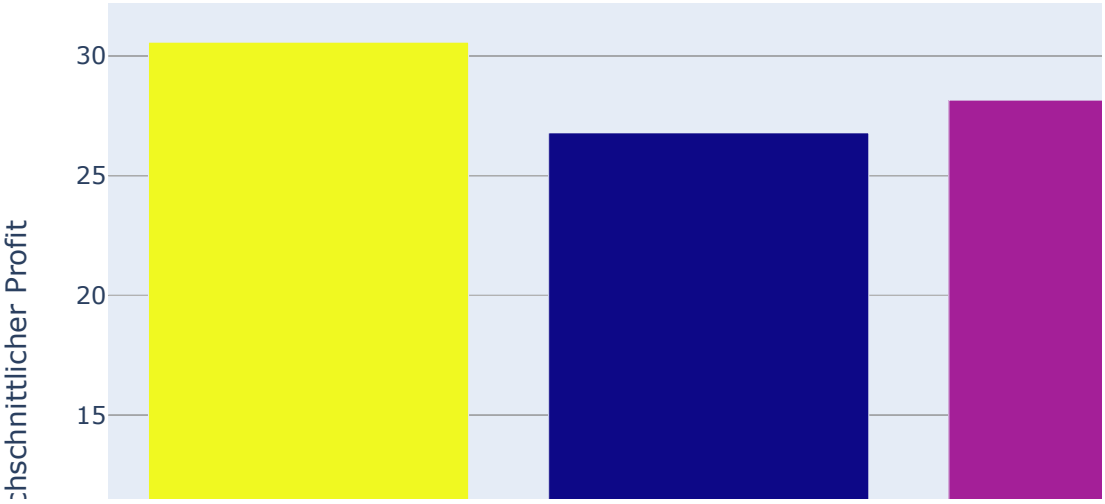
rechnerische Korrelationsberechnung nicht möglich, da sich die Werte dann auf das Order Date mit den einzelnen Tagen und nicht auf die vorgenommenen Gruppierungen beziehen würden

1.3.1 Diagramme zur Visualisierung der Zusammenhänge

a) Quartale

In [15]:

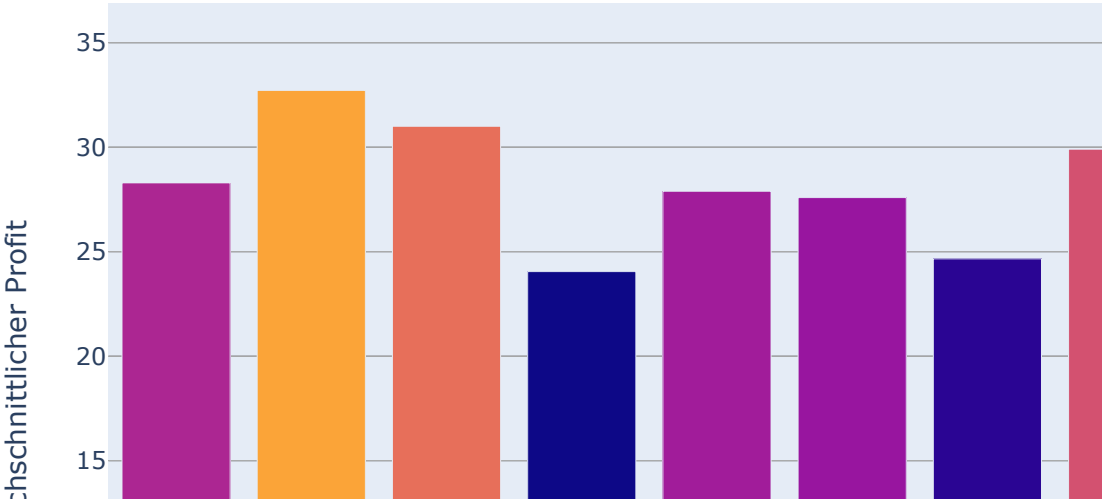
```
fig = px.bar(x=["Q1","Q2","Q3","Q4"], y=[q1.Profit.mean(),q2.Profit.mean(),q3.Profit.mean(),q4.Profit.mean()],color=[q1.Profit.mean(),q2.Profit.mean(),q3.Profit.mean(),q4.Profit.mean()], labels={"x":"Quartale","y":"Durchschnittlicher Profit"})
fig.show()
```



b) Monate

In [16]:

```
fig = px.bar(x=months2, y=month_means,color=month_means, labels={"x":"Monate","y":"Durc
hschnittlicher Profit"})
fig.show()
```



1.4 Fazit

Das Order Date ist genau wie im vorherigen Notebook das Ship Date kein geeignetes Attribut zur Vorhersage des Profits. Denn die einzelnen Tage liefern keine sinnvolle Information und bei den aggrgierten Monaten und Quartalen fallen keine Muster zwischen Profit und dem Datum auf. Daher wird das Order Date entfernt.

In [17]:

```
del df["Order Date"]
```

2. Analyse des Attributs Ship Mode

2.1 Verteilung der Transaktions pro Ship Mode

In [18]:

```
result,colour = countColumn("Ship Mode","All Ship Modes","Number of transactions")
fig = px.bar(result, x="All Ship Modes", y="Number of transactions", color=colour)
fig.show()
```

In [19]:

```
numberOfTransactions("Ship Mode")
```

Out[19]:

Ship Mode	Number of Transactions
Same Day	2701
First Class	7505
Second Class	10309
Standard Class	30775

2.2 Korrelation zwischen Profit und Ship Mode

2.2.1 Rechnerische Korrelationswerte

Wert aus Heatmap: 0

In [20]:

```
calculateAnovaScore("Ship Mode","Profit")
```

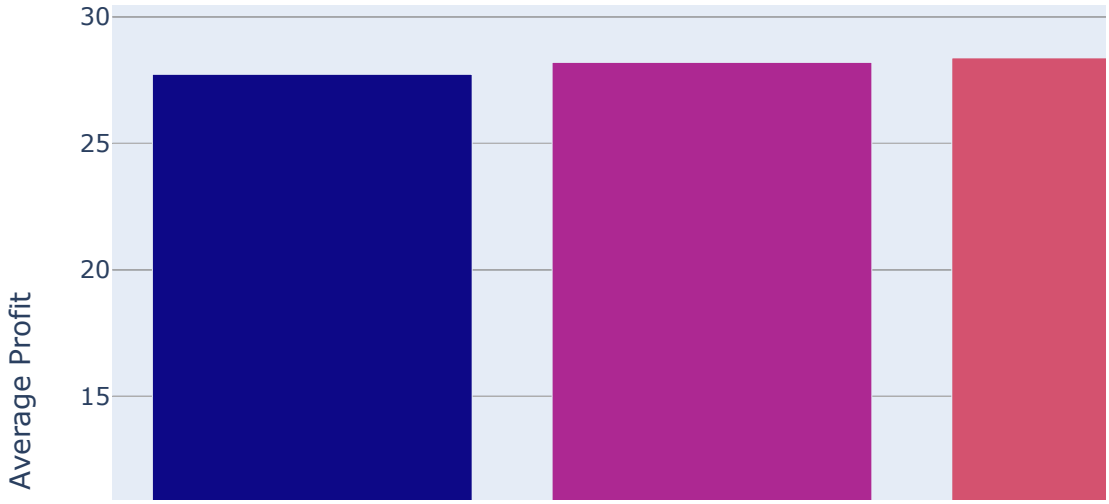
```
P-Wert für Anova ist:  0.953549120213493
F Score für Anova ist:  0.11126944003460387
```

2.2.2 Diagramme zur Visualisierung dieses Zusammenhangs

a) Balkendiagramm

In [21]:

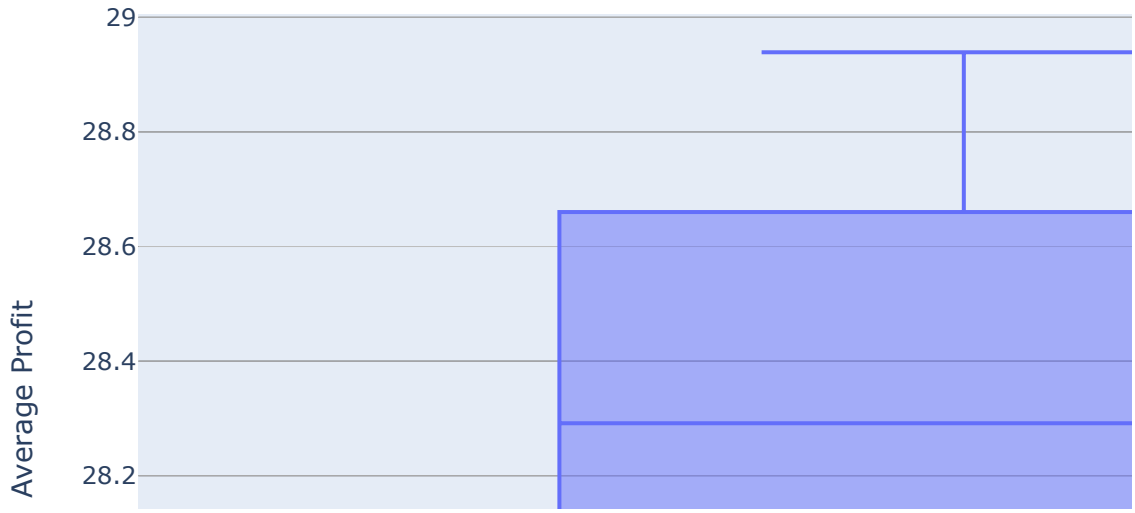
```
result,means = dfOfAverageMeans("Ship Mode","Profit", "All Ship Modes", "Average Profi
t")
fig = px.bar(result, x="All Ship Modes", y="Average Profit", color=means)
fig.show()
```



b) Box Plot

In [22]:

```
result,means = dfOfAverageMeans("Ship Mode","Profit", "All Ship Modes", "Average Profi
t")
fig = px.box(result, y="Average Profit")
fig.show()
```



2.3 Fazit

Das Attribut Ship Mode muss entfernt werden. Der p-Wert ist fast bei 1, da die verschiedenen Mittelwerte sich so gut wie gar nicht unterscheiden. Es gibt also im Hinblick auf den Profit keinen Unterschied, egal welcher Ship Mode ausgewählt wurde. Dies zeigt sich ebenfalls in dem Balkendiagramm und dem Korrelationswert von 0.

In [23]:

```
del df["Ship Mode"]
```

3. Analyse des Attributs Customer ID

3.1 Prüfen, ob Gruppen genug Transaktionen haben

In [24]:

```
numberOfTransactions("Customer ID")
```

Out[24]:

Customer ID	Number of Transactions
DK-2985	1
RC-9825	1
MG-7890	1
BG-1035	1
MG-7650	1
...	...
EM-13960	85
SW-20755	89
JG-15805	90
BE-11335	94
PO-18850	97

1590 rows × 1 columns

In [25]:

```
howOftenDoAmountsAppear("Customer ID")
```

Out[25]:

	Number of Transactions	counts
0	1	7
1	2	4
2	3	18
3	4	17
4	5	32
...	...	...
83	85	2
84	89	1
85	90	1
86	94	1
87	97	1

88 rows × 2 columns

Es gibt zwar einige Kunden IDs, die fast 100 Transaktionen haben. Dennoch hat die deutliche Mehrheit nur wenige. Außerdem gibt es über 1500 verschiedene Kunden IDs

3.2 Fazit

Es ergibt keinen Sinn die Customer ID als Vorhersage für den Profit zu nutzen. Einerseits gibt es zu viele verschiedene Gruppen mit zu wenigen Transaktionen und andererseits macht es im Sachzusammenhang keinen Sinn, dass man als Input in den Algorithmus eine spezifische Customer ID übergibt. Man möchte ja den generellen Profit vorhersagen und nicht den Profit, den man an einem einzelnen Customer erzielt. Das Attribut wird also entfernt.

In [26]:

```
del df["Customer ID"]
```

4. Customer name

4.1 Prüfen, ob Gruppen genug Transaktionen haben

In [27]:

```
numberOfTransactions("Customer Name")
```

Out[27]:

Customer Name	Number of Transactions
Michael Oakman	29
Nicole Brennan	31
Darren Budd	31
David Bremer	34
Andy Reiter	35
...	...
Patrick O'Brill	102
Bill Eplett	102
Gary Hwang	102
Steven Ward	106
Muhammed Yedwab	108

795 rows × 1 columns

4.2 Fazit

Auch wenn das Attribut Customer Name nochmal etwas aggregierter als die Customer ID ist und daher ggf. mit Profit korrelieren könnte, gilt auch hier die gleiche Argumentation wie bei Customer ID. Es ergibt im Sachzusammenhang keinen Sinn, dass der Name des Kunden als Parameter zur Bestimmung des Profits genutzt wird. Daher wird das Attribut entfernt.

In [28]:

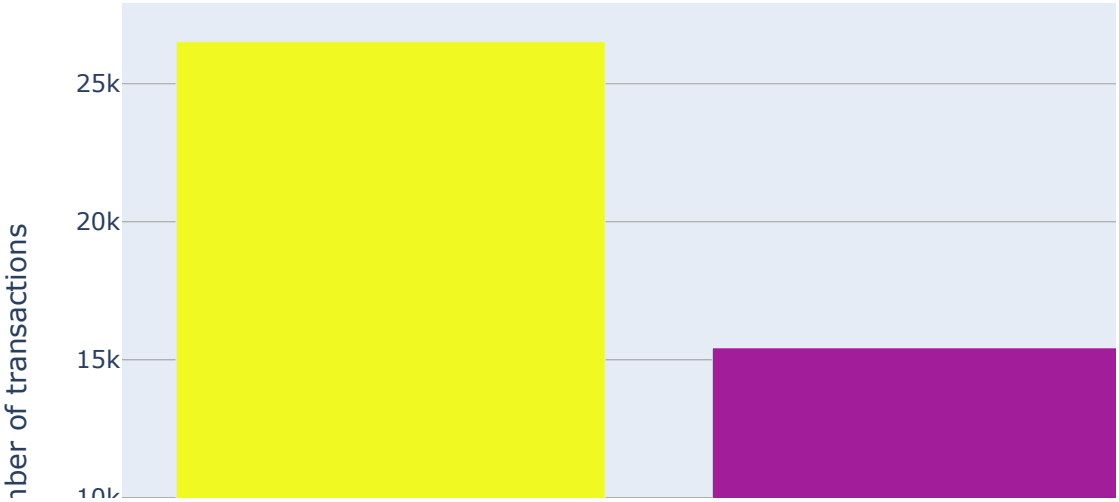
```
del df["Customer Name"]
```

5. Analyse des Attributs Segment

5.1 Prüfen ob jede Gruppe genug Transaktionen hat

In [29]:

```
result,colour = countColumn("Segment","All Segments","Number of transactions")
fig = px.bar(result, x="All Segments", y="Number of transactions", color=colour)
fig.show()
```



In [30]:

```
numberOfTransactions("Segment")
```

Out[30]:

Segment	Number of Transactions
Home Office	9343
Corporate	15429
Consumer	26518

5.2 Korrelation zwischen Segment und Profit

5.2.1 rechnerische BEstimmung der Korrelationswerte

Wert aus Heatmap: 0

In [31]:

```
calculateAnovaScore("Segment","Profit")
```

```
P-Wert für Anova ist:  0.8015632303031714
F Score für Anova ist:  0.2211923740432353
```

5.2.2 Diagramme zur Visualisierung dieses Zusammenhangs

a) Balkendiagramm

In [32]:

```
result,means = dfOfAverageMeans("Segment","Profit", "All Segments", "Average Profit")
fig = px.bar(result, x="All Segments", y="Average Profit", color=means)
fig.show()
```

Box Plot und Streudiagramm geben keine zusätzlichen Infrmationen

5.3 Fazit

Es gilt die gleiche Argumentation wie bei Ship Mode. Der Korrelationswert ist 0 und der p-Wert extrem hoch. Dies bedeutet, dass es keinerlei Korrelation/Zusammenhang zwischen Segment und Profit gibt. Dies bestätigt sich in den gleich hohen Balken des Diagramms. Daher wird auch Segment entfernt.

In [33]:

```
del df["Segment"]
```

Überarbeitetes DF der kundenbezogenen Attribute

In [34]:

```
df
```

Out[34]:

	Row ID	Profit
0	32298	762.1845
1	26341	-288.7650
2	25330	919.9710
3	13524	-96.5400
4	47221	311.5200
...	...	...
51285	29002	4.5000
51286	35398	-1.1100
51287	40470	11.2308
51288	9596	2.4000
51289	6147	1.8000

51290 rows × 2 columns