In [34]:
```
pip install mysql-connector-python
```

```
Requirement already satisfied: mysql-connector-python in ./opt/anaconda3/lib/python3.9/site-packages (8.0.31)
Requirement already satisfied: protobuf<=3.20.1,>=3.11.0 in ./opt/anaconda3/lib/python3.9/site-packages (from mysql-c
onnector-python) (3.20.1)
Note: you may need to restart the kernel to use updated packages.
```

In [35]:
```
import mysql.connector
from mysql.connector import Error
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [36]:
```
connection = mysql.connector.connect(user='root', password='Aishu.22112000',
                                     host='localhost',
                                     database='DMA_PROJECT')
```

In [37]:
```
sql_select_Query = "show tables"
cursor = connection.cursor()
cursor.execute(sql_select_Query)
records = cursor.fetchall()
for x in records:
  print(x)
```

```
('bill',)
('customer',)
('driver',)
('full_load_service',)
('goods',)
('goods_manager',)
('labour',)
('makes',)
('parcel_based_service',)
('refreshment_and_fuel',)
('request',)
('route',)
('stops_at',)
('trip',)
('trip_manager',)
('vehicle',)
('warehouse_incharge',)
('warehouses',)
('works_at',)
```

SQL Queries for Analytics

Customers who carry out the most business and least business with Boston Convenience

In [84]:
```
# Customers who carried out most business
sql_1 = 'select c.Customer_name, sum(b.total_freight) as tot_billed_USD from customer c, bill b'
sql_2=' where c.bill_id = b.bill_id group by c.Customer_name;'
sql_select_Query=sql_1+sql_2
cursor = connection.cursor()
cursor.execute(sql_select_Query)
records = cursor.fetchall()
df_cust = pd.DataFrame(records, columns = ['Customer_name', 'tot_billed_USD'])
df_cust.sort_values(by=['tot_billed_USD'],ascending=False).head(3)
```

Out[84]:

	Customer_name	tot_billed_USD
73	Zooveo	166929.93
41	Vinte	127515.28
52	Zazio	127370.67

In [61]:
```
# Customers who carried out Least business
df_cust.sort_values(by=['tot_billed_USD'],ascending=True).head(3)
```

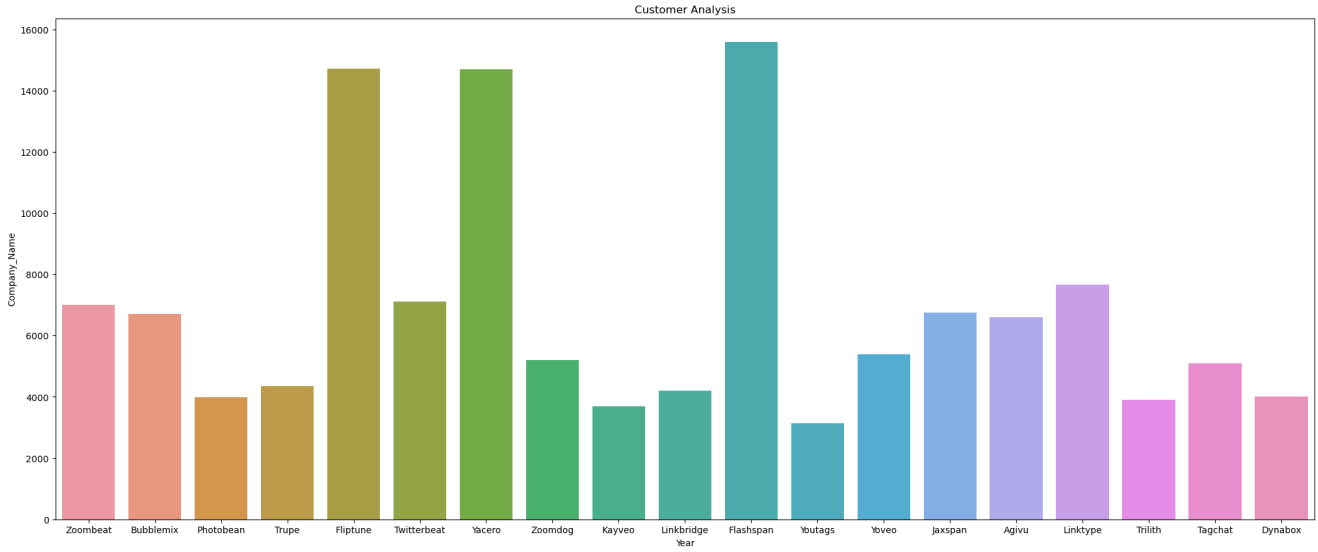
Out[61]:

	Customer_name	tot_billed_USD
24	Youtags	3132.68
18	Kayveo	3683.69
32	Trilith	3901.68

In [64]:
```
# creating the bar plot for all clients who chose us only for once or twice and we need to cater to their needs relative

fig = plt.figure(figsize = (25, 10))
df_above_avg=df_cust[df_cust['tot_billed_USD'] <(df_cust['tot_billed_USD'].mean()-df_cust['tot_billed_USD'].std())]
sns.barplot(data =df_above_avg, x=df_above_avg['Customer_name'], y= df_above_avg['tot_billed_USD'])

plt.xlabel("Year")
plt.ylabel("Company_Name")
plt.title("Customer Analysis")
plt.show()
```



In [59]:
```
df_above_avg
```

Out[59]:

	Customer_name	tot_billed_USD
1	Zoombeat	7015.87
3	Bubblemix	6710.94
4	Photobean	3988.21
6	Trupe	4342.23
9	Fliptune	14724.46
11	Twitterbeat	7108.75
13	Yacero	14696.85
17	Zoomdog	5195.37
18	Kayveo	3683.69
19	Linkbridge	4197.48
23	Flashspan	15581.94
24	Youtags	3132.68
25	Yoveo	5398.28
26	Jaxspan	6760.73
29	Agivu	6592.34
31	Linktype	7669.00
32	Trilith	3901.68
34	Tagchat	5085.05
35	Dynabox	4003.29

In [41]:
```
# Yearly Turnover
sql_1 = 'select sum(b.total_freight) as tot_billed_USD,YEAR(STR_TO_DATE(b.Bill_Date,"%m/%d/%Y")) as YR'
sql_2 = ' from customer c,bill b where c.bill_id = b.bill_id group by YR;'
sql_select_Query = sql_1 +sql_2
cursor = connection.cursor()
cursor.execute(sql_select_Query)
records = cursor.fetchall()
df_cust_yr = pd.DataFrame(records, columns = ['Revenue','Year']).sort_values(by=['Year'])
df_cust_yr
```

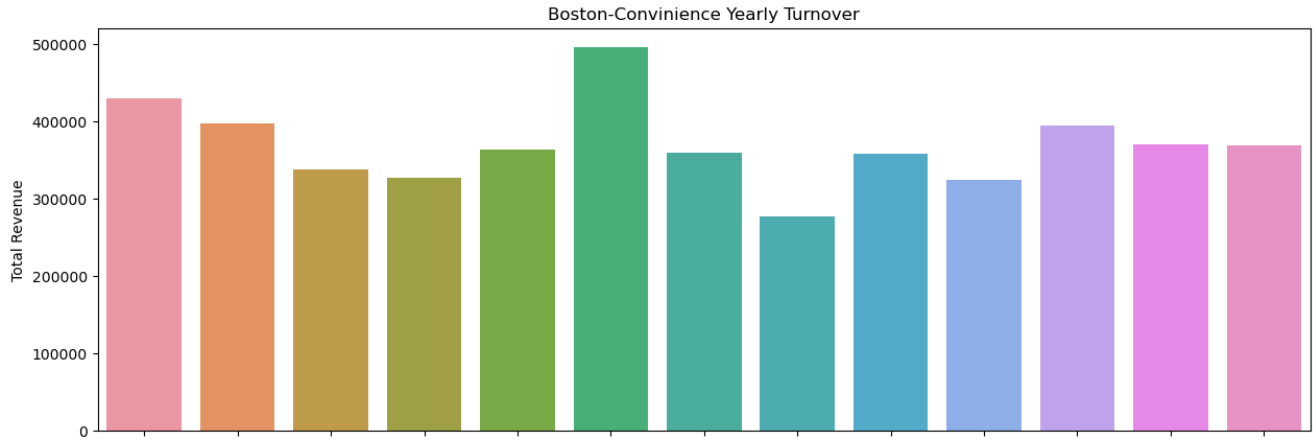
Out[41]:

	Revenue	Year
6	429832.85	2010
4	397275.59	2011
11	337143.98	2012
5	327420.40	2013
3	363704.00	2014
0	495975.59	2015
2	359951.62	2016
9	276724.35	2017
1	357620.50	2018
10	324081.53	2019
7	394891.99	2020
12	369943.35	2021
8	368791.68	2022

In [42]:
```
fig = plt.figure(figsize = (15, 5))

# creating the bar plot
sns.barplot(data =df_cust_yr, x=df_cust_yr['Year'], y= df_cust_yr['Revenue'])

plt.xlabel("Year")
plt.ylabel("Total Revenue")
plt.title("Boston-Convinience Yearly Turnover")
plt.show()
```



In [68]:
```
#Business Expansion - most frequently chosen pick-up and drop-off locations
sql_1 = 'select r.Pick_up_Location,count(c.Customer_name) as cust_count from customer c,makes m,request r '
sql_2 = 'where c.Invoice_Number=m.Invoice_Number and c.Employee_ID=m.Employee_ID and m.req_id=r.req_id '
sql_3 = 'and r.Pick_up_Location != "Boston" group by r.Pick_up_Location order by cust_count desc ;'
sql_select_Query = sql_1 + sql_2 + sql_3
cursor = connection.cursor()
cursor.execute(sql_select_Query)
records = cursor.fetchall()
df_cities_pick= pd.DataFrame(records, columns = ['Locations','Customers_Count']).sort_values(by=['Locations'])
sql_1 = 'select r.Drop_Location,count(c.Customer_name) as cust_count from customer c,makes m,request r '
sql_2 = 'where c.Invoice_Number=m.Invoice_Number and c.Employee_ID=m.Employee_ID and m.req_id=r.req_id '
sql_3 = 'and r.Drop_Location != "Boston" group by r.Drop_Location order by cust_count desc ;'
sql_select_Query = sql_1 + sql_2 + sql_3
cursor = connection.cursor()
cursor.execute(sql_select_Query)
records = cursor.fetchall()
df_cities_drop= pd.DataFrame(records, columns = ['Locations','Customers_Count']).sort_values(by=['Locations'])
df_cities=pd.merge(df_cities_pick,df_cities_drop,on='Locations',how='outer').fillna(0)
df_cities['Customer_count']=df_cities['Customers_Count_x'] + df_cities['Customers_Count_y']
df_cities.drop(['Customers_Count_x','Customers_Count_y'],axis=1,inplace=True)
```

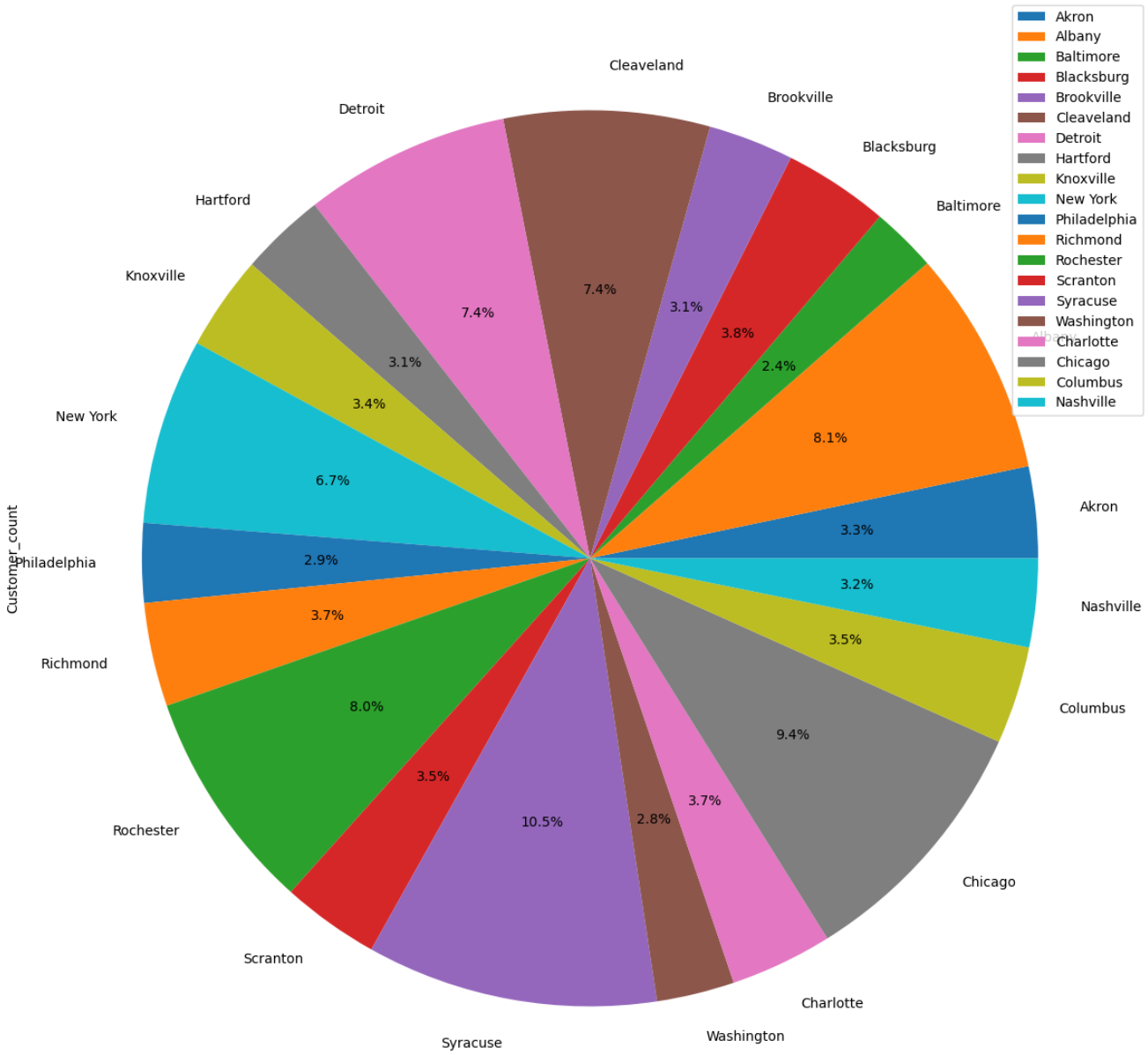
In [66]:
```
df_cities
```

Out[66]:

	Locations	Customer_count
0	Akron	60.0
1	Albany	148.0
2	Baltimore	43.0
3	Blacksburg	69.0
4	Brookville	56.0
5	Cleaveland	135.0
6	Detroit	135.0
7	Hartford	56.0
8	Knoxville	62.0
9	New York	122.0
10	Philadelphia	52.0
11	Richmond	68.0
12	Rochester	146.0
13	Scranton	64.0
14	Syracuse	191.0
15	Washington	51.0
16	Charlotte	67.0
17	Chicago	171.0
18	Columbus	64.0
19	Nashville	58.0

In [75]:
```
df_cities.plot.pie(y='Customer_count',labels=list(df_cities['Locations'].values),figsize=(15,15),autopct='%1.1f%%')
```

Out[75]: <AxesSubplot:ylabel='Customer_count'>



In [73]:
```
len(list(df_cities['Locations'].values))
```

Out[73]: 20