



# Boston Convenience - Transport Company

**Course:** IE6700 - Data Management for Analytics  
**Professor:** Dr. Venkat Krishnamurthy  
**Teaching Assistants:** Saharsh Desai, Zixi Xiao

**Done by: GROUP 17**

Valli Meenaa Vellaiyan (NUID: 002783394)

Hrithik Sarda (NUID: 002766048)

# PROBLEM STATEMENT

## ❖ Definition:

- Creating a database for a B2B business, i.e., a transport company called “Boston Convenience”, to store the details of services offered, requests made, trips taken, vehicles owned, and customer and employee information.
- The company offers four routes that travel via 21 pick-up and drop-off locations.
- The routes are as follows:

Route number	Location 1	Location 2	Location 3	Location 4	Location 5	Location 6	Location 7
1	Boston	New York	Philadelphia	Washington	Richmond	Charlotte	-
2	Boston	Albany	Syracuse	Rochester	Cleveland	Detroit	Chicago
3	Boston	Hartford	Scranton	Brookville	Akron	Columbus	-
4	Boston	New York	Baltimore	Blacksburg	Knoxville	Nashville	-


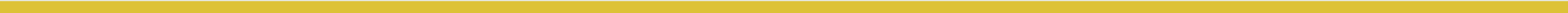
# PROBLEM STATEMENT

## ❖ Data utilization:

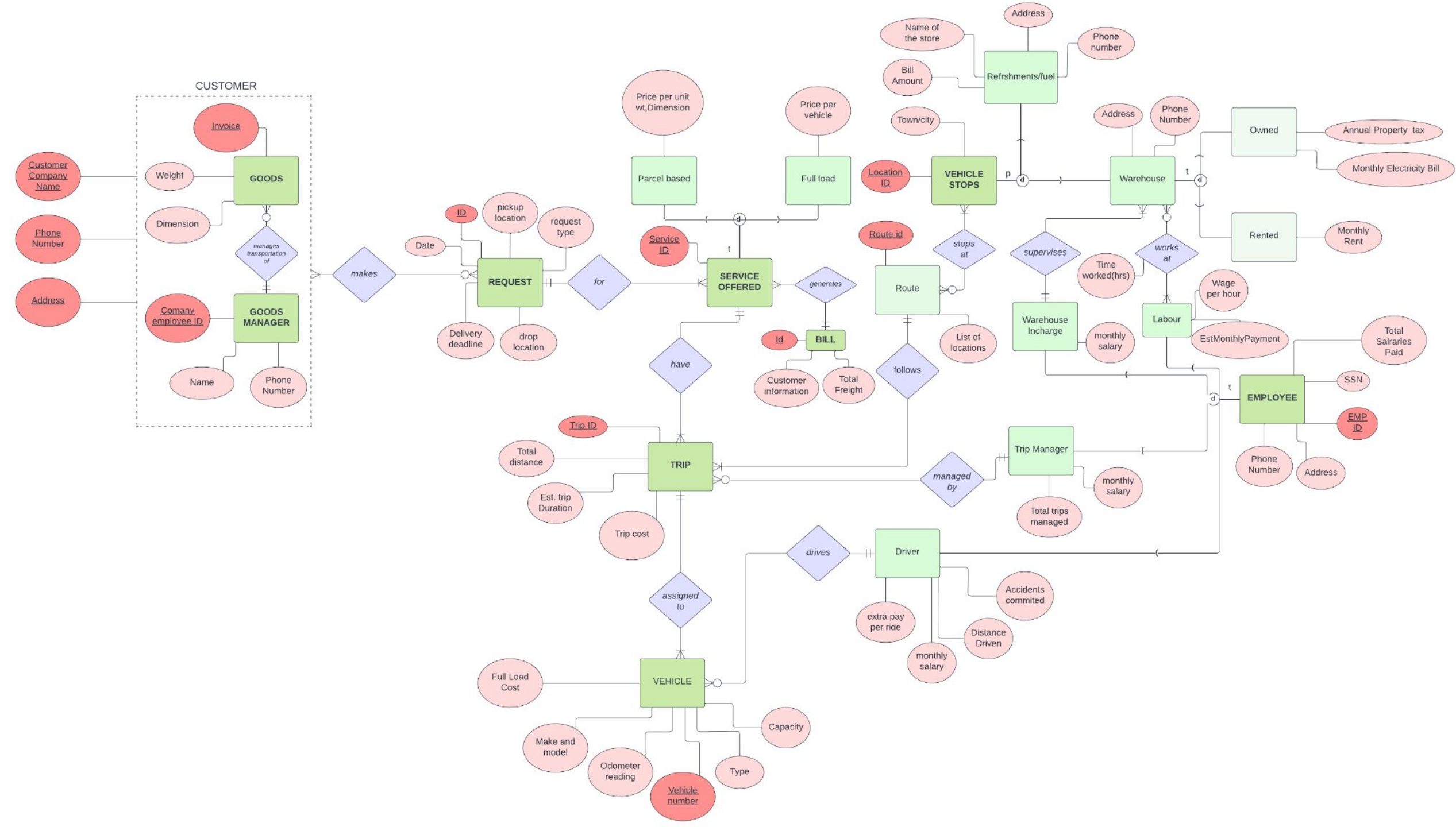
- Providing insights to make critical business decisions
- Employee management and ranking system
- Methods to implement inbound marketing
- Come up with key expansion opportunities







# **CONCEPTUAL DATA MODEL OF BOSTON CONVENIENCE TRANSPORT**



# EXPLANATION OF THE EER

- ❖ [Click here](#) for the PNG version of the EER.
- ❖ Customer makes a request for transportation
- ❖ Services are provided with respect to the type of request initiated by the customer (either **full load based** or **parcel based**)
- ❖ For a service offered, a bill is generated and trips is initiated
- ❖ One or more vehicles with drivers are assigned to a trip
- ❖ A trip is assigned to one of the available trip managers
- ❖ A trip follows a particular route based on the pick up and drop off locations
- ❖ Vehicle either stops for fuel/refreshment or at a particular warehouse (rented or owned) at the locations decided in the route
- ❖ Each warehouse has one warehouse incharge and at least one labour for loading and unloading purposes

## SCOPE OF ANALYTICS : Customer Information

- ❖ Performing analytics on our database can give us customers who carry out the most business and least business with Boston Convenience over the years or over a particular time period.

Least

- 1)
- 2) Send them a relatively better quote
- 3) Send them gifts or goodies

Higher

Highest business:

- 1) Increasing profit margins over time.
- 2) Additional income can be generated by offering additional services to them
- 3) Giving discounts and promotions to maintain long-term relations.



ness:  
eting



# SCOPE OF ANALYTICS : Business Expansion

- ❖ Upon performing further analytics to our database, we can obtain the locations that are potential hubs for business expansion.
  - 1) Top cities from our Analysis: Syracuse, Chicago.
  - 2) Establishing a new headquarter[a centre of operations] in these cities.
  - 3) Partnering with Emergent Transport companies in these cities.
  - 4) Marketing our services in these cities for increasing ClientBase.



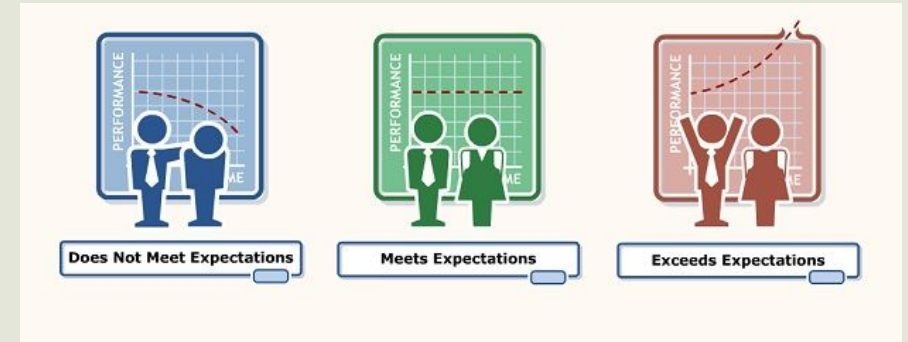


# SCOPE OF ANALYTICS : Employee Information

- ❖ The data can also be analyzed to find out which trip manager(s) has overlooked the most number of trips till date.

Actions:

- 1) Assigning high profile clients
- 2) Performance bonus



- ❖ We can find out the rash drivers based on the record of number of accidents committed by each driver till date

Actions:

- 1) Warning
- 2) Put under notice (assigning shorter trips, reduction in salary)



- ❖ Employee ranking system: Rank each employees in their specific Divisions for promotions/bonus or termination/decrement.

# ANALYTICS USING SQL

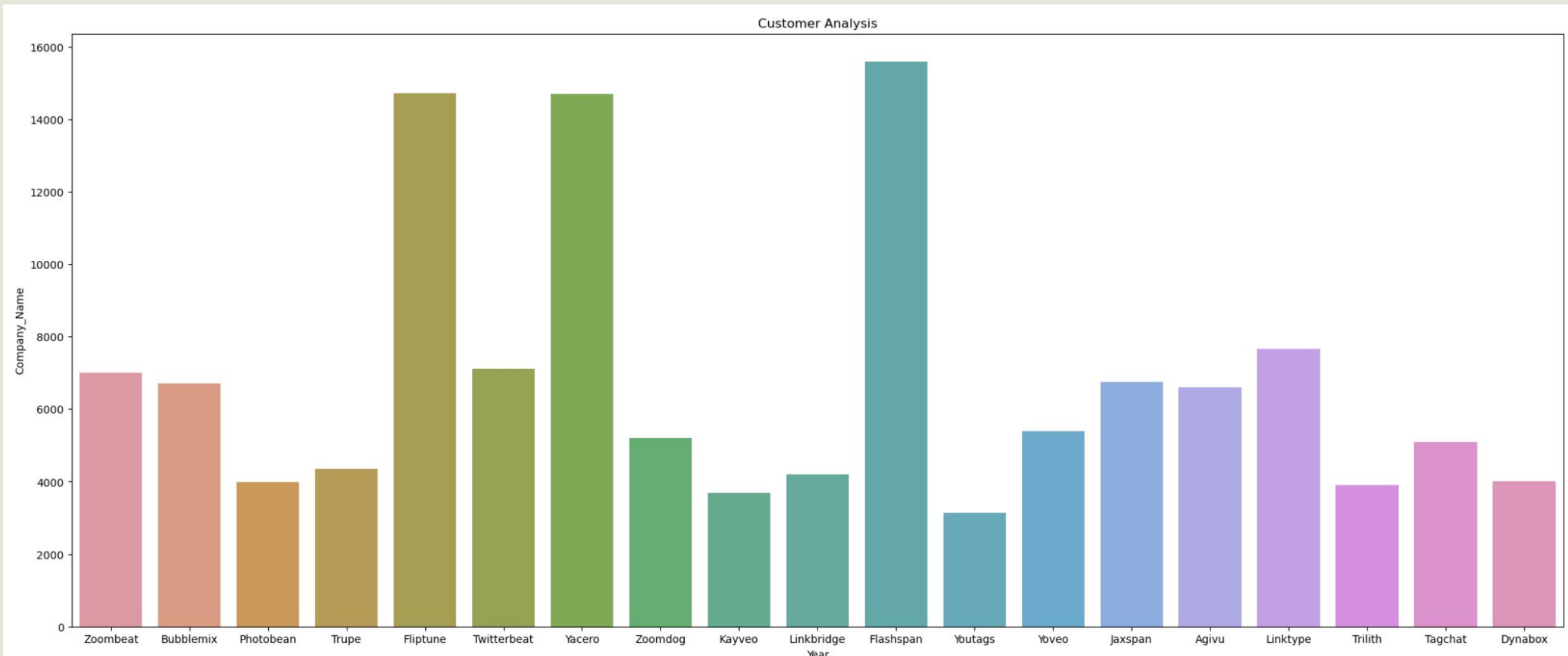
## Customers who carry out the most business and least business with Boston Convenience:

```
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'])
# Customers who carried out most business
df_cust.sort_values(by=['tot_billed_USD'],ascending=False).head(3)
# Customers who carried out Least business
df_cust.sort_values(by=['tot_billed_USD'],ascending=True).head(3)
# Creating the bar plot for all clients who chose us only for once or twice
# We need to cater to their needs relatively more when compared to other companies
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()
```

	Customer_name	tot_billed_USD
73	Zooveo	166929.93
41	Vinte	127515.28
52	Zazio	127370.67

	Customer_name	tot_billed_USD
24	Youtags	3132.68
18	Kayveo	3683.69
32	Trilith	3901.68

# BAR PLOT VISUALIZATION for clients who barely used our service





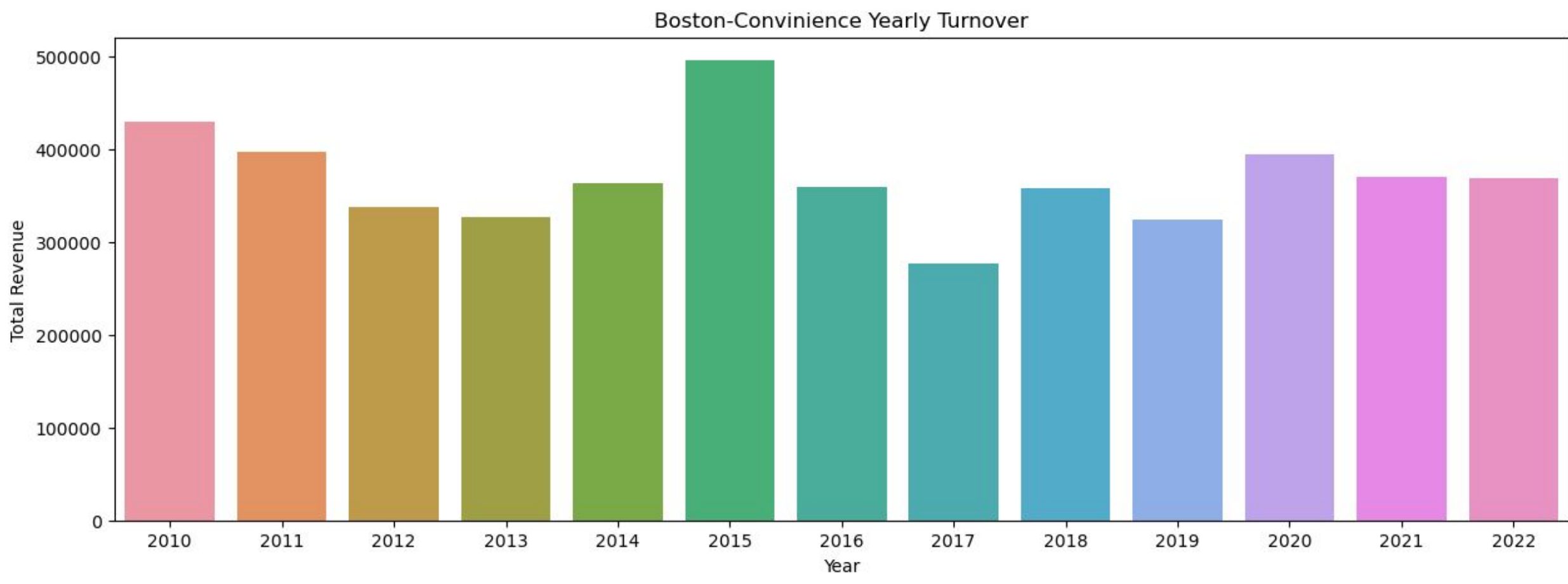
# ANALYTICS USING SQL

## 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
# creating the bar plot
fig = plt.figure(figsize = (15, 5))
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-Convenience Yearly Turnover")
plt.show()
```

	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

# BAR PLOT VISUALIZATION for the yearly turnover at our company



# ANALYTICS USING SQL

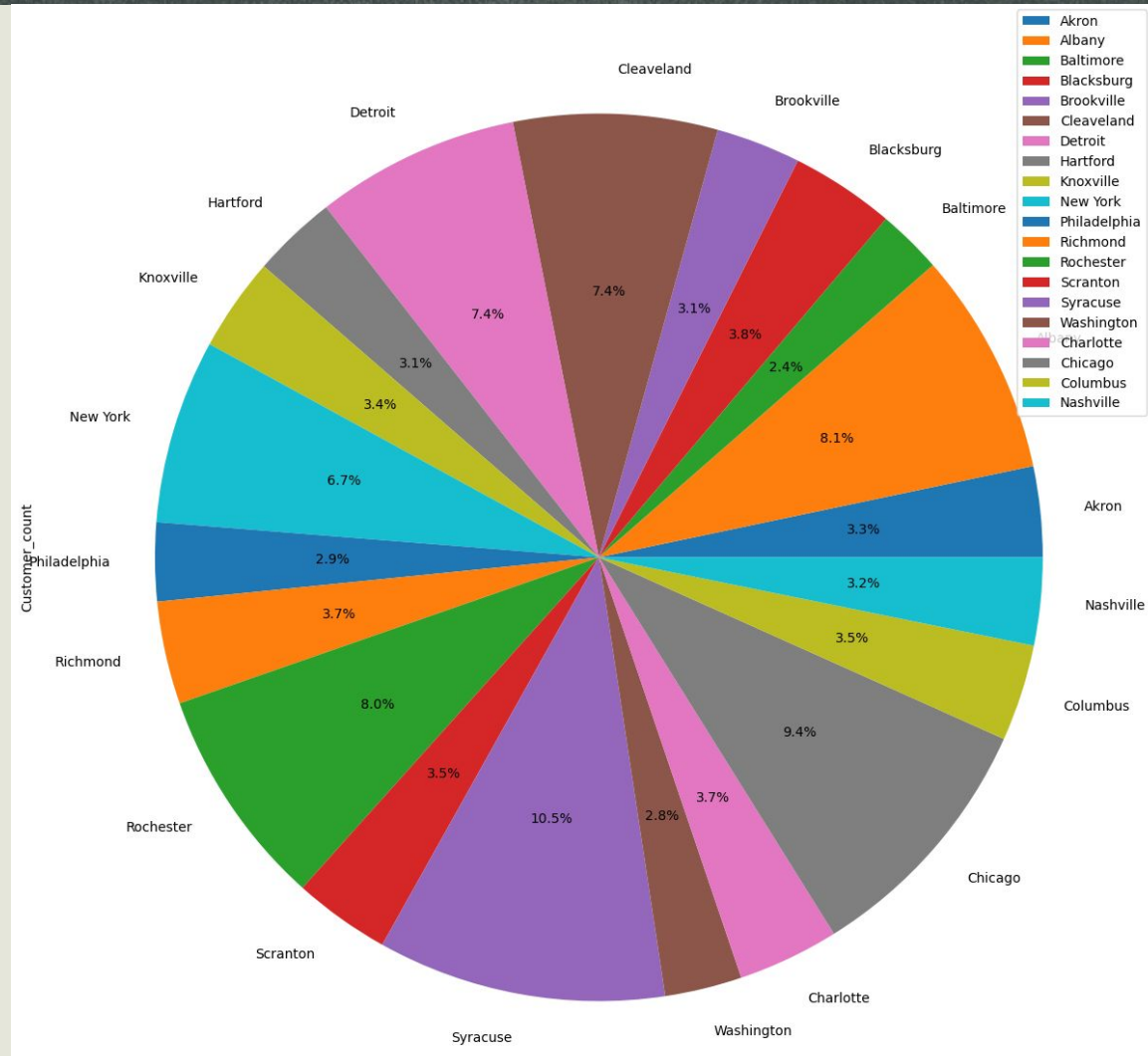
## Business Expansion: most frequently chosen pick-up and drop-off locations:

```
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)  
df_cities.sort_values(by=['Customer_count'],ascending=False)  
# Plotting a pie chart for the number of times each location is chosen for either pick-up or drop-off  
df_cities.plot.pie(y='Customer_count',labels=list(df_cities['Locations'].values),figsize=(15,15),autopct='%1.1f%%')
```

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



# PIE CHART VISUALIZATION for the number of times each location is chosen for either pick-up or drop-off



## IF TIME PERMITS.. ANALYTICS USING NOSQL (MONGODB)

- ❖ Type of truck preferred by majority of customers based on their odometer reading:

```
db.vehicle.find({}, {type:1, odometer_reading:1, _id:0}).sort({odometer_reading:-1}).limit(1);
```

```
> db.vehicle.find({}, {type:1, odometer_reading:1, _id:0}).sort({odometer_reading:-1}).limit(1);  
< { odometer_reading: 486603, type: 'Semi-trailer' }
```

- ❖ Currently employed driver who's committed most number of accidents till date:

```
db.driver.find({}, {DRV_id:1, accidents_committed:1, _id:0}).sort({accidents_committed:-1}).limit(1);
```

```
> db.driver.find({}, {DRV_id:1, accidents_committed:1, _id:0}).sort({accidents_committed:-1}).limit(1)  
< { DRV_id: 'DRV67', accidents_committed: 15 }
```



**THANK YOU**

**GOOD SIR**

This is the end of our presentation!