In [149…

```
pip install mysql-connector-python
```

```
Requirement already satisfied: mysql-connector-python in c:\users\shrey\anaconda3\lib\site
-packages (8.0.30)Note: you may need to restart the kernel to use updated packages.
Requirement already satisfied: protobuf<=3.20.1,>=3.11.0 in c:\users\shrey\anaconda3\lib\s
ite-packages (from mysql-connector-python) (3.20.1)
```

In [7]:

```
!pip install plotly
```

```
Collecting plotly
  Downloading plotly-5.9.0-py2.py3-none-any.whl (15.2 MB)
Collecting tenacity>=6.2.0
  Downloading tenacity-8.0.1-py3-none-any.whl (24 kB)
Installing collected packages: tenacity, plotly
Successfully installed plotly-5.9.0 tenacity-8.0.1
```

In [6]:

```
!pip install yagmail
```

```
Requirement already satisfied: yagmail in c:\users\shrey\anaconda3\lib\site-packages (0.1
5.280)
Requirement already satisfied: premailer in c:\users\shrey\anaconda3\lib\site-packages (fr
om yagmail) (3.10.0)
Requirement already satisfied: cssutils in c:\users\shrey\anaconda3\lib\site-packages (fro
m premailer->yagmail) (2.5.1)
Requirement already satisfied: lxml in c:\users\shrey\anaconda3\lib\site-packages (from pr
emailer->yagmail) (4.6.3)
Requirement already satisfied: requests in c:\users\shrey\anaconda3\lib\site-packages (fro
m premailer->yagmail) (2.26.0)
Requirement already satisfied: cssselect in c:\users\shrey\anaconda3\lib\site-packages (fr
om premailer->yagmail) (1.1.0)
Requirement already satisfied: cachetools in c:\users\shrey\anaconda3\lib\site-packages (f
rom premailer->yagmail) (5.2.0)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\users\shrey\anaconda3\lib\s
ite-packages (from requests->premailer->yagmail) (2.0.4)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\shrey\anaconda3\lib\site-pac
kages (from requests->premailer->yagmail) (2021.10.8)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\shrey\anaconda3\lib\site-
packages (from requests->premailer->yagmail) (1.26.7)
Requirement already satisfied: idna<4,>=2.5 in c:\users\shrey\anaconda3\lib\site-packages
(from requests->premailer->yagmail) (3.2)
```

In [1]:

```
#Importing libraries
import pandas as pd
import mysql.connector
import plotly.express as px
import plotly.graph_objects as go
import numpy
from datetime import date
import yagmail
```

In [2]:

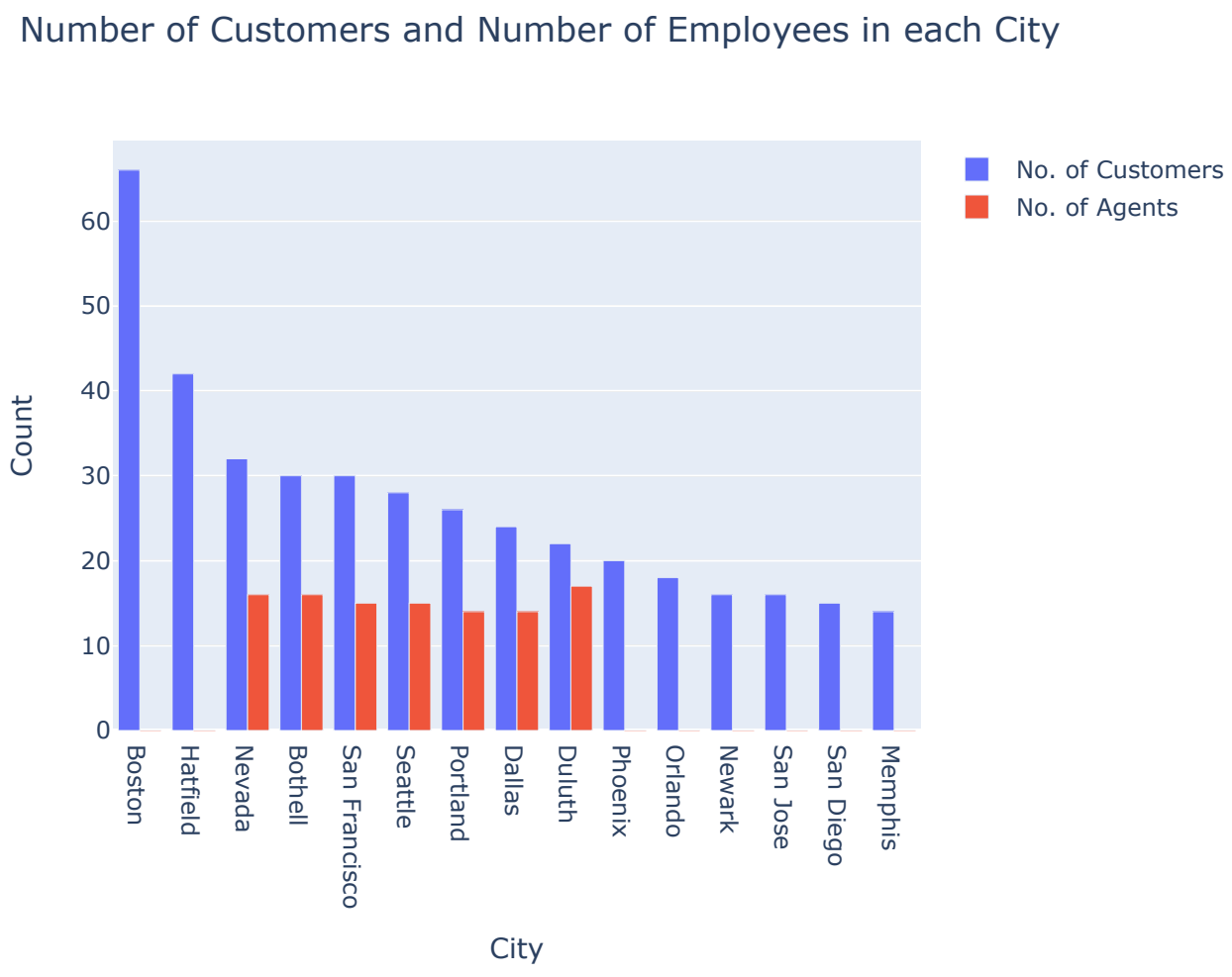
```
#mysql connection

mydb = mysql.connector.connect(
host= "localhost",
user= "root",
passwd= "mysql1234",
db= "Insurance"
)
```

```
myconn =mydb
```

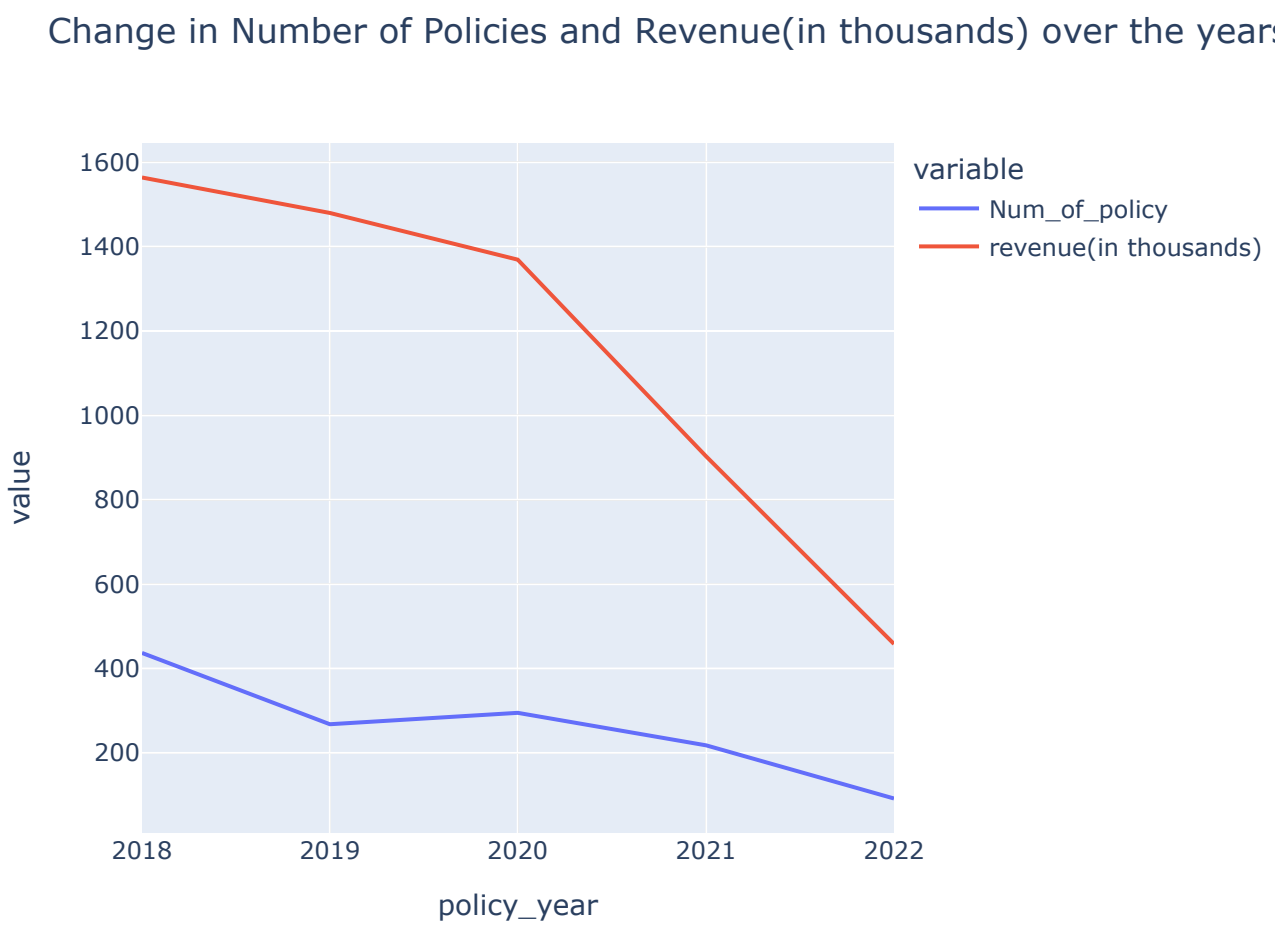
Visualization

In [3]:

```
#count of customers and agents
query="""
select c.city, count(distinct c.Customer_id) as num_of_customers, count(distinct a.Agent_i
from customer c
left join agent a on c.city=a.city group by city order by num_of_customers desc, num_of_ag
"""

df = pd.read_sql(query, myconn)

plot = go.Figure(data=[go.Bar(
    name = 'No. of Customers',
    x = df['city'],
    y = df['num_of_customers']),
    go.Bar(
    name = 'No. of Agents',
    x = df['city'],
    y = df['num_of_agents']
   )
])

plot.update_layout(title="Number of Customers and Number of Employees in each City", xaxis
plot.show()
```

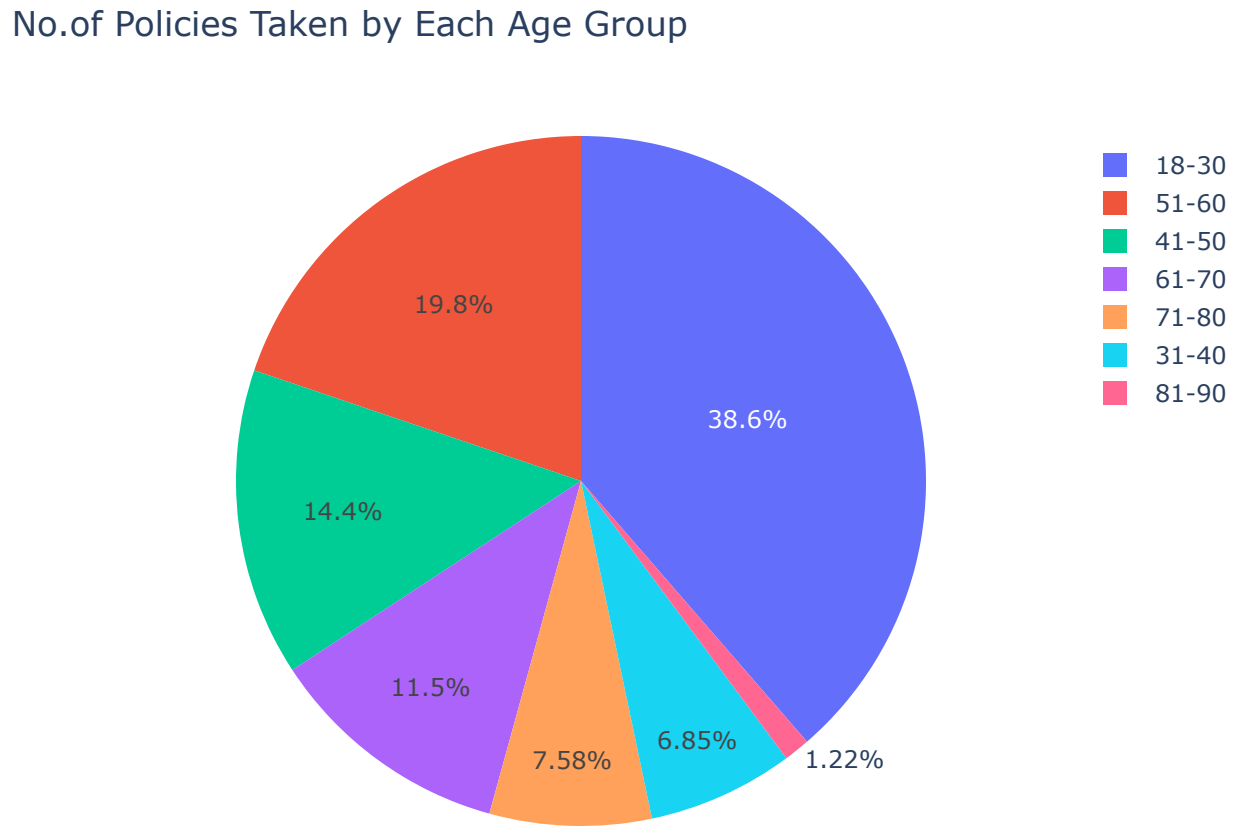


In [5]:
```
#year-wise analysis
query="""
select year(p.start_date) as policy_year, count(p.Policy_no) as Num_of_policy,sum(pa.amour
from policy p, customer_policy cp, payment pa
where p.Policy_no=cp.Policy_no and cp.Customer_id=pa.Customer_id
group by policy_year
"""

df_temp = pd.read_sql(query, myconn)
df_temp['policy_year']= pd.to_datetime(df_temp['policy_year'],format='%Y')
df_temp['revenue(in thousands)']=df_temp['revenue']/1000


fig = px.line(df_temp, x='policy_year', y=["Num_of_policy","revenue(in thousands)"],title=


fig.show()
```



In [6]:
```
#No.of Policies Taken by Respective Age Group
query="""
select age_group,count(Policy_no) as num_of_policy
from
( select
case when TIMESTAMPDIFF(year, DoB, date(now()))>= 18 and TIMESTAMPDIFF(year, DoB, date(nov
when TIMESTAMPDIFF(year, DoB, date(now()))>=31 and TIMESTAMPDIFF(year, DoB, date(now()))<4
when TIMESTAMPDIFF(year, DoB, date(now()))>=41 and TIMESTAMPDIFF(year, DoB, date(now()))<5
when TIMESTAMPDIFF(year, DoB, date(now()))>=51 and TIMESTAMPDIFF(year, DoB, date(now()))<6
when TIMESTAMPDIFF(year, DoB, date(now()))>=61 and TIMESTAMPDIFF(year, DoB, date(now()))<7
when TIMESTAMPDIFF(year, DoB, date(now()))>=71 and TIMESTAMPDIFF(year, DoB, date(now()))<8
when TIMESTAMPDIFF(year, DoB, date(now()))>=81 and TIMESTAMPDIFF(year, DoB, date(now()))<9
end as age_group,p.policy_no,c.DoB
```

```
from policy p, customer_policy cp, customer c
where p.Policy_no=cp.Policy_no and cp.Customer_id=c.Customer_id)d
group by age_group
"""

df2 = pd.read_sql(query, myconn)


fig = px.pie(df2, values='num_of_policy', names='age_group', title='No.of Policies Taken b
fig.show()
```



In [7]:

```
query="""
select health_plan,count(Policy_no) as num_of_policy, sum(premium) as premium
from
( select
case when premium>= 1000 and premium<2000 then 'basic plan'
when premium>=2000 and premium<3500 then 'silver plan'
when premium>=3500 and premium<4500 then 'gold plan'
when premium>=4500 then 'platinum plan'
end as health_plan,p.policy_no,p.premium
from policy p, health h
where p.Policy_no=h.Policy_no)d
group by health_plan
"""

df_h = pd.read_sql(query, myconn)

query_c="""
select car_plan,count(Policy_no) as num_of_policy, sum(premium) as premium
from
( select
```

```
case when premium>= 1000 and premium<1500 then 'basic plan'
when premium>1500 and premium<=2000 then 'silver plan'
when premium>2000 and premium<=3000 then 'gold plan'
when premium>3000 then 'platinum plan'
end as car_plan,p.policy_no,p.premium
from policy p, car c
where p.Policy_no=c.Policy_no)d
group by car_plan
"""

df_c = pd.read_sql(query_c, myconn)

query_ho="""
select home_plan,count(Policy_no) as num_of_policy, sum(premium) as premium
from
( select
case when premium>= 4000 and premium<5000 then 'basic plan'
when premium>=5000 and premium<9000 then 'silver plan'
when premium>=9000 and premium<12000 then 'gold plan'
when premium>=12000 then 'platinum plan'
end as home_plan,p.policy_no,p.premium
from policy p, home ho
where p.Policy_no=ho.Policy_no)d
group by home_plan
"""

df_ho = pd.read_sql(query_ho, myconn)

df_h.rename(columns = {'health_plan':'plan'}, inplace = True)
df_c.rename(columns = {'car_plan':'plan'}, inplace = True)
df_ho.rename(columns = {'home_plan':'plan'}, inplace = True)

df_h['Category'] = "health"
df_c['Category'] = "car"
df_ho['Category'] = "home"

df_ho_g=df_ho.groupby(['Category','plan']).sum()
df_h_g=df_h.groupby(['Category','plan']).sum()
df_c_g=df_c.groupby(['Category','plan']).sum()

df_merge=df_h_g.append(df_c_g).append(df_ho_g)

output = pd.pivot_table(data=df_merge,
                        index=['plan'],
                        columns=['Category'],
                        values='num_of_policy',
                        aggfunc='sum')
output
```

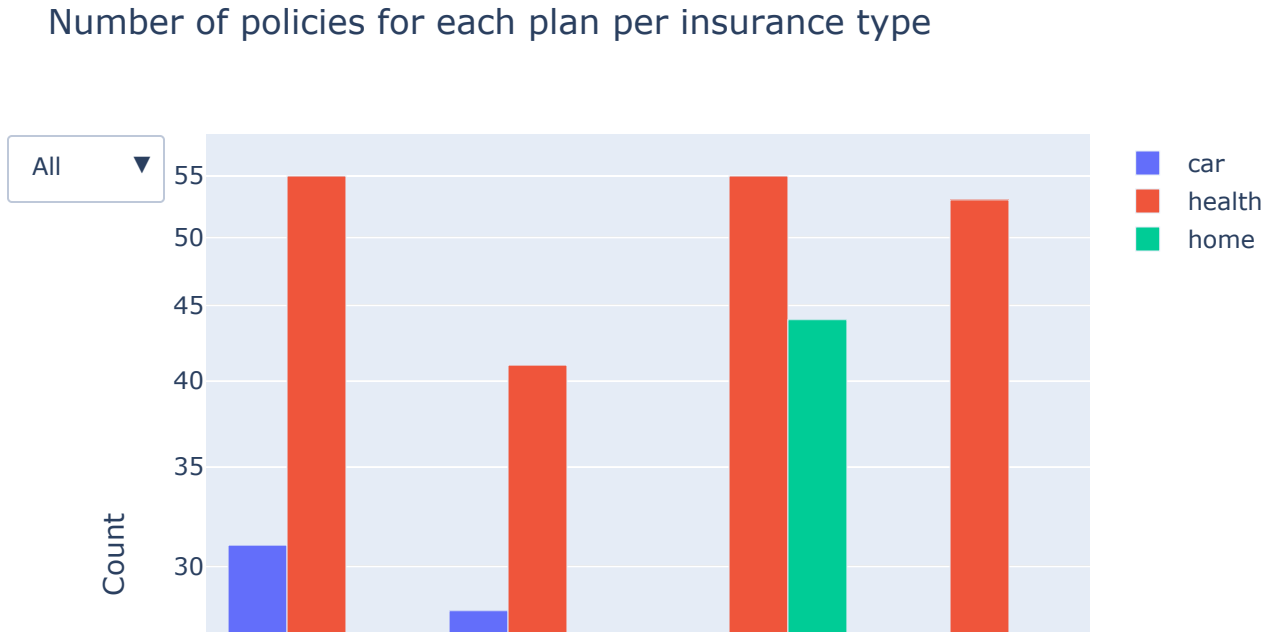
Out[7]:

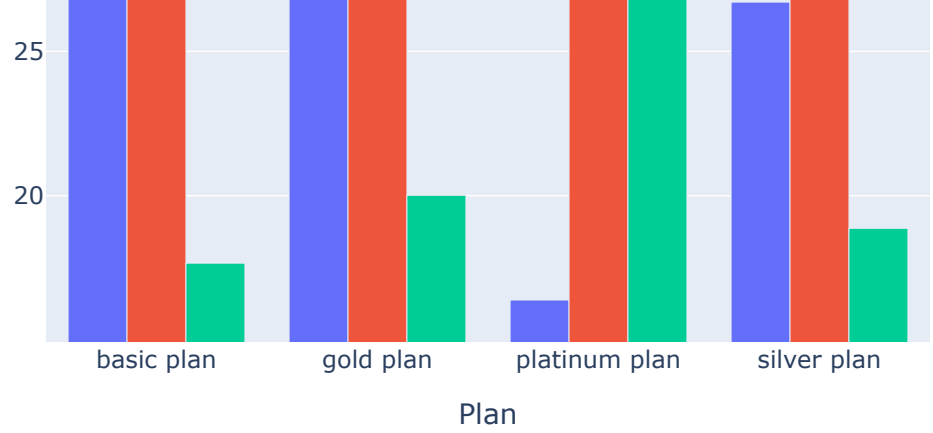
Category	car	health	home
plan			
basic plan	31	55	18
gold plan	28	41	20
platinum plan	17	55	44
silver plan	27	53	19

In [8]:

```
#Number of policies for each plan per insurance type

def multi_plot(df, title, addAll = True):
    fig = go.Figure()
```

```
    for column in df.columns.to_list():
        fig.add_trace(
            go.Bar(
                x = df.index,
                y = df[column],
                name = column
            )
        )

    button_all = dict(label = 'All',
                      method = 'update',
                      args = [{'visible': df.columns.isin(df.columns),
                               'title': 'All',
                               'showlegend':True}])

    def create_layout_button(column):
        return dict(label = column,
                    method = 'update',
                    args = [{'visible': df.columns.isin([column]),
                             'title': column,
                             'showlegend': True}])

    fig.update_layout(
        updatemenus=[go.layout.Updatemenu(
            active = 0,
            buttons = ([button_all] * addAll) + list(df.columns.map(lambda column: create_
            )
        ],
         yaxis_type="log"
    )
    # Update remaining layout properties
    fig.update_layout(
        title_text=title,
        height=600,
        xaxis_title="Plan",
        yaxis_title="Count"

    )

    fig.show()

multi_plot(output, title="Number of policies for each plan per insurance type")
```





Features

In [15]:

```
cursor = mydb.cursor()
today = date.today()
d1 = today.month
d2=  today.day

Message="""Wish you happy birthday from Insurance_Northeastern!!

Regards
Team Insurance_Northeastern """

#password = input("Type your password and press enter: ")

yag = yagmail.SMTP('karakata.s@northeastern.edu', '*********', host='smtp.office365.com',
query="Select DoB,Email from insurance.Customer;"

df_f = pd.read_sql(query, myconn)

for i in range(len(df_f)):
    m=(df_f.loc[i,"DoB"]).month
    d=(df_f.loc[i,"DoB"]).day
    e=(df_f.loc[i,"Email"])
    if m== d1 and d==d2:
      yag.send(
              to=e,
              subject="Happy Birthday!!",
              contents=Message,
              )
```

In [13]:

```
#Creating a cursor object using the cursor() method
conn = mysql.connector.connect(
   user='root', password='mysql1234', host='localhost', database='insurance')
cursor = conn.cursor()

table = input("""Please select the table number on which you want to insert the data

1.Payment
2.Customer
3.Policy
: """)


# Preparing SQL query to INSERT a record into the database.
if table=='1':
```

```
    insert_stmt = (
   """insert into payment(billing_id,bank_account_no,payment_Date,amount,customer_id)
   VALUES (%s, %s,%s, %s,%s)"""
   )

    billing_id = input("Type your billing_id and press enter: ")
    bank_account_no = input("Type your bank_account_no and press enter: ")
    payment_Date = input("Type your payment_Date and press enter: ")
    amount = input("Type your amount and press enter: ")
    customer_id = input("Type your customer_id and press enter: ")
    data = (billing_id,bank_account_no, payment_Date,amount,customer_id)

elif table=='2':
    insert_stmt = (
    """insert into customer(Customer_id,DOB,First_name,Last_name,Phone_num,email,City,Zipc
    VALUES (%s, %s,%s, %s,%s,%s, %s,%s, %s)"""
    )
    Customer_id = input("Type your Customer_id and press enter: ")
    DOB = input("Type your DOB and press enter: ")
    First_name = input("Type your First_name and press enter: ")
    Last_name = input("Type your Last_name and press enter: ")
    Phone_num = input("Type your Phone_num and press enter: ")
    email = input("Type your email and press enter: ")
    City = input("Type your City and press enter: ")
    Zipcode = input("Type your Zipcode and press enter: ")
    Agent_id = input("Type your Agent_id and press enter: ")
    data = (Customer_id,DOB,First_name,Last_name,Phone_num,email,City,Zipcode,Agent_id)


   # Executing the SQL command
cursor.execute(insert_stmt, data)

   # Commit your changes in the database
conn.commit()

print('Thank you, Data is now entered')

#6020
#46522708
#2025
#5000
#204
```

```
Please select the table number on which you want to insert the data

1.Payment
2.Customer
3.Policy
: 1
Type your billing_id and press enter: 6020
Type your bank_account_no and press enter: 46522708
Type your payment_Date and press enter: 2025
Type your amount and press enter: 5000
Type your customer_id and press enter: 204
Thank you, Data is now entered
```