```
def populate_customer(data):
                                                                                                                                                                                                                                        > 🔳 myenv
                                                                                                                                                                                                                                       sex = f"{data['SEX']}
                                    02
                                                                                                                                                                                                                                             index.html
                                                                   age = int(data['AGE'])
                                                                                                                                                                                                                                            customer id = create customer vertex(customer id, sex, age)
                                                                                                                                                                                                                                           e main.py
ertex
                                                                                                                                                                                                                                           ■ Salary_dataset.csv
rtex
                                                            def populate account(data):
                                                                                                                                                                                                                                  \sim OUTLINE
                                                                  account id = f"{data['ID']}'
                                                                   limit_bal = int(data['LIMIT_BAL'])
                                    (16)
vertex
                                                                   account_id = create_account_vertex(account_id, limit_bal, status)
                                                                   return(account id)
 vertex
                                    包
                                                            def populate_payment(data, month):
                                                                  payment_id = f"{data['ID']}-{month}"
if month == "April":
                                                                                                                                                                                                                                  > TIMELINE
                                                                                                                                                                                                                                  > ZIP EXPLORER
                                    Д
                                                 PROBLEMS (37) OUTPUT TERMINAL PORTS GITLENS COMMENTS
                                                                                                                                                                                     oxed{\sum} zsh - project_5 \ + \lor \ \ oxed{igwed} \ \ oxed{m{f m}} \ \ \cdots \ \ \lor SQL CONSOLE: MESSAGES
                                              ∨ TERMINAL
                                                 If this would cause problems for you,
please provide us feedback at https://github.com/pandas-dev/pandas/issues/
54466
                                                                                                                                                                                      assistance, please p
rovide the graph sch
ema or the part of t
he code where the gr
aph schema is define
d.
٦t
                                                                                                                                                                                                                           i r
nt
                                                 import pandas as pd
Timport pandas as pd
Timport ("gpa08cj4nq3gt7iustslmfdvtuctba9g", 1709940919, '2024-03-08 23:35
119')
cution for database po
                                                :19')
(myenv) (base) gideoncrawley@GidBook-Pro project_5 % python3 main.py
//Users/gideoncrawley/MSBA/MSBA/Machine Learning 5505/MLCC_lab/project_5/ma
in.py:1: DeprecationWarning:
Pyarrow will become a required dependency of pandas in the next major rele
ase of pandas (pandas 3.0),
(to allow more performant data types, such as the Arrow string type, and b
etter interoperablity with other libraries)
but was not found to be installed on your system.
If this would cause problems for you,
please provide us feedback at https://github.com/pandas-dev/pandas/issues/
                                                                                                                                                                                                                                  上区
ge of your data instance
                                    {. <mark>(</mark>)
JR ASSIGNMENTS
                                    (2)
                                                     import pandas as pd
KEN: ('as09g8gcd1la0kl0nlru7i4pqr52eega', 1709943081, '2024-03-09 00:11
                                                 TOKEN: ('as09g8gcd1la0kl0nlru7i4pqr52eega', 1709943081
:21')
(myenv) (base) gideoncrawley@GidBook-Pro project_5 %
ed
```

```
import pandas as pd
import pyTigerGraph as tg
host = "https://3314d527106244578c3eff59e7a1ce42.i.tgcloud.io"
graphname = "MLCC Lab"
username = "user 2"
password = "Tb1Yb8Kc6Vt6Jf3 "
secret = "s800no94cutspdqlaae55qfurvr7hsf1"
conn = tg.TigerGraphConnection(host=host, graphname=graphname,
username=username, password=password)
conn.apiToken = conn.getToken(secret)
print("TOKEN: ", conn.apiToken)
data = pd.read csv("/Users/gideoncrawley/MSBA/MSBA/Machine Learning
5505/Chapter 1 cleaned data.csv")
data = data.head()
def create month vertex(month):
    month \overline{id} = f^{\overline{u}} \{ month \}^{\overline{u}}
    attributes = {
```

```
"name": f"{month}"
    conn.upsertVertex("Month", month id, attributes)
    return (month id)
def create account vertex(account id, limit bal, status):
    account id = f"{account id}"
    attributes = {
        "limit bal": limit bal,
        "status" : status
    conn.upsertVertex("Account", account id, attributes)
    return(account id)
def create customer vertex(customer id, sex, age):
    customer id = f"{customer id}"
    attributes = {
        "sex": "male" if sex == 1 else "female",
        "age": age,
    conn.upsertVertex("Customer", customer id, attributes)
    return (customer id)
def create_billing vertex(bill id, bill amt):
    billing id = f''{bill id}"
    attributes = {
        "bill amt": bill amt
    conn.upsertVertex("Billing", billing id, attributes)
    return(billing id)
def create payment vertex (payment id, pay amt):
    payment id = f'' \{payment id\}''
    attributes = {
        "pay amt": pay amt
    conn.upsertVertex("Payment", payment id, attributes)
    return(payment id)
def populate month (month):
    month id = month
    name = month
    month id = create month vertex(month)
    return(month id)
def populate billing(data, month):
    billing id = f"{data['ID']}-{month}"
    if month == "April":
        bill amt = int(data['BILL AMT6'])
    elif month == "May":
        bill amt = int(data['BILL AMT5'])
    elif month == "June":
        bill amt = int(data['BILL AMT4'])
    elif month == "July":
```

```
bill amt = int(data['BILL AMT3'])
    elif month == "August":
        bill amt = int(data['BILL AMT2'])
    elif month == "September":
        bill amt = int(data['BILL AMT1'])
   bill id = create billing vertex(billing id, bill amt)
    conn.upsertEdge("Month", f"{month}", "invoiced", "Billing",
f"{billing id}")
    return (billing id)
def populate customer (data):
   customer id = f"{data['ID']}"
    sex = f"{data['SEX']}"
   age = int(data['AGE'])
    customer id = create customer vertex(customer id, sex, age)
   return(customer id)
def populate account (data):
   account id = f"{data['ID']}"
   limit bal = int(data['LIMIT BAL'])
    status = bool(data['default payment next month'])
    account_id = create_account vertex(account id, limit bal, status)
   return(account id)
def populate payment (data, month):
   payment id = f"{data['ID']}-{month}"
    if month == "April":
        pay amt = int(data['PAY AMT6'])
    elif month == "May":
        pay amt = int(data['PAY AMT5'])
    elif month == "June":
        pay amt = int(data['PAY AMT4'])
    elif month == "July":
       pay amt = int(data['PAY AMT3'])
    elif month == "August":
        pay amt = int(data['PAY AMT2'])
    elif month == "September":
        pay amt = int(data['PAY AMT1'])
   payment id = create payment vertex(payment id, pay amt)
    return (payment id)
def populate(data):
   month list = ['April', 'May', 'June', 'July', 'August', 'September']
    for month in month list:
        month = populate month(month)
        for index, row in data.iterrows():
            billing id = populate billing(row, month)
            customer id = populate customer(row)
            account id = populate account(row)
            payment id = populate payment(row, month)
            conn.upsertEdge("Billing", f"{billing id}",
"billed", "Account", f" {account_id}")
```

```
conn.upsertEdge("Customer", f"{customer id}", "holds",
"Payment", f"{payment_id}")
        conn.upsertEdge("Customer", f"{customer_id}", "belongs",
"Billing", f"{billing id}")
```

populate(data)