

Assignment – 7:

Tutedude: Course: Python

Module 12:

1. Installed PostgreSQL

2. Creating a Database

- Initial Default databases present:

```
SQL Shell (psql)
Server [localhost]:
Database [postgres]:
Port [5432]:
Username [postgres]:
Password for user postgres:

psql (17.4)
WARNING: Console code page (437) differs from Windows code page (1252)
8-bit characters might not work correctly. See psql reference
page "Notes for Windows users" for details.
Type "help" for help.

postgres=# \l
      List of databases
  Name | Owner | Encoding | Locale Provider | Collate | Ctype | Locale | ICU Rules | Access privileges
-----+-----+-----+-----+-----+-----+-----+-----+-----
 postgres | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 | | | =c/postgres +
 template0 | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 | | | postgres=Ctc/postgres
 template1 | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 | | | =c/postgres +
                                     postgres=Ctc/postgres
(3 rows)

postgres=#
```

- Created a new database named **demodb**:

```
SQL Shell (psql)
Port [5432]:
Username [postgres]:
Password for user postgres:

psql (17.4)
WARNING: Console code page (437) differs from Windows code page (1252)
8-bit characters might not work correctly. See psql reference
page "Notes for Windows users" for details.
Type "help" for help.

postgres=# \l
      List of databases
  Name | Owner | Encoding | Locale Provider | Collate | Ctype | Locale | ICU Rules | Access privileges
-----+-----+-----+-----+-----+-----+-----+-----+-----
 postgres | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 | | | =c/postgres +
 template0 | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 | | | postgres=Ctc/postgres
 template1 | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 | | | =c/postgres +
                                     postgres=Ctc/postgres
(3 rows)

postgres=# create database demodb;
CREATE DATABASE
postgres=# \l
      List of databases
  Name | Owner | Encoding | Locale Provider | Collate | Ctype | Locale | ICU Rules | Access privileges
-----+-----+-----+-----+-----+-----+-----+-----+-----
 demodb | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 | | | =c/postgres +
 postgres | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 | | | postgres=Ctc/postgres
 template0 | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 | | | =c/postgres +
 template1 | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 | | | postgres=Ctc/postgres
(4 rows)

postgres=#
```

- Deleting a database **testdb**:

```

SQL Shell (psql)
Username [postgres]:
Password for user postgres:

psql (17.4)
WARNING: Console code page (437) differs from Windows code page (1252)
8-bit characters might not work correctly. See psql reference
page "Notes for Windows users" for details.
Type "help" for help.

postgres=# \l
      List of databases
  Name | Owner  | Encoding | Locale Provider | Collate | Ctype | Locale | ICU Rules | Access privileges
-----+-----+-----+-----+-----+-----+-----+-----+-----
postgres | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 |  |  | =c/postgres +
template0 | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 |  |  | postgres=CTc/postgres
template1 | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 |  |  | =c/postgres +
                                           postgres=CTc/postgres
(3 rows)

postgres=# create database demodb;
CREATE DATABASE
postgres=# \l
      List of databases
  Name | Owner  | Encoding | Locale Provider | Collate | Ctype | Locale | ICU Rules | Access privileges
-----+-----+-----+-----+-----+-----+-----+-----+-----
demodb | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 |  |  |
postgres | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 |  |  | =c/postgres +
template0 | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 |  |  | postgres=CTc/postgres
template1 | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 |  |  | =c/postgres +
                                           postgres=CTc/postgres
(4 rows)

postgres=# \c demodb
You are now connected to database "demodb" as user "postgres".
demodb=# create database testdb;
CREATE DATABASE
demodb=# \l
      List of databases
  Name | Owner  | Encoding | Locale Provider | Collate | Ctype | Locale | ICU Rules | Access privileges
-----+-----+-----+-----+-----+-----+-----+-----+-----
demodb | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 |  |  |
postgres | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 |  |  | =c/postgres +
template0 | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 |  |  | postgres=CTc/postgres
template1 | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 |  |  | =c/postgres +
testdb | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 |  |  | postgres=CTc/postgres
(5 rows)

demodb=# drop database testdb;
DROP DATABASE
demodb=# \l
      List of databases
  Name | Owner  | Encoding | Locale Provider | Collate | Ctype | Locale | ICU Rules | Access privileges
-----+-----+-----+-----+-----+-----+-----+-----+-----
demodb | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 |  |  |
postgres | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 |  |  | =c/postgres +
template0 | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 |  |  | postgres=CTc/postgres
template1 | postgres | UTF8 | libc | English_United States.1252 | English_United States.1252 |  |  | =c/postgres +
                                           postgres=CTc/postgres
(4 rows)

demodb=#

```

- Create a Table and Insert data into the table:

```
SQL Shell (psql)

demodb=# CREATE TABLE students(name text, number int, age int);
CREATE TABLE
demodb=# \d
          List of relations
Schema |   Name   | Type  | Owner
-----+-----+-----+-----
public | students | table | postgres
(1 row)

demodb=# INSERT INTO studetns(name,number,age) values('Peter',02341,26);
ERROR:  relation "studetns" does not exist
LINE 1: INSERT INTO studetns(name,number,age) values('Peter',02341,2...
                  ^
demodb=# INSERT INTO students(name,number,age) values('Peter',02341,26);
INSERT 0 1
demodb=# INSERT INTO students(name,number,age) values('Samuel',03258,31);
INSERT 0 1
demodb=#
```

- Retrieving data from database and deleting contents in the table:

```
demodb=#
demodb=#
demodb=#
demodb=# SELECT * FROM Students;
  name | number | age
-----+-----+-----
 Peter |   2341 |  26
Samuel |   3258 |  31
(2 rows)
```

```
demodb=# SELECT name FROM students;
  name
-----
 Peter
Samuel
(2 rows)
```

```
demodb=# SELECT * FROM Students where name = 'Samuel';
  name | number | age
-----+-----+-----
Samuel |   3258 |  31
(1 row)
```

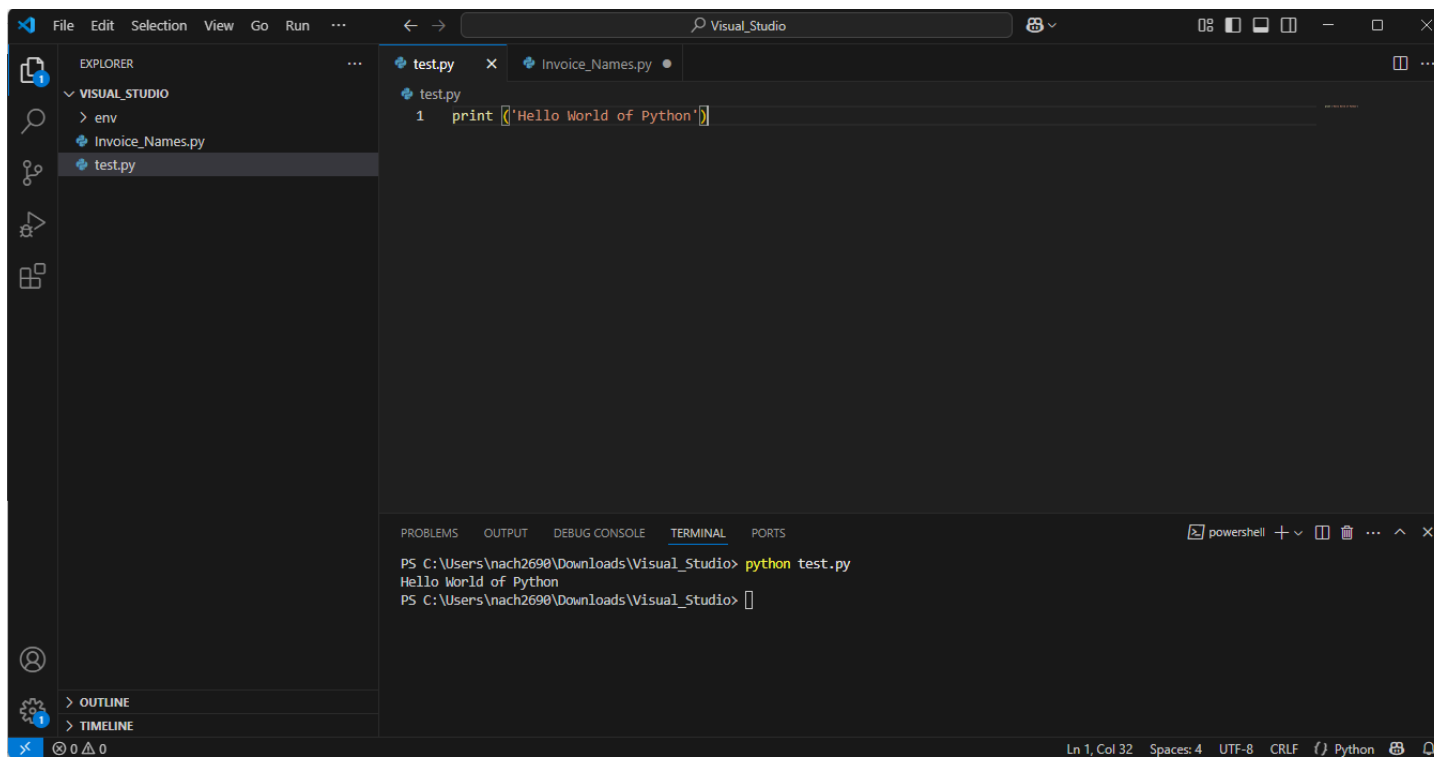
```
demodb=# SELECT * FROM Students where age < 30;
  name | number | age
-----+-----+-----
 Peter |   2341 |  26
(1 row)
```

```
demodb=# TRUNCATE TABLE students;
TRUNCATE TABLE
demodb=# \d
          List of relations
 Schema |   Name   | Type  | Owner
-----+-----+-----+-----
 public | students | table | postgres
(1 row)
```

```
demodb=# SELECT * FROM Students;
  name | number | age
-----+-----+-----
(0 rows)
```

```
demodb=#
```

- Setting up virtualenv & Testing Visual Code:

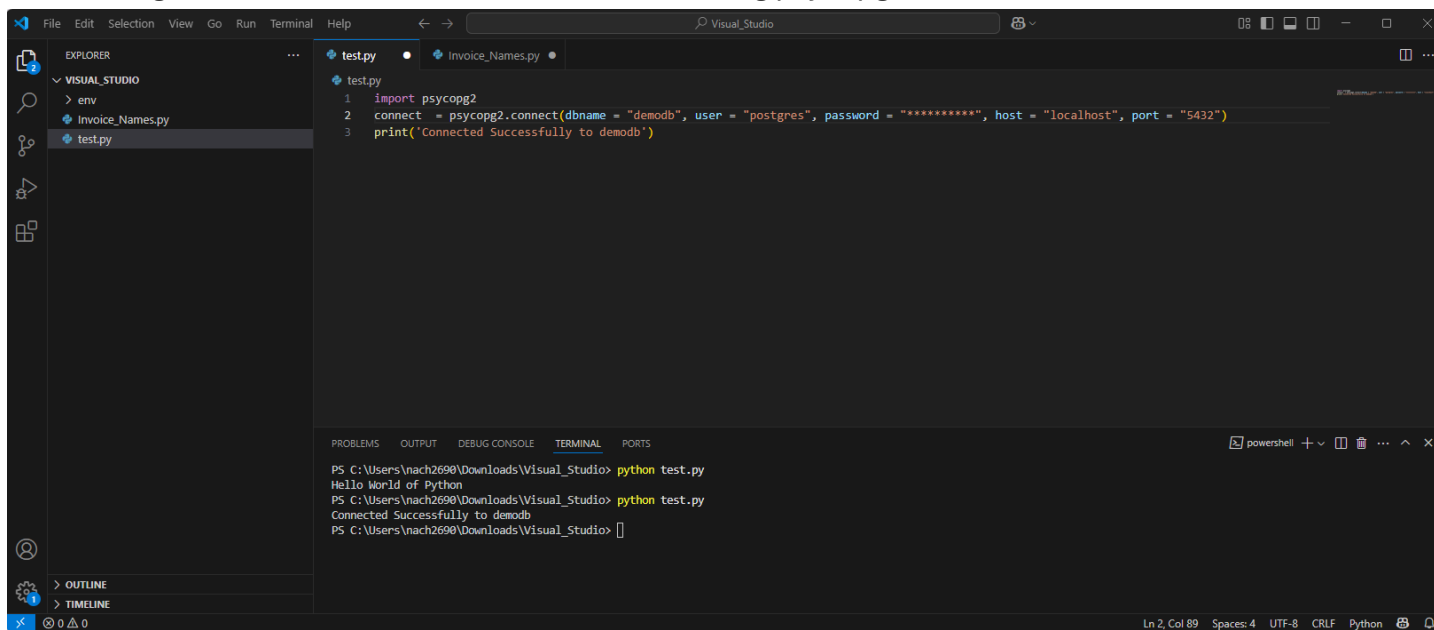


This screenshot shows the Visual Studio Code interface with a file explorer on the left containing 'env', 'Invoice_Names.py', and 'test.py'. The main editor displays 'test.py' with a single line of code: `1 print('Hello World of Python')`. The terminal at the bottom shows the command `python test.py` being executed, resulting in the output 'Hello World of Python'.

```
test.py
1 print('Hello World of Python')
```

```
PS C:\Users\nach2690\Downloads\Visual_Studio> python test.py
Hello World of Python
PS C:\Users\nach2690\Downloads\Visual_Studio>
```

- Connecting to the database in Visual Studio Code using psycopg2:



This screenshot shows the Visual Studio Code interface with a file explorer on the left containing 'env', 'Invoice_Names.py', and 'test.py'. The main editor displays 'test.py' with three lines of code: `1 import psycopg2`, `2 connect = psycopg2.connect(dbname = "demodb", user = "postgres", password = "*****", host = "localhost", port = "5432")`, and `3 print('Connected Successfully to demodb')`. The terminal at the bottom shows the command `python test.py` being executed, resulting in the output 'Connected Successfully to demodb'.

```
test.py
1 import psycopg2
2 connect = psycopg2.connect(dbname = "demodb", user = "postgres", password = "*****", host = "localhost", port = "5432")
3 print('Connected Successfully to demodb')
```

```
PS C:\Users\nach2690\Downloads\Visual_Studio> python test.py
Hello World of Python
PS C:\Users\nach2690\Downloads\Visual_Studio> python test.py
Connected Successfully to demodb
PS C:\Users\nach2690\Downloads\Visual_Studio>
```

- Creating table using python in Visual Studio Code using psycopg2: We can also see psql powershell if the table is created in the **demodb**.

The screenshot shows the Visual Studio Code interface with the following components:

- EXPLORER:** Shows the file structure with 'test.py' and 'Invoice_Names.py'.
- test.py:**

```

1 import psycopg2
2 conn = psycopg2.connect(dbname = "demodb", user = "postgres", password = " ", host = "localhost", port = "5432")
3 cursor = conn.cursor()
4 cursor.execute(''' CREATE TABLE employees(Name text, ID int, Age int); ''')
5 print('Table created Successfully')
6 conn.commit()
7 conn.close()
8

```
- TERMINAL:**

```

PS C:\Users\nach2690\Downloads\Visual_Studio> python test.py
Hello World of Python
PS C:\Users\nach2690\Downloads\Visual_Studio> python test.py
Connected Successfully to demodb
PS C:\Users\nach2690\Downloads\Visual_Studio> python test.py
Table created Successfully
PS C:\Users\nach2690\Downloads\Visual_Studio>

```
- SQL Shell (psql):**

```

demodb=# TRUNCATE TABLE students;
TRUNCATE TABLE
demodb=# \d
List of relations
Schema | Name | Type | Owner
-----+-----+-----+-----
public | students | table | postgres
(1 row)

demodb=# SELECT * FROM Students;
name | number | age
-----+-----+-----
(0 rows)

demodb=# \d
List of relations
Schema | Name | Type | Owner
-----+-----+-----+-----
public | employees | table | postgres
public | students | table | postgres
(2 rows)

demodb=#

```

- Inserting the data using python in Visual Studio Code using psycopg2: We can also see psql powershell if the employee is added in the **employees Table**.

The screenshot shows the Visual Studio Code interface with the following components:

- EXPLORER:** Shows the file structure with 'test.py' and 'Invoice_Names.py'.
- test.py:**

```

1 import psycopg2
2 def table():
3     conn = psycopg2.connect(dbname = "demodb", user = "postgres", password = " ", host = "localhost", port = "5432")
4     cursor = conn.cursor()
5     cursor.execute(''' CREATE TABLE employees(Name text, ID int, Age int); ''')
6     print('Table created Successfully')
7     conn.commit()
8     conn.close()
9
10 def data():
11     conn = psycopg2.connect(dbname = "demodb", user = "postgres", password = " ", host = "localhost", port = "5432")
12     cursor = conn.cursor()
13     cursor.execute(''' INSERT INTO employees(Name,ID,Age) values('Mark', 1457,37); ''')
14     print('Data Entered Successfully')
15     conn.commit()
16     conn.close()
17
18 data()

```
- TERMINAL:**

```

PS C:\Users\nach2690\Downloads\Visual_Studio> python test.py
Hello World of Python
PS C:\Users\nach2690\Downloads\Visual_Studio> python test.py
Connected Successfully to demodb
PS C:\Users\nach2690\Downloads\Visual_Studio> python test.py
Table created Successfully
PS C:\Users\nach2690\Downloads\Visual_Studio> python test.py
Data Entered Successfully
PS C:\Users\nach2690\Downloads\Visual_Studio>

```
- SQL Shell (psql):**

```

Schema | Name | Type | Owner
-----+-----+-----+-----
public | students | table | postgres
(1 row)

demodb=# SELECT * FROM Students;
name | number | age
-----+-----+-----
(0 rows)

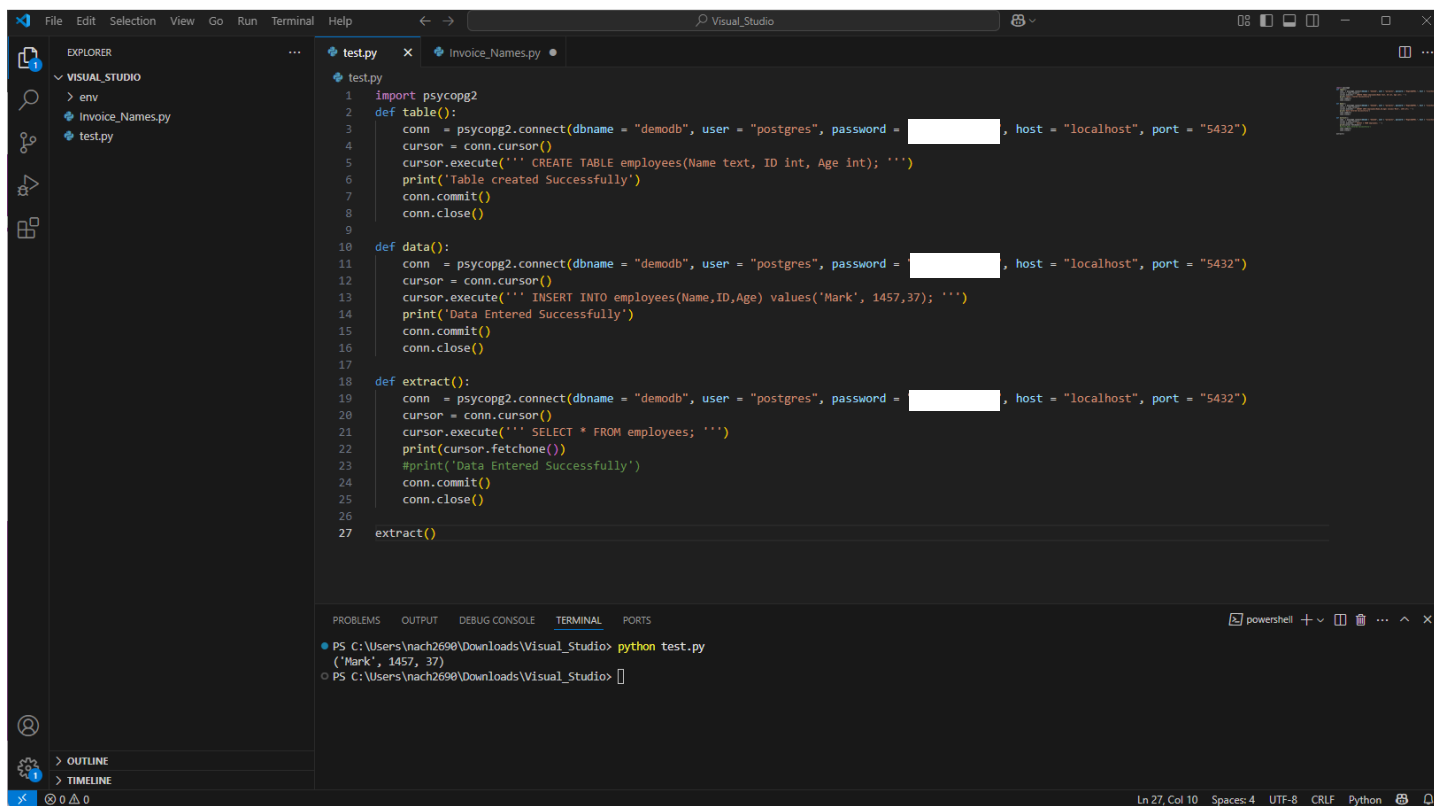
demodb=# \d
List of relations
Schema | Name | Type | Owner
-----+-----+-----+-----
public | employees | table | postgres
public | students | table | postgres
(2 rows)

demodb=# SELECT * FROM employees;
name | id | age
-----+-----+-----
Mark | 1457 | 37
(1 row)

demodb=#

```

- Extracting the data from the database in Visual Studio Code using psycopg2:

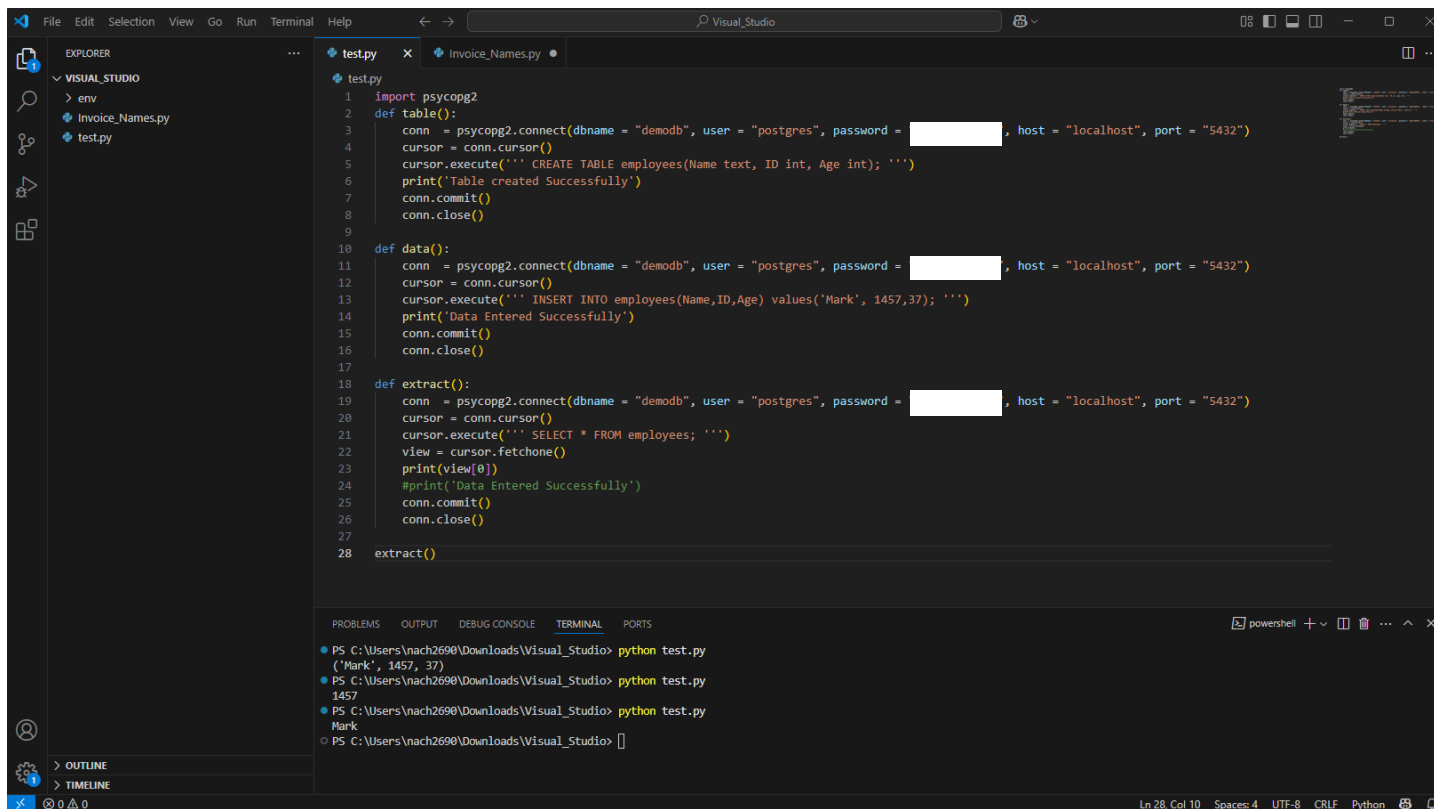


```
test.py
1 import psycopg2
2 def table():
3     conn = psycopg2.connect(dbname = "demodb", user = "postgres", password = " ", host = "localhost", port = "5432")
4     cursor = conn.cursor()
5     cursor.execute('CREATE TABLE employees(Name text, ID int, Age int);')
6     print('Table created Successfully')
7     conn.commit()
8     conn.close()
9
10 def data():
11     conn = psycopg2.connect(dbname = "demodb", user = "postgres", password = " ", host = "localhost", port = "5432")
12     cursor = conn.cursor()
13     cursor.execute('INSERT INTO employees(Name, ID, Age) values('Mark', 1457, 37);')
14     print('Data Entered Successfully')
15     conn.commit()
16     conn.close()
17
18 def extract():
19     conn = psycopg2.connect(dbname = "demodb", user = "postgres", password = " ", host = "localhost", port = "5432")
20     cursor = conn.cursor()
21     cursor.execute('SELECT * FROM employees;')
22     print(cursor.fetchone())
23     #print('Data Entered Successfully')
24     conn.commit()
25     conn.close()
26
27 extract()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\Nnach2690\Downloads\Visual_Studio> python test.py
('Mark', 1457, 37)
PS C:\Users\Nnach2690\Downloads\Visual_Studio>
```

- Selecting choice of positions in the tuple:

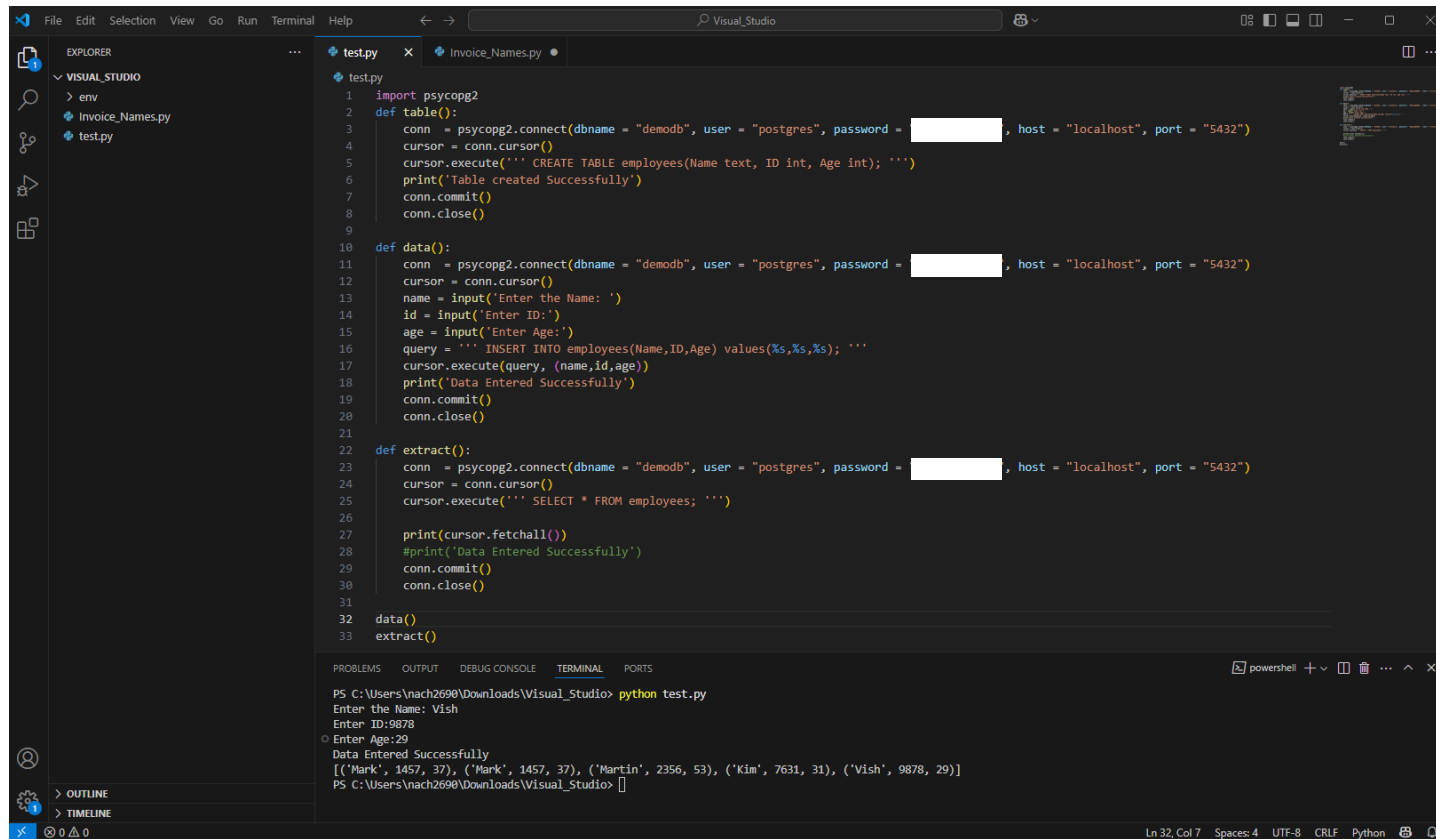


```
test.py
1 import psycopg2
2 def table():
3     conn = psycopg2.connect(dbname = "demodb", user = "postgres", password = " ", host = "localhost", port = "5432")
4     cursor = conn.cursor()
5     cursor.execute('CREATE TABLE employees(Name text, ID int, Age int);')
6     print('Table created Successfully')
7     conn.commit()
8     conn.close()
9
10 def data():
11     conn = psycopg2.connect(dbname = "demodb", user = "postgres", password = " ", host = "localhost", port = "5432")
12     cursor = conn.cursor()
13     cursor.execute('INSERT INTO employees(Name, ID, Age) values('Mark', 1457, 37);')
14     print('Data Entered Successfully')
15     conn.commit()
16     conn.close()
17
18 def extract():
19     conn = psycopg2.connect(dbname = "demodb", user = "postgres", password = " ", host = "localhost", port = "5432")
20     cursor = conn.cursor()
21     cursor.execute('SELECT * FROM employees;')
22     view = cursor.fetchone()
23     print(view[0])
24     #print('Data Entered Successfully')
25     conn.commit()
26     conn.close()
27
28 extract()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\Nnach2690\Downloads\Visual_Studio> python test.py
('Mark', 1457, 37)
PS C:\Users\Nnach2690\Downloads\Visual_Studio> python test.py
1457
PS C:\Users\Nnach2690\Downloads\Visual_Studio> python test.py
Mark
PS C:\Users\Nnach2690\Downloads\Visual_Studio>
```

- Adding the input from the user in Visual Studio Code using psycopg2: You can see the inputs in the terminal and the extracted fetchall output to see all the records and check the latest is added.



The screenshot shows the Visual Studio Code interface with a Python file named `test.py` open. The code defines three functions: `table()`, `data()`, and `extract()`. The `table()` function creates a table named `employees` with columns `Name`, `ID`, and `Age`. The `data()` function prompts the user for a name, ID, and age, and inserts the data into the `employees` table. The `extract()` function fetches all records from the `employees` table and prints them. The terminal output shows the execution of the script, with the user entering 'Vish' for the name, 9878 for the ID, and 29 for the age. The output displays the data entered successfully and the fetched records.

```
1 import psycopg2
2 def table():
3     conn = psycopg2.connect(dbname = "demodb", user = "postgres", password = " ", host = "localhost", port = "5432")
4     cursor = conn.cursor()
5     cursor.execute('CREATE TABLE employees(Name text, ID int, Age int);')
6     print('Table created Successfully')
7     conn.commit()
8     conn.close()
9
10 def data():
11     conn = psycopg2.connect(dbname = "demodb", user = "postgres", password = " ", host = "localhost", port = "5432")
12     cursor = conn.cursor()
13     name = input('Enter the Name: ')
14     id = input('Enter ID:')
15     age = input('Enter Age:')
16     query = 'INSERT INTO employees(Name,ID,Age) values(%s,%s,%s);'
17     cursor.execute(query, (name,id,age))
18     print('Data Entered Successfully')
19     conn.commit()
20     conn.close()
21
22 def extract():
23     conn = psycopg2.connect(dbname = "demodb", user = "postgres", password = " ", host = "localhost", port = "5432")
24     cursor = conn.cursor()
25     cursor.execute('SELECT * FROM employees;')
26
27     print(cursor.fetchall())
28     #print('Data Entered Successfully')
29     conn.commit()
30     conn.close()
31
32 data()
33 extract()
```

Terminal Output:

```
PS C:\Users\Nnach2698\Downloads\Visual_Studio> python test.py
Enter the Name: Vish
Enter ID:9878
Enter Age:29
Data Entered Successfully
[('Mark', 1457, 37), ('Mark', 1457, 37), ('Martin', 2356, 53), ('Kim', 7631, 31), ('Vish', 9878, 29)]
PS C:\Users\Nnach2698\Downloads\Visual_Studio>
```