TUTORIAL

PostgreSQL with Python

CRUD opt with postgreSQl in python

@_python.py_
Python | Programming | Projects

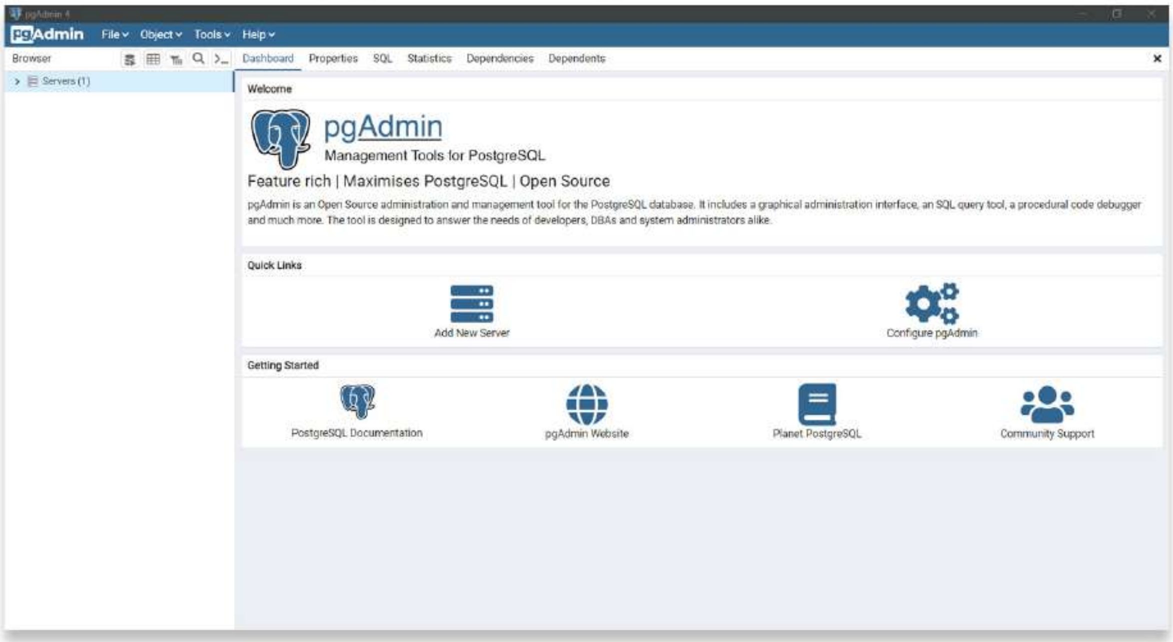
Arvind Singh
Python FullStack Developer

SAVE IT

02

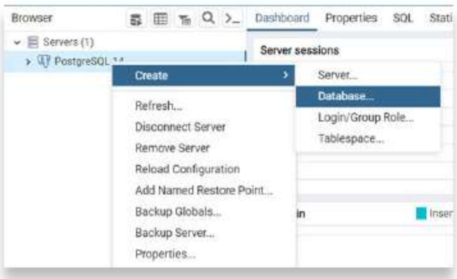
Download pgAdmin4

https://www.pgadmin.org/download/

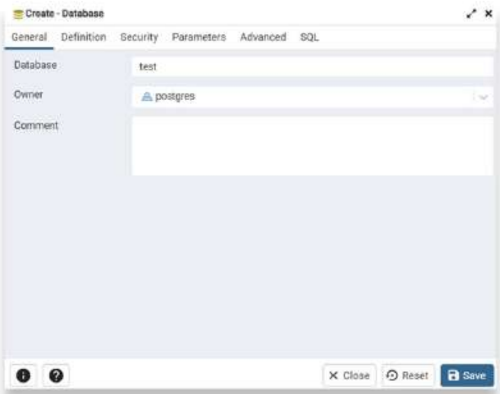


pgAdmin is an Open Source administration and management tool for the PostgreSQL database. It includes a graphical administration interface, an SQL query tool, a procedural code debugger and much more. The tool is designed to answer the needs of developers, DBAs and system administrators alike.

Create new database

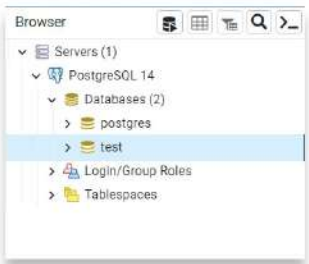


I create 'test' name database



Database: test
Owner: postgres
Comment:

Here two database postgres and test. we use test database



@_python.py_
Python | Programming | Projects

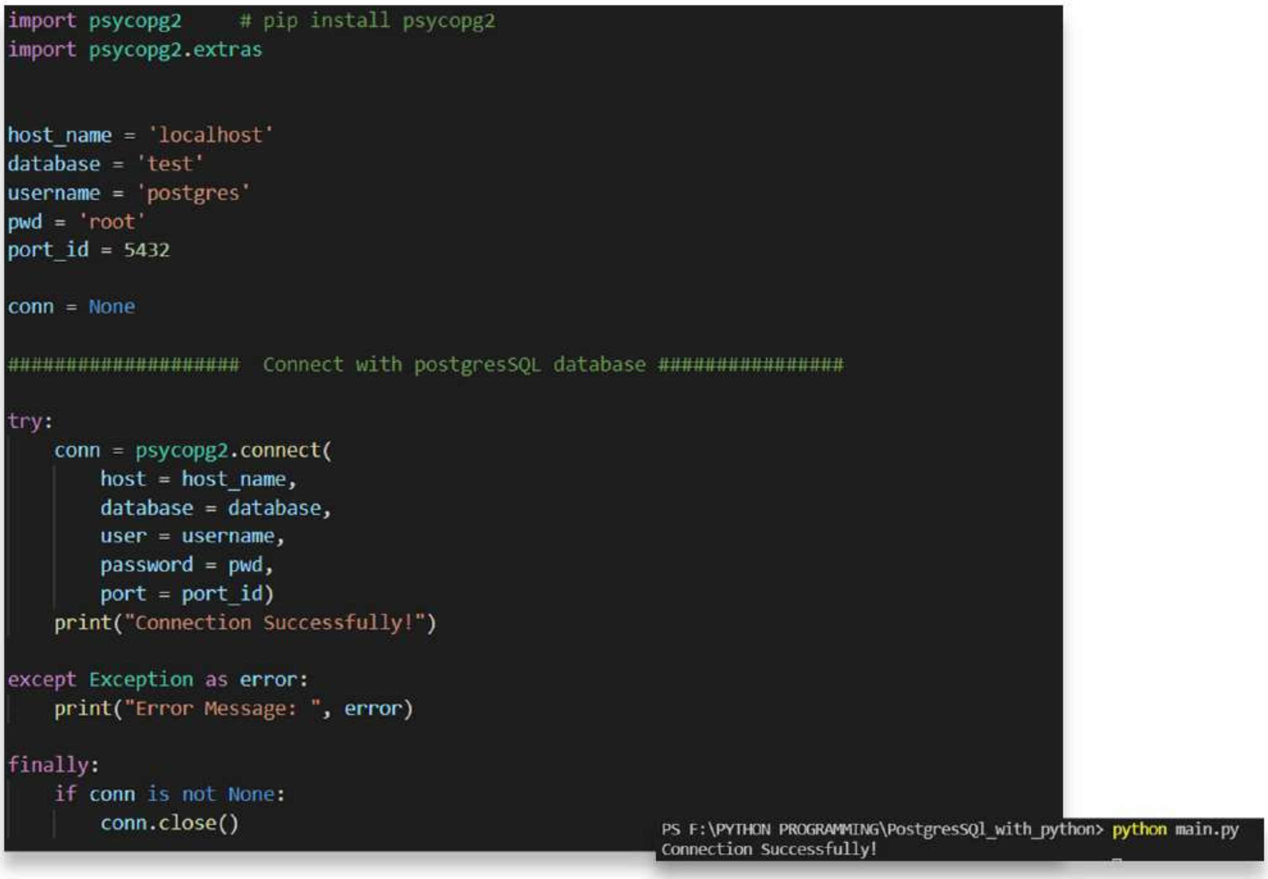
SAVE IT

03

1. Install psycopg2

run this command - pip install psycopg2

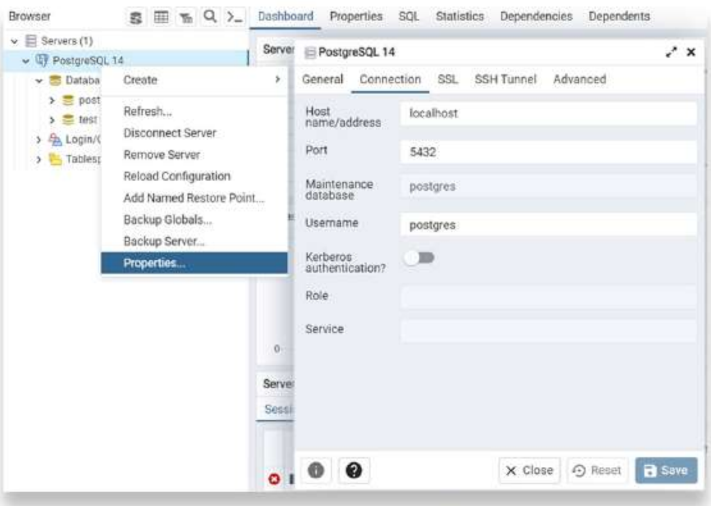
2. Connect with database.



```python
import psycopg2     # pip install psycopg2
import psycopg2.extras


host_name = 'localhost'
database = 'test'
username = 'postgres'
pwd = 'root'
port_id = 5432

conn = None

####################  Connect with postgresSQL database ################

try:
    conn = psycopg2.connect(
        host = host_name,
        database = database,
        user = username,
        password = pwd,
        port = port_id)
    print("Connection Successfully!")

except Exception as error:
    print("Error Message: ", error)

finally:
    if conn is not None:
        conn.close()
```

```
PS F:\PYTHON PROGRAMMING\PostgresSql_with_python> python main.py
Connection Successfully!
```

to check hostname, port and username
right click + Properties and go on Connection
password is same as you set in installation time.



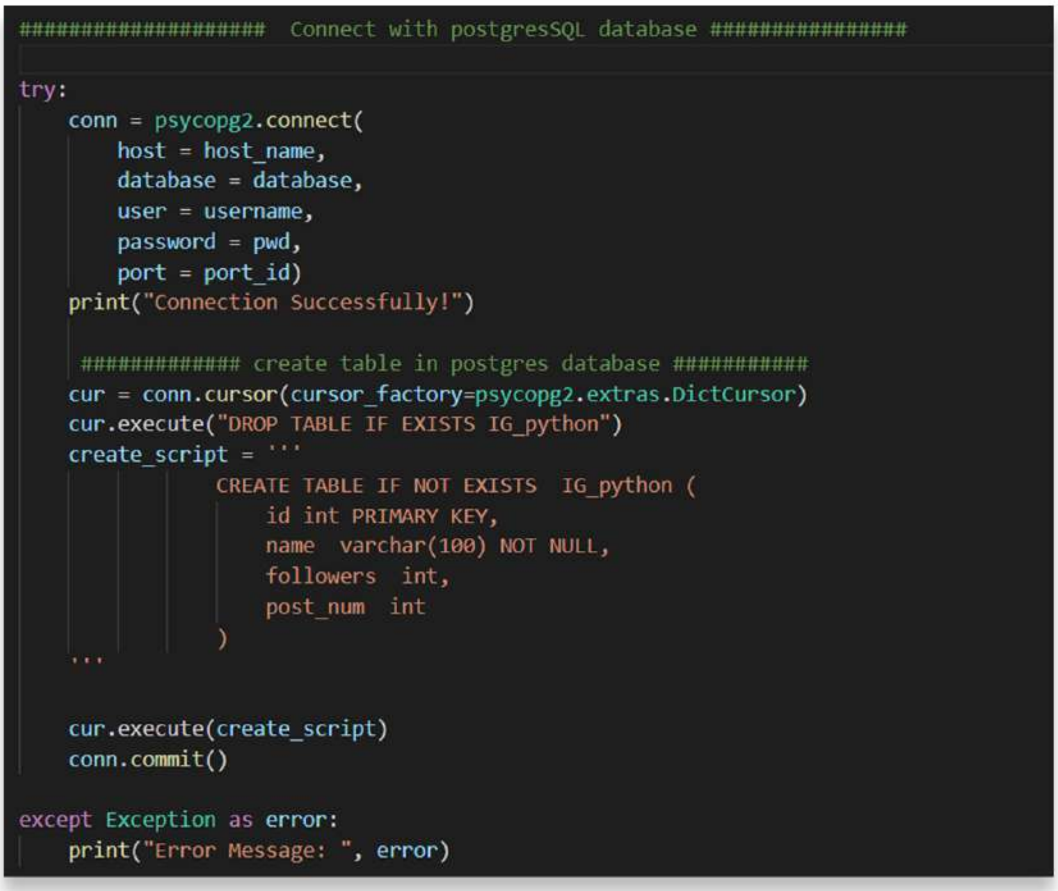
PostgreSQL 14

Host name/address: localhost
Port: 5432
Maintenance database: postgres
Username: postgres
Kerberos authentication?
Role
Service

@_python.py_
Python | Programming | Projects

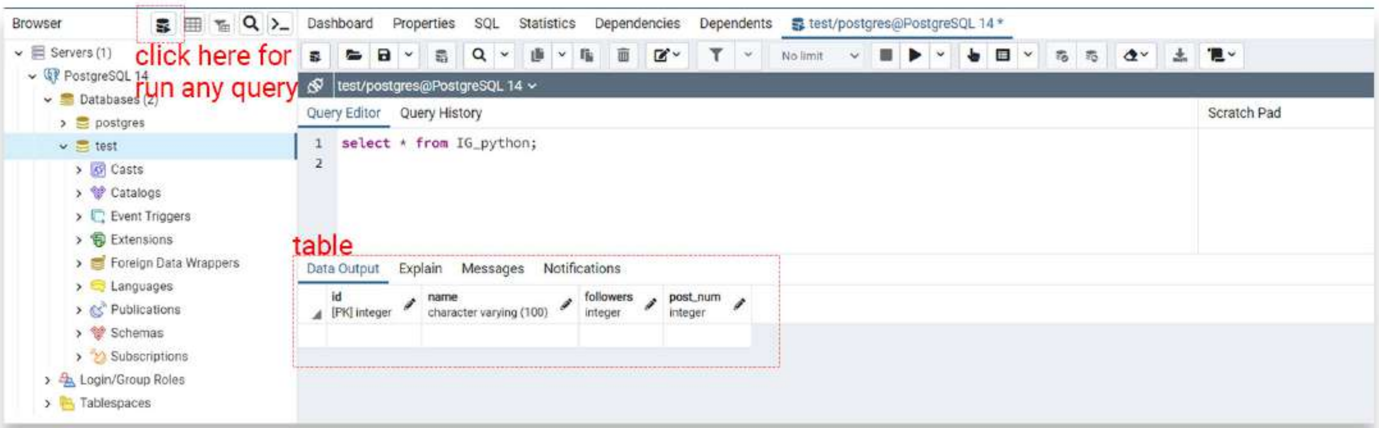
SAVE IT

3. Create table.



```
####################  Connect with postgresSQL database ################

try:
    conn = psycopg2.connect(
        host = host_name,
        database = database,
        user = username,
        password = pwd,
        port = port_id)
    print("Connection Successfully!")

     ############# create table in postgres database ###########
    cur = conn.cursor(cursor_factory=psycopg2.extras.DictCursor)
    cur.execute("DROP TABLE IF EXISTS IG_python")
    create_script = '''
                CREATE TABLE IF NOT EXISTS  IG_python (
                    id int PRIMARY KEY,
                    name  varchar(100) NOT NULL,
                    followers  int,
                    post_num  int
                )
    '''

    cur.execute(create_script)
    conn.commit()

except Exception as error:
    print("Error Message: ", error)
```

for check table in database run select query in pgAdmin



```
select * from IG_python;
```

id [PK] integer	name character varying (100)	followers integer	post_num integer

05

3. Insert data.

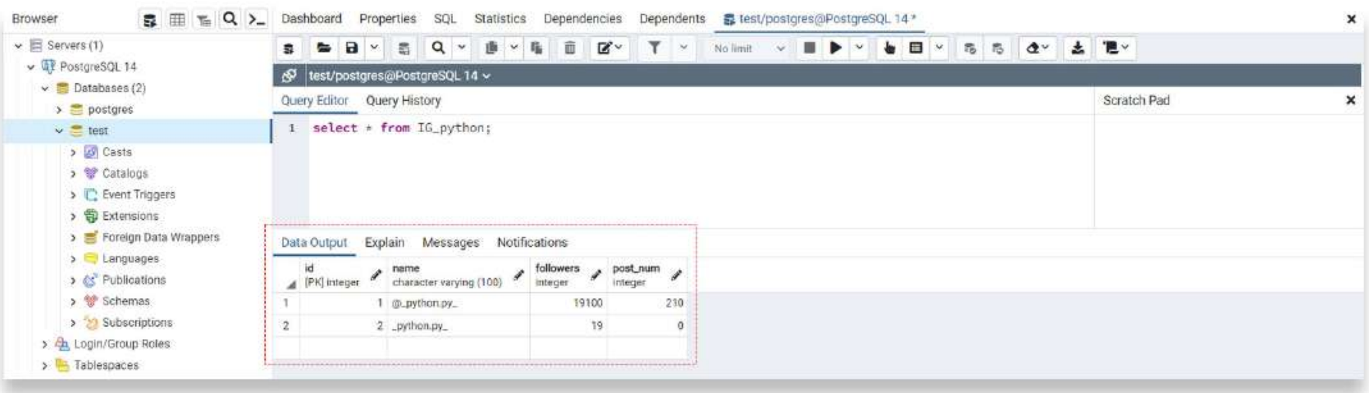
```python
########### Insert data in IG_python table ###############

insert_script = "INSERT INTO IG_python (id, name, followers, post_num) values (%s, %s, %s, %s)"
insert_values = [(1, '@_python.py_', 19100, 210), (2, '_python.py_', 19, 0)]
for insert_value in insert_values:
    cur.execute(insert_script, insert_value)
conn.commit()
```

Note: write code between in try and except

4. Select data.

```python
########### Insert data in IG_python table ###############

insert_script = "INSERT INTO IG_python (id, name, followers, post_num) values (%s, %s, %s, %s)"
insert_values = [(1, '@_python.py_', 19100, 210), (2, '_python.py_', 19, 0)]
for insert_value in insert_values:
    cur.execute(insert_script, insert_value)

############ Select data from IG_python table ##############
cur.execute('SELECT * FROM IG_python')
records = cur.fetchall()
print("Select data is : \n ",)
for record in records:
    print(record)
print()
conn.commit()
```

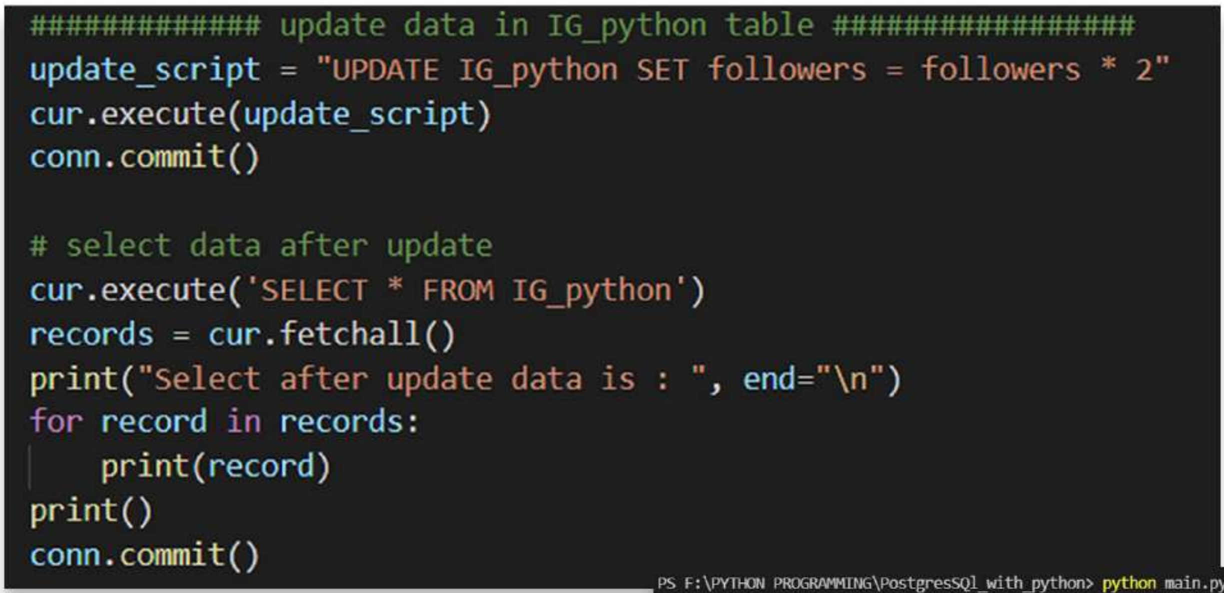
```
PS F:\PYTHON PROGRAMMING\PostgresSQl_with_python> python main.py
Connection Successfully!
Select data is :

[1, '@_python.py_', 19100, 210]
[2, '_python.py_', 19, 0]
```



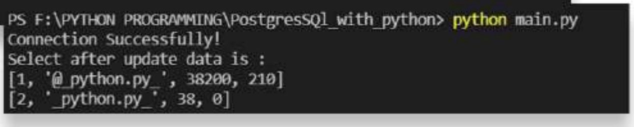
```sql
select * from IG_python;
```

	id [PK] integer	name character varying (100)	followers integer	post_num integer
1	1	@_python.py_	19100	210
2	2	_python.py_	19	0

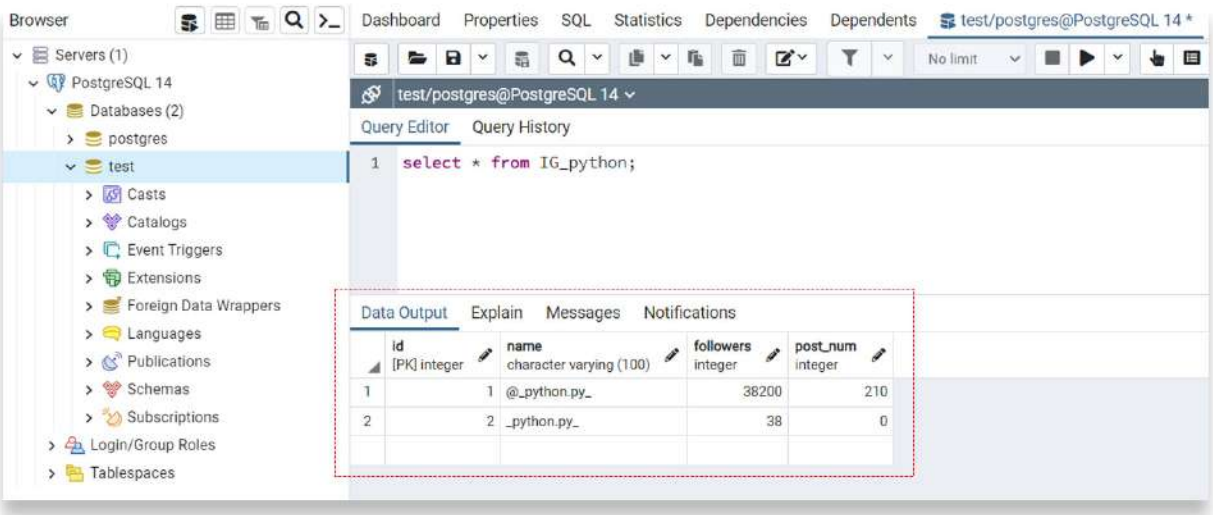
5. Update.



```
############## update data in IG_python table ##################
update_script = "UPDATE IG_python SET followers = followers * 2"
cur.execute(update_script)
conn.commit()

# select data after update
cur.execute('SELECT * FROM IG_python')
records = cur.fetchall()
print("Select after update data is : ", end="\n")
for record in records:
    print(record)
print()
conn.commit()
```



```
PS F:\PYTHON PROGRAMMING\PostgresSQl_with_python> python main.py
Connection Successfully!
Select after update data is :
[1, '@_python.py_', 38200, 210]
[2, '_python.py_', 38, 0]
```



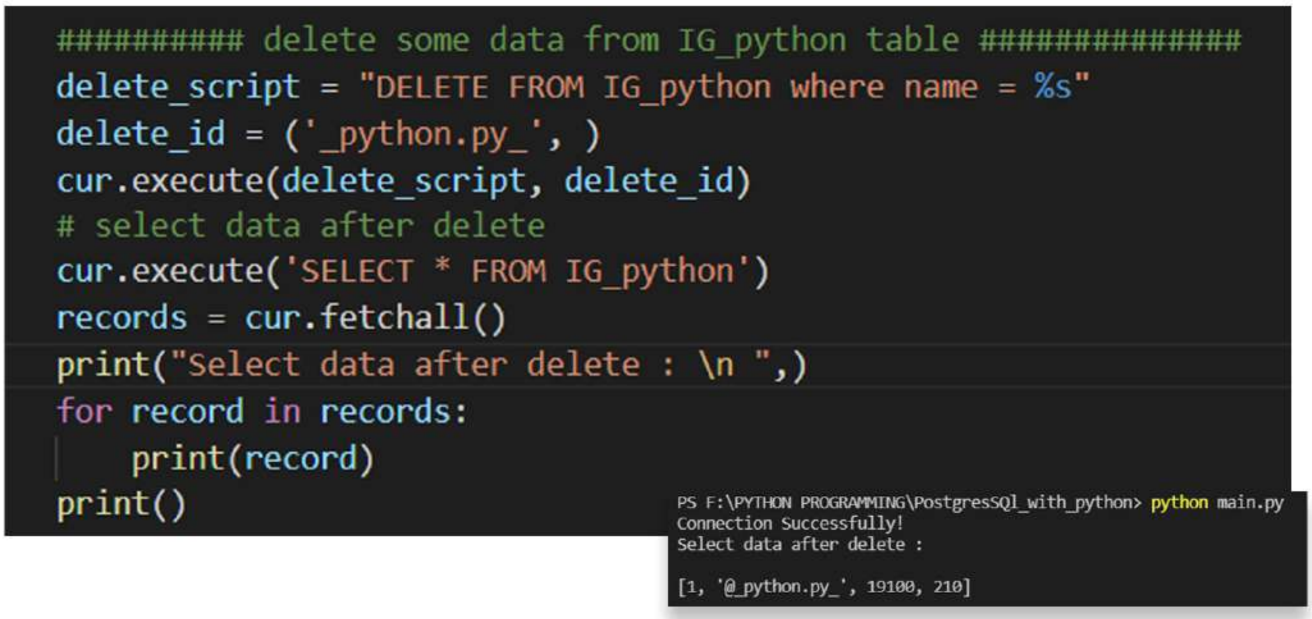
```
select * from IG_python;
```

	id [PK] integer	name character varying (100)	followers integer	post_num integer
1	1	@_python.py_	38200	210
2	2	_python.py_	38	0

@_python.py_
Python | Programming | Projects

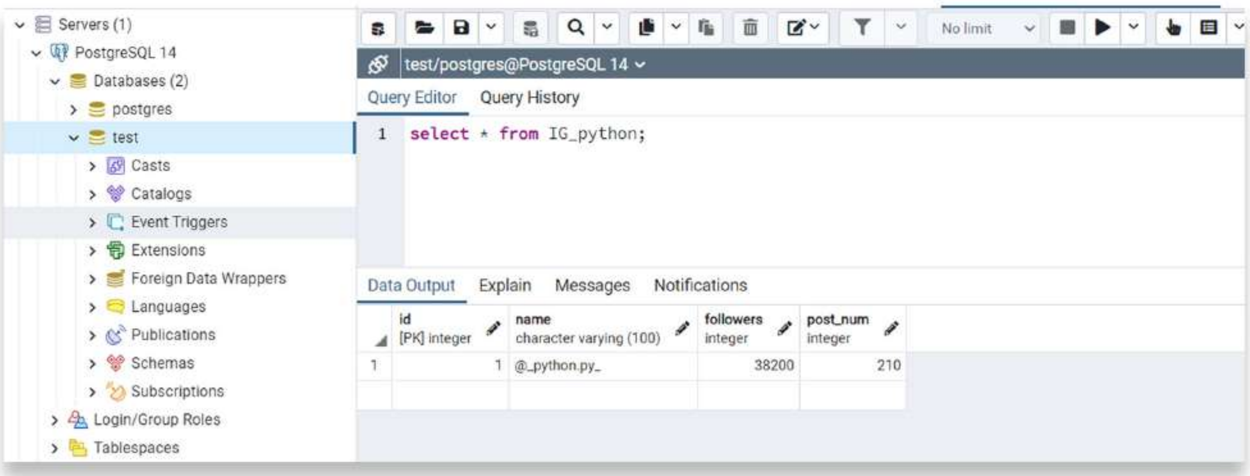
SAVE IT

6. Delete.



```python
########## delete some data from IG_python table ##############
delete_script = "DELETE FROM IG_python where name = %s"
delete_id = ('_python.py_', )
cur.execute(delete_script, delete_id)
# select data after delete
cur.execute('SELECT * FROM IG_python')
records = cur.fetchall()
print("Select data after delete : \n ",)
for record in records:
    print(record)
print()
```

```
PS F:\PYTHON PROGRAMMING\PostgresSql_with_python> python main.py
Connection Successfully!
Select data after delete :

[1, '@_python.py_', 19100, 210]
```

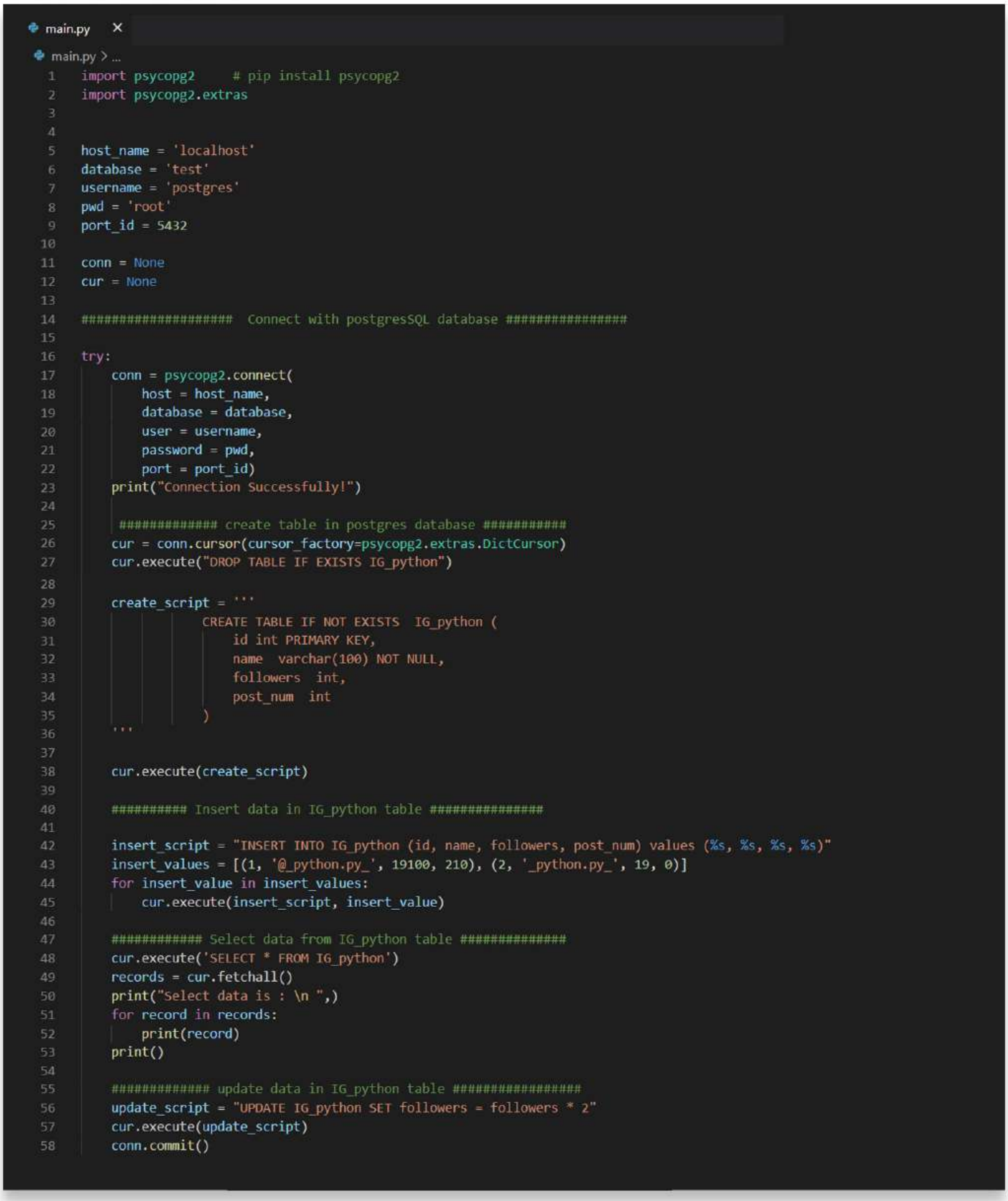


```sql
select * from IG_python;
```

	id [PK] integer	name character varying (100)	followers integer	post_num integer
1	1	@_python.py_	38200	210

08

Source Code

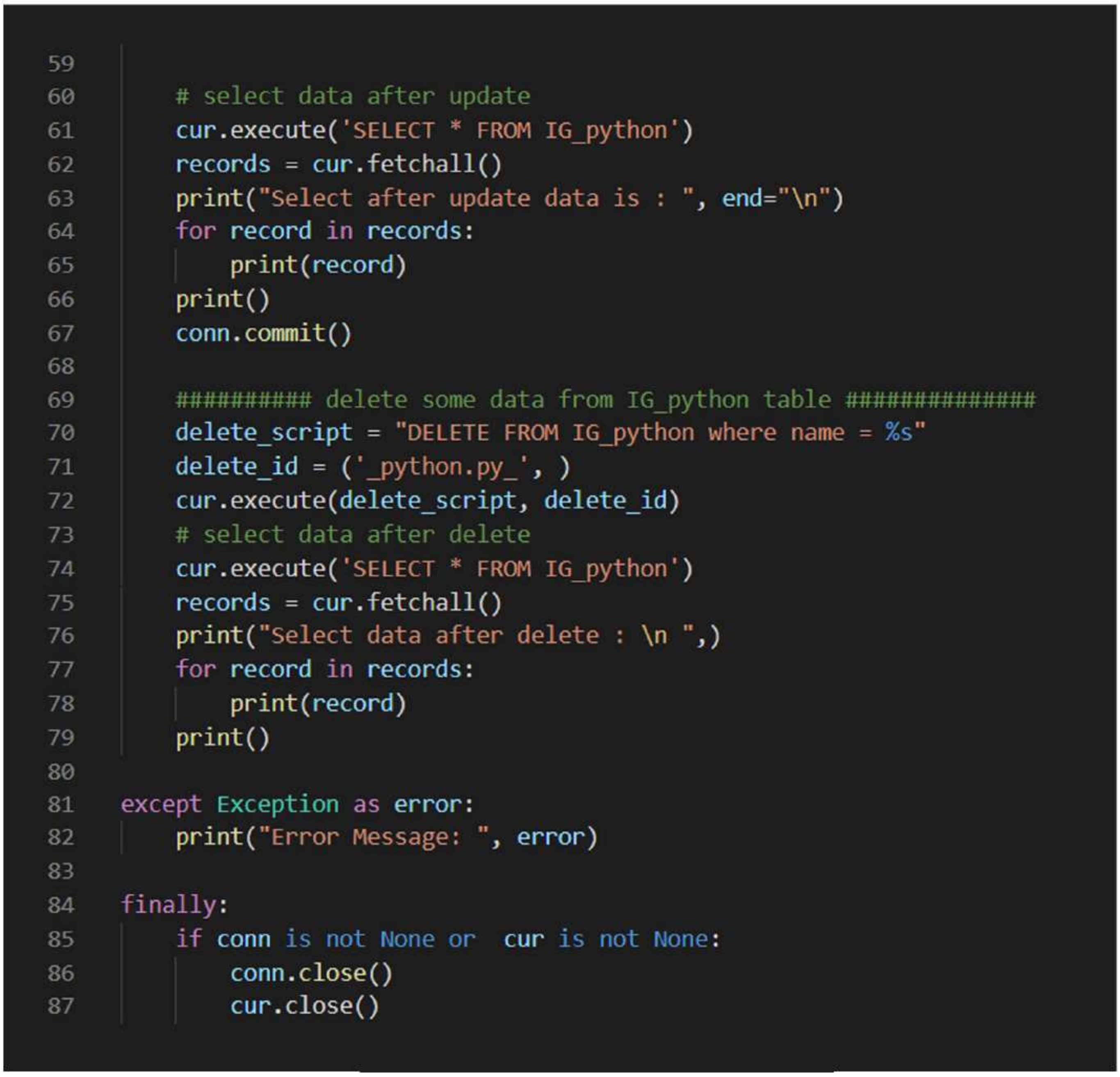


```python
import psycopg2     # pip install psycopg2
import psycopg2.extras


host_name = 'localhost'
database = 'test'
username = 'postgres'
pwd = 'root'
port_id = 5432

conn = None
cur = None

####################  Connect with postgresSQL database ################

try:
    conn = psycopg2.connect(
        host = host_name,
        database = database,
        user = username,
        password = pwd,
        port = port_id)
    print("Connection Successfully!")

     ############# create table in postgres database ###########
    cur = conn.cursor(cursor_factory=psycopg2.extras.DictCursor)
    cur.execute("DROP TABLE IF EXISTS IG_python")

    create_script = '''
                CREATE TABLE IF NOT EXISTS  IG_python (
                    id int PRIMARY KEY,
                    name  varchar(100) NOT NULL,
                    followers  int,
                    post_num  int
                )
    '''

    cur.execute(create_script)

    ########## Insert data in IG_python table ###############

    insert_script = "INSERT INTO IG_python (id, name, followers, post_num) values (%s, %s, %s, %s)"
    insert_values = [(1, '@_python.py_', 19100, 210), (2, '_python.py_', 19, 0)]
    for insert_value in insert_values:
        cur.execute(insert_script, insert_value)

    ############ Select data from IG_python table ##############
    cur.execute('SELECT * FROM IG_python')
    records = cur.fetchall()
    print("Select data is : \n ",)
    for record in records:
        print(record)
    print()

    ############# update data in IG_python table #################
    update_script = "UPDATE IG_python SET followers = followers * 2"
    cur.execute(update_script)
    conn.commit()
```

@_python.py_

Python | Programming | Projects

SAVE IT



```python

    # select data after update
    cur.execute('SELECT * FROM IG_python')
    records = cur.fetchall()
    print("Select after update data is : ", end="\n")
    for record in records:
        print(record)
    print()
    conn.commit()

    ########## delete some data from IG_python table ##############
    delete_script = "DELETE FROM IG_python where name = %s"
    delete_id = ('_python.py_', )
    cur.execute(delete_script, delete_id)
    # select data after delete
    cur.execute('SELECT * FROM IG_python')
    records = cur.fetchall()
    print("Select data after delete : \n ",)
    for record in records:
        print(record)
    print()

except Exception as error:
    print("Error Message: ", error)

finally:
    if conn is not None or  cur is not None:
        conn.close()
        cur.close()
```