

01



# PostgreSQL with Python

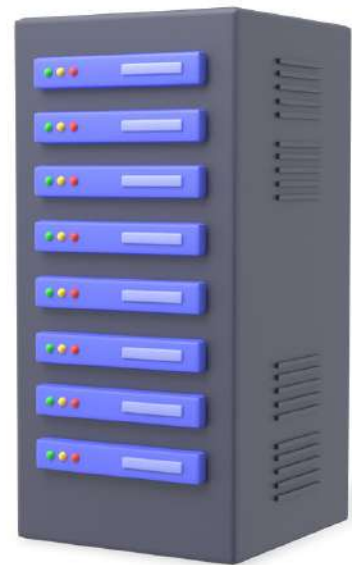
*CRUD opt with postgresql in python*



@\_python.py\_  
Python | Programming | Projects



**Arvind Singh**  
Python FullStack Developer

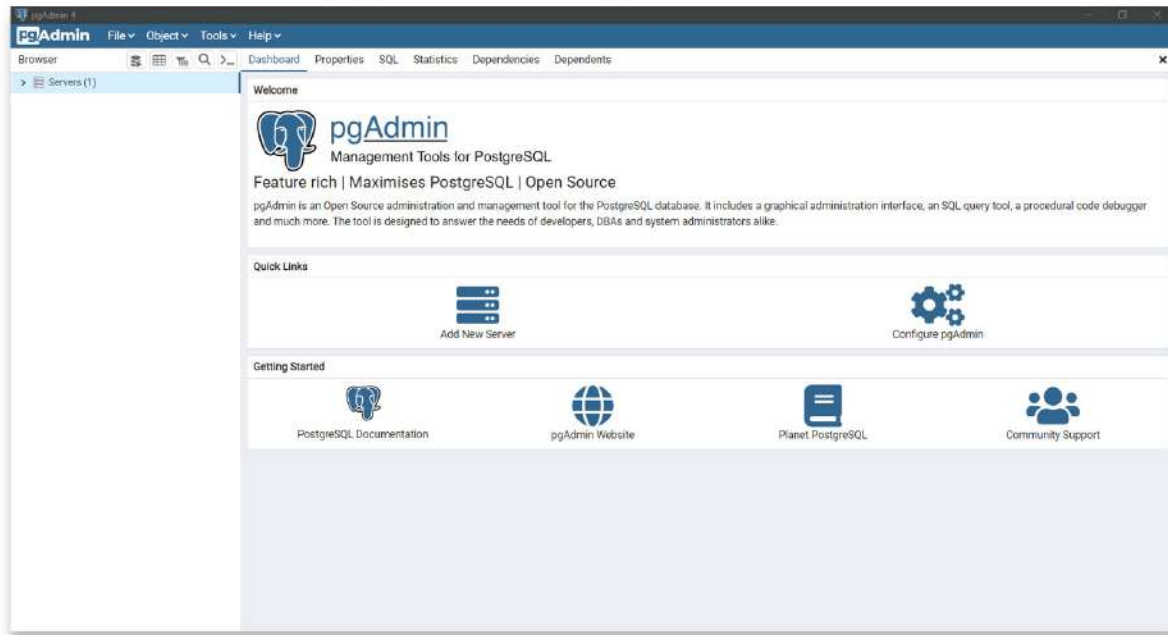


**SAVE IT** 

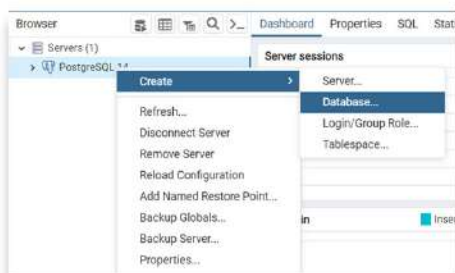
02

## Download pgAdmin4

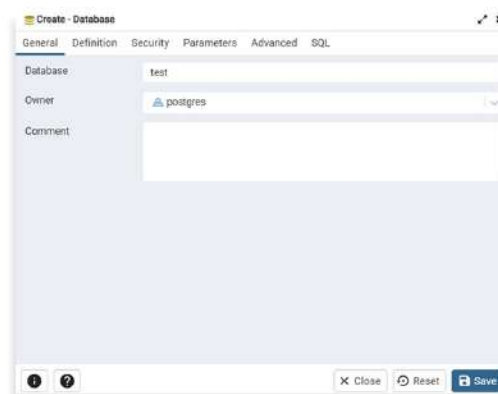
<https://www.pgadmin.org/download/>



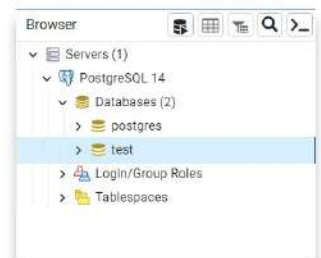
Create new database



I create 'test' name database



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.

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

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

conn = None

##### Connect with postgresQL database #####

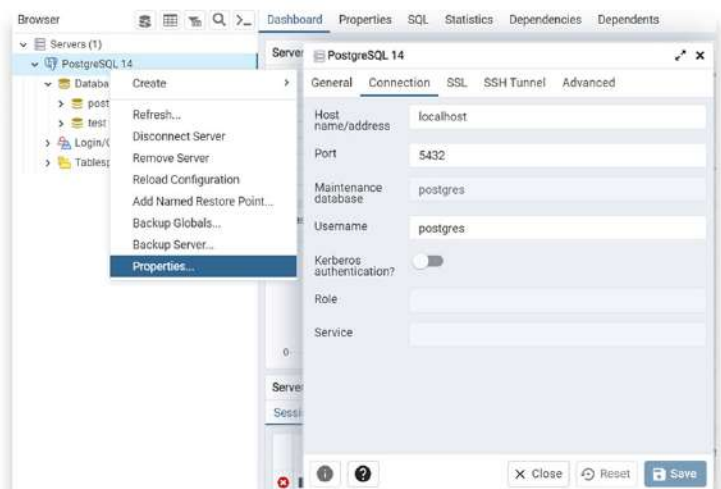
try:
    conn = psycopg2.connect(
        host = host_name,
        database = database,
        user = username,
        password = pwd,
        port = port_id)
    print("Connection Successfully!")

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

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

```
PS F:\PYTHON PROGRAMMING\PostgreSQL_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.



@\_python.py\_

Python | Programming | Projects

SAVE IT

# 04

## 3. Create table.

```
##### Connect with postgresQL 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

click here for [run any query](#)

table

The screenshot shows the pgAdmin interface. On the left, the 'Servers' tree is expanded to show the 'test' database. The 'Query Editor' is open, displaying a SQL query: 'select \* from IG\_python;'. The 'Data Output' tab is selected, showing the results of the query. The results are displayed in a table with four columns: 'id' (integer, primary key), 'name' (character varying (100)), 'followers' (integer), and 'post\_num' (integer). The table is currently empty.

| id | name | followers | post_num |
|----|------|-----------|----------|
|----|------|-----------|----------|



## 05

### 3. Insert data.

```
##### 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.

```
##### 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\PostgreSQL\_with\_python> python main.py  
Connection Successfully!  
Select data is :  
[1, '@\_python.py\_', 19100, 210]  
[2, '\_python.py\_', 19, 0]

The screenshot shows a PostgreSQL database browser interface. On the left, the 'test' database is selected. The 'IG\_python' table is highlighted. The table structure is shown as follows:

| id | name         | followers | post_num |
|----|--------------|-----------|----------|
| 1  | @_python.py_ | 19100     | 210      |
| 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\PostgreSQL_with_python> python main.py
Connection Successfully!
Select after update data is :
[1, '@_python.py_', 38200, 210]
[2, '_python.py_', 38, 0]
```

The screenshot shows a PostgreSQL database management tool interface. On the left, the 'Servers' tree is expanded to show the 'test' database. The 'Query Editor' is open, showing the query 'select \* from IG\_python;'. The 'Data Output' tab is selected, displaying the results of the query in a table format.

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



## 6. Delete.

```
##### 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\PostgreSQL_with_python> python main.py
Connection Successfully!
Select data after delete :
[1, '@_python.py_', 19100, 210]
```

The screenshot shows the PostgreSQL Enterprise Studio interface. On the left, the 'Servers' tree is expanded to show the 'test' database. The 'Query Editor' tab is active, displaying the query: `select * from IG_python;`. The 'Data Output' tab shows the results of the query in a table format.

| id | name         | followers | post_num |
|----|--------------|-----------|----------|
| 1  | @_python.py_ | 38200     | 210      |



# 08

## Source Code

```
main.py X
main.py > ...
1 import psycopg2 # pip install psycopg2
2 import psycopg2.extras
3
4
5 host_name = 'localhost'
6 database = 'test'
7 username = 'postgres'
8 pwd = 'root'
9 port_id = 5432
10
11 conn = None
12 cur = None
13
14 ##### Connect with postgresSQL database #####
15
16 try:
17     conn = psycopg2.connect(
18         host = host_name,
19         database = database,
20         user = username,
21         password = pwd,
22         port = port_id)
23     print("Connection Successfully!")
24
25     ##### create table in postgres database #####
26     cur = conn.cursor(cursor_factory=psycopg2.extras.DictCursor)
27     cur.execute("DROP TABLE IF EXISTS IG_python")
28
29     create_script = '''
30     CREATE TABLE IF NOT EXISTS IG_python (
31         id int PRIMARY KEY,
32         name varchar(100) NOT NULL,
33         followers int,
34         post_num int
35     )
36     '''
37
38     cur.execute(create_script)
39
40     ##### Insert data in IG_python table #####
41
42     insert_script = "INSERT INTO IG_python (id, name, followers, post_num) values (%s, %s, %s, %s)"
43     insert_values = [(1, '@_python.py_', 19100, 210), (2, '@_python.py_', 19, 0)]
44     for insert_value in insert_values:
45         cur.execute(insert_script, insert_value)
46
47     ##### Select data from IG_python table #####
48     cur.execute('SELECT * FROM IG_python')
49     records = cur.fetchall()
50     print("Select data is : \n ",)
51     for record in records:
52         print(record)
53     print()
54
55     ##### update data in IG_python table #####
56     update_script = "UPDATE IG_python SET followers = followers * 2"
57     cur.execute(update_script)
58     conn.commit()
```



@\_python.py\_

Python | Programming | Projects

SAVE IT



```

59
60     # select data after update
61     cur.execute('SELECT * FROM IG_python')
62     records = cur.fetchall()
63     print("Select after update data is : ", end="\n")
64     for record in records:
65         print(record)
66     print()
67     conn.commit()
68
69     ##### delete some data from IG_python table #####
70     delete_script = "DELETE FROM IG_python where name = %s"
71     delete_id = ('_python.py_', )
72     cur.execute(delete_script, delete_id)
73     # select data after delete
74     cur.execute('SELECT * FROM IG_python')
75     records = cur.fetchall()
76     print("Select data after delete : \n ",)
77     for record in records:
78         print(record)
79     print()
80
81 except Exception as error:
82     print("Error Message: ", error)
83
84 finally:
85     if conn is not None or cur is not None:
86         conn.close()
87         cur.close()

```

