POSTGRESQL - PYTHON INTERFACE

Installation

The PostgreSQL can be integrated with Python using psycopg2 module. sycopg2 is a PostgreSQL database adapter for the Python programming language. psycopg2 was written with the aim of being very small and fast, and stable as a rock. You do not need to install this module separately because its being shipped by default along with Python version 2.5.x onwards. If you do not have it installed on your machine then you can use yum command to install it as follows:

```
$yum install python-psycopg2
```

To use psycopg2 module, you must first create a Connection object that represents the database and then optionally you can create cursor object which will help you in executing all the SQL statements.

Python psycopg2 module APIs

Following are important psycopg2 module routines which can suffice your requirement to work with PostgreSQL database from your Python program. If you are looking for a more sophisticated application, then you can look into Python psycopg2 module's official documentation.

S.N.	API & Description
1	**psycopg2.connect(database="testdb", user="postgres", password="cohondob", host="127.0.0.1", port="5432")** This API opens a connection to the PostgreSQL database. If database is opened successfully, it returns a connection object.
2	**connection.cursor()** This routine creates a **cursor** which will be used throughout of your database programming with Python.
3	**cursor.execute(sql [, optional parameters])** This routine executes an SQL statement. The SQL statement may be parameterized (i.e., placeholders instead of SQL literals). The psycopg2 module supports placeholder using %s sign For example:cursor.execute("insert into people values (%s, %s)", (who, age))
4	**curosr.executemany(sql, seq_of_parameters)** This routine executes an SQL command against all parameter sequences or mappings found in the sequence sql.
5	**curosr.callproc(procname[, parameters])** This routine executes a stored database procedure with the given name. The sequence of parameters must contain one entry for each argument that the procedure expects.
6	**cursor.rowcount** This read-only attribute which returns the total number of database rows that have been modified, inserted, or deleted by the last last execute*().

7	**connection.commit()** This method commits the current transaction. If you don't call this method, anything you did since the last call to commit() is not visible from other database connections.
8	**connection.rollback()** This method rolls back any changes to the database since the last call to commit().
9	**connection.close()** This method closes the database connection. Note that this does not automatically call commit(). If you just close your database connection without calling commit() first, your changes will be lost!
10	**cursor.fetchone()** This method fetches the next row of a query result set, returning a single sequence, or None when no more data is available.
11	**cursor.fetchmany([size=cursor.arraysize])** This routine fetches the next set of rows of a query result, returning a list. An empty list is returned when no more rows are available. The method tries to fetch as many rows as indicated by the size parameter.
12	**cursor.fetchall()** This routine fetches all (remaining) rows of a query result, returning a list. An empty list is returned when no rows are available.

Connecting To Database

Following Python code shows how to connect to an existing database. If database does not exist, then it will be created and finally a database object will be returned.

```
#!/usr/bin/python

import psycopg2

conn = psycopg2.connect(database="testdb", user="postgres", password="pass123",
host="127.0.0.1", port="5432")

print "Opened database successfully"
```

Here, you can also supply database **testdb** as name and if database is successfully opened, then it will give following message:

```
Open database successfully
```

Create a Table

Following Python program will be used to create a table in previously created database:

```
#!/usr/bin/python

import psycopg2

conn = psycopg2.connect(database="testdb", user="postgres", password="pass123",
```

```
host="127.0.0.1", port="5432")
print "Opened database successfully"

cur = conn.cursor()
cur.execute('''CREATE TABLE COMPANY
       (ID INT PRIMARY KEY     NOT NULL,
       NAME           TEXT    NOT NULL,
       AGE            INT     NOT NULL,
       ADDRESS        CHAR(50),
       SALARY         REAL);''')
print "Table created successfully"

conn.commit()
conn.close()
```

When above program is executed, it will create COMPANY table in your **test.db** and it will display the following messages:

```
Opened database successfully
Table created successfully
```

INSERT Operation

Following Python program shows how we can create records in our COMPANY table created in above example:

```
#!/usr/bin/python

import psycopg2

conn = psycopg2.connect(database="testdb", user="postgres", password="pass123",
host="127.0.0.1", port="5432")
print "Opened database successfully"

cur = conn.cursor()

cur.execute("INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) \
      VALUES (1, 'Paul', 32, 'California', 20000.00 )");

cur.execute("INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) \
      VALUES (2, 'Allen', 25, 'Texas', 15000.00 )");

cur.execute("INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) \
      VALUES (3, 'Teddy', 23, 'Norway', 20000.00 )");

cur.execute("INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) \
      VALUES (4, 'Mark', 25, 'Rich-Mond ', 65000.00 )");

conn.commit()
print "Records created successfully";
conn.close()
```

When above program is executed, it will create given records in COMPANY table and will display the following two lines:

```
Opened database successfully
Records created successfully
```

SELECT Operation

Following Python program shows how we can fetch and display records from our COMPANY table created in above example:

```
#!/usr/bin/python

import psycopg2

conn = psycopg2.connect(database="testdb", user="postgres", password="pass123",
host="127.0.0.1", port="5432")
```

```
print "Opened database successfully"

cur = conn.cursor()

cur.execute("SELECT id, name, address, salary  from COMPANY")
rows = cur.fetchall()
for row in rows:
   print "ID = ", row[0]
   print "NAME = ", row[1]
   print "ADDRESS = ", row[2]
   print "SALARY = ", row[3], "\n"

print "Operation done successfully";
conn.close()
```

When above program is executed, it will produce the following result:

```
Opened database successfully
ID =  1
NAME =  Paul
ADDRESS =  California
SALARY =  20000.0

ID =  2
NAME =  Allen
ADDRESS =  Texas
SALARY =  15000.0

ID =  3
NAME =  Teddy
ADDRESS =  Norway
SALARY =  20000.0

ID =  4
NAME =  Mark
ADDRESS =  Rich-Mond
SALARY =  65000.0

Operation done successfully
```

UPDATE Operation

Following Python code shows how we can use UPDATE statement to update any record and then fetch and display updated records from our COMPANY table:

```
#!/usr/bin/python

import psycopg2

conn = psycopg2.connect(database="testdb", user="postgres", password="pass123",
host="127.0.0.1", port="5432")
print "Opened database successfully"

cur = conn.cursor()

cur.execute("UPDATE COMPANY set SALARY = 25000.00 where ID=1")
conn.commit
print "Total number of rows updated :", cur.rowcount

cur.execute("SELECT id, name, address, salary  from COMPANY")
rows = cur.fetchall()
for row in rows:
   print "ID = ", row[0]
   print "NAME = ", row[1]
   print "ADDRESS = ", row[2]
   print "SALARY = ", row[3], "\n"

print "Operation done successfully";
conn.close()
```

When above program is executed, it will produce the following result:

```
Opened database successfully
Total number of rows updated : 1
ID =  1
NAME =  Paul
ADDRESS =  California
SALARY =  25000.0

ID =  2
NAME =  Allen
ADDRESS =  Texas
SALARY =  15000.0

ID =  3
NAME =  Teddy
ADDRESS =  Norway
SALARY =  20000.0

ID =  4
NAME =  Mark
ADDRESS =  Rich-Mond
SALARY =  65000.0

Operation done successfully
```

DELETE Operation

Following Python code shows how we can use DELETE statement to delete any record and then fetch and display remaining records from our COMPANY table:

```
#!/usr/bin/python

import psycopg2

conn = psycopg2.connect(database="testdb", user="postgres", password="pass123",
host="127.0.0.1", port="5432")
print "Opened database successfully"

cur = conn.cursor()

cur.execute("DELETE from COMPANY where ID=2;")
conn.commit
print "Total number of rows deleted :", cur.rowcount

cur.execute("SELECT id, name, address, salary  from COMPANY")
rows = cur.fetchall()
for row in rows:
   print "ID = ", row[0]
   print "NAME = ", row[1]
   print "ADDRESS = ", row[2]
   print "SALARY = ", row[3], "\n"

print "Operation done successfully";
conn.close()
```

When above program is executed, it will produce the following result:

```
Opened database successfully
Total number of rows deleted : 1
ID =  1
NAME =  Paul
ADDRESS =  California
SALARY =  20000.0

ID =  3
NAME =  Teddy
ADDRESS =  Norway
SALARY =  20000.0
```

```
ID =  4
NAME =  Mark
ADDRESS =  Rich-Mond
SALARY =  65000.0

Operation done successfully
```