Database Programming

DigitalSkola
Uncover The World Of Digital Skills With Us

What will We Learn Today?

1. Reading file to python
2. Writing file to python
3. DB Programming in Python

File

- A file is a contiguous set of bytes used to store data.
- This data is organized in a specific format and can be anything as simple as a text file or as complicated as a program executable.
- In the end, these byte files are then translated into binary 1 and 0 for easier processing by the computer.
- After read from or write to a file, it needs to close so that the resources that are tied with the file are freed.
- Hence, in Python, a file operation takes place in the following order:
    1. Open a file
    2. Read or write (perform operation)
    3. Close the file

File Paths

When you access a file on an operating system, a file path is required. The file path is a string that represents the location of a file. It's broken up into two major parts:

- **Folder Path:** the file folder location on the file system where subsequent folders are separated by a forward slash / (Unix) or backslash \ (Windows)
- **File Name:** the actual name of the file

File Paths

Here's a quick example. Let's say you have a file located within a file structure like this:

```
/
|
├── path/
|   |
|   ├── to/
|   |   └── cats.gif
|   |
|   └── dog_breeds.txt
|
└── animals.csv
```

Example:

Let's say you wanted to access the cats.gif file, and your current location was in the same folder as path. In order to access the file, you need to go through the path folder and then the to folder, finally arriving at the cats.gif file. The Folder Path is path/to/. The File Name is cats. The File Extension is .gif. So the full path is path/to/cats.gif

Opening a File Stream

The first part of read/write a file is to create a stream to the file.

- A Stream is one way path for information to travel
- Code: varName = open(fileName, mode)

Mode	Description
r	Reading (default)
w	Writing (if file exists, content is wiped)
a	Append (if file exists, writes are appended)
r+	Reading and Writing
t	Text (default)
b	Binary

Example:
fileWrite = open("example.txt","w")
input = open("\databaseprogramming\example.txt","r")

Writing Files

To write to a file use the file stream name like a variable with the write function.

- Code: write(str)

Example:
output = open(“example.txt”)
output.write(“Welcome to Digital Skola!\n”)

Reading Files

Reading a file is similar to writing a file in that you use the file stream name followed by one of the read functions.

Code:

- read() - reads all characters from a file
- readline() - reads all characters up to and including \n
- readlines() - reads all lines

Example:

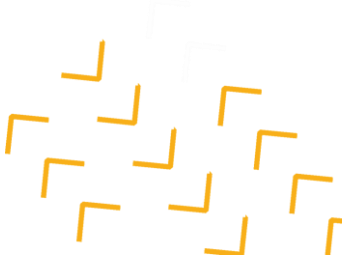
```
output = open("example.txt")
print(output.read())
```

Closing a File Stream

- Clean up is a necessary and essential step
- Just like when you make a mess, life will go on but there are implications of not cleaning up.
- Ensure you close a stream after you have processed all of your data.

Code:

```
output = open("example.txt")
print(output.read())
output.close()
```



Safer

- The best way to close a file is by using the with statement.
- This ensures that the file is closed when the block inside the with statement is exited.
- No need explicitly call the close()

Example:

```
with open('example.txt', 'r') as testwritefile:
    print(testwritefile.read())
```

DigitalSkola
Uncover The World Of Digital Skills With Us

Let's Practice!

DigitalSkola
Uncover The World Of Digital Skills With Us

DB Programming in Python

Introduction

- The python programming language has powerful features for database programming
- Python supports various databases like MySQL, Oracle, PostgreSQL, etc.
- Python also supports Data Definition Language (DDL), Data Manipulation Language (DML) and Data Query Statements.

Installing Psycopg2 using PIP

- To communicate with PostgreSQL using Python you need to install psycopg, an adapter provided for python programming, the current version of this is psycopg2.
- psycopg2 was written with the aim of being very small and fast, and stable as a rock. It is available under PIP (package manager of python)

```
#for install psycopg2
!pip install psycopg2
```

Database Connection

The connection class of the psycopg2 represents/handles an instance of a connection. You can create new connections using the connect() function. This accepts the basic connection parameters such as dbname, user, password, host, port and returns a connection object. Using this function, you can establish a connection with the PostgreSQL.

Database Connection

Database Connection

```
#for install psycopg2
#!pip install psycopg2

import psycopg2

#establishing the connection
conn = psycopg2.connect(host="digitalskoladb.c04me33o8tni.ap-southeast-1.rds.amazonaws.com", port = 5432, database="sandbox", user="group_1", password="12345")

#Creating a cursor object using the cursor() method
cursor = conn.cursor()

#Executing an SQL function using the execute() method
cursor.execute("select version()")

# Fetch a single row using fetchone() method.
data = cursor.fetchone()
print("Connection established to: ",data)

#Closing the connection
conn.close()
```

```
/usr/local/lib/python3.7/dist-packages/psycopg2/__init__.py:144: UserWarning: The psycopg2 wheel package will be renamed from release 2.8; in order to keep insta
  """)
Connection established to:  ('PostgreSQL 12.5 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.3 20140911 (Red Hat 4.8.3-9), 64-bit',)
```

Create Table

To create a table using python you need to execute the CREATE TABLE statement using the execute() method of the Cursor of pyscopg2. While executing this you need to specify the name of the table, column names and their data types.

DigitalSkola
Uncover The World Of Digital Skills With Us

Create Table

CREATE TABLE

```
#establishing the connection
conn = psycopg2.connect(host="digitalskoladb.c04me33o8tni.ap-southeast-1.rds.amazonaws.com", port = 5432, database="sandbox", user="group_1", password="12345")

#Creating a cursor object using the cursor() method
cursor = conn.cursor()

#Droping buses table if already exists.
cursor.execute("DROP TABLE IF EXISTS sandbox.batch_2.buses")

#Creating table as per requirement
sql ='''create table sandbox.batch_2.buses (
        id int,
        origin varchar,
        destination varchar,
        time time
)'''
cursor.execute(sql)
print("Table created successfully........")

# Commit your changes in the database
conn.commit()

#Closing the connection
conn.close()
```

```
Table created successfully........
```

Insert Data

To insert record into an existing table in PostgreSQL using the INSERT INTO statement. While executing this, you need to specify the name of the table, and values for the columns in it.

DigitalSkola
Uncover The World Of Digital Skills With Us

Insert Data

INSERT TABLE

```
#establishing the connection
conn = psycopg2.connect(host="digitalskoladb.c04me33o8tni.ap-southeast-1.rds.amazonaws.com", port = 5432, database="sandbox", user="group_1", password="12345")

#Creating a cursor object using the cursor() method
cursor = conn.cursor()

sql ='''insert into sandbox.batch_2.buses values
        (100, 'Munich', 'Rome', '13:00'),
        (200, 'Munich', 'Rome', '15:30'),
        (300, 'Munich', 'Rome', '20:00')'''

cursor.execute(sql)

# Commit your changes in the database
conn.commit()
print("Records inserted........")

#Closing the connection
conn.close()
```

```
Records inserted........
```

Select Data

READ Operation on any database means to fetch some useful information from the database. You can fetch data from PostgreSQL using the fetch() method provided by the psycopg2. The Cursor class provides two methods namely fetchall() and fetchone() where:

- The fetchall() method retrieves all the rows in the result set of a query and returns them as list of tuples. (If we execute this after retrieving few rows, it returns the remaining ones).
- The fetchone() method fetches the next row in the result of a query and returns it as a tuple.

DigitalSkola
Uncover The World Of Digital Skills With Us

Select Data

SELECT DATA

```
#establishing the connection
conn = psycopg2.connect(host="digitalskoladb.c04me33o8tni.ap-southeast-1.rds.amazonaws.com", port = 5432, database="sandbox", user="group_1", password="12345")

#Creating a cursor object using the cursor() method
cursor = conn.cursor()

sql ='''select * from sandbox.batch_2.buses'''

cursor.execute(sql)

#Fetching 1st row from the table
result = cursor.fetchone();
print(result)

#Fetching the next row from the table/ retrieves all the row in the result se of a query
result = cursor.fetchall();
print(result)

#Commit your changes in the database
conn.commit()

#Closing the connection
conn.close()
```

```
(100, 'Munich', 'Rome', datetime.time(13, 0))
[(200, 'Munich', 'Rome', datetime.time(15, 30)), (300, 'Munich', 'Rome', datetime.time(20, 0))]
```

Update Table

To modify the contents of existing records of a table in PostgreSQL you can using the UPDATE statement. To update specific rows, you need to use the WHERE clause along with it.

Update Table

UPDATE TABLE

```
#establishing the connection
conn = psycopg2.connect(host="digitalskoladb.c04me33o8tni.ap-southeast-1.rds.amazonaws.com", port = 5432, database="sandbox", user="group_1", password="12345")

#Creating a cursor object using the cursor() method
cursor = conn.cursor()

#Fetching all the rows before the update
print("Contents of the Employee table: ")
cursor.execute('''select * from sandbox.batch_2.buses''')
print(cursor.fetchall())

sql ='''update sandbox.batch_2.buses
        set id = 100, origin = 'Munich', destination = 'Roma', time = '14:00'
        where id = 100'''

cursor.execute(sql)
print("Table updated...... ")

#Fetching all the rows after the update
print("Contents of the Employee table after the update operation: ")
cursor.execute('''select * from sandbox.batch_2.buses''')
print(cursor.fetchall())

#Commit your changes in the database
conn.commit()

#Closing the connection
conn.close()
```

```
Contents of the Employee table:
[(100, 'Munich', 'Rome', datetime.time(13, 0)), (200, 'Munich', 'Rome', datetime.time(15, 30)), (300, 'Munich', 'Rome', datetime.time(20, 0))]
Table updated......
Contents of the Employee table after the update operation:
[(200, 'Munich', 'Rome', datetime.time(15, 30)), (300, 'Munich', 'Rome', datetime.time(20, 0)), (100, 'Munich', 'Roma', datetime.time(14, 0))]
```

Delete Data

To delete the records in an existing table using the DELETE FROM statement of PostgreSQL database. To remove specific records, you need to use WHERE clause along with it.

Delete Data

Delete Data

```
[6]  #establishing the connection
     conn = psycopg2.connect(host="digitalskoladb.c04me33o8tni.ap-southeast-1.rds.amazonaws.com", port = 5432, database="sandbox", user="group_1", password="12345")

     #Creating a cursor object using the cursor() method
     cursor = conn.cursor()

     #Retrieving contents of the table
     print("Contents of the table: ")
     cursor.execute('''select * from sandbox.batch_2.buses''')
     print(cursor.fetchall())

     #Deleting records
     cursor.execute('''delete from sandbox.batch_2.buses where id = 100''')

     #Retrieving data after delete
     print("Contents of the table after delete operation ")
     cursor.execute("select * from sandbox.batch_2.buses")
     print(cursor.fetchall())

     #Commit your changes in the database
     conn.commit()

     #Closing the connection
     conn.close()
```

```
Contents of the table:
[(200, 'Munich', 'Rome', datetime.time(15, 30)), (300, 'Munich', 'Rome', datetime.time(20, 0)), (100, 'Munich', 'Roma', datetime.time(14, 0))]
Contents of the table after delete operation
[(200, 'Munich', 'Rome', datetime.time(15, 30)), (300, 'Munich', 'Rome', datetime.time(20, 0))]
```

Drop Table

To drop a table from PostgreSQL database using the DROP TABLE statement. But you need to be very careful while deleting any existing table because the data lost will not be recovered after deleting a table.

Drop Table

Drop Table

```
[7] #establishing the connection
    conn = psycopg2.connect(host="digitalskoladb.c04me33o8tni.ap-southeast-1.rds.amazonaws.com", port = 5432, database="sandbox", user="group_1", password="12345")

    #Creating a cursor object using the cursor() method
    cursor = conn.cursor()

    #Droping buses table if already exists
    cursor.execute("drop table sandbox.batch_2.buses")
    print("Table dropped... ")

    #Commit your changes in the database
    conn.commit()

    #Closing the connection
    conn.close()
```

```
Table dropped...
```

DigitalSkola
Uncover The World Of Digital Skills With Us

Let's Practice!

DigitalSkola

Uncover The World Of Digital Skills With Us

Thank YOU