Blobs, python, and MySQL

We know we can read/write text and binary files ... how about interacting with blobs (binary large objects) and MySQL? Pretty common technique for some media files. [To be sure you could use Java Media library or Python's media libraries, too, to write movies, pdfs, to/from python alone or with sql.]

You may want to use python to save media files ... convert to base64:

```
import base64

video_stream = "hello"

with open('file.webm', 'wb') as f_vid:
    f_vid.write(base64.b64encode(video_stream))

with open('file.webm', 'rb') as f_vid:
    video_stream = base64.b64decode(f_vid.read())

print video_stream
```

Python & MySQL

Note that the below hardcodes the connection data - better to use a config file to store and retrieve your own settings.

```
import mysql.connector
from mysql.connector import Error
from mysql.connector import errorcode

def convertToBinaryData(filename):
    #Convert digital data to binary format
    with open(filename, 'rb') as file:
        binaryData = file.read()
    return binaryData

def insertBLOB(emp_id, name, photo, biodataFile):
    print("Inserting BLOB into python_employee table")

    try:
        connection = mysql.connector.connect(host='localhost',
                             database='python_db',
                             user='Bears',
                             password='goBears')

        cursor = connection.cursor(prepared=True)

        sql_insert_blob_query = """ INSERT INTO `python_employee`
                          (`id`, `name`, `photo`, `biodata`) VALUES (%s,%s,%s,%s)"""

        empPicture = convertToBinaryData(photo)
        file = convertToBinaryData(biodataFile)

        # Convert data into tuple format
        insert_blob_tuple = (emp_id, name, empPicture, file)

        result  = cursor.execute(sql_insert_blob_query, insert_blob_tuple)
        connection.commit()
        print ("Image and file inserted successfully as a BLOB", result)

    except mysql.connector.Error as error :
        connection.rollback()
```

```
        print("Failed inserting BLOB data into MySQL table {}".format(error))

    finally:
        #closing database connection.
        if(connection.is_connected()):
            cursor.close()
            connection.close()
            print("MySQL connection is closed")
```

E.g.,
`insertBLOB(1, "Eric", "D:\images\richard_photo.png", "D:\images\richard_bioData.txt")`

Python & MySQL

iPads and mobile devices use SQLite:

```
import sqlite3
conn = sqlite3.connect('database.db')
cursor = conn.cursor()

with open("...", "rb") as input_file:
    ablob = input_file.read()
    cursor.execute("INSERT INTO notes (id, file) VALUES(0, ?)", [sqlite3.Binary(ablob)])
    conn.commit()

with open("Output.bin", "wb") as output_file:
    cursor.execute("SELECT file FROM notes WHERE id = 0")
    ablob = cursor.fetchone()
    output_file.write(ablob[0])

cursor.close()
conn.close()
```

Google Cloud (never used this)

https://google-cloud-python.readthedocs.io/en/0.32.0/storage/blobs.html

filename: BlobsAndPython.rtf