

Assignment -2

Connect with Database Assignment

| | |
|---------------------|-----------------|
| Assignment Date | 09 October 2022 |
| Student Name | Mohan P |
| Student Roll Number | 621319104031 |
| Maximum Marks | 2 Marks |

Question-1:

1. Create User table with user with email , username, roll number, password.

The screenshot shows the IBM Db2 on Cloud web interface. The left sidebar contains navigation icons for Data objects, My script, and a search bar. The main area displays a script editor with the following SQL command:

```
1 CREATE TABLE USER ("USEREMAIL" VARCHAR(35), "USERNAME" VARCHAR(35), "ROLLNUMBER" INT, "PASSWORD" VARCHAR(100))
```

The interface includes a 'Run all' button and a 'Syntax assistant' toggle. The bottom status bar shows the Windows taskbar with the date 10/18/2022 and time 5:08 AM.

2. Perform UPDATE, DELETE Queries with User table

The screenshot shows the IBM Db2 on Cloud web interface with the following SQL commands in the script editor:

```
1 INSERT INTO USER VALUES('abc@gmail.com','RAM','6211','Abc@ibm');
2 INSERT INTO USER VALUES('def@gmail.com','sita','6212','Def@ibm');
```

The 'History' tab is active, displaying the execution results of the queries:

| Script | Date | Status | Runtime |
|--|--------------------------|--------|---------|
| Untitled - 1 | Oct 14, 2022 10:14:42 AM | 2 | 0.018 s |
| INSERT INTO USER VALUES('abc@gmail.com','RAM','6211','Abc@ibm') | | ✓ | 0.010 s |
| INSERT INTO USER VALUES('def@gmail.com','sita','6212','Def@ibm') | | ✓ | 0.008 s |

Table View:

| USERNAME | ROLLNUMBER | PASSWORD |
|---------------|------------|----------|
| adn@gmail.com | 6211 | Adn@dem |
| adn@gmail.com | 6212 | Adn@dem |
| adn@gmail.com | 6213 | Adn@dem |

UPDATE:

| Script | Date | Status | Runtime |
|---|------------------------|---------|---------|
| UPDATE USER SET USERNAME='ISACK' WHERE USERNAME='adn@gmail.com' | 06/16/2022 10:22:22 AM | Success | 0.007 s |

Table View:

| USERNAME | ROLLNUMBER | PASSWORD |
|---------------|------------|----------|
| adn@gmail.com | 6211 | Adn@dem |
| adn@gmail.com | 6212 | Adn@dem |
| adn@gmail.com | 6213 | Adn@dem |

DELETE:

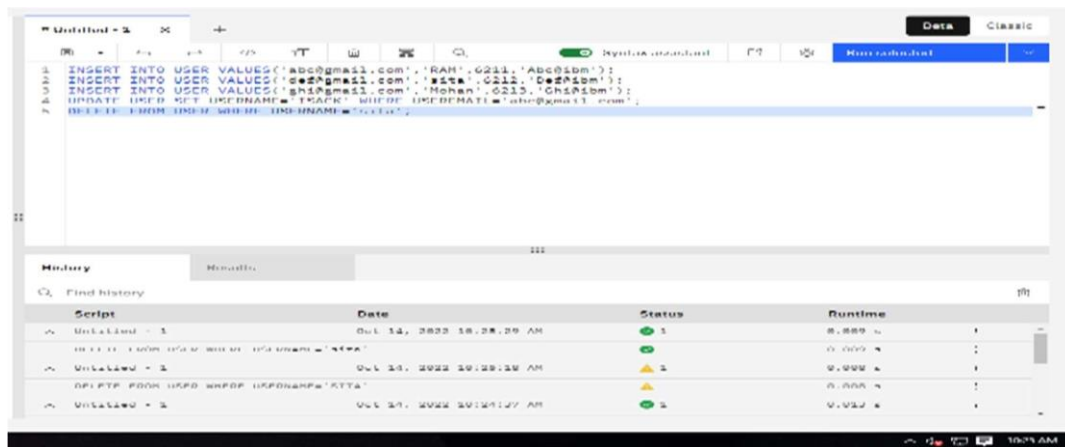
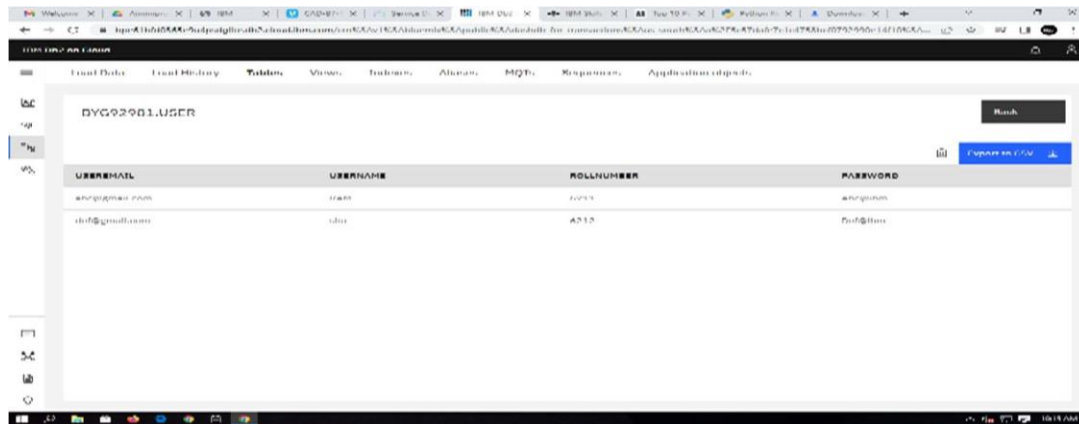


TABLE View:



3. Connect python with db2.

Solution:

```

import ibm_db conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=6667d8e9-9d4d-4ccb-ba32-21d
a3bb5aafe.clogj3sd0tgu01qde00.databases.appdomain.cloud;PORT=30376;SECURITY=SSL;SS
LS erverCertificate=DigiCertGlobalRootCA.crt;PROTOCOL=TCPIP;UID= rnf06984
;PWD=' VWqiPBgxELVtAn32
','')

```

4. Create a flask app with the registration page. Login page and the welcome page. By default load the registration page once the user enters all the fields, store the data in database and navigate to login page. Authenticate user username and password. If the user is valid so the welcome page.

Solution:

```
app.py from flask import Flask, render_template, request, redirect, url_for,
session

import ibm_db
import bcrypt conn
=
ibm_db.connect("DATABASE=bludb;HOSTNAME=;PORT=;SECURITY=SSL;SSLServerCertific
ate=DigiCer tGlobalRootCA.crt;UID=;PWD=\",\"") # url_for('static', filename='style.css')

app = Flask(_name_)
app.secret_key = 'C21FGSBAPOK43K5VSIDFB2'

@app.route("/",methods=['GET'])
def home():
    if 'email' not in session:
        return redirect(url_for('login'))
    return render_template("home.html",name='Home')

@app.route("/register",methods=['GET','POST'])
def register(): if request.method ==
'POST': email = request.form['email']
username = request.form['username']
rollNo = request.form['rollNo']
password = request.form['password']

if not email or not username or not rollNo or not password:
    return render_template('register.html',error='Please fill all fields')

hash=bcrypt.hashpw(password.encode('utf-8'),bcrypt.gensalt())

query = "SELECT * FROM USER WHERE email=? OR
rollNo=?" stmt = ibm_db.prepare(conn, query)
ibm_db.bind_param(stmt,1,email)
ibm_db.bind_param(stmt,2,rollNo) ibm_db.execute(stmt)
isUser = ibm_db.fetch_assoc(stmt)

if not isUser:
    insert_sql = "INSERT INTO User(username,email,PASSWORD,rollNo) VALUES
(?,?,?,?)"      prep_stmt      =      ibm_db.prepare(conn,      insert_sql)
ibm_db.bind_param(prepare_stmt, 1, username) ibm_db.bind_param(prepare_stmt, 2,
email) ibm_db.bind_param(prepare_stmt, 3, hash) ibm_db.bind_param(prepare_stmt, 4,
rollNo) ibm_db.execute(prepare_stmt)
    return render_template('register.html',success="You can login")
```

```

else:
    return render_template('register.html',error='Invalid
Credentials') return render_template('register.html',name='Home')

@app.route("/login",methods=['GET','POST'])
def login():
    if request.method == 'POST':
        email = request.form['email']
        password = request.form['password']

        if not email or not password:
            return render_template('login.html',error='Please fill all fields')
        query = "SELECT * FROM USER WHERE
email=?" stmt = ibm_db.prepare(conn, query)
        ibm_db.bind_param(stmt,1,email)
        ibm_db.execute(stmt) isUser =
        ibm_db.fetch_assoc(stmt)
        print(isUser,password)
        if not isUser:
            return render_template('login.html',error='Invalid Credentials')

        isPasswordMatch = bcrypt.checkpw(password.encode('utf-
8'),isUser['PASSWORD'].encode('utf-
8'))

        if not isPasswordMatch:
            return render_template('login.html',error='Invalid Credentials')

        session['email'] = isUser['EMAIL']
        return redirect(url_for('home'))

    return render_template('login.html',name='Home')
@app.route('/logout') def
logout():
    session.pop('email', None)
    return redirect(url_for('login'))

```

OUTPUT:



Register

{{success}}

{{error}}

[Already have an account? Login](#)

Activate Windows
Go to Settings to activate Windows.



Register

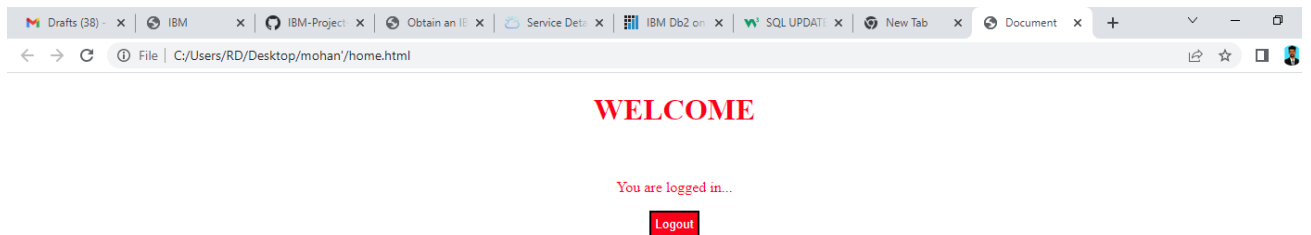
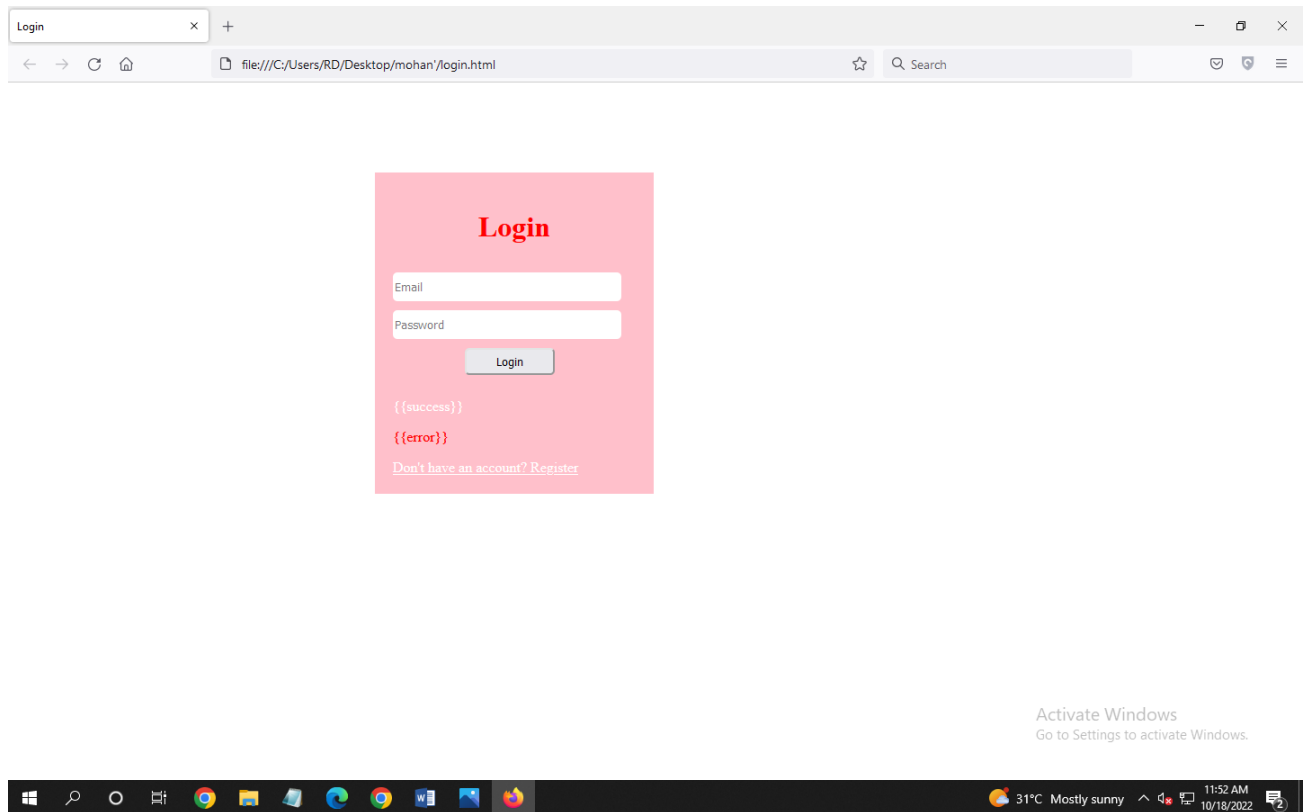
{{success}}

{{error}}

[Already have an account? Login](#)

Activate Windows
Go to Settings to activate Windows.





Database:

