

## Assignment -2

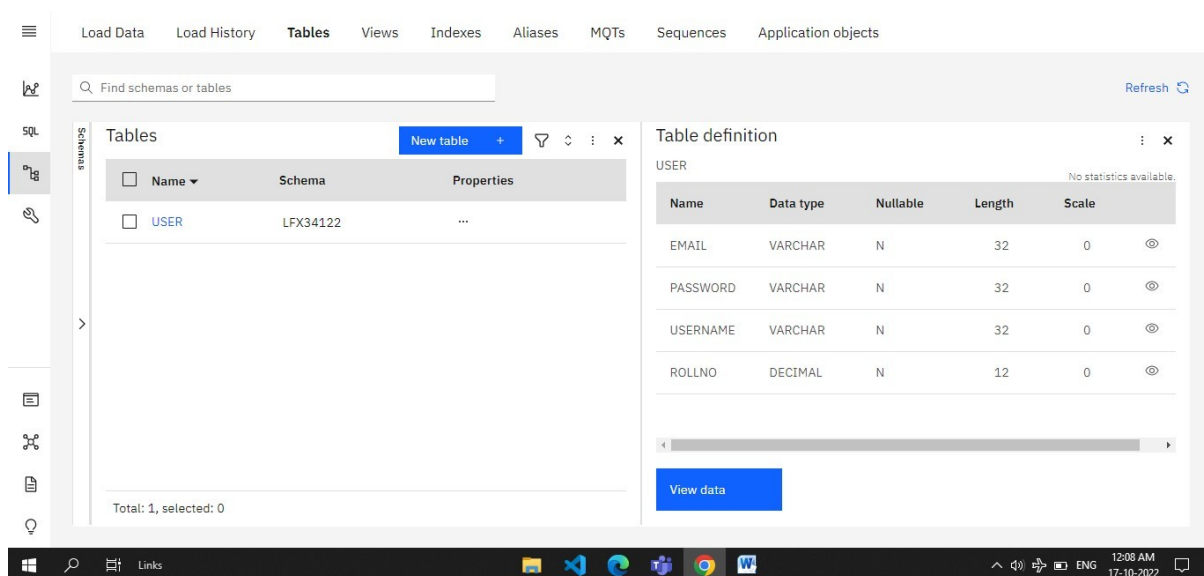
### Python Programming

Assignment Date	23 September 2022
Student Name	Santhosh Kumar S
Student Roll Number	2019115089
Maximum Marks	2 Marks

### Question 1:

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

### Solution 1:



### Question 2:

Perform UPDATE, DELETE Queries with user table

### Solution 2:

#### INSERT

INSERT INTO USER

VALUES('Vishnutheep','vishnutheep@gmail.com','vishnu',2019115123)

;

INSERT INTO USER

VALUES('Siva','sivakumar@gmail.com','siva',2019115089);

INSERT INTO USER

VALUES('Akshay','akshay@gmail.com','akshay',2019115013);

INSERT INTO USER

VALUES('Santhosh','santhosh@gmail.com','santhosh',2019115089);

The screenshot shows the IBM Db2 on Cloud SQL editor interface. The main window displays a script with four INSERT statements. Below the script, a 'History' table shows the execution details of these statements.

Script	Date	Status	Runtime
Untitled - 1	Oct 31, 2022 4:37:59 PM	5	0.039 s
INSERT INTO USER VALUES('Vishnutheep','vishnutheep@gmail.com','vishnu',2019115123);		✓	0.007 s
INSERT INTO USER VALUES('Siva','sivakumar@gmail.com','siva',2019115089);		✓	0.014 s
INSERT INTO USER VALUES('Akshay','akshay@gmail.com','akshay',2019115013);		✓	0.006 s
INSERT INTO USER VALUES('Santhosh','santhosh@gmail.com','santhosh',2019115089);		✓	0.006 s

The screenshot shows the IBM Db2 on Cloud SQL editor interface displaying the 'USER' table. The table has four columns: EMAIL, PASSWORD, USERNAME, and ROLLNO. The data is as follows:

EMAIL	PASSWORD	USERNAME	ROLLNO
akshay@gmail.com	12355	akshay	2019115013
santhosh@gmail.com	22345	santhosh	2019115089
siva@gmail.com	12345	siva	2019115098
vishnu@gmail.com	12344	vishnu	2019115123

UPDATE

update user set email='vishnutheep@gmail.com' where rollno=2019115124

## Output:

The screenshot shows a web application interface for a database. At the top, there is a navigation bar with links: Load Data, Load History, **Tables**, Views, Indexes, Aliases, MQTs, Sequences, and Application objects. On the left, there is a sidebar with icons for a menu, a chart, SQL, a table, a link, a document, and a lightbulb. The main content area displays a table titled 'QNV09796.USER'. Above the table, there is a 'Back' button and an 'Export to CSV' button with a download icon. The table has four columns: EMAIL, PASSWORD, USERNAME, and ROLLNO. It contains five rows of data.

EMAIL	PASSWORD	USERNAME	ROLLNO
akshay@gmail.com	12355	akshay	2019115013
santhosh@gmail.com	22345	santhosh	2019115089
siva@gmail.com	12345	siva	2019115098
vishnuthEEP@gmail.com	12344	vishnu	2019115123

## DELETE

delete from user where rollno = 2019115123

## OUTPUT:

The screenshot shows the same web application interface as before, but the table 'QNV09796.USER' now only contains three rows of data, as the row with rollno 2019115123 has been deleted.

EMAIL	PASSWORD	USERNAME	ROLLNO
akshay@gmail.com	12355	akshay	2019115013
santhosh@gmail.com	22345	santhosh	2019115089
siva@gmail.com	12345	siva	2019115098

## Question 3:

Connect python code to db2.

## **Solution 3:**

```
from flask import Flask, render_template, request, redirect,
url_for, session
import ibm_db
app = Flask(__name__)
```

```

app.secret_key = 'a'

conn = ibm_db.connect("DATABASE=bludb;
HOSTNAME=fbd88901-ebdb-4a4f-a32e-
9822b9fb237b.clogj3sd0tgtu0lqde00.databases.appdomain.
cloud; PORT=32731; SECURITY=SSL;
SSLServerCertificate=DigiCertGlobalRootCA.crt;
UID=rfh77431;PWD=1WClhcJWgdCoeAk5","")

if(conn):

    print("CONNECTED SUCCESSFULLY")

    print("Connection : "+str(conn))

    sql="SELECT * FROM USER WHERE rollno=2019115123"

    stmt = ibm_db.prepare(conn,sql)

    ibm_db.execute(stmt)

    acc = ibm_db.fetch_assoc(stmt)

    if acc:

        print(acc)

if __name__ == '__main__':

    app.run()

```

```

1  from flask import Flask, render_template, request, redirect, url_for, session
2  import ibm_db
3  app = Flask(__name__)
4  app.secret_key = 'a'
5  conn = ibm_db.connect("DATABASE=bludb; HOSTNAME=fbd88901-ebdb-4a4f-a32e-9822b9fb237b.clogj3sd0tgtu0lqde00.databases.appdomain.
6  cloud; PORT=32731; SECURITY=SSL;
7  SSLServerCertificate=DigiCertGlobalRootCA.crt;
8  UID=rfh77431;PWD=1WClhcJWgdCoeAk5","")
9  if(conn):
10     print("CONNECTED SUCCESSFULLY")
11     print("Connection : "+str(conn))
12     sql="SELECT * FROM USER WHERE rollno=2019115123"
13     stmt = ibm_db.prepare(conn,sql)
14     ibm_db.execute(stmt)
15     acc = ibm_db.fetch_assoc(stmt)
16     if acc:
17         print(acc)
18
19 if __name__ == '__main__':
20     app.run()

```

## OUTPUT:

```
vichu@pop-os:~/Education/Sem-7/IBM-docs/Assignment/Assignment2$ python3 temp.py
CONNECTED SUCCESSFULLY
Connection : <ibm_db.IBM_DBConnection object at 0x7fe225b4d390>
{'EMAIL': 'vishnutheep@gmail.com', 'PASSWORD': 'vishnu', 'USERNAME': 'Vishnutheep', 'ROLLNO': '2019115123'}
* Serving Flask app 'temp' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

## QUESTION - 4:

Create a Flask App with registration page, login page and welcome page. If the user is valid show the welcome page.

## SOLUTION:

```
from flask import Flask, render_template, request, redirect,
url_for, session
import ibm_db
import re

app = Flask(__name__)
app.secret_key = 'a'

conn = ibm_db.connect("DATABASE=bludb;
HOSTNAME=fbd88901-ebdb-4a4f-a32e-
9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.
cloud; PORT=32731; SECURITY=SSL;
SSLServerCertificate=DigiCertGlobalRootCA.crt;
UID=rfh77431;PWD=1WClhcJWgdCoeAk5","")

@app.route('/', methods = ['GET', 'POST'])
def login():
    global userid
    msg = ""
    if request.method == 'POST':
```

```

email = request.form['email']

password = request.form['password']

sql = "SELECT * FROM USERS WHERE EMAIL=? AND
PASSWORD=?"

stmt = ibm_db.prepare(conn,sql)
ibm_db.bind_param(stmt,1,email)
ibm_db.bind_param(stmt,2,password)
ibm_db.execute(stmt)
acc = ibm_db.fetch_assoc(stmt)
print(acc)
if acc:
    session['loggedin'] = True
    session['id'] = acc['USERNAME']
    userid = acc['USERNAME']
    session['username'] = acc['USERNAME']
    msg = acc['USERNAME']

    return render_template('dashboard.html', msg =
msg)
else:
    msg = "Incorrect username/password!!"
    return render_template('login.html', msg = msg)

@app.route('/register',methods =['GET', 'POST'])
def register():
    msg = ""
    if request.method == 'POST':
        username = request.form['username']
        email = request.form['email']
        password = request.form['password']
        sql = "SELECT * FROM USERS WHERE USERNAME=?"

```

```

stmt = ibm_db.prepare(conn,sql)
ibm_db.bind_param(stmt,1,username)
ibm_db.execute(stmt)
acc = ibm_db.fetch_assoc(stmt)
print(acc)
if acc:
    msg = "Account already exists !!"
elif not re.match(r'^@]+@[^@]+\.[^@]+',email):
    msg = "Invalid Email address"
elif not re.match(r'[A-Za-z0-9]+',username):
    msg = "Name must contain only characters and
numbers !!"
else:
    sql = "INSERT INTO USERS VALUES (?,?,?)"
    stmt = ibm_db.prepare(conn,sql)
    ibm_db.bind_param(stmt,1,username)
    ibm_db.bind_param(stmt,2,email)
    ibm_db.bind_param(stmt,3,password)
    ibm_db.execute(stmt)
    msg = "Successgully registered !!Login to continue"
    return render_template('login.html', msg = msg)
elif request.method == 'POST':
    msg = "Please fill out the form !"
    return render_template('register.html', msg = msg)
if __name__ == '__main__':
    app.run()

```

## OUTPUT:

REGISTRATION FORM

Username

Vishnu

Email

vishnutheep@gmail.com

Password

\*\*\*\*\*

SUBMIT

Already have an account? [Login here](#)

LOGIN FORM

Successfully registered !![Login to continue](#)

Username

vishnutheep@gmail.com

Password

\*\*\*\*\*

SUBMIT

Don't have an account? [Sign Up here](#)

Welcome Vishnu