

Assignment - 3

Assignment Date	29 September 2022
Student Name	Emmanuel. A
Student Roll No	812419104017
Maximum Marks	2 Marks

Question-1:

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

Solution:

Excel comma file

Sales Team Review							
Salesperson	Region	Current	Previous 2017 Sales	Total Sales	January 2018 Sales	Percent Change	
Jeffrey Burke	California	\$	28,000	\$	2,100	\$	45,148 17%
Amy Perreault	North Carolina	\$	21,118	\$	3,321	\$	21,212 0%
Mark Rayne	Missouri/Iowa	\$	25,092	\$	3,533	\$	22,456 20%
Justin Kay	California	\$	21,809	\$	1,913	\$	24,819 13%
Randy Graham	South Carolina	\$	21,162	\$	1,197	\$	22,005 14%
Christina Foster	Delaware	\$	21,308	\$	3,500	\$	17,517 33%
Judy Green	Texas	\$	21,533	\$	1,617	\$	22,951 6%
Paula Hall	Virginia	\$	21,714	\$	2,418	\$	18,007 18%
	Total	\$	181,033	\$	15,400	\$	172,118

User table in Db2

Service Details - IBM Cloud x IBM Db2 on Cloud x IBM x +

bs2ipcul0apon0uf80lte.db2.cloud.ibm.com/cm%3Av1%3Abuemi%3Apublic%3Adashdb-for-transactions%3Aeu-gb%3Aa%2F95df8426c3654866805d628e020bb10%3Ab2b03ed2-bde4-4e...

IBM Db2 on Cloud

Load Data Load History Tables Views Indexes Aliases MQTs Sequences Application objects

Find schemas or tables Refresh

Schemas

Tables

New table +

Table definition

USER

No statistics available

Name	Data type	Nullable	Length	Scale
USERNAME	VARCHAR	Y	23	0
EMAIL	VARCHAR	Y	31	0
PASSWORD	VARCHAR	Y	10	0
ROLL_NUMBER	VARCHAR	Y	7	0
COLUMN_4	VARCHAR	Y	5	0

View data

Total: 2, selected: 0

94°F Mostly sunny 14:06 24-09-2022

Data in User table

Service Details - IBM Cloud x IBM Db2 on Cloud x IBM x +

bs2ipcul0apon0uf80lte.db2.cloud.ibm.com/cm%3Av1%3Abuemi%3Apublic%3Adashdb-for-transactions%3Aeu-gb%3Aa%2F95df8426c3654866805d628e020bb10%3Ab2b03ed2-bde4-4e...

IBM Db2 on Cloud

Load Data Load History Tables Views Indexes Aliases MQTs Sequences Application objects

KNJ63348.USER

Back

Export to CSV

	USERNAME VARCHAR(23)	EMAIL VARCHAR(31)	PASSWORD VARCHAR(10)	ROLL_NUMBER VARCHAR(7)	COLUMN_4 VARCHAR(5)
1	aquarius	svetmak08@nealheardtrainers.com	moxVWuixTM	ATF1009	
2	batdonut	texakazi@hacktoy.com	tMXpLcNnPY	ATF1008	
3	cookiesfly	r4m4h@getmail.lt	GGofNmDYwM	ATF1005	
4	lee	anthonylee69@timetmail.com	LiBkvFuwp	ATF1010	
5	messier81	tinkergiri@asifboot.com	k1h3Wnldsc	ATF1003	
6	novastick	bgoldman191@hacktoy.com	mvKpJ2h0R	ATF1002	
7	raspberry	gary441@tigo.tk	W4UEQFSAu	ATF1004	
8	rollsodya	kokorinsanja@gmailvn.net	V'imWt*Ql	ATF1007	
9	tracksjuno	katyskat@gmailvn.net	ThTSMujcNW	ATF1006	
10	xylophone	hedob40703@dnitem.com	88pnH31RmN	ATF1001	
11					

94°F Mostly sunny 14:05 24-09-2022

Question-2:

Perform INSERT, SELECT, DELETE Queries in user table

Solution:

Insert Query

The screenshot shows the IBM Db2 on Cloud web interface. On the left, a sidebar lists 'Data objects' and 'My script'. The 'My script' tab is active, showing a list of templates including 'Template - Insert Statement'. The main editor area contains the following SQL query:

```
1 INSERT INTO user (Username, Email)
2 VALUES ('db2class','db@gmail.com');
3
4
```

Below the editor, a 'History' section displays a table of executed queries:

Script	Date	Status	Runtime
Template - Insert Statement	Sep 24, 2022 3:05:19 PM	1	0.011 s
INSERT INTO user (Username, Email) VALUES ('db2class','db@gmail.com')			0.011 s

Insert Query Output

The screenshot shows the 'Tables' view in the IBM Db2 on Cloud interface. The table 'KNJ63348.USER' is selected. The table structure is as follows:

USERNAME	EMAIL	PASSWORD	ROLL_NUMBER	COLUMN_4
aquarius	svetmak0@nealheardtrainers.com	moXVWuixTM	ATF1009	
batdonut	texakazi@hacktoy.com	TMXpLcNnPY	ATF1008	
cookiesfly	r4m4h@gmail.it	GGofNmDYwM	ATF1005	
db2class	db@gmail.com			
lee	anthonylee69@timetmail.com	LiBkivFuwp	ATF1010	
messier81	tinkergirl@asifboot.com	kthJWnidsc	ATF1003	
novastick	bgoldman191@hacktoy.com	nvKpJU2h0R	ATF1002	
raspberry	gary441@tjgo.tk	W4U1EQF5Au	ATF1004	

Update Query

The screenshot shows a SQL editor interface. On the left, there's a sidebar with a search bar and a list of templates. The main area displays a query: `UPDATE user SET Username = 'anto' WHERE Username='lee';`. Below the query, there's a 'History' section with a table of executed scripts.

History Table:

Script	Date	Status	Runtime
Template - Update Statement	Sep 24, 2022 3:14:06 PM	1	0.005 s
UPDATE user SET Username = 'anto' WHERE Username='lee'			0.005 s

Update Query Output

The screenshot shows a SQL query output window. At the top, there's a search bar and a 'Back' button. Below, there's a table with the following columns: USERNAME, EMAIL, PASSWORD, ROLL_NUMBER, and COLUMN_4. The table contains 10 rows of data.

Table Data:

USERNAME	EMAIL	PASSWORD	ROLL_NUMBER	COLUMN_4
anto	anthonylee69@timetmail.com	LiBkqvFuwp	ATF1010	
aquarius	svetmak08@nealheardtrainers.com	moxVWuixTM	ATF1009	
batdonut	texakazi@hacktoy.com	IMXpLcNnPY	ATF1008	
cookiesfly	r4m4h@gmail.it	GGofNmDfwM	ATF1005	
db2class	db@gmail.com			
messier81	tinkergirl@asifboot.com	kthJWnidsc	ATF1003	
novastick	bgoldman191@hacktoy.com	nvKpjU2hOR	ATF1002	
raspberry	gary441@rigo.tk	W4UIEQFSAu	ATF1004	

Delete Query

Filter scripts

Template - Delete Statement

Template - Insert Statement

Template - Insert Statement

Template - Select Statement

Template - SQL Stored Procedure

Template - Update Statement

OwnerSystem

CategoryTemplates

Size66B

Saved6/17/2021

ScriptUPDATE DB2ADMIN.TABLE...

Show more

Template - User Defined Function

Template - Insert ...

* Template - Upda...

* Template - ...

BetaClassic

Syntax assistant

Run all

1DELETE FROM user

2WHERE Username = 'db2class';

3

History

Results

Find history

Script	Date	Status	Runtime
Template - Delete Statement	Sep 24, 2022 3:38:56 PM	1	0.005 s
DELETE FROM user WHERE Username = 'db2class'			0.005 s

Delete Query Output

Export to CSV

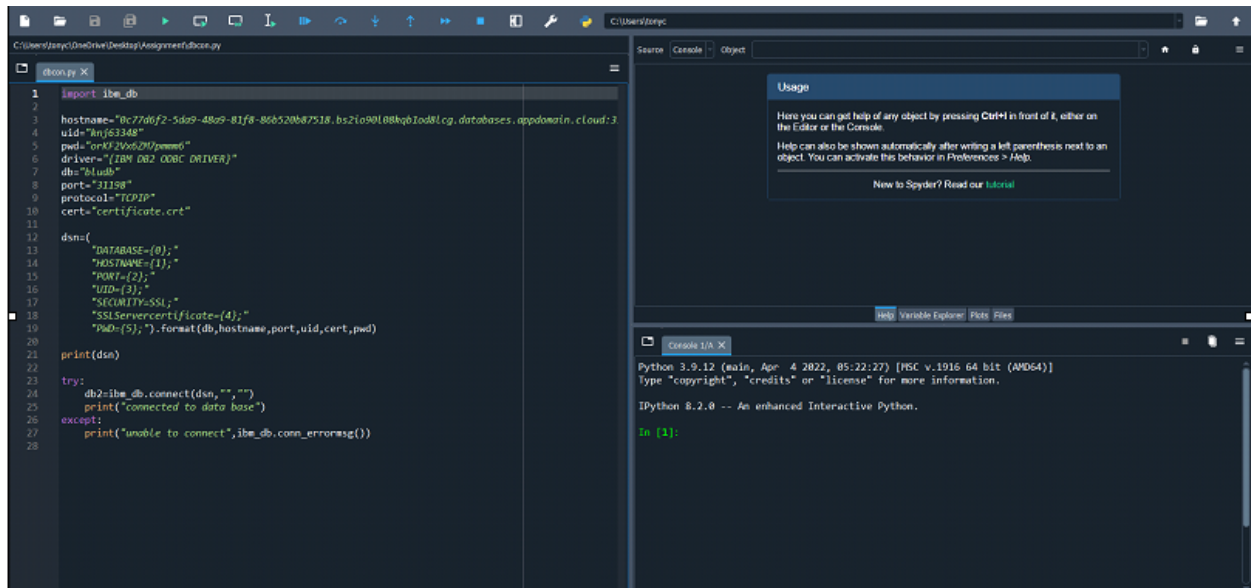
USERNAME	EMAIL	PASSWORD	ROLL_NUMBER	COLUMN_4
anto	anthonylee69@tinetmail.com	LiBkqvFuwp	ATF1010	
aquarius	svetmak08@nealheardtrainers.com	moxVWuixTM	ATF1009	
batdonut	texakazi@hacktoy.com	lMXpLcNnPY	ATF1008	
cookiesfly	r4m4h@gmail.it	GGofNmDYwM	ATF1005	
messier81	tinkergirl@asifboot.com	kIhJWnldsc	ATF1003	
novastick	bgoldman191@hacktoy.com	nvKpjU2h0R	ATF1002	
raspberry	gary441@tigo.tk	W4UITEQF5Au	ATF1004	
rollsyoda	kokorinsanja@gmailvn.net	V'inWl^TqI	ATF1007	

Question-3:

Connect Python to DB2

Solution:

Source Code for Python to DB2 Connection



```
1 import ibm_db
2
3 hostname="0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08qbtad8lcy.databases.appdomain.cloud:3
4 uid="hn63348"
5 pwd="0nX329A5D0/pmw6"
6 driver="(IBM DB2 ODBC DRIVER)"
7 db="bludb"
8 port="21198"
9 protocol="TCP/IP"
10 cert="certificate.crt"
11
12 dsn=(
13     "DATABASE={0};"
14     "HOSTNAME={1};"
15     "PORT={2};"
16     "UID={3};"
17     "SECURITY=SSL;"
18     "SSLServerCertificate={4};"
19     "PWD={5};".format(db,hostname,port,uid,cert,pwd)
20
21 print(dsn)
22
23 try:
24     db2=ibm_db.connect(dsn,"","")
25     print("connected to data base")
26 except:
27     print("unable to connect",ibm_db.conn_errorsg())
28
```

Usage

Here you can get help of any object by pressing **Ctrl+H** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in **Preferences > Help**.

[New to Spyder? Read our tutorial](#)

Python 3.9.12 (main, Apr 4 2022, 05:22:27) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license()" for more information.

IPython 8.2.0 -- An enhanced Interactive Python.

In [1]:

Question-4:

Create a flask app with login page and 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 show the welcome page

Solution:

Source code for login page

```

<?php
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>Login</title>
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0/css/bootstrap.min.css">
<script defer src="https://use.fontawesome.com/releases/v5.13.1/js/all.js"></script>
</head>

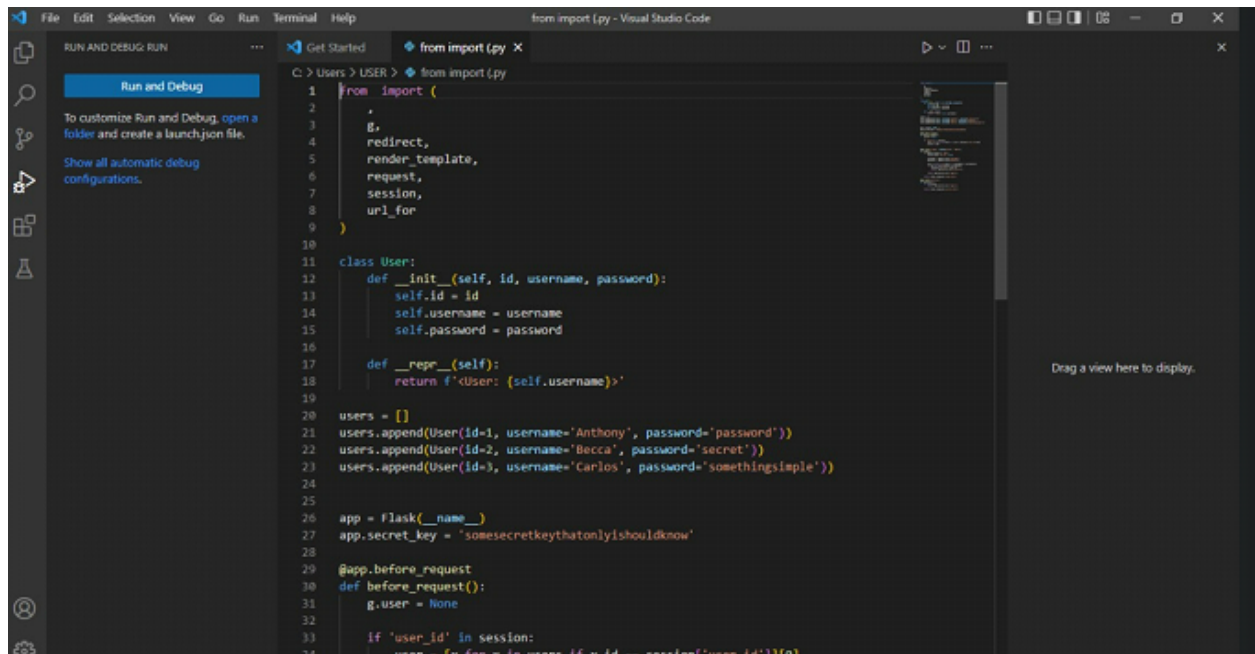
<body>
<section class="hero is-info is-fullheight">
<div class="hero-body">
<div class="container">
<div class="columns is-centered">
<div class="column is-half">
<form method="POST">
<div class="field">
<label class="label">Username</label>
<div class="control has-icons-left has-icons-right">
<input class="input" type="text" name="username" placeholder="Username">
<span class="icon is-small is-left">
<i class="fas fa-user"></i>
</span>
</div>
</div>
<div class="field">
<label class="label">Password</label>
<div class="control has-icons-left">
<input class="input" type="password" name="password" placeholder="Password">
<span class="icon is-small is-left">
<i class="fas fa-lock"></i>
</span>
</div>
</div>
<div class="field">
<p class="control">
<button type="submit" class="button is-success">
Login
</button>
</p>
</div>

```

Source code for welcome page

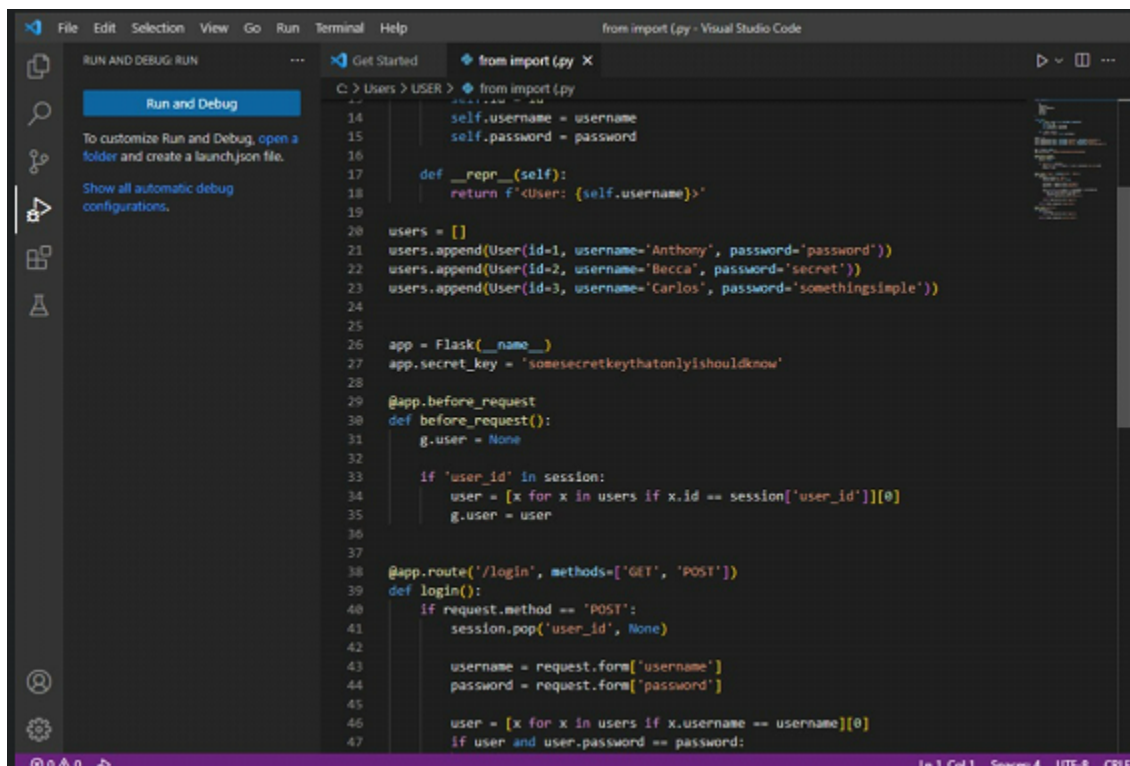
```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Profile Page</title>
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bulma@0.8.0/css/bulma.min.css">
    <script defer src="https://use.fontawesome.com/releases/v5.3.1/js/all.js"></script>
  </head>
  <body>
    <section class="hero is-success is-fullheight">
      <div class="hero-body">
        <div class="container">
          <h1 class="title">
            {{g.user.username}}'s Profile
          </h1>
          <h2 class="subtitle">
            You are user number #{{g.user.id}}
          </h2>
        </div>
      </div>
    </section>
  </body>
</html>
```

Flask code



```
1 from import (
2     '
3     g,
4     redirect,
5     render_template,
6     request,
7     session,
8     url_for
9 )
10
11 class User:
12     def __init__(self, id, username, password):
13         self.id = id
14         self.username = username
15         self.password = password
16
17     def __repr__(self):
18         return f'User: {self.username}'
19
20 users = []
21 users.append(User(id=1, username='Anthony', password='password'))
22 users.append(User(id=2, username='Becca', password='secret'))
23 users.append(User(id=3, username='Carlos', password='somethingimple'))
24
25
26 app = Flask(__name__)
27 app.secret_key = 'somesecretkeythatonlyishouldknow'
28
29 @app.before_request
30 def before_request():
31     g.user = None
32
33     if 'user_id' in session:
```

Flask code



```
14         self.username = username
15         self.password = password
16
17     def __repr__(self):
18         return f'User: {self.username}'
19
20 users = []
21 users.append(User(id=1, username='Anthony', password='password'))
22 users.append(User(id=2, username='Becca', password='secret'))
23 users.append(User(id=3, username='Carlos', password='somethingimple'))
24
25
26 app = Flask(__name__)
27 app.secret_key = 'somesecretkeythatonlyishouldknow'
28
29 @app.before_request
30 def before_request():
31     g.user = None
32
33     if 'user_id' in session:
34         user = [x for x in users if x.id == session['user_id']][0]
35         g.user = user
36
37
38 @app.route('/login', methods=['GET', 'POST'])
39 def login():
40     if request.method == 'POST':
41         session.pop('user_id', None)
42
43         username = request.form['username']
44         password = request.form['password']
45
46         user = [x for x in users if x.username == username][0]
47         if user and user.password == password:
```

Login Page Output



Welcome Page Output