



ADVANCE DATABASE MANAGEMENT SYSTEM

TOPIC: UNIVERSITY MANAGEMENT SYSTEM

INSTRUCTOR: REZWAN AHMED

SECTION: C

GROUP NAME: RJ

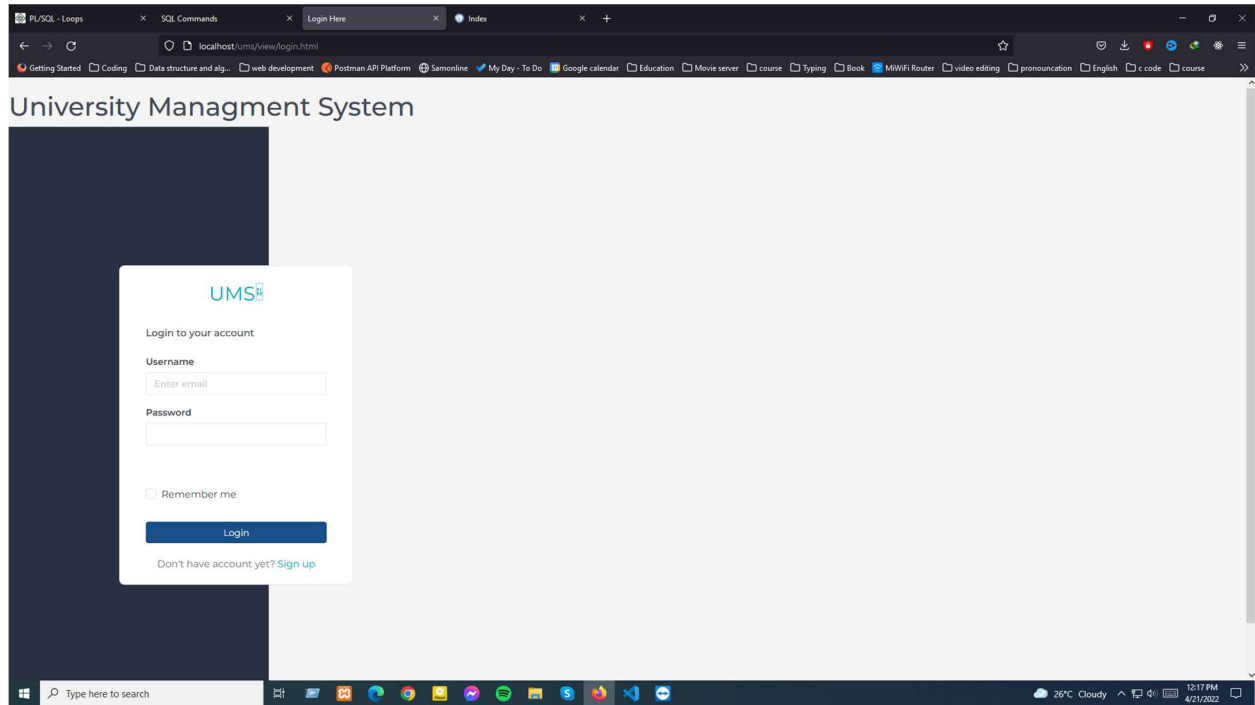
DEPARTMENT: BSc. CSE

GROUP MEMBERS:

NAME	ID
JONY, MD SHAHEEN ALAM	19-40211-1
BASAK, RITU	19-40179-1

ADMIN PANEL

Login:



The screenshot shows a web browser window with the URL `localhost/ums/view/login.html`. The page title is "University Management System". A login form is centered on the page, featuring the UMS logo at the top. The form includes a "Login to your account" heading, a "Username" field with a placeholder "Enter email", a "Password" field, a "Remember me" checkbox, a "Login" button, and a "Sign up" link for users without an account. The browser's taskbar at the bottom shows various application icons and the system clock indicating 12:17 PM on 4/21/2022.

University Management System

UMS

Login to your account

Username

Enter email

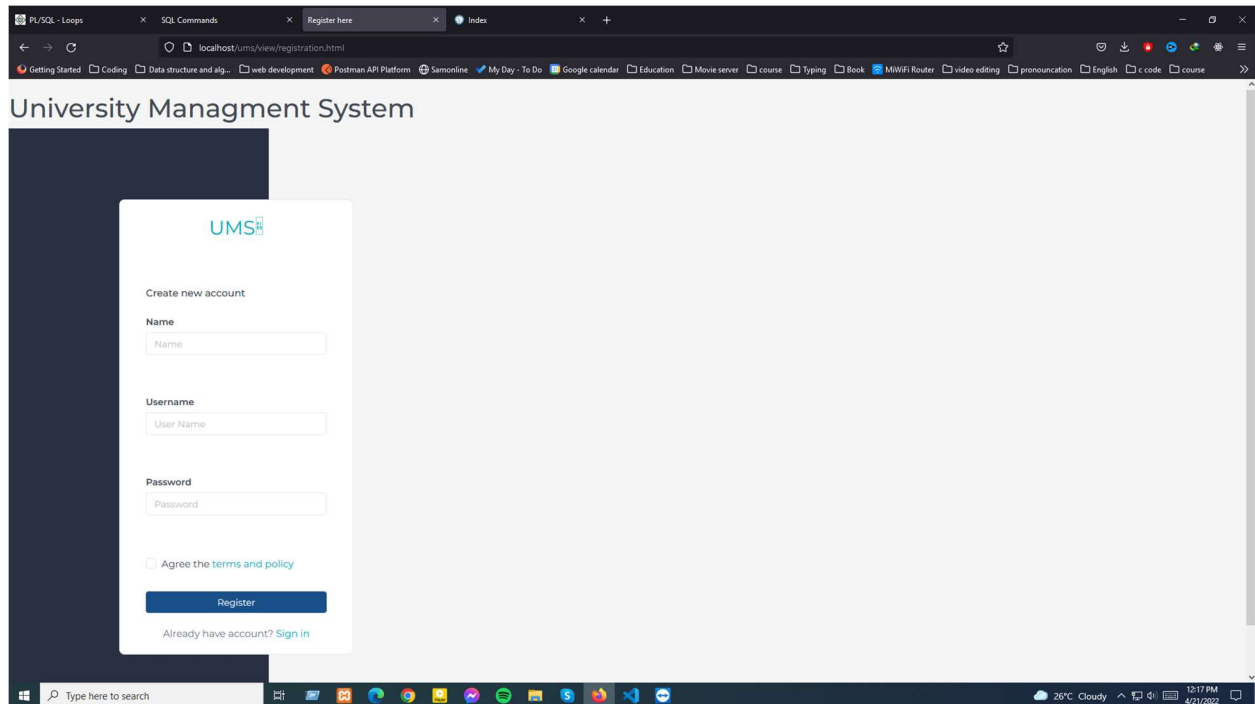
Password

☐ Remember me

Login

Don't have account yet? [Sign up](#)

Registration:



The screenshot shows the same web browser window, but the URL is `localhost/ums/view/registration.html`. The page title remains "University Management System". A registration form is centered on the page, featuring the UMS logo at the top. The form includes a "Create new account" heading, a "Name" field with a placeholder "Name", a "Username" field with a placeholder "User Name", a "Password" field, an "Agree the terms and policy" checkbox, a "Register" button, and a "Sign in" link for users who already have an account. The browser's taskbar at the bottom shows various application icons and the system clock indicating 12:17 PM on 4/21/2022.

University Management System

UMS

Create new account

Name

Name

Username

User Name

Password

Password

☐ Agree the [terms and policy](#)

Register

Already have account? [Sign in](#)

Dashboard:

The screenshot shows the UMS-ADMIN dashboard. On the left is a sidebar menu with options: Dashboard, Add Student, View Student, Add Teacher, View Teacher List, Add Library Book, Edit Library Book, Add Course, Delete Course, and Logout. The main content area is titled 'Welcome shaheen' and displays four statistics: Faculties/Schools (4), Academic Programs (19), Total Graduates (32,189), and Convocations (20). Below these is a 'User List' table with columns NAME, USERNAME, and PASSWORD. The table lists six users: shaheen, karim, shuvo, karim, abdur Rahman, and Ritu Basak, all with the password 1234. At the bottom, a paragraph explains that the University Management System is a process of several manages using structured data to define relationships between data groups.

NAME	USERNAME	PASSWORD
shaheen	shaheen	1234
karim	karim	1234
shuvo	shuvo	1234
karim	karim	1234
abdur Rahman	abdurrahman	1234
Ritu Basak	ritu	1234

Add data:

The screenshot shows the 'Add Student' form in the UMS system. The form is titled 'Add Student' and is set against a blue gradient background. It contains five input fields: STUDENT ID, NAME, EMAIL, DEPARTMENT, and SECTION. A 'Register' button is located at the bottom of the form. On the left side of the form, there is a logo for 'University Management System' and a button labeled 'UMS'.

STUDENT ID:

NAME:

EMAIL:

DEPARTMENT:

SECTION:

SEARCHING & ADVANCE SEARCHING

1. Display department name where student id is 1114.

```
Select Department_name
from Department
where Department_id=(select Department_id
from Student
where Student_id=1114)
```

Results Explain Describe Saved SQL History

DEPARTMENT_NAME
ECONOMICS

1 rows returned in 0.01 seconds [CSV Export](#)

2. Find out in which library When to jump book is available and select the location of that library.

```
Select Library_name, Location
from Library
where Books_name='When to jump'
```

Results Explain Describe Saved SQL History

LIBRARY_NAME	LOCATION
Saya_nazrul	Modhur_canteen

1 rows returned in 0.01 seconds [CSV Export](#)

3. Find out the Email_no of Ritu or Jony.

```
Select Email_no
from student_information
where student_id in(select student_id
from Student
where first_name in('Ritu','Jony'))
```

Results Explain Describe Saved SQL History

EMAIL_NO
ritubsk@gmail.com

1 rows returned in 0.00 seconds [CSV Export](#)

4. Find out the section name of Data structure.

```
Select section
from course_information
where course_id= (select course_id
from course
where Course_name='Data Structure')
```

Results Explain Describe Saved SQL History

SECTION
DD9093-C

1 rows returned in 0.00 seconds [CSV Export](#)

5. Find the city and Country of a faculty with faculty_id-6021.

```
Select City, Country
from Faculty_information
where Passport_no = (select passport_no
from Faculty
where faculty_id=6021)
```

Results Explain Describe Saved SQL History

CITY	COUNTRY
DHAKA	BANGLADESH

1 rows returned in 0.01 seconds [CSV Export](#)

6. Find out the faculty_id and passport_no of faculty whose name starts with R and ends with A.

```
Select faculty_id, faculty_name , passport_no
from faculty
where faculty_name like 'J%n'
```

Results Explain Describe Saved SQL History

FACULTY_ID	FACULTY_NAME	PASSPORT_NO
6025	Juena Nowsin	yy--8382728969

1 rows returned in 0.00 seconds [CSV Export](#)

SEQUENCES

Library Table:

```
create sequence library_seq  
  increment by 1  
  start with 1021  
  maxvalue 100000
```

```
SELECT LIBRARY_ID  
FROM LIBRARY
```

Results Explain Describe Saved

LIBRARY_ID
1021
1022
1023
1024
1025

5 rows returned in 0.00 seconds

Course Information Table:

```
create sequence course_information_seq  
  increment by 2  
  start with 50001  
  maxvalue 100000
```

```
SELECT COURSE_ID  
FROM COURSE_INFORMATION
```

Results Explain Describe Saved

COURSE_ID
50001
50003
50005
50007
50009

5 rows returned in 0.00 seconds

Faculty Table:

```
create sequence faculty_seq  
  increment by 1  
  start with 6021  
  maxvalue 100000
```

```
SELECT FACULTY_ID  
FROM FACULTY
```

Results Explain Describe Saved

FACULTY_ID
6021
6022
6023
6024
6025

5 rows returned in 0.00 seconds

Student_information Table:

```
create sequence Student_information_seq  
  increment by 1  
  start with 1111  
  maxvalue 100000
```

```
SELECT STUDENT_ID  
FROM STUDENT_INFORMATION
```

Results Explain Describe Saved

STUDENT_ID
1111
1112
1113
1114
1115

5 rows returned in 0.00 seconds

VIEW

1.

```
CREATE OR REPLACE VIEW VIEW_FACULTY
AS SELECT FACULTY_ID "F_ID", FACULTY_NAME
"F_NAME", PASSPORT_NO "PASS_NO"
FROM FACULTY
```

```
SELECT * FROM VIEW_FACULTY
```

Results Explain Describe Saved SQL History

F_ID	F_NAME	PASS_NO
6021	Rezwan Ahmed	ba--19562376
6022	Nabil Hasan	bb--37343672
6023	Rifat Tasnim Anannya	aa--672827822
6024	Nazmul Hossain	ga--982876328
6025	Juena Nowsin	yy--8382728969

5 rows returned in 0.03 seconds

[CSV Export](#)

2.

```
CREATE OR REPLACE VIEW STD_DEPT
AS SELECT D.DEPARTMENT_NAME AS D_NAME,
S.FIRST_NAME AS STUDENT_NAME
FROM STUDENT S
JOIN DEPARTMENT D ON S.DEPARTMENT_ID =
D.DEPARTMENT_ID
```

```
SELECT * FROM STD_DEPT
```

Results Explain Describe Saved SQL

D_NAME	STUDENT_NAME
PHARMACY	om
CSE	Shaheen Alam
ENGLISH	Ritu
EEE	Junak
ECONOMICS	Suraiya
BBA	Nusrat

6 rows returned in 0.02 seconds

[CSV](#)

3.

```
CREATE OR REPLACE VIEW VIEW_COURSE
AS SELECT COURSE_ID "C_ID",
COURSE_NAME "C_NAME"
FROM COURSE
```

```
SELECT * FROM VIEW_COURSE
```

Results Explain Describe Saved SQL

C_ID	C_NAME
50001	Advance WebTech
50003	Advance .Net
50005	Data Structure
50007	Java
50041	Advance Database

5 rows returned in 0.00 seconds

[CSV](#)

4.

```
CREATE OR REPLACE VIEW VIEW_LIB
AS
SELECT LIBRARY_NAME AS L_NAME,
BOOKS_NAME AS B_NAME, LOCATION AS
ADDRESS
FROM LIBRARY
WHERE CAPACITY = 3000;
```

```
SELECT * FROM VIEW_LIB
```

Results Explain Describe Saved SQL History

L_NAME	B_NAME	ADDRESS
Saya_nazrul	When_to_jump	Modhur_canteen

1 rows returned in 0.00 seconds

[CSV Export](#)

Procedure

1.

```
CREATE OR REPLACE PROCEDURE INSERT_STUDENT
(STUDENT_ID IN NUMBER, FIRST_NAME IN VARCHAR2, LAST_NAME IN VARCHAR2,
ADDRESS IN VARCHAR2, DEPARTMENT_ID IN NUMBER)
IS
BEGIN
INSERT INTO STUDENT VALUES(STUDENT_ID, FIRST_NAME, LAST_NAME, ADDRESS,
DEPARTMENT_ID);
END;
```

```
BEGIN
INSERT_STUDENT(1116, 'om', 'basak', 'tangail', 2021);
END;
```

2.

```
DECLARE
VAR_NAME VARCHAR2(20);
PROCEDURE UPDATE_NAME(NAME IN VARCHAR2)
IS
BEGIN
UPDATE FACULTY SET FACULTY_NAME=NAME;
END;
BEGIN
VAR_NAME := 'Juena Nowsin';
UPDATE_NAME(VAR_NAME);
DBMS_OUTPUT.PUT_LINE('FACULTY NAME IS UPDATED!');
END;
```

3.

```
DECLARE
VAR_CAPACITY NUMBER;
PROCEDURE UPDATE_CAPACITY(CAPACITY IN NUMBER)
IS
BEGIN
UPDATE LIBRARY SET CAPACITY=CAPACITY;
END;
BEGIN
VAR_CAPACITY := 400;
UPDATE_CAPACITY(VAR_CAPACITY);
DBMS_OUTPUT.PUT_LINE('CAPACITY IS UPDATED!');
END;
```

Function

1.

```
CREATE OR REPLACE FUNCTION DEPT_PROG
RETURN NUMBER
IS
TOTAL_PROG NUMBER(5) :=0;
BEGIN
SELECT COUNT(*) INTO TOTAL_PROG
FROM DEPARTMENT WHERE NO_OF_PROGRAM >150;
RETURN TOTAL_PROG;
END;
DECLARE
N NUMBER(10);
BEGIN
N := DEPT_PROG();
DBMS_OUTPUT.PUT_LINE(N || ' NO. OF RECORDS OF PROGRAMS ARE GREATER THAN 150');
END;
```


2.

```
CREATE OR REPLACE FUNCTION F_COURSE
RETURN VARCHAR2
IS
DETAILS VARCHAR2(20) :=0;
BEGIN
SELECT COUNT(*) INTO DETAILS
FROM COURSE WHERE COURSE_NAME = 'Java';
RETURN DETAILS;
END;

DECLARE
V VARCHAR2(20);
BEGIN
V := F_COURSE();
DBMS_OUTPUT.PUT_LINE(V);
END;
```

3.

```
CREATE OR REPLACE FUNCTION F_LIBRARY
RETURN NUMBER
IS
DETAILS NUMBER(20) :=0;
BEGIN
SELECT SUM(CAPACITY) INTO DETAILS
FROM LIBRARY;
RETURN DETAILS;
END;

DECLARE
V NUMBER(20);
BEGIN
V := F_LIBRARY();
```

```
DBMS_OUTPUT.PUT_LINE('SUM OF LIBRARY CAPACITY IS : ' || V);  
END;
```

Trigger

1.

```
CREATE OR REPLACE TRIGGER INSERT_TRIG_STD  
AFTER INSERT ON STUDENT  
FOR EACH ROW  
BEGIN  
IF INSERTING THEN  
DBMS_OUTPUT.PUT_LINE('1 RECORD IS INSERTED ON ' || SYSDATE());  
END IF;  
END;
```

2.

```
CREATE OR REPLACE TRIGGER DELETE_TRIG_STUDENT  
AFTER DELETE ON STUDENT  
FOR EACH ROW  
BEGIN  
IF DELETING THEN  
DBMS_OUTPUT.PUT_LINE('1 RECORD IS DELETED ON ' || SYSDATE());  
END IF;  
END;
```

```
DELETE FROM STUDENT WHERE FIRST_NAME = 'Puja';
```

3.

```
###OLD & NEW###
```

```
create table BACKUP_STDNT(  
Student_id NUMBER(10),
```

```
First_name varchar2(25),  
Last_name varchar2(25),  
Address varchar2(30),  
Department_id NUMBER(10), TIMESTAMP DATE  
)
```

```
CREATE TRIGGER BCKUP_TRIG_ST  
BEFORE UPDATE ON STUDENT  
FOR EACH ROW  
BEGIN  
INSERT INTO BACKUP_STDNT VALUES(:OLD.STUDENT_ID, :OLD.FIRST_NAME,  
:OLD.LAST_NAME,:OLD.ADDRESS,:OLD.DEPARTMENT_ID,SYSDATE);  
END;
```

```
UPDATE STUDENT SET LAST_NAME='Sanjida' WHERE LAST_NAME = 'Priya';
```

```
SELECT * FROM BACKUP_STDNT;
```

4.

```
create table BCKUP_DEPARTMENT(  
Department_id number(20),  
Department_name varchar2(20),  
No_of_program number(10),  
D_NAME VARCHAR2(20),TIMESTAMP DATE  
)
```

```
CREATE OR REPLACE TRIGGER BCKUP_TRIG_DEPT  
BEFORE UPDATE ON DEPARTMENT  
FOR EACH ROW  
BEGIN  
INSERT INTO BCKUP_DEPARTMENT  
VALUES(:OLD.DEPARTMENT_ID,:OLD.DEPARTMENT_NAME, :OLD.NO_OF_PROGRAM,  
:NEW.DEPARTMENT_NAME,SYSDATE);
```

END;

DELETE FROM DEPARTMENT WHERE NO_OF_PROGRAM = 176;

SELECT * FROM BCKUP_DEPARTMENT

PACKAGE

CREATE OR REPLACE PACKAGE P_STUDENT AS

PROCEDURE DISPLAY_SNAME(S_ID STUDENT.STUDENT_ID%TYPE);

PROCEDURE DISPLAY_ADD(S_ID STUDENT.STUDENT_ID%TYPE);

END P_STUDENT;

CREATE OR REPLACE PACKAGE BODY P_STUDENT AS

PROCEDURE DISPLAY_SNAME(S_ID STUDENT.STUDENT_ID%TYPE)

IS

S_NAME STUDENT.STUDENT_ID%TYPE;

BEGIN

SELECT FIRST_NAME INTO S_NAME

FROM STUDENT

WHERE STUDENT_ID = S_ID;

DBMS_OUTPUT.PUT_LINE('STUDENT NAME : ' || S_NAME);

END DISPLAY_SNAME;

PROCEDURE DISPLAY_ADD(S_ID STUDENT.STUDENT_ID%TYPE)

IS

S_ADD STUDENT.ADDRESS%TYPE;

BEGIN

SELECT ADDRESS INTO S_ADD

FROM STUDENT

WHERE STUDENT_ID = S_ID;

DBMS_OUTPUT.PUT_LINE('STUDENT ADDRESS : ' || S_ADD);

END DISPLAY_ADD;

END P_STUDENT;

BEGIN

P_STUDENT.DISPLAY_SNAME('1112');

P_STUDENT.DISPLAY_ADD('1112');

END;

EXCEPTION

DECLARE

S_ID STUDENT.STUDENT_ID%TYPE := STUDENT_NO;

INVAL_STUDENT_ID EXCEPTION;

F_NAME.FIRST_NAME%TYPE;

BEGIN

IF S_ID < 1 THEN

RAISE INVAL_STUDENT_ID

ELSE

SELECT FIRST_NAME INTO F_NAME FROM STUDENT

WHERE STUDENT_ID = S_ID;

DBMS_OUTPUT.PUT_LINE(F_NAME);

END IF;

EXCEPTION

WHEN INVAL_STUDENT_ID

DBMS_OUTPUT.PUT_LINE('INVALID INFO!');

WHEN NO_DATA_FOUND THEN

DBMS_OUTPUT.PUT_LINE('STUDENT DOES NOT EXIST!');

WHEN OTHERS THEN

DBMS_OUTPUT.PUT_LINE('SOMETHING WENT WRONG!');

END;