DBMS LAB CYCLE 2021

1. Familiarization of DDL Commands

a) Create a database for the college.

b) Create 2 tables

student and staff tables with following fields respectively.

STUDENT			STAFF	
Attribute name	**Data type(size)**		**Attribute name**	**Data type(size)**
rollno	int(3)		Eid	int(5)
name	varchar(20)		name	varchar(15)
age	int(5)		age	int(5)
branch	varchar(10)		branch	varchar(10)
semester	int(10)		designation	varchar(20)

c) List out the tables present in the college database.

d) Show the structure of student table, staff table.

e) Insert values into student table and staff table (at least 3 rows).

f) Alter the student table by adding a column called 'contact number'(int fileld) and insert values into the added filed.

- ✓ by droping a coloumn named 'contact number'
- ✓ modify the existing column named 'semester'
  - # by modidying its data type from 'int' to 'varchar'
  - # by modifying the width of the column from 10 to 5
  - # modifying the constraint of 'semester' colomn from NULL to NOT NULL.

g) Retrieve all data present in student table.

h) Rename student table as 'student details' and staff table as 'staff details'.

i) Delete all data present in the student table and staff table.

j) Drop student table as well as staff table.

k) Drop college database.

2. Familiarization of DML Commands

Create 2 tables employee and department with the corresponding field and constraints given below.

EMPLOYEE	
Eno	primary key and first letter is 'E'
Ename	NOT NULL
Salary	should not be zero

DNO	foreign key referencing DNO of DEPARTMENT
DOJ	
MNGRNO	
JOB	
ADDRESS	
CITY	values must be 'cochin', 'Bombay', 'madrass', 'Delhi'
PINCODE	

where Eno=employee number, dno=department number, mngrno=manager number, doj=date of joing

DEPARTMENT	
DNO	PRIMARY KEY
DNAME	NOT NULL
CNT_EMP	should not be greater than 15
DEPT_HOD	

here CNT_EMP= employee count, DEPT_HOD= head of the department

a) insert values into employee and department tables

EMPLOYEE									
Eno	**Ename**	**Salary**	**Dno**	**DOJ**	**MNGRNO**	**Job**	**Address**	**city**	**PIN**
E1	Lini	40000	D10	1/1/1990	12	Sales	Vytila	Cochin	048622
E20	Anu	50000	D20	30/9/1998	33	Commerce	Kollam	Bombai	665356
E15	Giri	60000	D30	1/9/1999	12	Sales	Kerala	Delhi	62235
E16	Lulu	50000	D15	1/9/1997	-	Agriculture	Kerala	Madras	64633
E12	Sini	40000	D10	1/1/1998	22	finance	Alappuzha	Delhi	241156

DEPARTMENT			
Dno	**Dname**	**CNT-EMP**	**Dept-HOD**
D10	Sales	2	Sreela
D20	Agriculture	1	Vinod
D15	Finance	1	Sreeni
D30	Commerce	1	Greena

b) Display all the employee details & department details.

c) update the 'city' and 'salary' of emplyee whose Eid=E12 to 'cochin' and '70000'.

d) Display all the employee details & department details.

e) List the name of employees joined after 1-1-1998 and working in department number d10.

f) List all employees working in department other than department number d30.

g) List the name of employees working in department 'sales'.

h) List the name of employee who does not have a manager.

i) Display employees details whose city=’cochin’.

j) List the HOD’s of different department.

k) Find out who is the HOD of department D20.

l) Delete employee whose Eid=E15 from employee table.

m) Display details of employee table.

n) Delete employees whose city=’Delhi’.

o) Display details of employee table.

p) Delete all the employees from employee table.

q) Display details of employee table.

3. Familiarization of TCL Commands

Create a database for bank & create a table with name ‘savings-account’. The fields are CID, cname, balance, date of joining.

a) Add 2 records to the ‘savings-account’ table.
b) Display the values of ‘savings-account’ table.
c) Make the changes permanently.
d) Add 2 more records to the ‘savings-account’ table.
e) Display all the records of ‘savings-account’ table.
f) Modify the balance amount by adding the interest of 6%..
g) Display all the records of ‘savings-account’ table.
h) Abandon the last changes.
i) Display all the records of ‘savings-account’ table.
j) Add a marker to the changed state as ‘A’.
k) Add two more records to the ‘’savings-account’ table.
l) Display all the records of ‘savings-account’ table.
m) Modify the balance amount by adding the interest of 6%.
n) Display all the records of ‘savings-account’ table.
o) Add a marker to the changed state as ‘B’.
p) Delete one record from the ‘savings-account table.
q) Display all the records of ‘savings-account’ table.
r) Abandon the last deletion (ie, recover the table with deleted row).
s) Display all the records of ‘savings-account’ table.
t) Abandon to save point/marker ‘A’.
u) Display all the records of ‘savings-account’ table.

4. View

Create a staff table with arguments staffed, sname, salary, sdept, scategory. Scategory can take 2 values (teaching/ non teaching) only.

a) Insert the following values into staff table.

Staffid	sname	salary	sdept	scategory
2	John	60000	CS	teaching
1	Hentry	50000	EC	teaching
10	Jimmy	60000	CS	teaching
5	Anu	20000	CS	non teaching
11	Jithin	50000	EC	teaching
14	Jinu	20000	EC	non teaching
20	Seetha	10000	ME	non teaching

b) Display the details of staff table.

c) Create a view named 'faculty' for teaching staff.

d) Display the contents of 'faculty' view.

e) Create a view named 'non-faculty' for non teaching staff.

f) Display the contents of 'non-faculty' view.

g) Update the salary of non-teaching staff whose staffid=20 to 15000 in corresponding view.

h) Display the contents of 'non-faculty' view.

i) Display the contents of staff table.

j) Delete the details of staff whose staffid=11 from 'faculty' view.

k) Display the contents of 'faculty' view.

l) Create a view named 'viewEC' for staffs in EC department with fieldnames id, name, salary, dept, category respectively.

m) Display the contents of 'viewEC'.

n) Create a view named 'depttoppers' for keeping the information of highest salaried staffs in each department.Net alary is calculated by adding the interest of 5%.

5. Complex Queries

Create 3 tables student (<u>SID</u>,sname,sage), course (<u>CID</u>,Cname) and student-course (SID,CID).

a) Display details of all the 3 tables.

b) Find out student ID (SID) who are enrolled in course name 'DSA' or 'DBMS'.

c) Find out names of students who are either enrolled in 'DSA' or 'DBMS'.

d) Find out the names of students who are neither enrolled in 'DSA' nor in 'DBMS'.

e) Find out the names of students who are enrolled in course ID 'C1'.

6. GROUP BY & HAVING CLAUSE

Create a table called ‘members’ to store member’s details of a family in the ‘family’ database. The attributes of the table are (mno, mname, mage, role, gender).This family consists of twins.

a) Insert following values into ‘members’ table.

mno	mname	mage	role	gender
1	Joseph	60	Father	Male
2	Mary	55	Mother	Female
3	Lilly	16	Daughter	Female
4	Cleetus	15	Son	Male
5	Kevin	20	Son	Male
6	Nevin	20	Son	Male

b) Returns all the gender entries from the ‘members’ table.

c) Retrieve unique values for genders.

d) Returns all member’s name, age, role from ‘members’ table.

e) Select any one child who is eligible for voting.

f) Find out the total no. of males and females in the given family.

g) Select all the eligible candidate’s details for voting in the given family.

7. JOIN operations

a) Create student table with following fields

column name	**data type**	**size**
rollno	varchar	4
name	varchar	15
address	varchar	15
phone	number	10
age	number	3

b) Insert the following values into student table

rollno	**name**	**address**	**phone**	**age**
1	Ram	Delhi	9961253564	18
2	Ramesh	Gurgaon	9962363564	18
3	Sujit	Rohtak	9961253222	20
4	suresh	delhi	9961663564	18

c) Create studentcourse table with following fields

column name	**data type**	**size**
courseid	number	3

rollno	number	3

d) Insert the following values into studentcourse table

courseid	**rollno**
1	1
2	2
2	3
3	4

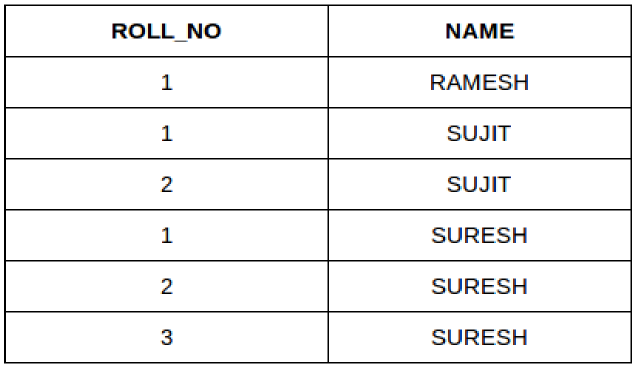
f) Display all the values of employee table.

g) Display all the values of customer table.

h) select NAME and Age from Student table and COURSEID from StudentCourse table. (cross join)

i) each row of the student table is joined with itself and all other rows depending on some conditions(eg: a.ROLL_NO < b.ROLL_NO). (self join)

o/p for given eg:



ROLL_NO	NAME
1	RAMESH
1	SUJIT
2	SUJIT
1	SURESH
2	SURESH
3	SURESH

j) Show the names and age of students enrolled in different courses. (equi join)

k) Perform natural join on ‘student’ and ‘studentcourse’ table.

l) Perform left join on ‘student’ and ‘studentcourse’ table.

m) Perform right join on ‘student’ and ‘studentcourse’ table.

n) Perform full outer join on ‘student’ and ‘studentcourse’ table.

8. SET operations

Based on the following given tables, write query for the following operations

Books			
ID	**Title**	**price**	**status**
1	The witcher	100	available
2	Harry potter	200	available
3	Nineteen eighty four	200	available

1	The witcher	100	available

1	Iron man	100	Not available
2	Harry potter	200	available

a) Returns all distinct rows from 2 tables. (UNION)
b) Returns all rows from 2 tables. (UNION ALL)
c) Returns all distinct rows common to both tables (INTERSECT)
d) Returns all rows common to both tables (INTERSECT ALL)
e) Returns all distinct rows from 'books' table that is not in 'movies' table.(EXCEPT)
f) Returns all rows from 'books' table that is not in 'movies' table. (EXCEPT ALL)

9. Procedures

I.

a) Create a table called 'employee' with the following attributes

employee	
Field names	**Type**
id	Number(3)
name	Varchar(20)

b) Show the structure of the table.

c) Insert the following values into employee table.

id	**name**
101	Nithya
102	maya

d) Display all the values in 'employee' table.

e) Create a procedure to insert a number as 'id' to the 'employee' table. 'name' field of the table should accept default value as the 'user' of the given computer.

f) Display all the values in 'employee' table.

g) Create a procedure to include a new field called 'age' to the 'employee' table and also insert values for this field in all the existing rows as 20, 30, 18 respectively.

h) Create a procedure for employee to get the employee details where age>20.

i) Create a procedure to get employee details using 'id'.

II. Create a procedure to add 2 numbers.

III. Create a procedure to find largest among the given numbers.

10. Functions

Consider the 'bank' table and do the following operations.

Bank		
Acc_no	**B_name**	**balance**

101	SBI	25000
102	SBI	5000
103	FEDERAL	10000
104	AXIS	15000
105	CANARA	50000

a) Create a function for withdrawing money from an account in a bank management system which uses bank table. The minimum balance the account should hold is 500.

b) Display all the values in ‘bank’ table.

c) Create a function for depositing money to an account in a bank management system which uses bank table.

d) Display all the values in ‘bank’ table.

11. Cursors

Consider the ‘student table’ of a particular class and do the following operations.

Roll_No.	**Name**	**M1**	**M2**	**M3**	**Total**	**Percentage**	**grade**
15	Jenny	20	30	20	0	0	0
41	Reena	98	90	85	0	0	0
22	Leena	40	45	60	0	0	0
23	Boban	50	30	20	0	0	0

I. Implicit cursor

a) Create a cursor for calculating the total marks and percentage of the student, grade of the student in a student management system which uses a student table. Grade the student according to the following rules.

Total Marks	**Grade**
>=250	Distinction
180-250	First class
120-179	Second class
180-80	Third class
<80	fail

b) Display all the details of student table.

II. Explicit cursor

a) Create a cursor to find student who got the highest mark from the ‘student’ table and display the particular student details in the following format. Remark for a highest scored student is ‘topper of the class’.

Roll no:

Student name:

Total marks:

Grade:

Remarks:

b) Display all the details of student table.

12. Triggers

a) Create a table called 'reservations' with following fields.

reservations		
Name	**Null**	**Type**
Fight_id	Not null	Char(6)
Customer_phone	Not null	number

b) Create another table called 'flights' with the following fileds. Make 'flight_id' the primary key in 'flights' table.

flights		
Flight_id	Not null	Char(6)
Seats	Not null	number

c) Show the structure of 'reservations' table.

d) Show the structure of 'flights' table.

e) Insert the following rows into 'flights' table.

Flight_id	**Seats**
ACO529	120
ACO530	0

f) Display all the details of 'flights' table.

g) Create a trigger **RES_TRG** that will ensure that when a new row is inserted into the **RESERVATIONS** table, the flight id is in the **FLIGHTS** table and that the number of seats on this flight, SEATS is greater than 0.

Here are the details of how the trigger should behave:

- If flight id is not in the flights table it should raise application error 'Invalid flight id'.
- If flight id is in the flights table, (for example AC0529) but SEATS = 0, then it should raise application error 'Flight AC0529 has no seats left'.
- If flight id is in the flights table and SEATS > 0, then it should update the appropriate row in flights table by setting SEATS = SEATS – 1 for this flight.