GALGOTIAS UNIVERSITY

LAB MANUAL

Data Base Management System

Name of School: SCHOOL OF COMPUTING SCIENCE & ENGINEERING

Department: COMPUTER SCIENCE & ENGINEERING

Year: 2025-2026

GALGOTIAS UNIVERSITY
(Under the Uttar Pradesh Private Universities Act No. 12 of 2019)

SUBJECT	**Data Base Management System**	**PROGRAMME**	**B. Tech.**
SUBJECT CODE	**E2UC302B**	**SEMESTER**	**3**
CREDITS	**1**	**DURATION OF SEMESTER**	**13 Weeks**
PREREQUISITE SUBJECTS	**NA**	**SESSION DURATION**	**2 Hrs per Week**

Vision

"To be recognized globally as a premier School of Computing Science and Engineering for imparting quality and value-based education within a multi-disciplinary and collaborative research-based environment."

Mission

The mission of the school is to:

M1: Develop a strong foundation in fundamentals of computing science and engineering with responsiveness towards emerging technologies.

M2: Establish state-of-the-art facilities and adopt education 4.0 practices to analyze, develop, test and deploy sustainable ethical IT solutions by involving multiple stakeholders.

M3: Foster multidisciplinary collaborative research in association with academia and industry through focused research groups, Centre of Excellence, and Industry Oriented R&D Labs.

PROGRAM EDUCATIONAL OBJECTIVES

The Graduates of Computer Science and Engineering shall:

PEO1: be engaged with leading Global Software Services and Product development companies handling projects in cutting edge technologies.

PEO2: serve in technical or managerial roles at Government firms, Corporates and contributing to thesociety as successful entrepreneurs through startup.

PEO3: undertake higher education, research or academia at institutions of transnational reputation.

PROGRAMME SPECIFIC OUTCOME (PSO):

The students of Computer Science and Engineering shall:

PSO1: Have the ability to work with emerging technologies in computing requisite to Industry 4.0.

PSO2: Demonstrate Engineering Practice learned through industry internship and research project to solve live problems in various domains.

PROGRAMME OUTCOME (PO):

PO1 Computing Science knowledge: Apply the knowledge of mathematics, statistics, computing science and information science fundamentals to the solution of complex computer application problems.

PO2 Problem analysis: Identify, formulate, review research literature, and analyze complex computing science problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and computer sciences.

PO3 Design/development of solutions: Design solutions for complex computing problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4 Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5 Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern computing science and IT tools including prediction and modeling to complex computing activities with an understanding of the limitations.

PO6 IT specialist and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional computing science and information science practice.

PO7 Environment and sustainability: Understand the impact of the professional computing science solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8 Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the computing science practice.

PO9 Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10 Communication: Communicate effectively on complex engineering activities with the IT analyst community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11 Project management and finance: Demonstrate knowledge and understanding of the computing science and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12 Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

List of Experiments

LAB PLAN FOR THEORY COURSES (15 weeks * 2 Hours = 30 Classes)		
1	L	Draw an E-R diagram and convert entities and relationships to a relation table for a given scenario. (Two assignments shall be carried out i.e., consider two different scenarios (e.g. bank, College)
2	L	Implementation of DDL commands of SQL with suitable examples. (a) Create table (b) Alter table (c) Drop Table

3	L	Implementation of DML commands of SQL with suitable examples. (a) Insert table (b) Update table (c) Delete Table
4	L	Implementation of different types of functions with suitable examples. Number Function Aggregate Function Character Function Conversion Function Date Function
5	L	Implementation of different types of operators in SQL.
6	L	Perform the following: a. Creating Tables (With and Without Constraints (Key/Domain) b. Creating Tables (With Referential Integrity Constraints)
7	L	For a given set of relation schemes, create tables and perform the following Queries: a. Simple Queries b. Queries with Aggregate functions (Max/Min/Sum/Avg/Count) c. Queries with Aggregate functions (group by and having clause) d. Queries involving- Date Functions, String Functions, Math Functions
8	L	For a given set of relation schemes, create tables and perform the following Queries: a. Inner Join b. Outer Join c. Natural Join
9	L	. For a given set of related tables perform the following: - a. Creating Views b. Dropping views c. Selecting from a view
10	L	Implementation of Group by & Having Clause, Order by Clause, Indexing.
11	L	Given the table EMPLOYEE (EmpNo, Name, Salary, Designation, DeptID) write a cursor to select the five highest-paid employees from the table.
12	L	For a given set of related tables perform the following: a. Begin Transactions b. End Transaction
13	L	For a given set of related tables perform the following: a. Create roles b. Assign Privileges c. Revoke Privileges
14	L	Perform the following: Inserting/Updating/Deleting Records in a Table, Saving (Commit) and Undoing (rollback)

Course Outcomes

Upon fruitful completion of this course, students will be able to

CO1	Conceptual understanding of DBMS, ability to define and manipulate data, understanding data independence, and the overall structure of databases.
CO2	Ability to design and query relational databases using SQL, ensure data integrity, and apply relational algebra and calculus concepts in practical scenarios.
CO3	Apply database Normalization techniques up to BCNF for the removal of anomalies.
CO4	Ability to manage and ensure the consistency of transactions, implement concurrency control mechanisms, recover from transaction failures.

CO-PO-PSO MAPPING:

COs#/ POs	PO 1	PO 2	PO 3	PO 4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO 1	PSO2
CO1	3	2	1	1	1	-	-	-	-	-	-	-	-	1
CO2	3	2	2	2	2	-	-	-	-	-	-	-	1	1
CO3	2	3	2	2	2	-	-	-	-	-	-	-	1	1
CO4	2	2	1	1	2	-	-	-	-	-	-	-	2	-

Continuous Assessment Pattern

Practical	ETE	Total
Lab (25), Lab Exam 25)	0	50

Rubrics for Practical

S. No.	Rubrics - Parts	Marks
1	Performance	**5**
2	Result	**5**
3	File	**5**
4	Viva	**10**
Total		**25**

Experiment 1

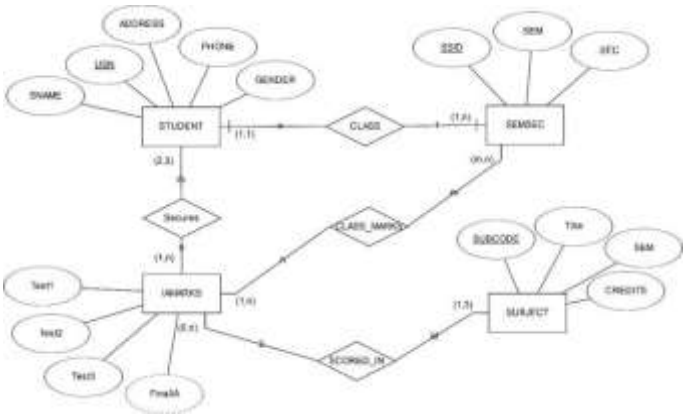
Aim: Consider following databases and draw ER diagram and convert entities and relationships to relation table for a given scenario. Solution:

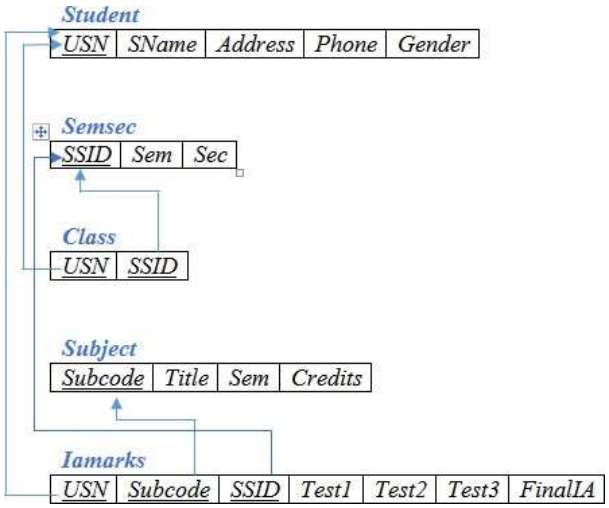
1. **COLLEGE DATABASE:**
   STUDENT (*USN, SName, Address, Phone, Gender*) SEMSEC (*SSID, Sem, Sec*)
   CLASS (*USN, SSID*)
   SUBJECT (*Subcode, Title, Sem, Credits*)
   IAMARKS (*USN, Subcode, SSID, Test1, Test2, Test3, FinalIA*)

2. **COMPANY DATABASE:**

EMPLOYEE (*SSN, Name, Address, Sex, Salary, SuperSSN, DNo*)
DEPARTMENT (*DNo, DName, MgrSSN, MgrStartDate*) DLOCATION (*DNo,DLoc*)
PROJECT (*PNo, PName, PLocation, DNo*)
WORKS_ON (*SSN, PNo, Hours*)

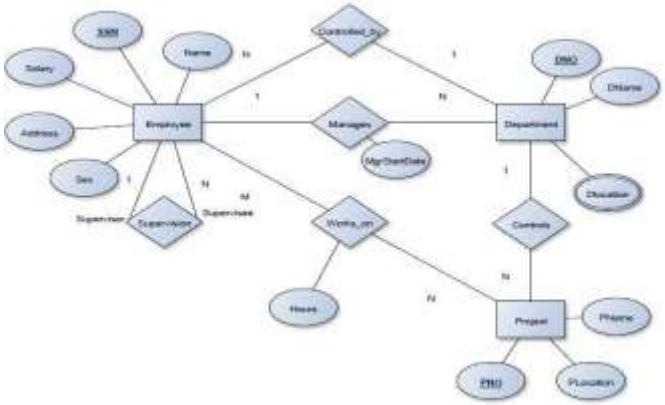
College Database: E-R Diagram

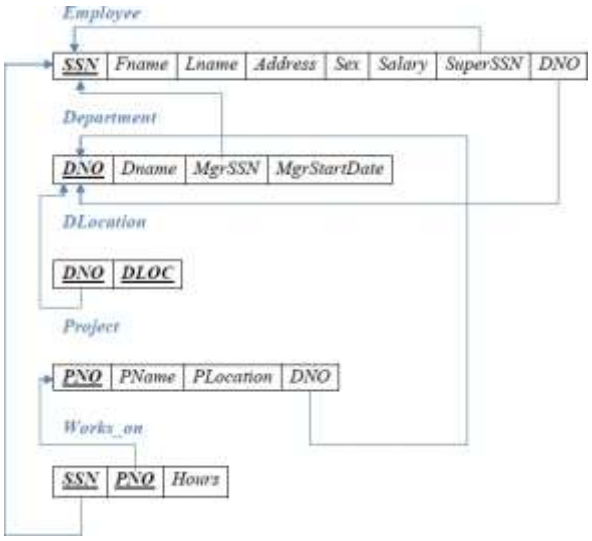




Mapping entities and relationships to relation table (Schema Diagram)

COMPANY DATABASE: E-R Digram





Experiment 2

AIM: Implementation of DDL commands of SQL with suitable examples. (a) Create table (b) Alter table (c) Drop Table.

1. CREATE:

(a)CREATE TABLE: This is used to create a new relation (table)

Syntax: CREATE TABLE <relation_name/table_name > (field_1 data_type(size),field_2 data_type(size), .. .); Example:

SQL> CREATE TABLE Student (sno NUMBER (3), sname CHAR (10), class CHAR (5)); 2.

ALTER:

(a) ALTER TABLE ...ADD...: This is used to add some extra fields into existing relation.

Syntax: ALTER TABLE relation_name ADD (new field_1 data_type(size), new field_2 data_type(size),..); Example:
SQL>ALTER TABLE std ADD (Address CHAR(10));
(b) ALTER TABLE...MODIFY...: This is used to change the width as well as data type of fields of existing relations.

Syntax: ALTER TABLE relation_name MODIFY (field_1 newdata_type(Size), field_2 newdata_type(Size), ... field_newdata_type(Size));
Example:SQL>ALTER TABLE student MODIFY(sname VARCHAR(10),class VARCHAR(5)); c)
ALTER TABLE..DROP .... This is used to remove any field of existing relations.

Syntax: ALTER TABLE relation_name DROP COLUMN (field_name);
Example:SQL>ALTER TABLE student DROP column (sname);
d)ALTER TABLE..RENAME...: This is used to change the name of fields in existing relations.

Syntax: ALTER TABLE relation_name RENAME COLUMN (OLD field_name) to (NEW field_name); Example:
SQL>ALTER TABLE student RENAME COLUMN sname to stu_name;
3. DROP TABLE: This is used to delete the structure of a relation. It permanently deletes the records in the table.

Syntax: DROP TABLE relation_name; Example:
SQL>DROP TABLE std;
4. RENAME: It is used to modify the name of the existing database object. Syntax: RENAME TABLE old_relation_name TO new_relation_name; Example: SQL>RENAME TABLE std TO std1;

1.Create a table EMPLOYEE with following schema:
(Emp_no, E_name, E_address, E_ph_no, Dept_no, Dept_name,Job_id , Salary) **2.**
Add a new column; HIREDATE to the existing relation.
3. Change the datatype of JOB_ID from char to varchar2.

4. Change the name of column/field Emp_no to E_no.

5. Modify the column width of the job field of emp table

Experiment 3

AIM: Implementation of DML commands of SQL with suitable examples. (a) Insert table (b) Update table (c) Delete Table

1. INSERT INTO: This is used to add records into a relation. These are three type of INSERT INTO queries which are as **a)**

Inserting a single record

Syntax: INSERT INTO < relation/table name> (field_1,field_2……field_n)VALUES (data_1,data_2, data_n); ***Example:***
SQL>INSERT INTO student (sno,sname,class,address)VALUES (1,'Ravi','M.Tech','Palakol');

b) Inserting a single record

Syntax: INSERT INTO < relation/table name>VALUES (data_1,data_2, ... data_n); ***Example:***
SQL>INSERT INTO student VALUES (1,'Ravi','M.Tech','Palakol');

c) Inserting all records from another relation

Syntax: INSERT INTO relation_name_1 SELECT Field_1,field_2,field_n FROM relation_name_2 WHERE field_x=data;
Example: SQL>INSERT INTO std SELECT sno,sname FROM student WHERE name = 'Ramu';

d) Inserting multiple records

Syntax: INSERT INTO relation_name field_1,field_2, ... field_n) VALUES
(&data_1,&data_2, ... &data_n);
Example: SQL>INSERT INTO student (sno, sname, class,address) VALUES (&sno,'&sname','&class','&address');
Enter value for sno: 101 Enter value for name: Ravi Enter value for class: M.Tech Enter value for name: Palakol

2. UPDATE-SET-WHERE: This is used to update the content of a record in a relation.

Syntax: SQL>UPDATE relation name SET Field_name1=data,field_name2=data, WHERE field_name=data; ***Example:*** SQL>UPDATE student SET sname = 'kumar' WHERE sno=1;

3. DELETE-FROM: This is used to delete all the records of a relation but it will retain the structure of that relation.

a) DELETE-FROM: This is used to delete all the records of relation.

Syntax: SQL>DELETE FROM relation_name;
Example: SQL>DELETE FROM std;

b) DELETE -FROM-WHERE: This is used to delete a selected record from a relation.

Syntax: SQL>DELETE FROM relation_name WHERE condition;
Example: SQL>DELETE FROM student WHERE sno = 2;

5. TRUNCATE: This command will remove the data permanently. But structure will not be removed.

Experiment 5

Aim: Perform the following:

a. Altering a Table, Dropping/Truncating/Renaming Tables.

b. Adding a column, Changing column data type, size

c. Dropping a column

Consider Dept table

DEPTNO	DNAME	LOC

Perform the following:

1. Rename the table dept as department
2. Add a new column PINCODE with not null constraints to the existing table DEPT
3. All constraints and views that reference the column are dropped automatically, along with the column.
4. Rename the column DNAME to DEPT_NAME in dept table
5. Change the data type of column loc as CHAR with size 10
6. Delete table

SOLUTION:

Create Table

SQL> CREATE TABLE DEPT(DEPTNO INTEGER, DNAME VARCHAR(10),LOC VARCHAR(4), PRIMARY KEY(DEPTNO));

1. Rename the table dept as department

   SQL> ALTER TABLE DEPT RENAME TO DEPARTMENT;
   Table altered.

2. Add a new column PINCODE with not null constraints to the existing table DEPT SQL> ALTER

   TABLE DEPARTMENT ADD(PINCODE NUMBER(6) NOT NULL); Table altered.

   SQL> DESC DEPARTMENT;

Name	Null?	Type
DEPTNO	NOT NULL	NUMBER(38)
DNAME		VARCHAR2(10)
LOC		VARCHAR2(4)
PINCODE	NOT NULL	NUMBER(6)

3. All constraints and views that reference the column are dropped automatically, along with the column.

   SQL> ALTER TABLE DEPARTMENT DROP column LOC CASCADE CONSTRAINTS;

   Table altered.

   SQL> desc dept

Name	Null?	Type
DEPTNO	NOT NULL	NUMBER(38)
DNAME		VARCHAR2(10)
PINCODE	NOT NULL	NUMBER(6)

4. Rename the column DNAME to DEPT_NAME in dept table

   SQL> ALTER TABLE DEPT RENAME COLUMN DNAME TO DEPT_NAME ;

   Table altered.
   SQL> DESC DEPARTMENT;

Name	Null?	Type
DEPTNO	NOT NULL	NUMBER(38)
DEPT_NAME		VARCHAR2(10) LOC VARCHAR2(4)
PINCODE	NOT NULL	NUMBER(6)

5. Change the datatype of columm loc as CHAR with size 10

SQL> ALTER TABLE DEPARTMENT MODIFY LOC CHAR(10) ;
Table altered.
SQL> DESC DEPARTMENT; Name Null?
Type

DEPTNO NOT NULL NUMBER(38)
DEPT_NAME VARCHAR2(10) LOC
CHAR(10)
PINCODE NOT NULL NUMBER(6)

6. Delete table

SQL> DROP TABLE DEPARTMENT;
Table dropped.

Experiment 5A

Consider Employee table

EMPNO	EMP_NAME	DEPT	SALARY	DOJ	BRANCH
E101	Amit	oduction	45000	12-Mar-00	Bangalore
E102	Amit	HR	70000	03-Jul-02	Bangalore
E103	sunita	anagemen	120000	11-Jan-01	mysore
E105	sunita	IT	67000	01-Aug-01	mysore
E106	mahesh	Civil	145000	20-Sep-03	Mumbai

Perform the following

1. **Display all the fields of employee table**
2. **Retrieve employee number and their salary**
3. **Retrieve average salary of all employee**
4. **Retrieve number of employee**
5. **Retrieve distinct number of employee**
6. **Retrieve total salary of employee group by employee name and count similar names**
7. **Retrieve total salary of employee which is greater than >120000**
8. **Display name of employee in descending order**
9. **Display details of employee whose name is AMIT and salary greater than 50000;**

1. Display all the fields of employee table SQL>
select * from employee;

EMPN	EMP_NAME	DEPT	SALARY O	DOJ	BRANCH
E101	Amit	Production	45000	12-MAR-00	Bangalore
E102	Amit	HR	70000	03-JUL-02	Bangalore
E103	sunita	Management	120000	11-JAN-01	mysore
E105	sunita	IT	67000	01-AUG-01	mysore

E106 mahesh Civil 145000 20-SEP-03 Mumbai

2. Retrieve employee number and their salary

SQL> select empno, salary from employee;

EMPNO SALARY

E101 45000

E102 70000

E103 120000

E105 67000

E106 145000

3. Retrieve average salary of all employee

SQL> select avg(salary) from employee;

AVG(SALARY)

89400

4. Retrieve number of employee

SQL> select count(*) from employee;

COUNT(*)

5

5. Retrieve distinct number of employee

SQL> select count(DISTINCT emp_name) from employee;

COUNT(DISTINCTEMP_NAME)

3

6. Retrieve total salary of employee group by employee name and count similar names

SQL> SELECT EMP_NAME, SUM(SALARY),COUNT(*) FROM
EMPLOYEE 2 GROUP BY(EMP_NAME);

EMP_NAME SUM(SALARY) COUNT(*)

---

mahesh 145000 1 sunita 187000 2 Amit 115000
2

7. Retrieve total salary of employee which is greater than >120000

SQL> SELECT EMP_NAME, SUM(SALARY) FROM
EMPLOYEE 2 GROUP BY(EMP_NAME)
3 HAVING SUM(SALARY)>120000;

EMP_NAME SUM(SALARY)

--- ---

mahesh 145000
sunita 187000

8. Display name of employee in descending order

```
SQL> select emp_name from employee 2
  order by emp_name desc;
EMP_NAME

----------------------------


sunita sunita
mahesh
Amit
Amit
```

Experiment 5B For

a given tables

EMPLOYEE	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
	John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
	Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
	Alicia	J	Zelaya	999887777	1968-07-19	3321 Castle, Spring, TX	F	25000	987654321	4
	Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
	Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
	Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
	Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
	James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	null	1

DEPARTMENT	DNAME	DNUMBER	MGRSSN	MGRSTARTDATE
	Research	5	333445555	1988-05-22
	Administration	4	987654321	1995-01-01
	Headquarters	1	888665555	1981-06-19

Create tables and perform the following

1. How the resulting salaries if every employee working on the 'Research' Departments is given a 10 percent raise.
2. Find the sum of the salaries of all employees of the 'Accounts' department, as well as the maximum salary, the minimum salary, and the average salary in this department
3. Retrieve the name of each employee Controlled by department number 5 (use EXISTS operator).
4. Retrieve the name of each dept and number of employees working in each department which has at least 2 employees
5. Retrieve the name of employees who born in the year 1990's
6. Retrieve the name of employees and their dept name (using JOIN)

SOLUTION

```
SQL>  CREATE  TABLE  DEPARTMENT(  DNO
          VARCHAR2 (20) PRIMARY KEY,
          DNAME VARCHAR2 (20), MGRSTARTDATE DATE);

SQL> DESC DEPARTMENT;
          Name                                  Null?      Type

          ----------------------------------------------------------- -------- ------------------------------------

          DNO                                   NOT NULL VARCHAR2(20)
          DNAME                                          VARCHAR2(20)
```

MGRSTARTDATE DATE

```
SQL> CREATE TABLE EMPLOYEE( SSN
VARCHAR2 (20) PRIMARY KEY,
FNAME VARCHAR2 (20),
LNAME VARCHAR2 (20),
ADDRESS VARCHAR2 (20),
SEX CHAR (1),
SALARY INTEGER,
SUPERSSN REFERENCES EMPLOYEE (SSN), DNO
REFERENCES DEPARTMENT (DNO));

SQL> DESC EMPLOYEE;
Name Null? Type

---

SSN NOT NULL VARCHAR2(20)
FNAME VARCHAR2(20)
LNAME VARCHAR2(20)
ADDRESS VARCHAR2(20)
SEX CHAR(1)
SALARY NUMBER(38)
SUPERSSN VARCHAR2(20)
DNO NUMBER(38)

SQL> ALTER TABLE DEPARTMENT
2 ADD MGRSSN REFERENCES EMPLOYEE (SSN); Table

altered.

SQL> DESC DEPARTMENT;
Name Null? Type

---

DNO NOT NULL VARCHAR2(20) DNAME
VARCHAR2(20)
MGRSTARTDATE DATE
MGRSSN VARCHAR2(20)

INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSECE01','JOHN','SCOTT','BANGALORE','M', 450000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSCSE01','JAMES','SMITH','BANGALORE','M', 500000); INSERT INTO EMPLOYEE
(SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSCSE02','HEARN','BAKER','BANGALORE','M', 700000); INSERT INTO EMPLOYEE
(SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSCSE03','EDWARD','SCOTT','MYSORE','M', 500000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSCSE04','PAVAN','HEGDE','MANGALORE','M', 650000); INSERT INTO EMPLOYEE
(SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSCSE05','GIRISH','MALYA','MYSORE','M', 450000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSCSE06','NEHA','SN','BANGALORE','F', 800000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSACC01','AHANA','K','MANGALORE','F', 350000);
```

INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSACC02','SANTHOSH','KUMAR','MANGALORE','M', 300000); INSERT INTO EMPLOYEE
(SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSISE01','VEENA','M','MYSORE','M', 600000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSIT01','NAGESH','HR','BANGALORE','M', 500000);

INSERT INTO DEPARTMENT VALUES (1,'ACCOUNTS','01-JAN- 01','RNSACC02');
INSERT INTO DEPARTMENT VALUES (2,'IT','01-AUG-16','RNSIT01'); INSERT INTO
DEPARTMENT VALUES (3,'ECE','01-JUN-08','RNSECE01'); INSERT INTO DEPARTMENT
VALUES (4,'ISE','01-AUG-15','RNSISE01'); INSERT INTO DEPARTMENT VALUES (5,'CSE','01-JUN-
02','RNSCSE05');

Note: update entries of employee table to fill missing fields SUPERSSN and DNO UPDATE EMPLOYEE
SET SUPERSSN=NULL, DNO='3' WHERE SSN='RNSECE01';
UPDATE EMPLOYEE SET SUPERSSN='RNSCSE02', DNO='5' WHERE SSN='RNSCSE01';
UPDATE EMPLOYEE SET SUPERSSN='RNSCSE03', DNO='5' WHERE SSN='RNSCSE02';
UPDATE EMPLOYEE SET SUPERSSN='RNSCSE04', DNO='5' WHERE SSN='RNSCSE03';
UPDATE EMPLOYEE SET DNO='5', SUPERSSN='RNSCSE05' WHERE SSN='RNSCSE04';
UPDATE EMPLOYEE SET DNO='5', SUPERSSN='RNSCSE06' WHERE SSN='RNSCSE05';
UPDATE EMPLOYEE SET DNO='5', SUPERSSN=NULL WHERE SSN='RNSCSE06';
UPDATE EMPLOYEE SET DNO='1', SUPERSSN='RNSACC02' WHERE SSN='RNSACC01';
UPDATE EMPLOYEE SET DNO='1', SUPERSSN=NULL WHERE SSN='RNSACC02';
UPDATE EMPLOYEE SET DNO='4', SUPERSSN=NULL WHERE SSN='RNSISE01';
UPDATE EMPLOYEE SET DNO='2', SUPERSSN=NULL WHERE SSN='RNSIT01';

1. **How the resulting salaries if every employee working on the 'Research' Departments is given a 10 percent raise.**

```
SQL> SELECT E.FNAME,E.LNAME, 1.1*E.SALARY AS INCR_SAL 2 FROM
  EMPLOYEE1 E,DEPARTMENT D,EMPLOYEE1 W
  3 WHERE E.SSN=W.SSN
  4 AND E.DNO=D.DNUMBER
  5 AND D.DNAME='research';
```

FNAME	LNAME	SALARY	DNO	DNUMBER	INC_SAL
john	smith	30000	5	5	33000
franklin	wong	40000	5	5	44000
ramesh	narayan	780000	5	5	858000
joyce	english	25000	5	5	27500

2. **Find the sum of the salaries of all employees of the 'Accounts' department, as well as the maximum salary, the minimum salary, and the average salary in this department**

```
SQL> SELECT SUM(E.SALARY),MAX(E.SALARY),MIN(E.SALARY),
AVG(E.SALARY)FROM EMPLOYEE1 E,DEPARTMENT D WHERE
E.DNO=D.DNUMBER AND D.DNAME='RESEARCH';
```

SUM(E.SALARY)	MAX(E.SALARY)	MIN(E.SALARY)	AVG(E.SALARY)
875000	780000	25000	218750

3. **Retrieve the name of each employee Controlled by department number 5 (use EXISTS operator).**

```
SQL> SELECT

 E.FNAME,E.LNAME 2 FROM

 EMPLOYEE1 E
3     WHERE EXISTS(SELECT DNO FROM EMPLOYEE1 WHERE E.DNO=5);
```

FNAME	LNAME
john	smith
franklin	wong
ramesh	narayan
joyce	english

4. **Retrieve the name of each dept and number of employees working in each department which has at least 2 employees**

```
SELECT DNAME, COUNT(*)
FROM EMPLOYEE E, DEPARTMENT D WHERE
D.DNO=E.DNO
AND D.DNO IN (SELECT E1.DNO FROM
EMPLOYEE E1
GROUP BY E1.DNO
having count(*)>2 )
ORDER BY DNO;
```

5. **Retrieve the name of employees who born in the year 1990's**

```
SELECT E.FNAME,E.LNAME,E.BDATE FROM EMPLOYEE1 E WHERE BDATE LIKE '196%';
```

FNAME	LNAME	BDATE
john	smith	1965-jan-09

6. **Retrieve the name of employees and their dept name (using JOIN)**

```
SELECT E.FNAME, E.LNAME, DNAME
FROM EMPLOYEE E NATURAL JOIN DEPARTMENT D ON E,DNO=D.DNO;
```

Experiment 6

Perform the String Functions, Date functions and Mathematical functions supported by Oracle

```
SQL> select ascii('t') from dual;

ASCII('T')
----------
```

```
SQL> select ascii('a') from dual;

ASCII('A')
----------

        97

SQL> select ascii('A') from dual;

ASCII('A')
----------

        65

SQL> select ascii('Z') from dual;

ASCII('Z')
----------

        90

SQL> select ascii('z') from dual;

ASCII('Z')
----------

       122


SQL> SELECT UPPER('bldea sb arts and kcp science college') from dual;

UPPER('BLDEASBARTSANDKCPSCIENCECOLLEG
-------------------------------------

BLDEA SB ARTS AND KCP SCIENCE COLLEGE

SQL> select LOWER('welcome to dbms lab') from dual;

LOWER('WELCOMETODBM
-------------------

welcome to dbms lab
SQL> select LOWER('WELCOME TO DBMSLAB') from dual;

LOWER('WELCOMETODB
------------------

welcome to dbmslab


SQL> SELECT REPLACE('HELLO','H','K') FROM DUAL;

REPLA
```

KELLO

SQL> SELECT REPLACE('COMPUTER','C','K') FROM

DUAL; REPLACE(

KOMPUTER

SQL> SELECT REPLACE('HELLO','L','A') FROM DUAL;

REPLA

HEAAO

SQL> SELECT TRIM('A' FROM 'ANACONDA') FROM

DUAL; TRIM('

--

NACOND

SQL> SELECT LTRIM('ANACONDA','A') FROM DUAL;

LTRIM('

NACONDA

SQL> SELECT LTRIM('ANIL','A') FROM

DUAL; LTR

NI

L

SQL> SELECT RTRIM('ANITA','A') FROM DUAL;

RTRI

ANIT

SQL> SELECT RTRIM('ANACONDA','A') FROM DUAL;

RTRIM('

ANACOND

SQL> SELECT RTRIM('ANACONDA ','A') FROM DUAL;

RTRIM('ANAC

ANACONDA

Date Functions

SQL> SELECT CURRENT_DATE FROM DUAL;

CURRENT_D

14-AUG-19

SQL> SELECT EXTRACT(YEAR FROM SYSDATE) FROM

DUAL; EXTRACT(YEARFROMSYSDATE)

2019

SQL> SELECT EXTRACT(DAY FROM SYSDATE) FROM DUAL;

EXTRACT(DAYFROMSYSDATE)

14

SQL> SELECT EXTRACT(MONTH FROM SYSDATE) FROM

DUAL; EXTRACT(MONTHFROMSYSDATE)

8

SQL> SELECT SYSDATE FROM

DUAL; SYSDATE

14-AUG-19

Mathematical Functions

SQL> select ABS(-100) from dual; ABS(-100)

100

SQL> select ABS(-6) from dual;

ABS(-6)

6

SQL> select FLOOR(2345.78) FROM DUAL;

FLOOR(2345.78)

2345

```
SQL> SELECT GREATEST(23,67,90,123,78,50) FROM DUAL;
GREATEST(23,67,90,123,78,50)
----------------------------

                        123

SQL> SELECT LEAST(34, 21,67,11,89,9) FROM DUAL;

LEAST(34,21,67,11,89,9)
-----------------------
                      9

SQL> SELECT LENGTH('RAJESHWARI') FROM DUAL;
LENGTH('RAJESHWARI')
--------------------

                  10

SQL> SELECT LENGTH(17245637) FROM DUAL;
LENGTH(17245637)
----------------

                8

SQL> SELECT SQRT(16) FROM DUAL;
  SQRT(16)
----------

         4

SQL> SELECT SQRT(99) FROM DUAL;
  SQRT(99)
----------

9.94987437
SQL> SELECT POWER(2,4) FROM
DUAL; POWER(2,4)
----------

        16

SQL> SELECT POWER(2,10) FROM
DUAL; POWER(2,10)
-----------

       1024

SQL> SELECT power(2,10) FROM
DUAL; POWER(2,10)
-----------
```

1024

SQL> SELECT ROUND(5.86) FROM

DUAL; ROUND(5.86)

6

SQL> SELECT ROUND(1001.6) FROM
DUAL; ROUND(1001.6)

1002

SQL> SELECT ROUND(1001.3) FROM
DUAL; ROUND(1001.3)

1001

SQL> SELECT SIN(90) FROM
DUAL; SIN(90)

.893996664

SQL> SELECT COS(45) FROM
DUAL; COS(45)

.525321989

SQL> SELECT TAN(30) FROM DUAL;
TAN(30)

-6.4053312

SQL> SELECT TAN(90) FROM
DUAL; TAN(90)

-1.9952004

SQL> SELECT TAN(180) FROM DUAL;
TAN(180)

1.33869021

SQL> SELECT SIGN(-128) FROM
DUAL; SIGN(-128)

-1

```
SQL> SELECT SIGN(10) FROM
  DUAL; SIGN(10)
----------

         1

SQL> SELECT SIGN(0) FROM
   DUAL; SIGN(0)
----------

         0

SQL> SELECT LN(100) FROM
   DUAL; LN(100)
----------

4.60517019

SQL> SELECT LN(10) FROM
    DUAL; LN(10)
----------

2.30258509

SQL> SELECT LOG(10,100) FROM
DUAL; LOG(10,100)
----------

         2

SQL> SELECT LOG(100,10) FROM
DUAL; LOG(100,10)
----------

         .5

SQL> SELECT MOD(4,3) FROM

  DUAL; MOD(4,3)
----------

         1

SQL> SELECT MOD(4,2) FROM DUAL;

  MOD(4,2)
----------

         0
```

SQL> SELECT EXP(2) FROM DUAL;

EXP(2)

7.3890561

SQL> SELECT EXP(-2) FROM

DUAL; EXP(-2)

.135335283

SQL> SELECT EXP(0) FROM

DUAL; EXP(0)

1

Experiment 7

Aim: Perform the following:

a. Creating Tables (With and Without Constraints (Key/Domain)

b. Creating Tables (With Referential Integrity Constraints) For a given EMPLOYEE tables

EMPLOYEE	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
	John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
	Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
	Alicia	J	Zelaya	999887777	1968-07-19	3321 Castle, Spring, TX	F	25000	987654321	4
	Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
	Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
	Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
	Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
	James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	null	1

Perform the Following

1. Creating Views (With and Without Check Option),
2. Selecting from a View
3. Dropping Views,

SOLUTION:

```
SQL> CREATE TABLE EMPLOYEE ( SSN
     VARCHAR2 (20) PRIMARY KEY,
     FNAME VARCHAR2 (20),
     LNAME VARCHAR2 (20),
     ADDRESS VARCHAR2 (20),
     SEX CHAR (1),
     SALARY INTEGER,
     SUPERSSN REFERENCES EMPLOYEE (SSN), DNO
     REFERENCES DEPARTMENT (DNO));
```

SQL> DESC EMPLOYEE; Name

	Null?	Type
SSN	NOT NULL	VARCHAR2(20)
FNAME		VARCHAR2(20)
LNAME		VARCHAR2(20)
ADDRESS		VARCHAR2(20)
SEX		CHAR(1)
SALARY		NUMBER(38)
SUPERSSN		VARCHAR2(20)
DNO		NUMBER(38)

INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSECE01','JOHN','SCOTT','BANGALORE','M', 450000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSCSE01','JAMES','SMITH','BANGALORE','M', 500000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES ('RNSCSE02','HEARN','BAKER','BANGALORE','M', 700000); INSERT INTO EMPLOYEE
(SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSCSE03','EDWARD','SCOTT','MYSORE','M', 500000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES ('RNSCSE04','PAVAN','HEGDE','MANGALORE','M', 650000); INSERT INTO EMPLOYEE
(SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSCSE05','GIRISH','MALYA','MYSORE','M', 450000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSCSE06','NEHA','SN','BANGALORE','F', 800000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSACC01','AHANA','K','MANGALORE','F', 350000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSACC02','SANTHOSH','KUMAR','MANGALORE','M', 300000); INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSISE01','VEENA','M','MYSORE','M', 600000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY) VALUES
('RNSIT01','NAGESH','HR','BANGALORE','M', 500000);

Creating View

The query that defines the sales_staffview references only rows in department 5. Furthermore, the CHECK OPTION creates the view with the constraint (named sales_staff_cnst) that INSERT and UPDATE statements issued against the view cannot result in rows that the view cannot select.

1. Creating Views (With and Without Check Option)

```
SQL> CREATE VIEW sales_staff AS
  2  SELECT fname, ssn, dno
  3  FROM employee
  4  WHERE dno =5
  5  WITH CHECK OPTION CONSTRAINT sales_staff_cnst; View
```

created.

2. Selecting from a View

SQL> select * from sales_staff;

3. **Drop View**

SQL>DROP VIEW sales_staff;

Experiment 8

Aim: Given the table EMPLOYEE (EmpNo, Name, Salary, Designation, DeptID) write a cursor to select the five highest paid employees from the table.

Solution:
EMPLOYEE (EmpNo, Name, Salary, Designation, DeptID)

SOLUTION:

```
CREATE TABLE EMPLOYEE (EMPNO
INTEGER PRIMARY KEY, NAME
VARCHAR(20),
SALARY NUMBER(7,2),
DESIGNATION VARCHAR(10),
DEPTID INTEGER);
```

```
get e:/p8.sql; 1
declare  2   i
number:=0;
3 cursor ec is select empno,name,salary from employee order by gross_salary desc; 4 r ec%rowtype;
5 begin
6 open ec;
7 loop
8 exit when i=5; 9 fetch ec into r;
10 dbms_output.put_line(r.emp_no||' '||r.employee_name||' '||r.salary); 11 i:=i+1;
12 end loop;
13 close ec;
14* end; 15 .

SQL> /

1 rajesh 31000

2 paramesh 15000

3 pushpa 14000

4 vijaya 14000

5 keerthi 13000

PL/SQL procedure successfully completed.
```

Experiment 10

Aim : Write a Pl/SQL program to print integers from 1 to 10 by using PL/SQL FOR loop

SOLUTION:

PL/SQL Block

```
SET SERVEROUTPUT ON SIZE 1000000;
DECLARE
  n_times NUMBER := 10;
BEGIN
  FOR n_i IN 1..n_times LOOP
  DBMS_OUTPUT.PUT_LINE(n_i); END LOOP;
END;

/
```

Output Table

Value Added List of Experiments

1	Create a Database for registering a new user for the generation of electricity bills for a customer.
2	Design a Database for checking daily items sold by the Online Retail Application.
3	Design a database for the issue and return of items in Inventory Control Management.
4	Design a database to calculate fines on pending books in a Library Management System.
5	Design a database to check student's fee details in Student Database Management.
6	Create a database for providing leave to employees in a Payroll Management System.
7	Design a database to check transport availability in a Voice-based Transport Enquiry System.

8	SMS-based Remote Server Monitoring System.
9	Design a database to book doctor appointments in a Hospital Management System.
10	Create a database to check the total of blood availability for a Blood Donation Management System.

Name of the Course Lead: **Signature**