FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)®

HORMIS NAGAR, MOOKKANNOOR, ANGAMALY-683577

FOCUS ON EXCELLENCE

20MCA134 ADVANCED DBMS LAB

LABORATORY RECORD

Name: FAIKKA P.A

Branch: MASTER OF COMPUTER APPLICATIONS

Semester: 2 Batch: A Roll No:41

University Register Number: FIT24MCA-2041

JUNE 2025

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)®

HORMIS NAGAR, MOOKKANNOOR, ANGAMALY-683577

FOCUS ON EXCELLENCE

CERTIFICATE

This is to certify that this is a Bonafide record of the Practical work done and submitted to APJ Abdul Kalam Technological University in the partial fulfilment for the award of the Master Of Computer Applications by **FAIKKA P.A (FIT24MCA-2041)** *in the* **20MCA134 ADVANCED DBMS LAB** *of the Federal Institute of Science and Technology during the academic year 2024-2025.*

Signature of Staff in Charge	Signature of H O D
Ms. Anju L.	Dr. Deepa Mary Mathews

Date of University practical examination ……

Signature of Internal Examiner	Signature of External Examiner

CONTENT

EXPERIMENT 1

Creation of a database using DDL commands including integrity constraints.

QUESTION-1

1. Create a table called student with the following values and Write a SQL command which will show the entire **STUDENT** table.

REGD.NO	NAME	BRANCH
0001	Ram	CSE
0002	Hari	MECH
0003	Pradeep	EEE
0004	Deepak	ETC

Program Code:

SQL> create table student(reg_no integer primary key,sname varchar(20),mark integer check(mark between 0 and 100));

Table created.

SQL> insert into student values(41,'faikka',80);

1 row created.

SQL> insert into student values(40,'dhyan',50);

1 row created.

SQL> insert into student values(38,'delna',85);

1 row created.

SQL> insert into student values(39,'devu',75);

1 row created.

SQL> insert into student values(42,'nesrin',60);

1 row created.

SQL> select * from student;

Output:

REG_NO SNAME MARK

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

41 faikka	80
40 dhyan	50
38 delna	85
39 devu	75
42 nesrin	60

QUESTION-2

2. Create a table EMPLOYEE with following schema:

(Emp_no, E_name, E_address, E_ph_no, Dept_no, Dept_name,Job_id , Salary)

a) Add a new column; HIREDATE to the existing relation.

b) Change the datatype of JOB_ID from varchar to integer.

c) Change the name of column/field Emp_no to E_no.

d) Modify the column width of the Employee name field of emp table.

Program Code:

```
CREATE TABLE EMPLOYEE (Emp_no INT PRIMARY KEY,E_name VARCHAR(100),E_address VARCHAR(255),E_ph_no VARCHAR(15),Dept_no INT,Dept_name VARCHAR(100),Job_id VARCHAR(20),Salary DECIMAL(10, 2));

DESC EMPLOYEE

ALTER TABLE EMPLOYEE ADD HIREDATE DATE;

DESC EMPLOYEE

ALTER TABLE EMPLOYEE MODIFY Job_id INT;

DESC EMPLOYEE

ALTER TABLE EMPLOYEE RENAME COLUMN Emp_no TO E_no;

DESC EMPLOYEE

ALTER TABLE EMPLOYEE MODIFY E_name VARCHAR(150);

DESC EMPLOYEE
```

Output:

```
Table created.

TABLE EMPLOYEE
```

Column	Null?	Type
EMP_NO	NOT NULL	NUMBER
E_NAME	-	VARCHAR2(100)
E_ADDRESS	-	VARCHAR2(255)
E_PH_NO	-	VARCHAR2(15)
DEPT_NO	-	NUMBER
DEPT_NAME	-	VARCHAR2(100)
JOB_ID	-	VARCHAR2(20)
SALARY	-	NUMBER(10,2)

```
Table altered.

Table altered.

TABLE EMPLOYEE
```

Column	Null?	Type
E_NO	NOT NULL	NUMBER
E_NAME	-	VARCHAR2(150)
E_ADDRESS	-	VARCHAR2(255)
E_PH_NO	-	VARCHAR2(15)
DEPT_NO	-	NUMBER

DEPT_NAME	-	VARCHAR2(100)
JOB_ID	-	NUMBER
SALARY	-	NUMBER(10,2)
HIREDATE	-	DATE

QUESTION-3

3. Write a query in sql to create a table employee and department.

Employee(empno, ename, deptno, job, hiredate) Department(deptno,dname,loc

Include the following constraints on column of emp table.

a) to make the empno as primary key of the table

b) to ensure that the ename column does not contain NULL values and

c) the job column to have only UPPERCASE entries

d) put the current date as default date in hire date column in case data is not supplied for the column.

Include the following constraints on column of Department table

a) to make deptno as primary key.

b) to ensure dname,loc coloumns does not contain NULL values

c) Also enforce REFERENTIAL INTEGRITY, declare deptno field of dept table as primary key and deptno field of emp table as foreign key.

Program Code:

SQL> create table employ41(emp_no integer,emp_name varchar(50),dept_no integer,job varchar(50),hiredate date);

Table created.

a. SQL> alter table employ41 modify(emp_no integer primary key);

Table altered.

b. SQL> alter table employ41 modify(emp_name varchar(50)not null);

Table altered.

c. SQL> alter table employ41 modify(job varchar(50)check(job=UPPER(job)));

Table altered.

SQL> desc employ41

SQL> create table dept41(dept_no integer,dept_name varchar(50),loc varchar(20));

Table created.

Program Code:

a. SQL> alter table dept41 modify(dept_no integer primary key);

Table altered.

Program Code:

b. SQL> alter table dept41 modify(dept_name varchar(50)not null,loc varchar(20)not null);

Table altered.

Program Code:

c. SQL> alter table employ41 modify(dept_no integer references dept41(dept_no));

Table altered.

Output:

Name	Null?	Type
EMP_NO	NOT NULL	NUMBER(38)
EMP_NAME	NOT NULL	VARCHAR2(50)
DEPT_NO		NUMBER(38)
JOB		VARCHAR2(50)
HIREDATE		DATE

SQL> desc dept41

Name	Null?	Type
DEPT_NO	NOT NULL	NUMBER(38)
DEPT_NAME	NOT NULL	VARCHAR2(50)
LOC	NOT NULL	VARCHAR2(20)

SQL> desc employ41

Name	Null?	Type
EMP_NO	NOT NULL	NUMBER(38)
EMP_NAME	NOT NULL	VARCHAR2(50)
DEPT_NO		NUMBER(38)
JOB		VARCHAR2(50)
HIREDATE		DATE

EXPERIMENT 2

Implementation of DML commands.

QUESTION-4

Create a table EMPLOYEE with following schema:
(Emp_no, E_name, E_address, E_ph_no, Dept_no, Dept_name,Job_id , Salary)
Write SQL queries for following question:
1. Insert aleast 5 rows in the table.
2. Display all the information of EMP table.
3. Display the record of each employee who works in department D10.
4. Update the city of Emp_no-12 with current city as Nagpur.
5. Display the details of Employee who works in department MECH.
6. Delete the email_id of employee James.
7. Display the complete record of employees working in SALES Department.
8. Find out the employee id, names, salaries of all the employees
9. Find the names of the employees who have a salary greater than or equal to 4800

Program Code:

```
SQL>  create table emp41(e_no integer primary key,e_name varchar(20),e_address
varchar(15),e_ph_no varchar(20),dept_no varchar(15),dept_name varchar(10),job_id
integer,salary varchar(15));

Table created.

1. SQL> insert into emp41 values(102,'anju','palakal',9087652312,'B11','MECH',13,25000);

1 row created.

Commit complete.
SQL>  insert into emp41 values(110,'manju','kottakal',7809876543,'D10','SALES',20,35000);

1 row created.

Commit complete.
SQL> insert into emp41 values(51,'manju','kodvathan',8765903212,'D10','CIVIL',20,5000);

1 row created.

Commit complete.
SQL> insert into emp41 values(41,'james','mundeth',9023456121,'C22','CS',25,4500);

1 row created.

Commit complete.
SQL> insert into emp41 values(22,'dhyan','ullasam',9112345621,'C32','cs',18,3000);

1 row created.
```

Commit complete.

Program Code:

2. SQL> select * from emp41;

E_NO	E_NAME	E_ADDRESS	E_PH_NO	DEPT_NO	DEPT_NAME	JOB_ID	SALARY
102	anju	palakal	9087652312	B11	MECH	13	25000
110	manju	kottakal	7809876543	D10	SALES	20	35000
51	manju	kodvathan	8765903212	D10	CIVIL	20	5000
41	james	mundeth	9023456121	C22	CS	25	4500
22	dhyan	ullasam	9112345621	C32	cs	18	3000

Program Code:

3. select * from emp41 where dept_no='D10';

Output:

E_NO	E_NAME	E_ADDRESS	E_PH_NO	DEPT_NO	DEPT_NAME	JOB_ID	SALARY
110	manju	kottakal	7809876543	D10	SALES	20	35000
51	manju	kodvathan	8765903212	D10	CIVIL	20	5000

Program Code:

4. update emp41 SET city='nagpur' where e_name='james';

1 row updated.

Output:

E_NO	E_NAME	E_ADDRESS	E_PH_NO	DEPT_NO	DEPT_NAME	JOB_ID	SALARY	CITY
102	anju	palakal	9087652312	B11	MECH	13	25000	
110	manju	kottakal	7809876543	D10	SALES	20	35000	
51	manju	kodvathan	8765903212	D10	CIVIL	20	5000	
41	james	mundeth	9023456121	C22	CS	25	4500	nagpur
22	dhyan	ullasam	9112345621	C32	cs	18	3000	

Program Code:

5. select * from emp41 where dept_name='MECH';

Output:

E_NO	E_NAME	E_ADDRESS	E_PH_NO	DEPT_NO	DEPT_NAME	JOB_ID	SALARY	CITY
102	anju	palakal	9087652312	B11	MECH	13	25000	

Program Code:

6. SQL> alter table emp41 modify(email varchar(20));

Table altered.

SQL> update emp41 SET email='james@gmail.com' where e_name='james';

1 row updated.

Commit complete.

SQL> update emp41 SET email='' where e_name='james';

1 row updated.

select * from emp41 where e_name='james';

Output:

E_NO	E_NAME	E_ADDRESS	E_PH_NO	DEPT_NO	DEPT_NAME	JOB_ID	SALARY	CITY	EMAIL
41	james	mundeth	9023456121	C22	CS	25	4500	nagpur	

Program Code:

7. select * from emp41 where dept_name='SALES';

Output:

E_NO	E_NAME	E_ADDRESS	E_PH_NO	DEPT_NO	DEPT_NAME	JOB_ID	SALARY	CITY	EMAIL
110	manju	kottakal	7809876543	D10	SALES	20	35000		

Program Code:

8. select e_no,e_name,salary from emp41;

Output:

E_NO	E_NAME	SALARY
102	anju	25000

110 manju	35000
51 manju	5000
41 james	4500
22 dhyan	3000

Program Code:

9. select * from emp41 where salary>=4800;

Output:

E_NO	E_NAME	E_ADDRESS	E_PH_NO	DEPT_NO	DEPT_NAME	JOB_ID	SALARY	CITY	EMAIL
102	anju	palakal	9087652312	B11	MECH	13	25000		
110	manju	kottakal	7809876543	D10	SALES	20	35000		
51	manju	kodvathan	8765903212	D10	CIVIL	20	5000		

QUESTION-5

(Exercise on updating records in table)
Create Client_master with the following fields(ClientNO, Name, Address, City, State, bal_due)
(a) Insert five records
(b) Find the names of clients whose bal_due> 5000 .
(c) Change the bal_due of ClientNO " C123"to Rs. 5100
(d) Change the name of Client_master to Client12 .
(e) Display the bal_due heading as "BALANCE"

Program Code:

SQL> create table client_master41(client_no varchar(10) primary key,name varchar(15),address varchar(20),city varchar(15),state varchar(15),bal_due integer);

SQL> desc client_master41

Output:

Name	Null?	Type
CLIENT_NO	NOT NULL	VARCHAR2(10)
NAME		VARCHAR2(15)
ADDRESS		VARCHAR2(20)
CITY		VARCHAR2(15)
STATE		VARCHAR2(15)
BALANCE		NUMBER(38)

Program Code:

a) SQL> insert into client_master41 values('c123','anju','palakkal','angamaly','kerala',2500);

1 row created.

SQL> insert into client_master41
values('c451','dhyan','sadhgamaya','ambalamkunnu','kerala',5500);

1 row created.

SQL> insert into client_master41
values('c342','faikka','kotayilkudy','mudickal','kerala',5300);

1 row created.

SQL> insert into client_master41 values('c672','maya','mundeth','perumbavoor','kerala',6000);

1 row created.

SQL> insert into client_master41 values('c523','delna','meledath','thrissur','kerala',1000);

1 row created.

SQL> set linesize 200

SQL> select * from client_master41;

Output:

CLIENT_NO	NAME	ADDRESS	CITY	STATE	BAL_DUE
c123	anju	palakkal	angamaly	kerala	2500
c451	dhyan	sadhgamaya	ambalamkunnu	kerala	5500
c342	faikka	kotayilkudy	mudickal	kerala	5300
c672	maya	mundeth	perumbavoor	kerala	6000
c523	delna	meledath	thrissur	kerala	1000

Program Code:

b) SQL> select name from client_master41 where bal_due>5000;

Output:

```
SQL> select name from client_master41 where bal_due>5000;

NAME
---------------
dhyan
faikka
maya
```

Program Code:

c) SQL> update client_master41 set bal_due=5100 where client_no='c123';

1 row updated.

SQL> select * from client_master41 where client_no='c123';

Output:

CLIENT_NO	NAME	ADDRESS	CITY	STATE	BAL_DUE
c123	anju	palakkal	angamaly	kerala	5100

Program Code:

d) SQL> alter table client_master41 rename to client_12;

Table altered

Output:

```
SQL> desc client_12
 Name                                      Null?    Type
 ----------------------------------------- -------- ------------------------------------
 CLIENT_NO                                 NOT NULL VARCHAR2(10)
 NAME                                               VARCHAR2(15)
 ADDRESS                                            VARCHAR2(20)
 CITY                                               VARCHAR2(15)
 STATE                                              VARCHAR2(15)
 BALANCE                                            NUMBER(38)
```

Program Code:

e) SQL> alter table client_12 rename column bal_due to balance;

Table altered.

SQL> select * from client_12;

Output:

CLIENT_NO	NAME	ADDRESS	CITY	STATE	BALANCE
c123	anju	palakkal	angamaly	kerala	5100
c451	dhyan	sadhgamaya	ambalamkunnu	kerala	5500
c342	faikka	kotayilkudy	mudickal	kerala	5300
c672	maya	mundeth	perumbavoor	kerala	6000
c523	delna	meledath	thrissur	kerala	1000

QUESTION-6

(Rollback and Commit commands)
Create Teacher table with the following fields(Name, DeptNo, Date of joining, DeptName, Location,Salary)

(a) Insert five records
(b) Give Increment of 25% salary for Mathematics Department .
(c) Perform Rollback command
(d) Give Increment of 15% salary for Commerce Department
(e) Perform commit command

Program Code:

create table teacher41(name varchar(15),dept_no integer,date_of_joining date,dept_name varchar(15),address varchar(15),salary integer);

Table created.

SQL> insert into teacher41 values('Anju',12,'2-aug-25','cs','meprathupady',25000);

1 row created.

SQL> insert into teacher41 values('Richu',24,'3-june-2018','Mathematics','plathoor',2000);

1 row created.

SQL> insert into teacher41 values('Shahana',18,'12-may-2015','Commerce','meledath',15000);

1 row created.

SQL> insert into teacher41 values('Deepa',12,'5-aug-2010','cs','chenthara',30000);

1 row created.

SQL> insert into teacher41 values('Manju',24,'9-april-2015','Mathematics','mundeth',5000);

1 row created.

Program Code:

a) SQL> select * from teacher41;

Output:

NAME	DEPT_NO	DATE_OF_J	DEPT_NAME	ADDRESS	SALARY
Anju	12	02-AUG-25	cs	meprathupady	25000
Richu	24	03-JUN-18	Mathematics	plathoor	2000
Shahana	18	12-MAY-15	Commerce	meledath	15000
Deepa	12	05-AUG-10	cs	chenthara	30000
Manju	24	09-APR-15	Mathematics	mundeth	5000

b) SQL> update teacher41 set salary=salary+(0.25*salary) where dept_name='Mathematics';

2 rows updated.

SQL> select * from teacher41 where dept_name='Mathematics';

Output:

NAME	DEPT_NO	DATE_OF_J	DEPT_NAME	ADDRESS	SALARY
Richu	24	03-JUN-18	Mathematics	plathoor	2500
Manju	24	09-APR-15	Mathematics	mundeth	6250

b) SQL> rollback;

Output:

Rollback complete.

SQL> update teacher41 set salary=salary+(0.15*salary) where dept_name='Commerce';

1 row updated.

c) SQL> select * from teacher41 where dept_name='Commerce';

Output:

NAME	DEPT_NO	DATE_OF_J	DEPT_NAME	ADDRESS	SALARY
Shahana	18	12-MAY-15	Commerce	meledath	17250

e) SQL> commit;

Commit complete.

QUESTION-7

(Exercise on order by and group by clauses)
Create Sales table with the following fields(Sales No, Salesname, Branch, Salesamount, DOB)
(a) Insert five records
(b) Calculate total salesamount in each branch
(c) Calculate average salesamount in each branch .
(d) Display all the salesmen, DOB who are born in the month of December as day in character
format i.e. 21-Dec-09
(e) Display the name and DOB of salesman in alphabetical order of the month.

Program Code:

SQL> create table sales41(sales_no integer primary key,sales_name varchar(20),branch varchar(20),sales_amnt integer,dob date);

Table created.

a) SQL> insert into sales41 values(110,'abc','sales',10000,'12-jun-2010');

1 row created.

Commit complete.
SQL> insert into sales41 values(132,'def','production',15000,'8-jan-2015');

1 row created.

Commit complete.
SQL> insert into sales41 values(96,'ghi','billing',30000,'11-may-2020');

1 row created.

Commit complete.
SQL> insert into sales41 values(58,'jkl','stitching',20000,'22-nov-2022');

1 row created.

Commit complete.
SQL> insert into sales41 values(125,'xyx','sales',25000,'1-dec-2024');

1 row created.

Commit complete.

SQL> select * from sales41;

Output:

SALES_NO	SALES_NAME	BRANCH	SALES_AMNT	DOB
110	abc	sales	10000	12-JUN-10
132	def	production	15000	08-JAN-15
96	ghi	billing	30000	11-MAY-20
58	jkl	stitching	20000	22-NOV-22
125	xyx	sales	25000	01-DEC-24

b) SQL> select branch, sum(sales_amnt) as totsales_amnt from sales41 group by branch;

output:

BRANCH	TOTSALES_AMNT
sales	35000
stitching	20000
billing	30000
production	15000

c) SQL> select branch, avg(sales_amnt) as totsales_amnt from sales41 group by branch;

output:

BRANCH	TOTSALES_AMNT
sales	17500
stitching	20000
billing	30000
production	15000

EXPERIMENT -3

Implementation of different types of operators in SQL.

QUESTION-8

Create an Emp table with the following fields:(EmpNo, EmpName, Job, Basic, DA, HRA,PF, GrossPay, NetPay) Hint:(PF is calculated as 10% of basic salary) (Calculate DA as 30% of Basic and HRA as 40% of Basic)
(a) Insert Five Records and calculate GrossPay and NetPay.
(b) Display the employees whose Basic is lowest in each department .
(c) If NetPay is less than <Rs. 10,000 add Rs. 1200 as special allowances .
(d) Display the employees whose GrossPay lies between 10,000 & 20,000
(e) Display all the employees who earn maximum salary .

Program Code:

```
SQL> create table emp41(emp_no integer primary key,ename varchar(10),job
varchar(10),basic integer,da varchar(10),hra varchar(10),pf varchar(10),grosspay
varchar(10),netpay varchar(10));

Table created.

SQL> desc emp41
```

Output:

```
 Name                                      Null?    Type
 ----------------------------------------- -------- ------------------------------
 EMP_NO                                    NOT NULL NUMBER(38)
 ENAME                                              VARCHAR2(10)
 JOB                                                VARCHAR2(10)
 BASIC                                              NUMBER(38)
 DA                                                 VARCHAR2(10)
 HRA                                                VARCHAR2(10)
 PF                                                 VARCHAR2(10)
 GROSSPAY                                           VARCHAR2(10)
 NETPAY                                             VARCHAR2(10)
```

a) SQL> insert into emp41(emp_no,ename,job,basic) values(101,'arun','manager',15000);

1 row created.

SQL> insert into emp41(emp_no,ename,job,basic) values(110,'maya','accountant',10000);

1 row created.

SQL> insert into emp41(emp_no,ename,job,basic) values(105,'dhyan','developer',20000);

1 row created.

```
SQL> insert into emp41(emp_no,ename,job,basic) values(125,'delna','developer',25000);

1 row created.

SQL> insert into emp41(emp_no,ename,job,basic) values(102,'devu','tester',30000);

1 row created.

SQL> select * from emp41;
```

Output:

```
 EMP_NO ENAME JOB         BASIC    DA HRA      PF      GROSSPAY       NETPAY
---------- ---------- ---------- ---------- ---------- ---------- ----------
       101 arun       manager    15000
       110 maya       accountant 10000
       105 dhyan      developer  20000
       125 delna      developer  25000
       102 devu       tester     30000
```

```
SQL> update emp41 set da=.03*basic,hra=.04*basic,pf=.01*basic;

5 rows updated.

SQL> update emp41 set grosspay=basic+da+hra;

5 rows updated.

SQL> update emp41 set netpay=grosspay-pf;

5 rows updated.

SQL> select * from emp41;
```

```
 EMP_NO ENAME  JOB       BASIC    DA       HRA   PF       GROSSPAY NETPAY
---------- ---------- ---------- ---------- ---------- ---------- ---------- ----------
       101 arun       manager    15000   450       600       150      16050      15900
       110 maya       accountant 10000   300       400       100      10700      10600
       105 dhyan      developer  20000   600       800       200      21400      21200
       125 delna      developer  25000   750      1000       250      26750      26500
       102 devu       tester     30000   900      1200       300      32100      31800
```

b) SQL> select job,min(basic) as lowest_basic from emp41 group by job;

```
JOB        LOWEST_BASIC
---------- ------------
tester     30000
manager    15000
developer  20000
accountant 3000
```

c) SQL> update emp41 set netpay=netpay+1200 where netpay<10000;

1 row updated.

d) SQL> select ename from emp41 where grosspay between 10000 and 20000;

Output:

ENAME

Arun

Program Code:

e) SQL> select max(basic) as highest_basic from emp41;

Output:

HIGHEST_BASIC

30000

EXPERIMENT 4:

Implementation of different types of functions with suitable examples.

QUESTION-9

Create a table EMPLOYEE with following schema:
(Emp_no, E_name, E_address, E_ph_no, Dept_no, Dept_name,Job_id, Designation , Salary)
Write SQL statements for the following query.

1. List the E_no, E_name, Salary of all employees working for MANAGER.
2. Display all the details of the employee whose salary is more than the Sal of any IT PROFF..
3. List the employees in the ascending order of Designations of those joined after 1981.
4. List the employees along with their Experience and Daily Salary.
5. List the employees who are either 'CLERK' or 'ANALYST' .
6. List the employees who joined on 1-MAY-81, 3-DEC-81, 17-DEC-81,19-JAN-80 .
7. List the employees who are working for the Deptno 10 or20.
8. List the Enames those are starting with 'S' .
9. Dislay the name as well as the first five characters of name(s) starting with 'H'
10. List all the emps except 'PRESIDENT' & 'MANAGR" in asc order of Salaries.

Program Code:

```
SQL> create table EMPLOYEEE41(Emp_no integer primary key,E_name
varchar(15),E_address varchar(20),E_ph_no number(10,0),Dept_no integer,Dept_name
varchar(10),Job_id varchar(5),Designation varchar(10),Salary number(8,0));

SQL> insert into EMPLOYEEE41 values(1, 'Sooraj', 'Nandanam', '859058296', 10, 'HR',
'MANAGER', 'HR Manager', 50000, '15-MAR-1985');

SQL> insert into EMPLOYEEE41 values(2, 'Abhay', 'Pallath', '9876543210', 20, 'IT', 'IT
PROFF', 'SEngineer', 55000, '01-MAY-1981');

SQL> insert into EMPLOYEEE41 values(3, 'Harshan', 'Pine Road', '9605080027', 20, 'IT', 'IT
PROFF', 'IT Support', 40000,'3-DEC-1981');

SQL> insert into EMPLOYEEE41 values(4, 'Sona', 'Maple Drive', '6259847621', 30,
'Finance', 'CLERK', 'AccClerk', 35000, '17-DEC-1981');

SQL> insert into EMPLOYEEE41 values(5, 'Hardik', 'Boulevard', '8547210352', 10, 'HR',
'CLERK', 'HR Assist', 30000, '19-JAN-1980');

SQL> insert into EMPLOYEEE41 values(6, 'Micheal', 'Cedar Lane', '8605080027', 40,
'Sales', 'ANALYST', 'SaleAnalys', 55000,'15-MAR-2000');

SQL> insert into EMPLOYEEE41 values(8, 'Alice', 'Willow Way', '8596547896', 10, 'HR',
'PRESIDENT', 'CEO', 100000, '3-APR-1989');
```

1. SQL> select Emp_no, E_name, Salary from EMPLOYEE41 where Designation = 'MANAGER';

Output:

EMP_NO	E_NAME	SALARY
1	Sooraj	50000

Program Code:

2. SQL>select * from EMPLOYEE41 where Salary >(select MAX(Salary) from EMPLOYEE where Dept_name='IT');

Output:

EMP_NO	E_NAME	E_ADDRESS	E_PH NO	DEPT_NO	DEPT_NAME	JOB_ID	DESIGNATION	SALARY	JOIN_DATE
8	Alice	Willow Way	859654 7896	10	HR	PRESIDENT	CEO	100000	03-APR-89

Program Code:

3. SQL>select E_name,Designation,Hiredate from EMPLOYEE41 where Hiredate > TO_DATE('31-DEC-1981','DD-MON_YYYY') order by Designation asc;

Output:

E_NAME	DESIGNATION	HIREDATE
Micheal	ANALYST	15-MAR-2000
Alice	PRESIDENT	03-MAR-1989

Program Code:

4. SQL>select E_name,ROUND(MONTHS_BETWEEEN(SYSDATE,Hiredate)/12,1) as Experience,Round(Salary/30,2) as Daily_Salary from EMPLOYEE41;

Output:

E_NAME	EXPERIENCE YEARS	DAILY SALARY
Sooraj	40.1	1666.67
Abhay	43.0	1833.33
Harshan	43.4	1333.33
Sona	43.3	1166.67
Hardik	45.3	1000.00
Micheal	25.1	1833.33
Alice	36.1	3333.33

5. SQL>select * from EMPLOYEE41 where Designation in ('CLERK','ANALYST');

Output:

EMP_NO	E_NAME	DEPT_NAME	DESIGNATION
4	Sona	Finance	CLERK
5	Hardik	HR	CLERK
6	Micheal	Sales	ANALYST

Program Code:

6. SQL>select from EMPLOYEE41 where Hiredate in(TO_DATE('01-MAY-81','DD-MON-YY'),TO_DATE('03-DEC-81', 'DD-MON-YY'),TO_DATE('17-DEC-81', 'DD-MON-YY'),TO_DATE('19-JAN-80', 'DD-MON-YY');

Output:

E_NAME	HIREDATE
Abhay	01-MAY-81
Harshan	03-DEC-81
Sona	17-DEC-81
Hardik	19-JAN-80

Program Code:

7. SQL>select * from EMPLOYEE41 where Dept_no in(10,20);

Output:

E_NAME	DEPT_NO
Sooraj	10
Abhay	20
Harshan	20
Hardik	10
Alice	10

Program Code:

8. SQL>select E_name from EMPLOYEE41 where E_name like 'S%';

Output:

E_NAME
Sooraj
Sona

Program Code:

9. SQL>select E_name,SUBSTR(E_name,1,5) as First5 from EMPLOYEE41;

Output:

E_NAME FIRST5
Harshan Harsh
Hardik Hardi

Program Code:

10. SQL>select * from EMPLOYEE41 where Designation NOT IN('PRESIDENT','MANAGER') order by Salary asc;

Output:

E_NAME DESIGNATION SALARY
---------- ---------- ----------
Hardik CLERK 30000
Sona CLERK 35000
Harshan IT Support 40000
Abhay SEngineer 55000
Micheal ANALYST 55000

QUESTION-10

10. Consider Employee table

EMPNO	EMP_NAME	DEPT	SALARY	DOJ	BRANCH
E101	Amit	roduction	45000	12-Mar-00	Bangalore
E102	Amit	HR	70000	03-Jul-02	Bangalore
E103	sunita	anagemer	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;

Program Code:

SQL> create table employe41(e_no integer primary key,e_name varchar(10),dept varchar(10),salary integer,doj varchar(10),branch varchar(10));

Table created.

Program Code:

1. SQL>select * from EMPLOYEE;

Output:

EMPNO	EMP_NAME	DEPT	SALARY	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

Program Code:

2. SQL> select EMPNO, SALARY from EMPLOYEE;

Output:

EMPNO	SALARY
E101	45000
E102	70000
E103	120000
E105	67000
E106	145000

Program Code:

3. SQL>select AVG(SALARY) as AVG_SALARY from EMPLOYEE;

Output:

AVG_SALARY
89400

Program Code:

4. SQL> select COUNT(*) as TOTAL_EMPLOYEES from EMPLOYEE;

Output:

```
TOTAL_EMPLOYEES
---------------
5
```

Program Code:

5. SQL> select COUNT(DISTINCT EMP_NAME) as DISTINCT_NAMES from EMPLOYEE;

Output:

```
DISTINCT_NAMES
--------------
3
```

Program Code:

6. SQL>select EMP_NAME, COUNT(*) as COUNT, SUM(SALARY) asTOTAL_SALARY from EMPLOYEE group by EMP_NAME;

Output:

```
EMP_NAME  COUNT    TOTAL_SALARY
--------  -------  ------------
Amit      2        115000
Sunita    2        187000
Mahesh    1        145000
```

Program Code:

7. SQL>select * from EMPLOYEE where SALARY > 120000;

Output:

```
EMPNO  EMP_NAME  DEPT   SALARY  DOJ        BRANCH
-----  --------  -----  ------  ---------  ------
106    mahesh    Civil  145000  20-Sep-03  Mumbai
```

Program Code:

8. SQL> select EMP_NAME from EMPLOYEE order by EMP_NAME desc;

Output:

```
EMP_NAME
----------
sunita
sunita
mahesh
Amit
Amit
```

Program Code:

9. SQL> select * from EMPLOYEE where EMP_NAME = 'Amit' and SALARY > 50000;

Output:

EMPNO	EMP_NAME	DEPT	SALARY	DOJ	BRANCH
E102	Amit	HR	70000	03-Jul-02	Bangalore

QUESTION-11

11. Create a table called Employee with the following structure.

Name	Type
Empno	Number
Ename	Varchar2(20)
Job	Varchar2(20)
Mgr	Number
Sal	Number

a) Display lowest paid employee details under each department.
b) Display number of employees working in each department and their department number.
c) Using built-in functions, display number of employees working in each department and their department name from dept table. Insert deptname to dept table and insert deptname for each row, do the required thing specified above.
d) List all employees which start with either B or C.
e) Display only these ename of employees where the maximum salary is greater than or equal to 5000.
f) Calculate the average salary for each different job.
g) Show the average salary of each job excluding manager.
h) Show the average salary for all departments employing more than three people.
f) How many days between day of birth to current date.
i) List all employee names, salary and 15% rise in salary.

j) Display lowest paid emp details under each manager
k) Display the average monthly salary bill for each deptno.
l) Show the average salary for all departments employing more than two people.
m) By using the group by clause, display the eid who belongs to deptno 05 along with average salary.
n) Count the number of employees in department 20
o) Find the minimum salary earned by clerk.
p) Find minimum, maximum, average salary of all employees.
q) List the minimum and maximum salaries for each job type.
r) List the employee names in descending order.
s) List the employee id, names in ascending order by empid.

Program Code:

```
SQL> create table Employee Empno  number primary key, Ename varchar(20), Job
varchar(20), Mgr number, Sal number);
SQL>INSERT INTO Employee VALUES (101, 'Biju',   'Clerk',      201, 3000);
SQL>INSERT INTO Employee VALUES (102, 'Celine', 'Manager',    200, 8000);
SQL>INSERT INTO Employee VALUES (103, 'Rahul',  'Analyst',    201, 6000);
SQL>INSERT INTO Employee VALUES (104, 'Renu',   'Clerk',      202, 2800);
SQL>INSERT INTO Employee VALUES (105, 'Chetan', 'Manager',    200, 9000);
SQL>INSERT INTO Employee VALUES (106, 'Sneha',  'Developer', 201, 6500);
```

Program Code:

a) SQL>select Job, MIN(Sal) as Min_Salary from Employee group by Job;
Output:

```
Job          Min_Salary
--------     ----------
Clerk        2800
Manager      8000
Analyst      6000
Developer    6500
```

Program Code:

b) SQL> select Mgr, COUNT(*) as No_of_Employees from Employee group by Mgr;

Output:

```
Mgr     No_of_Employees
-------  ---------------
200      2
201      3
202      1
```

Program Code:

c) SQL> select * from Employee where Ename like 'B%' OR Ename like 'C%';

Output:

Empno	Ename	Job	Mgr	Sal
101	Biju	Clerk	201	3000
102	Celine	Manager	200	8000
105	Chetan	Manager	200	9000

Program Code:

d) SQL> select Ename from Employee where Sal >= 5000;

Output:

Ename
Celine
Rahul
Chetan
Sneha

Program Code:

e) SQL> select Job, AVG(Sal) as Avg_Salary from Employee group by Job;

Output:

Job	Avg_Salary
Clerk	2900
Manager	8500
Analyst	6000
Developer	6500

Program Code:

f) SQL> select Job, AVG(Sal) as Avg_Salary from Employee where Job != 'Manager' group by Job;

Outtput:

Job	Avg_Salar
Clerk	2900
Analyst	6000
Developer	6500

Program Code:

g) SQL> select Job, AVG(Sal) as AvgSal from Employee group by Job having COUNT(*) > 1;

Output:

Job	AvgSal
Clerk	2900
Manager	8500

Program Code:

h) SQL> select Ename, Sal, Sal*1.15 as New_Salary from Employee;

Output:

Ename	Sal	New_Salary
Biju	3000	3450
Celine	8000	9200
Rahul	6000	6900
Renu	2800	3220
Chetan	9000	10350

Program Code:

i) SQL> select Mgr, MIN(Sal) as Min_Salary from Employee group by Mgr;

Output:

Mgr	Min_Salary
200	8000
201	3000
202	2800

Program Code:

j) SQL> select Mgr, AVG(Sal) as Avg_Sal from Employee group by Mgr;

Output:

Mgr	Avg_Sal
200	8500
201	5166.67
202	2800

Program Code:

k) SQL> select Mgr,AVG(Sal) as AvgSal from Employee group by Mgr having count(*) > 2;

Output:

Mgr	Avg_Sal
201	5166.67

Program Code:

l) SQL> select Mgr, AVG(Sal) as Avg_Sal from Employee where Mgr = 201 group by Mgr;

Output:

Mgr	Avg_Sal
201	5166.67

Program Code:

m) SQL> select COUNT(*) as Emp_Count from Employee where Mgr = 200;

Output:

Emp_Count
2

Program Code:

o) SQL> select MIN(Sal) as Min_Salary from Employee where Job = 'Clerk';

Output:

Min_Salary
2800

Program Code:

p) SQL> select MIN(Sal), MAX(Sal), AVG(Sal) from Employee;

Output:

MIN(Sal)	MAX(Sal)	AVG(Sal)
2800	9000	5883.33

Program Code:

q) SQL> select Job, MIN(Sal), MAX(Sal) from Employee group by Job;

Output:

Job	MIN(Sal)	MAX(Sal)
Clerk	2800	3000
Manager	8000	9000
Analyst	6000	6000
Developer	6500	6500

Progam Code:

r) SQL> select Ename from Employee order by Ename desc;

Output:

Ename
Sneha
Renu
Rahul
Chetan
Celine
Biju

Program Code:

s) SQL> select Empno, Ename from Employee order by Empno;

Output:

Empno	Ename
101	Biju
102	Celine
103	Rahul
104	Renu
105	Chetan
106	Sneha

EXPERIMENT- 5

Implementation of different types of functions with suitable examples

QUESTION-12

1112.Create a table EMPLOYEE with following schema:
(Emp_no, E_name, E_address, E_ph_no, Dept_no, Dept_name,Job_id , Salary)

1. Display all the dept numbers available with the dept and emp tables avoiding duplicates.
2. Display all the dept numbers available with the dept and emp tables.
3. Display all the dept numbers available in emp and not in dept tables and vice versa.

Program Code:

```
SQL>create table EMPLOYEE (Emp_no number primary key, E_name varchar(20),E_address varchar(30), E_ph_no number(10),Dept_no number,Dept_name varchar(20),Job_id varchar(10),Salary number(10,2));

SQL>insert into EMPLOYEE values (101, 'Gautham', 'Kottayam', 859058296, 10, 'HR', 'MGR', 50000);
SQL>insert into EMPLOYEE values(102, 'Abhay', 'Kochi', 9876543210, 20, 'IT', 'DEV', 40000);
SQL>insert into EMPLOYEE values (103, 'Celine', 'Thrissur', 9605080027, 30, 'Finance', 'CLK', 30000);
SQL>insert into EMPLOYEE values (104, 'Arjun', 'Alappuzha', 8547210352, 40, 'Sales', 'ANL', 45000);

SQL> create table DEPT (Dept_no number primary key, Dept_name varchar(20));

SQL> insert into DEPT values (10, 'HR');

SQL>insert into DEPT values (20, 'IT');

SQL>insert into DEPT values (50, 'Marketing');

SQL>insert into DEPT values (60, 'Support');
```

Program Code:

1. SQL> select Dept_no from EMPLOYEE UNION select Dept_no from DEPT;

Output:

```
DEPT_NO
---------
10
20
30
40
50
60
```

Program Code:

2. SQL> select Dept_no from EMPLOYEE UNION ALL select Dept_no from DEPT;

Output:

DEPT_NO

10
20
30
40
10
20
50
60

Program Code:

3. SQL> select Dept_no from EMPLOYEE MINUS select Dept_no from DEPT;
SQL> select Dept_no from DEPT MINUS select Dept_no from EMPLOYEE;

Output:

DEPT_NO

30
40

DEPT_NO

50
60

EXPERIMENT-6

Implementation of Join,Views,Set operations.

QUESTION-13

1213.Consider the following schema: Sailors (sid, sname, rating, age) Boats (bid, bname, color) Reserves (sid, bid, day(date))

a) Find all the information of sailors who have reserved boat number 101.
b) Find the name of boat reserved by Bob.
c) Find the names of sailors who have reserved a red boat, and list in order of age.
d) Find the names of sailors who have reserved at least one boat.
e) Find the ids and names of sailors who have reserved two different boats on the same day.
f) Find the ids of sailors who have reserved a red boat or a green boat.
g) Find the name and the age of the youngest sailor.
h) Count the number of different sailor names.
i) Find the average age of sailors for each rating level.
j) Find the average age of sailors for each rating level that has at least two sailors.

Program Code:

```
SQL>create table Sailors (sid number primary key,Sname varchar(20),rating number,age number);
SQL>insert into Sailors values (1, 'Bob', 5, 25);
SQL>insert into Sailors values (2, 'Alice', 3, 22);
SQL>insert into Sailors values (3, 'Charlie', 4, 27);
SQL>insert into Sailors values (4, 'David', 2, 24);
SQL>insert into Sailors values (5, 'Eve', 5, 20);

SQL> create table Boats (bid number primary key,bname varchar(20),color varchar(10));
SQL>insert into Boats values (101, 'Boat1', 'Red');
SQL>insert into Boats values (102, 'Boat2', 'Blue');
SQL>insert into Boats values (103, 'Boat3', 'Green');
SQL>insert into Boats values (104, 'Boat4', 'Red');

SQL> create table Reserves (sid number,bid number day date primary key (sid, bid, day),foreign key (sid) references Sailors(sid),foreign key (bid) references Boats(bid));
SQL> insert into Reserves values (1, 101, TO_DATE('2025-04-17', 'YYYY-MM-DD'));
SQL>insert into Reserves values (2, 102, TO_DATE('2025-04-17', 'YYYY-MM-DD'));
SQL>insert into Reserves values (3, 103, TO_DATE('2025-04-16', 'YYYY-MM-DD'));
SQL>insert into Reserves values (4, 101, TO_DATE('2025-04-17', 'YYYY-MM-DD'));
SQL>insert into Reserves values (5, 104, TO_DATE('2025-04-16', 'YYYY-MM-DD'));
SQL>insert into Reserves values (1, 103, TO_DATE('2025-04-15', 'YYYY-MM-DD'));
SQL>insert into Reserves values (2, 101, TO_DATE('2025-04-16', 'YYYY-MM-DD'));
```

Program Code:

a) SQL> select S.* from Sailors S join Reserves R on S.sid = R.sid where R.bid = 101;

Output:

sid	sname	rating	age
1	Bob	5	25
4	David	2	24

Program Code:

b) SQL> select B.bname from Boats B join Reserves R on B.bid = R.bid join Sailors S on R.sid = S.sid where S.sname = 'Bob';

Output:

Bname

Boat1
Boat3

Program Code:
c) SQL> select S.sname from Sailors S join Reserves R on S.sid = R.sid join Boats B on R.bid = B.bid where B.color = 'Red' order by S.age;

Output:

Sname

Eve
David
Bob

Program Code:

d) SQL> select DISTINCT S.sname from Sailors S join Reserves R on S.sid = R.sid;

Output:

Sname

Bob
Alice
Charlie
David
Eve

Program Code:

e) SQL> select R1.sid, S.sname from Reserves R1, Reserves R2 join Sailors S on R1.sid = S.sid where R1.sid = R2.sid and R1.bid != R2.bid and R1.day = R2.day;

Output:

sid	sname
1	Bob
2	Alice

Program Code:

f) SQL> select DISTINCT R.sid from Reserves R join Boats B on R.bid = B.bid where B.color in ('Red', 'Green');

Output:

Sid
1
2
3
4

Program Code:

g) SQL> select S.sname, S.age from Sailors S where S.age = (select MIN(age) from Sailors);

Output:

sname	age
Eve	20

Program Code:

h) SQL> select COUNT(DISTINCT sname) as distinct_sailor_names from Sailors;

Output:

distinct_sailor_names
5

Program Code:

i) SQL> select rating, AVG(age) as average_age from Sailors group by rating;

Output:

rating	average_age
5	22.5
3	22

4 27
2 24

Program Code:

j) SQL> select rating, AVG(age) as average_age from Sailors group by rating having COUNT(sid) >= 2;

Output:

rating	average_age
5	22.5

QUESTION-14

Create a view named EmployeeDetails that displays the employee ID, name, and salary from the Employees table.

Original Table: Employees (employee_id, name, salary, department_id)

Program Code:

SQL>create table Employees (employee_id number primary key,name varchar(50),salary number(10, 2),department_id number);
SQL>insert into Employees values (101, 'faikka', 50000, 10);
SQL>insert into Employees values (102, 'delna', 40000, 20);
SQL>insert into Employees values (103, 'devu', 45000, 30);
SQL>insert into Employees values (104, 'Arjun', 60000, 10);

SQL>create view EmployeeDetails as select employee_id, name, salary from Employees;

SQL> select * from EmployeeDetails;

Output:

employee_id	name	salary
101	faikka	50000
102	delna	40000
103	devu	45000
104	Arjun	60000

QUESTION-15

Write a SQL query to create a view called CustomerContacts that combines the

customer's first name, last name, and email address from the Customers table.

Original Table: Customers (customer_id, first_name, last_name, email)

Program Code:

SQL> create table Customers (customer_id number primary key,first_name varchar(30),last_name varchar(30),email varchar(50));

SQL> insert into Customers values (1, 'Alice', 'Mathew', 'alice@example.com');
SQL>insert into Customers values (2, 'Babu', 'Thomas', 'babu@example.com');
SQL>insert into Customers values (3, 'Sona', 'Joseph', 'sona@example.com');
SQL>insert into Customers values (4, 'David', 'Paul', 'david@example.com');
SQL> create view CustomerContacts as select first_name, last_name, email from Customers;

SQL> select * from CustomerContacts;

Output:

first_name	last_name	Email
Alice	Mathew	alice@example.com
Babu	Thomas	babu@example.com
Sona	Joseph	sona@example.com

QUESTION-16

Create a view named EmployeeSalaries that shows the employee ID, name, and salary along with the salary grade from the Employees and SalaryGrades tables.

Original Tables:
Employees (employee_id, name, salary_grade_id), SalaryGrades (salary_grade_id, min_salary, max_salary)

Program Code:

SQL>create table Employees (employee_id number primary key,name varchar(30),salary_grade_id number);

SQL>create table SalaryGrades (salary_grade_id number primary key,min_salary number,max_salary number);

SQL> insert into Employees values (101, 'faikka', 1);
SQL>insert into Employees values (102, 'delna', 2);
SQL>insert into Employees values (103, 'dhyan', 3);

SQL>insert into SalaryGrades values (1, 30000, 40000);
SQL>insert into SalaryGrades values (2, 40001, 50000);

SQL>insert into SalaryGrades values (3, 50001, 60000);

SQL>create view EmployeeSalaries as select E.employee_id,E.name,S.min_salary,S.max_salary,E.salary_grade_id from Employees E join SalaryGrades S on E.salary_grade_id = S.salary_grade_id;

SQL> select * from EmployeeSalaries;

Output:

employee_id	name	min_salary	max_salary	salary_grade_id
101	faikka	30000	40000	1
102	delna	40001	50000	2
103	dhyan	50001	60000	3

QUESTION-17

Create tables Employees (employee_id , name) ,Managers (manager_id, name)

a) Write a SQL query to retrieve the names of all employees and managers, ensuring that duplicate names are removed.
b) Create a query to find the common names between employees and managers.
c) Write a query to find the names of employees who are not managers.
d) Write a query to find the distinct names of all employees and managers, along with their resp/ective roles (employee/manager).

Program Code:

SQL> create table Employees (employee_id number primary key,name varchar(30));

SQL>create table Managers (manager_id number primary key,name varchar(30));

SQL>insert into Employees values (1, 'delna');
SQL>insert into Employees values (2, 'devu');
SQL>insert into Employees values (3, 'Celine');
SQL>insert into Employees values (4, 'arjun');

SQL>insert into Managers values (1, 'dhyan');
SQL>insert into Managers values (2, 'shahana');
SQL>insert into Managers values (3, 'Celine');
SQL>insert into Managers values (4, 'pt');

Program Code:

SQL> select DISTINCT name from (select name from Employees UNION select name from Managers);

Output:

Name
---
delna
devu
Celine
dhyan
Arjun
pt

Program Code:

b) SQL>select name from Employees intersect select name from Managers;

Output:

Name
---
devu
celine

Program Code:

c) SQL> select name from Employees MINUS select name from Managers;

Output:

Name

Delna
Arjun

Program Code:

d) SQL> select name, 'Employee' as role from Employees UNION select name, 'Manager' as role from Managers;

Output:

Name	role
Delna	employee
Devu	employee
Celine	employee
Arjun	Manager
Dhyan	Manager
Pt	Manager

EXPERIMENT-7

QUESTION- 18

PL/SQL program to swap the values of two numbers.

Program Code:

```
declare
a integer;
b integer;
temp integer;
begin
a:=&a;
        b:=&b;
        dbms_output.put_line('Before swapping: num1=' || a ||' num2=' ||b);
        temp:=a;
a:=b;

b:=temp;
dbms_output.put_line('after swapping: num1=' || a ||' num2=' ||b);
end;
```

Output:

```
SQL> @swap.sql
Enter value for a: 12
Enter value for b: 59
Before swapping: num1=12 num2=59
after swapping: num1=59 num2=1
```

QUESTION-19

PL/SQL program to determine the largest among three given numbers.

Program Code:

```
DECLARE
a integer;
b integer;
c integer;
maxx integer;
BEGIN
   a:=&a;
   b:=&b;
   c:=&c;
   if a>b && a>c then
   maxx:=a;
   else if b>c && b>a then
   maxx:=b;
   else
   maxx:=c;
   end if;
   dbms_output.put_line('Largest=' || maxx);
end;
```

Output:

```
SQL>@large.sql
Enter the value for a: 15
Enter the value for b: 25
Enter the value for c: 5
Largest=25
```

QUESTION-20

PL/SQL program to compute the sum of digits of a given number.

Program Code:

```
declare
n integer;
s integer:=0;
r integer;
begin
   n:=&n;
   while n>0 loop
   r:=mod(n,10);
   s:=s+r;
   n:=trunc(n/10);
   end loop;
   dbms_output.put_line('Sum =' || s);
end;
```

Output:

```
SQL> @sod.sql
Enter the value for n: 135
Sum = 9
```

QUESTION-21

Program Code:

```
declare
n integer;
s integer:=0;
r integer;
begin
   n:=&n;
   while n>0 loop
   r:=mod(n,10);
   s:=(s*10)+r;
   n:=trunc(n/10);
   end loop;
   dbms_output.put_line('Reverse =' || s);
end;
```

Output:

```
SQL> @rev.sql
Enter value for n: 345
Reverse =543
```

QUESTION-21

Write a PL/SQL program to calculate the net salary and annual salary, considering DA as 30% of basic, HRA as 10% of basic, and PF as:7% if the basic salary is less than 8000 10% if the basic salary is between 8000 and 16000.

Program Code:

```
declare
   basic number;
   da number;
   hra number;
   pf number;
   net_salary number;
   annual_salary number;
begin
   basic:= &basic;
   da := 0.3 * basic;
   hra := 0.1 * basic;

   if basic < 8000 then
      pf := 0.07 * basic;
   elsif basic <= 16000 then
      pf := 0.10 * basic;
   else
      pf := 0.12 * basic;
   end if;

   net_salary := basic + da + hra - pf;
   annual_salary := net_salary * 12;
   dbms_output.put_line('given basic: ' || basic);
   dbms_output.put_line('net salary: ' || net_salary);
   dbms_output.put_line('annual salary: ' || annual_salary);
end;
```

Output:

SQL> @salary.sql

Enter the value for basic: 9000

given basic: 9000

net salary: 11700

annual salary:140400

QUESTION-22

Write a PL/SQL program that accepts an account number, checks if the balance is below the minimum required balance, and deducts Rs.100/- from the balance if necessary. The program should be applied to the acct table.

Program Code:

```
declare
   v_acct_no number ;
   v_balance number;
begin
   v_acct_no:=&v_acct_no;
   dbms_output.put_line('account_no:'||v_acct_no);
   dbms_output.put_line('minimum balance required: rs.1500/-');
   select balance into v_balance from account where acct_no = v_acct_no;

   if v_balance < 1500 then
      update account set balance = balance - 100 where acct_no = v_acct_no;
      dbms_output.put_line('changes made in account. rs.100 deducted.');
   else
      dbms_output.put_line('balance sufficient. no changes needed.');
   end if;
end;
```

Output:

SQL>@balance.sql

ACCT_NO	BALANCE
101	4500
102	8000
103	1000

```
Account_no:103
Minimum balance required: Rs.1500/-
Changes made in account. Rs.100 deducted.


PL/SQL procedure successfully completed.
```

ACCT_NO	BALANCE
101	4500
102	8000
103	900

QUESTION-23

Write a PL/SQL function that computes and returns the maximum of two given values.

Program Code:

```
create or replace function max_two(a number, b number)
return number
as
begin
  return greatest(a, b);
end;
/

declare
  result number;
  a number;
  b number;
begin
a:=&a;
  b:=&b;
  result := max_two(a, b);
  dbms_output.put_line('maximum of '||a|| ' and '|| b || ' is: '|| result);
end;
```

Output:

```
Function created.

Statement processed.

Enter value for a: 15

Ener value for b: 45
maximum of 25 and 35 is: 45
```

QUESTION-24

Write a PL/SQL function to check whether a given string is a palindrome.

Program Code:

```
create or replace function is_palindrome(str in varchar)
return varchar2
as
  rev_str varchar(20) := '';
  i integer;
begin
  for i in reverse 1..length(str) loop
  rev_str := rev_str || substr(str, i, 1);
  end loop;

  if lower(str) = lower(rev_str) then
    return 'palindrome';
  else
    return 'not palindrome';
  end if;
end;
/
declare
  result  varchar(20);
string   varchar(20);
begin
  string := '&string';
  dbms_output.put_line('given string: ' || string);
  result := is_palindrome(string);
  dbms_output.put_line('result: ' || result);
end;
```

Output:

```
Statement processed.
Given string: malayalam
Result: palindrome
```

QUESTION-25

Write a PL/SQL function that returns the total count of customers in the customers table.

Program Code:

```
create or replace function customer_count
return number
as
  total number;
begin
  select count(*) into total from customers;
  return total;
end;
 /
declare
  count number;
begin
  count := customer_count;
dbms_output.put_line('total customers: ' || count);
end;
/
```

Output:

CUSTOMER_ID	FIRST_NAME	LAST_NAME	EMAIL
1	ananda	krishna	ananda123@gmail.com
2	kavya	Santhosh	kavya11@gmail.com
3	Anna	KS	anna97@gmail.com

```
Statement processed.
total customers: 3
```

QUESTION-26

Write a PL/SQL procedure to compute and display the sum of two numbers.

Program Code:

```
create or replace procedure sum_two(a in number, b in number)
as
  total number;
begin
  total := a + b;
  dbms_output.put_line('sum of ' ||a|| ' and ' || b  || ' =' || total);
end;
/

declare
a number;
b number;
begin
a:=&a;
b:=&b;
sum_two(a, b);
end;
/
```

Output:

```
Statement processed.
Sum of 25 and 20 = 45
```

QUESTION-27

Write a PL/SQL procedure to insert a student's roll number and name into the student table.

Program Code:

```
create or replace procedure insert_student(p_roll in number, p_name in varchar2)
as
begin
  insert into student(rollno, name) values (p_roll, p_name);
dbms_output.put_line('student inserted('||p_roll || ',' || p_name || ')');
end;
/

declare
roll number;
name varchar2(20);
begin
roll:=&roll;
name:='&name';
insert_student(roll, name);
end;
```

Output:

```
Statement processed.
student inserted(41,faikka)
```

QUESTION-28

Write a PL/SQL procedure to retrieve and display the count of instructors in a specified department.

Program Code:

```
create or replace procedure instructor_count(p_dept in varchar2)
as
  cnt number;
begin
  select count(*) into cnt from instructor where dept = p_dept;
dbms_output.put_line('instructors in ' || p_dept || ': ' || cnt);
end;
/

declare
begin
  instructor_count('mca');
end;
/
```

Output:

FID	DEPT	FNAME
1	mca	Anu
2	Mba	devu
3	Mca	deepa
4	Eee	Dhyan

```
Statement processed.
Instructors in mca: 2
```

QUESTION-29

Create a Customers table with attributes (CustId (Primary Key), CustName, City). Then, write a PL/SQL program using an explicit cursor to display all details from the Customers table.

Program Code:

```
declare
   cursor cust_cursor is select * from customers;
   rec customers%rowtype;
begin
   open cust_cursor;
   loop
      fetch cust_cursor into rec;
      exit when cust_cursor%notfound;
      dbms_output.put_line('id: ' || rec.custid ||
                           ', name: ' || rec.custname ||
                           ', city: ' || rec.city);
   end loop;
   close cust_cursor;
end;
/
```

Output:

ID: 1, Name: Faikka, City:Ernakulam

ID: 2, Name: Devu, City: Kottayam

ID: 3, Name: Dhyan, City: Kollam

QUESTION-30

Write a PL/SQL program using an explicit cursor to display details of employees working in the MCA department.

Program Code:

```
declare
   cursor emp_cursor is select * from instructor where dept = 'mca';
   rec instructor%rowtype;
begin
   open emp_cursor;
   loop
      fetch emp_cursor into rec;
      exit when emp_cursor%notfound;
      dbms_output.put_line('id: ' || rec.fid || ', name: ' || rec.fname);
   end loop;
   close emp_cursor;
end;
```

Output:

ID: 1, Name: Anu

ID: 2, Name: Deepa

QUESTION-31

Create a table Teacher with following attributes

Teacher(T_id, T_name, Join_date, Department). Write a trigger that verifies the joining date when a new row is inserted in the 'teacher' table. Joining date should be greater than or equal to current date.

Program Code:

```
create table teacher ( t_id number primary key, t_name varchar2(20), join_date date,department varchar2(20));

create or replace trigger trg_check_join_date
before insert on teacher
for each row
begin
   if :new.join_date < trunc(sysdate) then
      raise_application_error(-20001, 'joining date cannot be earlier than today.');
   end if;
end;
/
insert into teacher values (1, 'john', current_date, 'mca');
insert into teacher values (2, 'anu', to_date('2024-05-01', 'yyyy-mm-dd'), 'mba');
```

Output:

```
Trigger TRG_CHECK_JOIN_DATE compiled

Elapsed: 00:00:00.017

SQL> INSERT INTO Teacher VALUES (1, 'John', current_date, 'MCA')


1 row inserted.

SQL> INSERT INTO Teacher VALUES (2, 'Anu', TO_DATE('2024-05-01', 'YYYY-MM-DD'), 'MBA')

ORA-20001: Joining date cannot be earlier than today.
ORA-06512: at "SQL_KWFL2H41G6E1C4VBO8LXOAE73K.TRG_CHECK_JOIN_DATE", line 3
ORA-04088: error during execution of trigger 'SQL_KWFL2H41G6E1C4VBO8LXOAE73K.TRG_CHECK_JOIN_DATE'
```

EXPERIMENT 8

Configuration of NoSQL database

QUESTION-32

Comparison between relational and non-relational (NoSQL) databases and the configuration of NoSQL Databases.

Feature	Relational (SQL) Databases	Non-Relational (NoSQL) Databases
Data Model	Structured tables with rows and columns	Flexible models: document, key-value, wide-column, graph
Schema	Fixed schema; predefined structure	Dynamic schema; allows for unstructured or semi-structured data
Scalability	Vertical scaling (adding more power to a single server)	Horizontal scaling (adding more servers to distribute the load)
Query Language	Structured Query Language (SQL)	Varies by database (e.g., MongoDB uses its own query language)
Transactions	Strong ACID compliance (Atomicity, Consistency, Isolation, Durability)	Some support for ACID; others favor BASE (Basically Available, Soft state, Eventual consistency)
Examples	MySQL, PostgreSQL, Oracle, Microsoft SQL Server	MongoDB, Cassandra, Redis, Couchbase, Neo4j
Use Cases	Complex queries, multi-row transactions, structured data	Large volumes of diverse data, real-time analytics, content management, IoT, big data applications
Data Integrity	Enforced through constraints and relationships	Application-level enforcement; less emphasis on strict data integrity
Flexibility	Less flexible; changes require altering the schema	Highly flexible; easy to add new fields or data types without affecting existing data
Performance	Optimized for complex queries and joins	Optimized for high-speed read/write operations and large-scale data handling

CONFIGURATION OF NOSQL DATABASES

Configuring a NoSQL database involves several key considerations to ensure optimal performance, scalability, and reliability:

1. Data Modeling:
    - Document Stores (e.g., MongoDB): Design collections and documents to align with application access patterns.
    - Key-Value Stores (e.g., Redis): Determine appropriate keys and values for efficient retrieval.
    - Wide-Column Stores (e.g., Cassandra): Define column families and clustering keys based on query requirements.
    - Graph Databases (e.g., Neo4j): Model nodes and relationships to represent entities and their connections.
2. Sharding and Partitioning:
    - Distribute data across multiple nodes or clusters to balance load and enhance performance.
    - Choose appropriate sharding keys to ensure even data distribution.
3. Replication:
    - Implement replication strategies to enhance data availability and fault tolerance.
    - Configure the number of replicas based on desired redundancy and read performance.
4. Consistency Models:
    - Select a consistency level that balances performance and data accuracy (e.g., eventual consistency, strong consistency).
    - Configure settings per the database's capabilities and application requirements.
5. Security Settings:
    - Set up authentication and authorization mechanisms to control access.
    - Enable encryption for data at rest and in transit.
    - Regularly update and patch the database software to address vulnerabilities.
6. Monitoring and Maintenance:
    - Utilize monitoring tools to track performance metrics, resource utilization, and potential issues.
    - Schedule regular backups and implement disaster recovery plans.
    - Perform routine maintenance tasks such as compaction, indexing, and cleanup.
7. Deployment Considerations:
    - Decide between on-premises, cloud-based, or hybrid deployment models.
    - For cloud deployments, leverage managed services to reduce operational overhead.
8. Scaling Strategies:
    - Plan for horizontal scaling by adding more nodes to handle increased load.
    - Implement auto-scaling features if supported, to dynamically adjust resources based on demand.

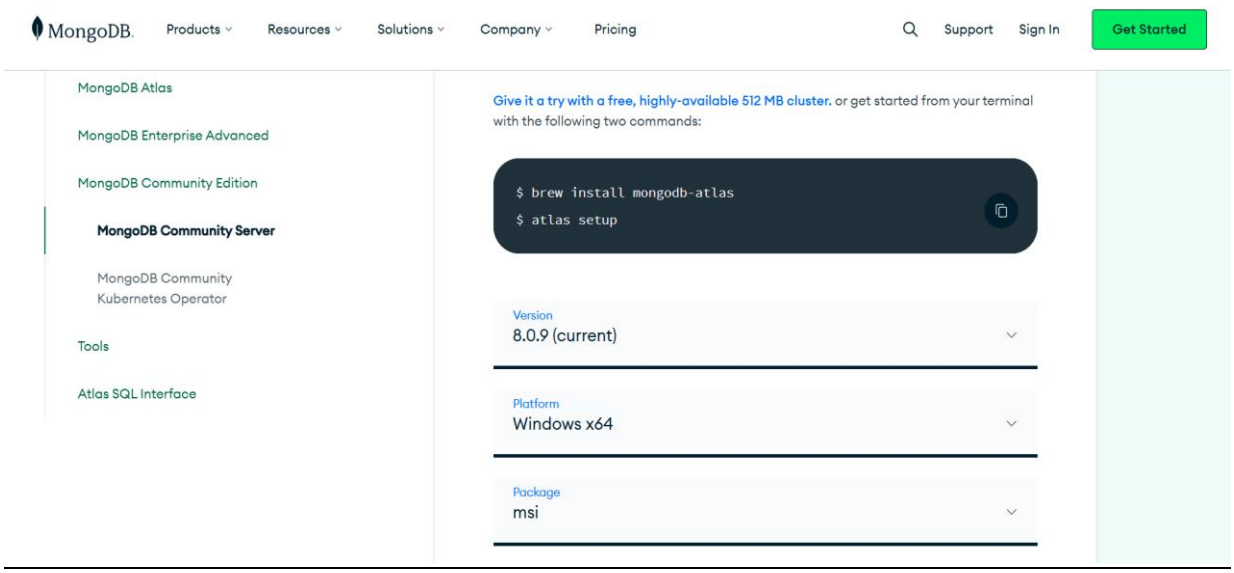
EXPERIMENT 9

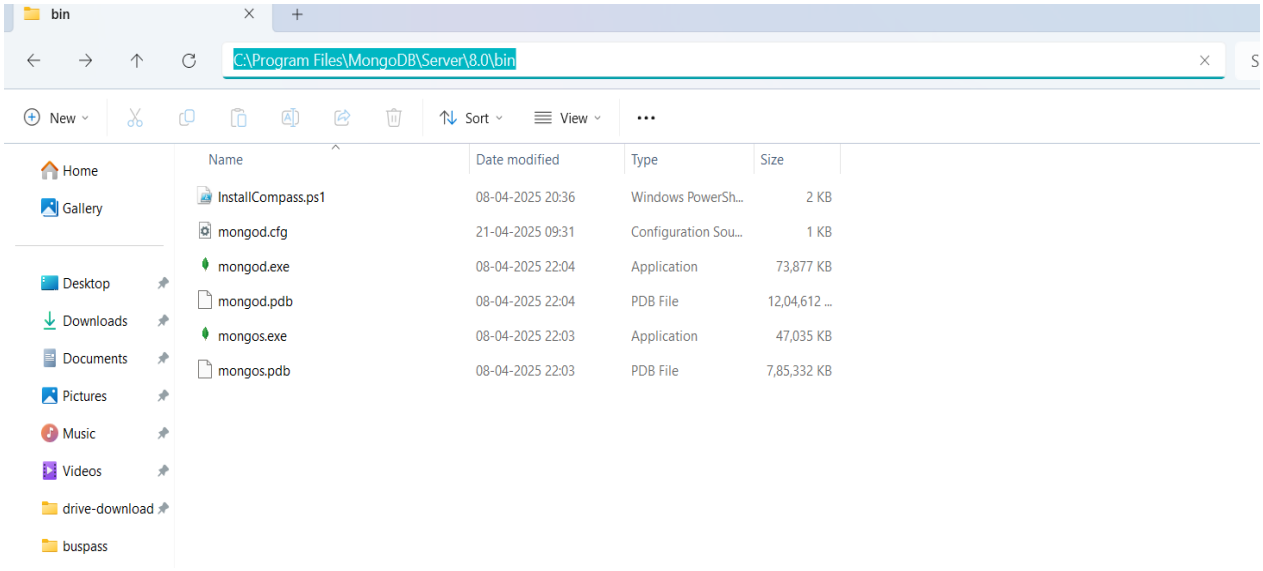
PROGRAMS USING MONGODB

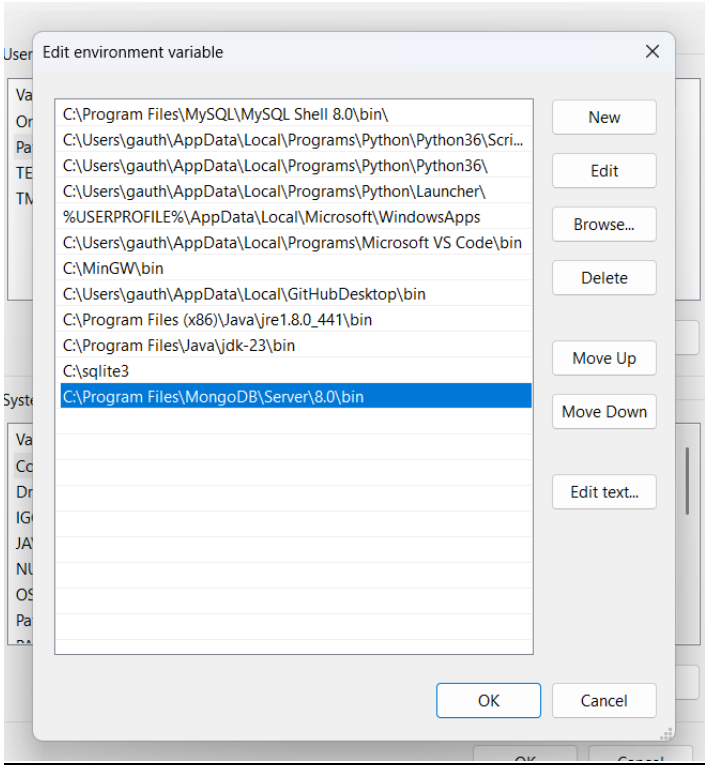
QUESTION 33

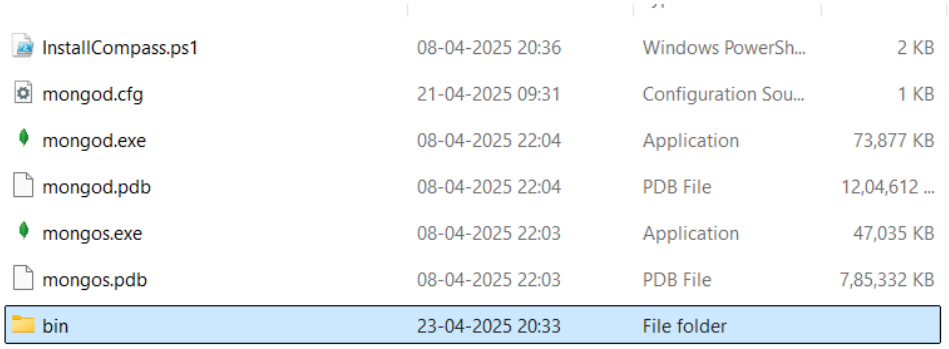
Install the MongoDB and configure it.

OUTPUT










```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\gauth> mongosh
Current Mongosh Log ID: 68189ced0859ba1fbdb5f898
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.5.0
Using MongoDB:          8.0.8
Using Mongosh:          2.5.0

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

------
   The server generated these startup warnings when booting
   2025-04-30T16:25:19.178+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
------

test>
```

QUESTION 34

Create a collection student consists of details like rollno, name, phoneno, marks, address, year of course etc.

PROGRAM CODE

```
test> use college

switched to db college

college> db.createCollection("student41")

{ ok: 1 }

college> db.student41.find()
```

QUESTION 35

Insert the details of the multiple students (atleast 5) in the form of documents in the student collection.

PROGRAM CODE

```
college>
db.student41.insertOne({rollno:1,name:"Faikka",phoneno:8590451893,marks:94,address:"Kottayam",year:2024})

{

  acknowledged: true,

  insertedId: ObjectId('6818b16f0859ba1fbdb5f89e')

}

college>
db.student41.insertOne({rollno:2,name:"Delna",phoneno:8547459618,marks:90,address:"Thrissur",year:2024})

{

  acknowledged: true,

  insertedId: ObjectId('6818b1d70859ba1fbdb5f89f')

}
```

```
college>
db.student41.insertOne({rollno:3,name:"Devu",phoneno:9605080027,marks:85,address:"Kollam",year:2024})
{
  acknowledged: true,
  insertedId: ObjectId('6818b20e0859ba1fbdb5f8a0')
}
college>
db.student41.insertOne({rollno:4,name:"Dhyan",phoneno:9446491760,marks:94,address:"Kannur",year:2022})
{
  acknowledged: true,
  insertedId: ObjectId('6818b2520859ba1fbdb5f8a1')
}
college>
db.student41.insertOne({rollno:5,name:"Anfi",phoneno:8547210352,marks:92,address:"Thrissur",year:2024})
{
  acknowledged: true,
  insertedId: ObjectId('6818b2830859ba1fbdb5f8a3')
}
```

QUESTION 36

Retrieve the fields rollno, name, phoneno, marks, city for all the documents in the collection student.

PROGRAM CODE

```
college> db.student41.find()
```

OUTPUT


```
[
  {
    _id: ObjectId('6818b16f0859ba1fbdb5f89e'),
    rollno: 1,
    name: 'Faikka',
    phoneno: 8590451893,
    marks: 94,
    address: 'Kottayam',
    year: 2024
  },
  {
    _id: ObjectId('6818b1d70859ba1fbdb5f89f'),
    rollno: 2,
    name: 'Delna',
    phoneno: 8547459618,
    marks: 90,
    address: 'Thrissur',
    year: 2024
  },
  {
    _id: ObjectId('6818b20e0859ba1fbdb5f8a0'),
    rollno: 3,
    name: 'Devu',
    phoneno: 9605080027,
    marks: 85,
    address: 'Kollam',
    year: 2024
  },
```

```
  {
    _id: ObjectId('6818b2520859ba1fbdb5f8a1'),
    rollno: 4,
    name: 'Dhyan',
    phoneno: 9446491760,
    marks: 94,
    address: 'Kannur',
    year: 2022
  },
  {
    _id: ObjectId('6818b2830859ba1fbdb5f8a3'),
    rollno: 5,
    name: 'Anfi',
    phoneno: 8547210352,
    marks: 92,
    address: 'Thrissur',
    year: 2024
  }
]
```

QUESTION 37

Display the details of students who achieved a score more than 90 and are from ‘Thrissur’.

PROGRAM CODE

```
college> db.student41.find({ marks: { $gt: 90 }, address: "Thrissur" })
```

OUTPUT

```
[
 {
   _id: ObjectId('6818b2830859ba1fbdb5f8a3'),
   rollno: 5,
   name: 'Anfi',
   phoneno: 8547210352,
   marks: 92,
   address: 'Thrissur',
   year: 2024
  }
]
```

QUESTION 38

Update the phone number of Devu in the student collection. Retrieve the updated Information.

PROGRAM CODE

```
college> db.student41.updateOne({name:"Devu"},{ $set: {phoneno:9876542130}})
college> db.student41.find({ name: "Devu" })
```

OUTPUT

```
[
  {
    _id: ObjectId('6818b20e0859ba1fbdb5f8a0'),
    rollno: 3,
    name: 'Devu',
    phoneno: 9876542130,
    marks: 85,
    address: 'Kollam',
    year: 2024
  }]
```

QUESTION 39

Update the year of course in all the documents in the student collection to 2021. Also retrieve the updated information.

PROGRAM CODE

```
college> db.student41.updateMany({},{$set:{year:2021}})

college> db.student41.find()
```

OUTPUT

```
[
  {
    _id: ObjectId('6818b16f0859ba1fbdb5f89e'),
    rollno: 1,
    name: 'Faikka',
    phoneno: 8590451893,
    marks: 94,
    address: 'Kottayam',
    year: 2021
  },
  {
    _id: ObjectId('6818b1d70859ba1fbdb5f89f'),
    rollno: 2,
    name: 'Delna',
    phoneno: 8547459618,
    marks: 90,
    address: 'Thrissur',
    year: 2021
  },
  {
    _id: ObjectId('6818b20e0859ba1fbdb5f8a0'),
```

```
    rollno: 3,
    name: 'Devu',
    phoneno: 9876542130,
    marks: 85,
    address: 'Kollam',
    year: 2021
  },
  {
    _id: ObjectId('6818b2520859ba1fbdb5f8a1'),
    rollno: 4,
    name: 'Dhyan',
    phoneno: 9446491760,
    marks: 94,
    address: 'Kannur',
    year: 2021
  },
  {
    _id: ObjectId('6818b2830859ba1fbdb5f8a3'),
    rollno: 5,
    name: 'Anfi',
    phoneno: 8547210352,
    marks: 92,
    address: 'Thrissur',
    year: 2021
  }
]
```

QUESTION 40

Display the contact address of ‘Dhyan’.

PROGRAM CODE

```
college> db.student41.find({name:"Dhyan"},{_id:0,phoneno:1})
```

OUTPUT

```
[ { phoneno: 9446491760 } ]
```

QUESTION 41

Delete the details of the student whose name is ‘Dhyan’ from the student collection.

PROGRAM CODE

```
college> db.student41.deleteOne({ name: "Dhyan" })
```

QUESTION 42

Retrieve the number of students per department from the student collection.

PROGRAM CODE

```
college> db.student41.aggregate([{$group:{_id:"$department", count:{$sum:1}}}])
```

OUTPUT

```
[ { _id: null, count: 5 } ]
```

QUESTION 43

Arrange the name of the students in ascending order along with all the columns.

PROGRAM CODE

```
college> db.student41.find().sort({ name: 1 })
```

OUTPUT


```
[
  {
    _id: ObjectId('6818b2830859ba1fbdb5f8a3'),
    rollno: 5,
    name: 'Anfi',
    phoneno: 8547210352,
    marks: 92,
    address: 'Thrissur',
    year: 2021
  },

  {
    _id: ObjectId('6818b1d70859ba1fbdb5f89f'),
    rollno: 2,
    name: 'Delna',
    phoneno: 8547459618,
    marks: 90,
    address: 'Thrissur',
    year: 2021
  },
{
    _id: ObjectId('6818b20e0859ba1fbdb5f8a0'),
    rollno: 3,
    name: 'Devu',
    phoneno: 9876542130,
    marks: 85,
    address: 'Kollam',
    year: 2021
```

```
  }


  {
    _id: ObjectId('6818b16f0859ba1fbdb5f89e'),
    rollno: 1,
    name: 'Faikka',
    phoneno: 8590451893,
    marks: 94,
    address: 'Kottayam',
    year: 2021
  }
]
```

QUESTION 44

Rename city as town and add the detail of address consists of apartment no, street name and PIN.

PROGRAM CODE

```
college>db.student41.updateMany({},[{$set:{town:{apartment_no:"101",street:"Gandhinagar",pin:"680001"}}}])
college> db.student41.updateMany({},{$unset:{address:""}})
college> db.student41.find({rollno:1})
```

OUTPUT

```
[  {
    _id: ObjectId('6818b16f0859ba1fbdb5f89e'),
    rollno: 1,
    name: 'Faikka',
    phoneno: 8590451893,
    marks: 94,
```

```
  year: 2021,
  town: { apartment_no: '101', street: 'Gandhinagar', pin: '680001' }
}]
```