CHIT 1:

1. CREATE TABLE branch_master (
   branch_id INT PRIMARY KEY,
   branch_name VARCHAR(50) NOT NULL
   );
2. CREATE TABLE employee_master (
   emp_id INT PRIMARY KEY,
   first_name VARCHAR(50) NOT NULL,
   middle_name VARCHAR(50),
   last_name VARCHAR(50),
   department VARCHAR(50),
   manager_id INT
   );
3. INSERT INTO branch_master (branch_id, branch_name) VALUES
   ((101,"computer"),(102,"ENTC"),(103,"IT"),(104,"Mech"));
4. INSERT into emp_master (employee_id, first_name, last_name, middle_name, department, manager_id) values (1001,"vivek", "patil", "janardan", "Computer", 2001),(1002,"Niranjan", "Jagtap", "laxman", "Computer", 2001),(1003,"aman", "ingale", "santosh", "Computer", 2002),(1004,"varad", "shinde", "santosh", "Computer", 2004);
5. create index name_index on employee_master(first_name, last_name)
6. create view employeeDetails as SELECT first_name, last_name, department from employee_master;

CHIT NO. 2

1. create database university
2. use university
3. create table student(stud_id int PRIMARY KEY,
   department varchar(30),
   name varchar(30),
   sem int,
   yr int,
   credits int);
4. create table teaches (teacher_id int PRIMARY KEY,
   teacher_name varchar(30),
   salary float,
   department varchar (40));
5. insert into student(stud_id, s_name, department, sem, yr, credits) values
   (1001,"vivek", "Computer", 5,3, 2001),
   (1002,"Niranjan", "Computer",5,3, 2001),
   (1003,"aman","Computer", 5,3 ,2002),
   (1004,"varad", "Computer",5,3,2004);
6. insert into teaches(teacher_id, teacher_name, salary, department) values
   (0001, "shafali gupta", 20000, "COMPUTER"),
   (0002, "pradnya kasture", 50000, "Cyber"),
   (0003, "priyanka jadhav", 40000, "Data Science");
7. SELECT teacher_name,MAX(salary) from teaches;
8. delete from teaches where salary <= 20000;
9. select sum(salary) from teaches GROUP by department;

CHIT No. 3

1. CREATE table dept(deptId int PRIMARY KEY, deptName varchar(30));
2. create table emp(empId int PRIMARY key,
   empName varchar(30),
   empSal int,
   empDeptId int,
   foreign key empDeptId REFERENCES dept(deptId)
   );
3. alter table emp modify empName varchar(30) NOT NULL;
4. insert into dept(deptId, deptName) VALUES (2020, "Comp"),(2025,"IT"),(2021,"Mech");
5. insert into emp (empId, empName, empsal,empdeptid) values
   (101,"Vivek", 100000,2020),
   (102,"Niranjan", 340000,2025),
   (103,"Varad", 600000,2025),
   (104,"Aman", 200000,2021);
6. alter table dept add deptLoc varchar(30) UNIQUE;

CHIT No. 4

1. CREATE table dept(deptId int PRIMARY KEY, deptName varchar(30));
2. create table emp(empId int PRIMARY key,
   empName varchar(30),
   empSal int,
   empDeptId int,
   foreign key empDeptId REFERENCES dept(deptId)
   );
3. insert into dept(deptId, deptName) VALUES (2020, "Comp"),(2025,"IT"),(2021,"Mech");
4. insert into emp (empId, empName, empsal,empdeptid) values
   (101,"Vivek", 100000,2020),
   (102,"Niranjan", 340000,2025),
   (103,"Varad", 600000,2025),
   (104,"Aman", 200000,2021);
5. select * from emp where empDeptId in (2020, 10,30,2021);
6. select * from emp where empSal BETWEEN 100000 and 500000;
7. SELECT Count(*) as Number_Of_Employees from emp;
8. select avg(empsal) as average, empdeptId from emp GROUP BY empdeptId;
9. select * from emp ORDER BY empsal;

CHIT No. 5

1. create table customer(
   customer_id int PRIMARY KEY,
   first_name varchar(30)
   );
2. create table orders (
   order_id int PRIMARY KEY,
   amount int,
   customer_id int,
   FOREIGN KEY (customer_id) REFERENCES customer(customer_id)
   );
3. INSERT INTO customer (customer_id, first_name)

VALUES
(1, 'Alice'),
(2, 'Bob'),
(3, 'Charlie'),
(4, 'David'),
(5, 'Eva');

4. INSERT INTO orders (order_id, amount, customer_id)
   VALUES
   (101, 500, 1),
   (102, 1200, 2),
   (103, 750, 3),
   (104, 1500, 1),
   (105, 300, 5);
5. select * from customer c
   LEFT JOIN orders o
   ON c.customer_id=o.customer_id;
6. select * from customer c
   RIGHT JOIN orders o
   ON c.customer_id=o.customer_id;
7. select * from customer c
   CROSS JOIN orders o
   ON c.customer_id=o.customer_id;
8. #INNER by default
   select * from customer c
   JOIN orders o
   ON c.customer_id=o.customer_id;

CHIT No. 6

1. create table stud_marks(name varchar(30), marks int);
2. create table result(roll int,name varchar(30),class varchar(30));
3. DELIMITER $$

   create PROCEDURE proc_grade()

   BEGIN

   DECLARE v_name varchar(30);

DECLARE v_roll int;

DECLARE v_marks int;

DECLARE v_class varchar(30);

DECLARE done int DEFAULT 0;

DECLARE stud_cursor CURSOR FOR

select name,marks,roll from stud_marks;

DECLARE CONTINUE HANDLER for not FOUND set done=1;

OPEN stud_cursor;

```
read_loop: LOOP
  FETCH stud_cursor INTO v_name, v_marks, v_roll;
   IF done THEN
   LEAVE read_loop;
   END IF;
   IF v_marks BETWEEN 1500 AND 990 THEN
         SET v_class="distinction";
   ELSEIF      v_marks BETWEEN 989 AND 900 THEN
         SET v_class ="first_class";
   ELSEIF v_marks BETWEEN 899 AND 825 THEN
         SET v_class ="higher_second_class";
   ELSE
         SET v_class="fail";
         END IF;
   INSERT INTO stud_marks (roll, name, class) values (v_roll, v_marks, v_class);
   END LOOP;
  CLOSE stud_cursor;
END $$
DELIMITER ;
```

CHIT No. 7

```
1. #Create table n_rollcall, o_rollcall and add values in o_rollcall some values in n_rollcall
2. DELIMITER $$
create procedure exe()
begin

  DECLARE done int default 0;
  DECLARE v_roll int;
  DECLARE v_name varchar(30);
  DECLARE v_count int;
  DECLARE n_cursor cursor FOR
            SELECT name,roll from n_rollcall;
  declare CONTINUE HANDLER for
  not found set done = 1;

  OPEN n_cursor;
read_loop: LOOP
         FETCH n_cursor into v_name, v_roll;
```

```
if done THEN
        LEAVE read_loop;
        END IF;

SELECT count(*) into v_count
        from n_rollcall;
if v_count>0 then
        INSERT INTO o_rollcall (name, roll) values(v_name, v_roll);
END IF;
END LOOP;
CLOSE n_cursor;
end$$;
DELIMITER ;
```

CHIT No. 8

1. create table accounts(
   acc_no int PRIMARY KEY,
   cust_name varchar(40),
   balance int);
2. INSERT INTO accounts (acc_no,cust_name,balance) VALUES
   (101,"Vivek", 2000),
   (102,"Varad", 3000),
   (103,"Aman", 5000),
   (104,"Niranjan", 20000),
   (105,"Shailesh", 12500);

3. START TRANSACTION;
4. update accounts set balance = balance-5000 WHERE acc_no = 104;
5. UPDATE accounts set balance = balance+5000 where acc_no = 101;
6. SELECT * from accounts;
7. ROLLBACK;
8. SELECT * FROM accounts;

9. START TRANSACTION;
10. update accounts set balance = balance-5000 WHERE acc_no = 104;
11. UPDATE accounts set balance = balance+5000 where acc_no = 101;
12. SELECT * from accounts;
13. COMMIT;
14. SELECT * FROM accounts;

15. START TRANSACTION;
16. update accounts set balance = balance-5000 WHERE acc_no = 104;
17. SAVEPOINT debited;
18. UPDATE accounts set balance = balance+5000 where acc_no = 101;
19. SAVEPOINT credited;
20. ROLLBACK to debited;
21. SELECT * FROM accounts;