DBMS 2

```
create table customers(Acc_no int primary key,Name varchar(20),Balance int,City varchar(10));
desc customers;

insert into customers(Acc_no,Name,Balance,City) values
(1,'Ram',10000,'Pune'),(2,'Ravi',25000,'Nashik'),(3,'Sachin',30000,'Mumbai');
select * from customers;

create table loan(Loan_no int,Name varchar(20),Loan_amt int);

insert into loan(Loan_no,Name,Loan_amt) values(1,'Ram',10000);
insert into loan values(2,'Ravi',50000),(3,'Dipak',40000);
select * from loan;

select name from Customers union select name from loan;
select name from Customers where name in (select name from loan);
select name from customers where name not in(select name from loan);

create view cust_above_20000 as select*from customers where balance>=20000;
select * from cust_above_20000;

create view cust_nashik as select*from customers where Balance>=20000 and city='Nashik';
select * from cust_nashik;

create index name_index on customers(Name);
show index from customers;

create index index1 on customers(Acc_no,Name);
show index from customers;

select * from customers order by Balance asc;

select * from loan order by Loan_amt desc;

select Name,Loan_amt,Loan_amt*0.04*20 as InterestAfter20Yrs from loan;
```

DBMS 3

```
create table student_mark(RollNo int,Name varchar(20),Marks int,Branch varchar(20));
insert into student_mark
values(1,'Ravi',90,'Computer'),(2,'Vedika',70,'Computer'),(3,'Aarush',95,'IT'),(4,'Jyoti',60,'IT');
select *from student_mark;

select * from student_mark where Marks>=70;
```

```
select * from student_mark where Marks<70;

select * from student_mark where Marks=70 and Name='Vedika';

update student_mark set Name='Sachin' where Name='Ravi';
select * from student_mark;

delete from student_mark where Name='Aarush';
select * from student_mark;

select * from student_mark where Marks=(select max(Marks) from student_mark);

select Branch,min(Marks) as Minimum,max(Marks) as Maximum,avg(Marks) as Average,sum(Marks)
as Sum,count(Branch) as Count from student_mark group by Branch;
```

DBMS 4

```
create table Borrower(RollNo int,Name varchar(10),DOI date,NameofBook varchar(20),Status
varchar(10));
insert into Borrower values(1,'Mansi','2023-08-23','DBMS','I');
select * from Borrower;

create table Fine(RollNo int,Date date,Amt int);
select * from Fi

drop procedure if exists p_fine;

 delimiter//

 create procedure p_fine(IN rno,IN bname varchar(20))
 begin
 declare d1 date;
 declare daycnt int;
 declare fine_amt int;

 select DOI into d1 from Borrower
 where RollNo=rno and NameofBook=bname;

 select DATEDIFF(now(),d1) into daycnt;

   if(daycnt >= 15 and daycnt <= 30) then
     set fine_amt=daycnt*5;

   update Borrower set Status = 'R' where RollNo= rno;
```

```
    elseif(daycnt>30) then
      set fine_amt = daycnt*50;

      insert into Fine values(rno,now(),fine_amt);

      update Borrower set Status='R' where RollNo = rno;

    else

    update BORROWER set Status = 'R' where RollNo = rno;


  end if;

    end;
//
```

DBMS 5

```
create table tbl_area(radius INT, area float);
desc tbl_area;

select * from tbl_area;

drop procedure if exists find_circle_area;

delimiter //

create procedure find_circle_area()
begin
  declare r int;
  declare a float;

  set r = 5;
  loop_label: LOOP
    IF r > 9 THEN
      leave loop_label;CREA
    end if;

    set a = 3.14 * r * r;
    insert into tbl_area(radius, area) values (r, a);

    set r = r + 1;
  END LOOP loop_label;
```

```
END;
//

delimiter ;

call find_circle_area();

select * from tbl_area;
```

DBMS 6

```
create table studmarks(rollno int, name varchar(20), marks int);

insert into studmarks(rollno, name, marks) values
(1, 'JOHN', 850),
(2, 'HARRY', 1250),
(3, 'EMMA', 1450),
(4, 'LUCY', 950),
(5, 'TOM', 750);

create table result_v(rollno int, name varchar(20), class varchar(20));

drop procedure if exists proc_grade;

delimiter //

create procedure proc_grade()
begin
  DECLARE done INT DEFAULT 0;

  declare s_marks int;
  declare s_rollno int;
  declare s_name varchar(30);
  declare s_class varchar(20);

  declare s_student cursor for select rollno, name, marks from studmarks;

  DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;

  open s_student;

  read_loop: loop
    fetch s_student into s_rollno, s_name, s_marks;

    if done = 1 then
      leave read_loop;
```

```
    end if;

    if (s_marks <= 1500 and s_marks >= 990) then
      set s_class = 'DISTINCTION';
    elseif (s_marks <= 989 and s_marks >= 825) then
      set s_class = 'FIRST CLASS';
    elseif (s_marks <= 899 and s_marks >= 750) then
      set s_class = 'HIGHER SECOND CLASS';
    else
      set s_class = 'PASS';
    end if;

    insert into result_v(rollno, name, class) values (s_rollno, s_name, s_class);
  end loop;

  close s_student;
end;
//

delimiter ;

select * from result_v;

CALL proc_grade();

select * from result_v;
```

DBMS 7

```
create table N_RollCall(rollno int,name varchar(30));

create table O_RollCall(rollno int,name varchar(30));

insert into N_RollCall values(1,'Shweta'),(2,'Vaishnavi'),(3,'Ankita'),(4,'Utkarsha');

insert into O_RollCall values(2,'Vaishnavi'),(3,'Ankita'),(5,'Ruchita'),(6,'Nikita');

select * from N_RollCall;

select * from O_RollCall;

drop procedure if exists mycursor1;

delimiter //

create procedure mycursor1()
```

```
begin
  DECLARE done int default 0;
  DECLARE c_rollno int;
  DECLARE c_name char(20);

  DECLARE c_studentN CURSOR for SELECT rollno,name FROM O_RollCall where rollno not
in(select rollno from N_RollCall);

  DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
  begin
    OPEN c_studentN;
      read_loop:LOOP
        FETCH c_studentN into c_rollno,c_name;

        IF done = 1 THEN
          LEAVE read_loop;
        END IF;

        insert into N_RollCall(rollno,name) values(c_rollno,c_name);

        END LOOP;
        CLOSE c_studentN;
    END;
  END;
//

delimiter ;

call mycursor1();

select * from N_RollCall;

select * from O_RollCall;
```

DBMS 8

```
create table lib_book(bid int,bname varchar(20),qty int);

create table lib_audit(bid int,bname varchar(20),qty int);

insert into lib_book values(1,'SECRET',10),
(2,'HATE',30),
(3,'THE',50),
(4,'LOVE',45);
```

```
select * from lib_book;

select * from lib_audit;

delimiter //

CREATE TRIGGER Make_audit1
AFTER DELETE ON lib_book
FOR EACH ROW
BEGIN
INSERT INTO lib_audit(bid,bname,qty)
VALUES(OLD.bid,OLD.bname,OLD.qty);
END;
//

delimiter;

DELETE FROM lib_book where bid = 4 or bid = 2;

select * from lib_book;

select * from lib_audit;

delimiter //

CREATE TRIGGER Make_audit2
AFTER UPDATE ON lib_book
FOR EACH ROW
BEGIN
INSERT INTO lib_audit(bid,bname,qty)
VALUES(OLD.bid,OLD.bname,OLD.qty);
END;
//

delimiter ;

select * from lib_book;

select * from lib_audit;
```

DBMS 1A Mongodb

```
db.createCollection("Student1");

db.Student1.insert({RollNo:1,Name:"RAM"});
```

```
db.Student1.insert([{RollNo:2,Name:"RUCHITA",Class:"COMPUTER"},{RollNo:3,Name:"Kaveri"},{
RollNo:4,Name:"Pooja"}]);

db.Student1.find().pretty();

db.Student1.update({RollNo:2},{$set:{Class:"ENTC",Name:"ARYA"}});

db.Student1.find().pretty();

db.Student1.remove({RollNo:4});

db.Student1.find().pretty();

db.Student1.save({RollNo:4,Name:"DEV",Year:"TE"});

db.Student1.find().pretty();

db.Student1.find({$and:[{RollNo:1},{Name:"RAM"}]}).pretty()
```

DBMS 2B Mongodb

```
db.createCollection("emp");

db.emp.insert({ id: 1, name: "STEVE", age: 23, role: "MANAGER", project: "NIRVANA", salary:
40000 });
db.emp.insert({ id: 2, name: "TONY", age: 28, role: "DEVELOPER", project: "ALEXA", salary: 35000
});
db.emp.insert({ id: 3, name: "NATASHA", age: 30, role: "DESIGNER", project: "NIRVANA", salary:
32000 });
db.emp.insert({ id: 4, name: "GROOT", age: 21, role: "MARKETING", project: "CORTANA", salary:
38000 });

db.emp.find().pretty();

db.emp.aggregate([
  { $match: { project: "ALEXA" } },
  { $count: "People Working on Project:ALEXA" }
]);

db.emp.aggregate([
  {
    $group: {
      _id: "$role",
      totalPayment: { $sum: 1 },
      min_age: { $min: "$age" }
    }
```

```
  }
]);

db.emp.aggregate([
  {
    $group: {
      _id: "$role",
      totalPayment: { $sum: 1 },
      max_age: { $max: "$age" }
    }
  }
]);

db.emp.aggregate([
  {
    $group: {
      _id: "$role",
      avg_salary: { $avg: "$salary" }
    }
  }
]);

db.emp.aggregate([
  { $match: { age: { $gt: 30 } } }
]);

db.emp.aggregate([
  { $sort: { name: 1 } }
]);

db.emp.aggregate([{$unwind: "$tech"}]);

db.emp.createIndex({ "age": 1 });

db.emp.createIndex({ "age": 1, "salary": 1 });

db.emp.getIndexes();

db.emp.dropIndex({ "age": 1 });

db.emp.dropIndex({ "age": 1, "salary": 1 });

db.emp.getIndexes();
```

DBMS 3B mapReduce

```
db.books.insert([{bid:1,bname:"the achemist",author:"anna",price:200,status:"A"},
            {bid:2,bname:"abc",author:"andy",price:300,status:"A"},
            {bid:3,bname:"mno",author:"paul",price:180,status:"A"},
            {bid:4,bname:"pqr",author:"andy",price:250,status:"A"},
            {bid:5,bname:"xyz",author:"anna",price:400,status:"NA"}]);

db.books.find().pretty();

db.books.mapReduce(function(){emit(this.author,this.price);},
                function(author,prices){return Array.sum(prices)},
                {query:{status:"A"},
                out:"AuthorTotalPrices"});

db.AuthorTotalPrices.find().pretty()
```