DBMS SQL QUERIES

Week -2

Consider the Employee database given below. The primary keys are underlined. Assume relevant data types for attributes.

EMPLOYEE (Fname, Lname, SSN, Addrs, Sex, Salary, SuperSSN, Dno)

DEPARTMENT (Dname, Dnumber, MgrSSN, MgrStartDate)

PROJECT(Pno, Pname, Dnum)

WORKS_ON (ESSN, Pno, Hours)

1) Display all the details of all employees working in the company.
   select * from employee;
2) Display ssn, lname, fname, address of employees who work in department no 7.
   select ssn,lname,fname,address
   from employee
   where dno=7;
3) Retrieve the birthdate and address of the employee whose name is 'Franklin T.Wong'
   select bdate,address
   from employee
   where fname="Franklin" and mname="T" and lname="Wong";
4) Retrieve the name and salary of every employee
   select fname,mname,lname,salary
   from employee;
5) Retrieve all distinct salary values
   select distinct salary from employee;
6) Retrieve all employee names whose address is in 'Bellaire'
   select fname,mname,lname from employee where address="Bellaire";
7) Retrieve all employees who were born during the 1950s
   select fname from employee where bdate between #01-01-50# and #31-1259#;
8) Retrieve all employees in department 5 whose salary is between 50,000 and 60,000(inclusive)
   select * from employee where dno=5 and salary >=50000 and salary <=60000;

9) Retrieve the names of all employees who do not have supervisors
select fname,mname,lname from employee where superssn is null;

10) Retrieve SSN and department name for all employees
select e.ssn, d.dname from employee e, department d;

Week-3

Consider the insurance database given below. The primary keys are made bold and the data types are specified.
PERSON(**driver_id**:string , name:string , address:string)
CAR(**regno**:string , model:string , year:int)
ACCIDENT(**report_number**:int , accd_date:date , location:string)
OWNS(**driver_id**:string , **regno**:string)
PARTICIPATED(**driver_id**:string , **regno**:string , **report_number**:int , damage_amount:int)
1)Create the above tables by properly specifying the primary keys and foreign keys.
2)Enter at least five tuples for each relation.
3)Demonstrate how you
a.Update the damage amount for the car with specific regno in the accident with report number 12 to 25000.
b.Add a new accident to the database.
4)Find the total number of people who owned cars that were involved in accidents in the year 2008.
5)Find the number of accidents in which cars belonging to a specific model were involved.

create table person(driver_id varchar(10),name varchar(10),address varchar(10),primary key(driver_id));

SQL> create table car(regno varchar(10),model varchar(10),year int,primary key(regno));

SQL> create table accident(report_number int,accd_date date,location varchar(10),primary key(report_number));

SQL> create table owns(driver_id varchar(10),regno varchar(10),primary key(driver_id,regno),foreign key(driver_id) references person(driver_id),foreign key(regno) references car(regno));

```
SQL> create table participated(driver_id varchar(10),regno
varchar(10),report_number int,damage_amount int,primary
key(driver_id,regno,report_number),foreign key(driver_id) references
person(driver_id),foreign key(regno) references car(regno),foreign
key(report_number) references accident(report_number));
```

2.

```
SQL> insert into person values('&driver_id','&name','&address');

SQL> insert into  car values('&regno','&model',&year);

SQL> insert into accident
values(&report_number,'&accd_date','&location');

SQL> insert into owns values('&driver_id','&regno');

SQL> insert into participated
values('&driver_id','&regno',&report_number,&damage_amount);
```

3a.

```
SQL> update participated set damage_amount=25000 where
report_number=12 and regno='5';
```

3b.

```
SQL> insert into accident
values(&report_number,'&accd_date','&location');

SQL> insert into participated
values('&driver_id','&regno',&report_number,&damage_amount);
```

4.

```
SQL> select count(distinct o.driver_id) as People from owns
o,participated p,accident a where a.accd_date like
 '%08' and o.regno=p.regno and p.report_number=a.report_number;
```

5.

SQL> select count(*) as Totalcars from car c,participated p where c.regno=p.regno and c.model='Alto';

week-4

Consider the following relations for a order processing database application in a company.
CUSTOMER(**custno**:int , cname:string , city:string)
ORDER(**orderno**:int , odate:date , custno:int , ord_amt:int)
ORDER_ITEM(**orderno**:int , **itemno**:int , quantity:int)
ITEM(**itemno**:int , unitprice:int)
SHIPMENT(**orderno**:int , **warehouseno**:int , ship_date:date)
WAREHOUSE(**warehouseno**:int , city:string)
1)Create the above tables by properly specifying the primary keys and foreign keys.
2)Enter at least five tuples for each relation.
3)Produce a listing: custname , No_of_orders , Avg_order_amount , where the middle column is the total number of orders by the customer and the last column is the average order amount for that customer.
4)List the orderno for orders that were shipped from ***all*** the warehouses that the company has in a specific city.
5)Demonstrate the deletion of an item from the ITEM table and demonstrate a method of handling the rows in the ORDER_ITEM table that contains this particular item.
insert into customer values(&custno,'&cname','&city');

SQL> insert into order1 values(&orderno,'&odate',&custno,&ord_amt);

SQL> insert into item values(&itemno,&unitprice);

SQL> insert into order_item values(&orderno,&itemno,&quantity);

SQL> insert into warehouse values(&warehouseno,'&city');

SQL> insert into shipment
values(&orderno,&warehouseno,'&ship_date');

3.

SQL> select c.custno,count(*) as No_of_orders,avg(o.ord_amt) as Avg_order_amount from customer c,order1 o where o.custno=c.custno group by c.custno;

4.

SQL> select distinct s.orderno from shipment s where not exists((select warehouseno from warehouse where city='Bangalore') minus (select w.warehouseno from shipment w where w.orderno=s.orderno))and exists (select warehouseno from warehouse where city='Bangalore');

OR

SQL> select s.orderno from shipment s,warehouse w where s.warehouseno=w.warehouseno and w.city='Bangalore' group by orderno having count(*)=(select count(*) from warehouse where city='Bangalore') and not(count(*)=0);

delete from item where itemno=3;

week 5

Consider the following database of student enrollment in courses and books adopted for that course.
STUDENT(regno:string , name:string , major:string , bdate:date)
COURSE(courseno:int , cname:string , dept:string)
ENROLL(regno:string , courseno:int , sem:int , marks:int)
BOOK_ADOPTION(courseno:int , sem:int , book_isbn:int)
TEXT(book_isbn:int , book_title:string , publisher:string , author:string)
1)Create the above tables by properly specifying the primary keys and foreign keys.
2)Enter atleast five tuples for each relation.
3)Demonstrate how you add a new text book to the database and make this book to be adopted by some department.

4)Produce a list of text books (includes courseno , book_isbn , book_title) in the alphabetical order for courses offered by the 'CS' department that use more than two books.
5)List any department that has all its books published by a specific publisher.SQL> create table student(regno varchar(10),name varchar(10),major varchar(10),bdate date,primary key(regno));

SQL> create table course(courseno int,cname varchar(10),dept varchar(10),primary key(courseno));

SQL> create table enroll(regno varchar(10),courseno int,sem number(1),marks number(3),primary key(regno,courseno,sem),foreign key(regno) references student(regno),foreign key(courseno) references course(courseno));

SQL> create table text(book_isbn int,book_title varchar(10),publisher varchar(10),author varchar(10),primary key(book_isbn));

SQL> create table book_adoption(courseno int,sem int,book_isbn int,primary key(courseno,sem),foreign key(courseno) references course(courseno),foreign key(book_isbn) references text(book_isbn));

2.

SQL> insert into student values('&regno','&name','&major','&bdate');

SQL> insert into course values(&courseno,'&cname','&dept');

SQL> insert into enroll values('&regno',&courseno,&sem,&marks);

SQL> insert into text values(&book_isbn,'&book_title','&publisher','&author');

SQL> insert into book_adoption values(&courseno,&sem,&book_isbn);
4.

SQL> select b.courseno,t.book_isbn,t.book_title from book_adoption b,text t,course c where c.courseno=b.courseno and c.dept='CS' and b.book_isbn=t.book_isbn and c.courseno in (select courseno from (select courseno from book_adoption group by courseno,book_isbn) group by courseno having count(*)>=2) order by t.book_title;

5.

SQL> select distinct c.dept from course c where not exists ((select b.book_isbn

from book_adoption b,course c1 where c1.courseno=b.courseno and c1.dept=c.dept) minus (select book_isbn from text where publisher='BPB')) and exists (select b.book_isbn from book_adoption b,course c1 where c1.courseno=b.courseno and c1.dept=c.dept);

Week 6

The following are maintained by abook dealer.
AUTHOR(**author_id**:int , name:string , city:string , country:string)
PUBLISHER(**publisher_id**:int , name:string , city:string , country:string)
CATALOG(**book_id**:int , title:string , author_id:int , publisher_id:int , category_id:int , year:int , price:int)
CATEGORY(**category_id**:int , description:string)
ORDER_DETAILS(**order_no**:int , **book_id**:int , quantity:int)
1)Create the above tables by properly specifying the primary keys and foreign keys.
2)Enter at least five tuples for each relation.
3)Give the details of the authors who have 2 or more books in the catalog and the price of the books is greater than the average price of the books in the catalog and the year of publication is after 2000.
4)Find the author of the book that has maximum sales.
5)Demonstrate how you increase the price of books published by a specific publisher by 10%.

SQL> create table author(author_id int,name varchar(10),city varchar(10),country varchar(10),primary key(author_id));

SQL> create table publisher(publisher_id int,name varchar(10),city varchar(10),country varchar(10),primary key(publisher_id));

SQL> create table category(category_id int,description varchar(10),primary key(category_id));

SQL> create table catalog(book_id int,title varchar(10),author_id int,publisher_id int,category_id int,year int,price int,primary key(book_id),foreign key(author_id) references author(author_id),foreign key(publisher_id) references publisher(publisher_id),foreign

key(category_id) references category(category_id));

SQL> create table order_details(order_no int,book_id int,quantity int,primary key(order_no,book_id),foreign key(book_id) references catalog(book_id));

2.

SQL> insert into author values(&author_id,'&name','&city','&country');

SQL> insert into publisher values(&publisher_id,'&name','&city','&country');

SQL> insert into category values(&category_id,'&description');

SQL> insert into catalog values(&book_id,'&title',&author_id,&publisher_id,&category_id,&year, &price);

SQL> insert into order_details values(&order_no,&book_id,&quantity);

3.

SQL> select * from author where author_id in (select author_id from catalog where year>2000 and price>(select avg(price) from catalog) group by author_id having count(*)>=2);

4.

SQL> select a.author_id,a.name,a.city,a.country from author a,catalog c where c.author_id=a.author_id and c.book_id=(select book_id from order_details group by book_id having sum(quantity)=(select max(quantity) from (select sum(quantity) as quantity from order_details group by book_id)));

5.

SQL> update catalog set price=1.1*price where publisher_id=(select publisher_id from publisher where name='BPB');

Week 7

Consider the following database for a banking enterprise.
BRANCH(**branch_name**:string , branch_city:string , assets:real)
ACCOUNT(**accno**:int , branch_name:string , balance:real)
DEPOSITOR(**customer_name**:string , **accno**:int)
CUSTOMER(**customer_name**:string , customer_street:string , customer_city:string)
LOAN(**loan_number**:int , branch_name:string , amount:real)
BORROWER(**customer_name**:string , **loan_number**:int)
1)Create the above tables by properly specifying the primary keys and foreign keys.
2)Enter at least five tuples for each relation.
3)Find ***all*** the customers who have at least two accounts at the ***main*** branch.
4)Find all the customers who have an account at ***all*** the branches located in a specific city.
5)Demonstrate how you delete all account tuples at every branch located in a specific city.

create table branch(branch_name varchar(10),branch_city varchar(10),assets int,primary key(branch_name));

SQL> create table account(accno int,branch_name varchar(10),balance int,primary key(accno));

SQL> create table customer(customer_name varchar(10),customer_street varchar(10),customer_city varchar(10),primary key(customer_name));

SQL> create table depositor(customer_name varchar(10),accno int,primary key(customer_name,accno),foreign key(customer_name) references customer(customer_name),foreign key(accno) references account(accno) on delete cascade);

SQL> create table loan(loan_number int,branch_name varchar(10),amount int,primary key(loan_number),foreign

key(branch_name) references branch(branch_name));

SQL> create table borrower(customer_name varchar(10),loan_number int,primary key(customer_name,loan_number),foreign key(customer_name) references customer(customer_name),foreign key(loan_number) references loan(loan_number));

2.

SQL> insert into branch values('&branch_name','&branch_city',&assets);

SQL> insert into account values(&accno,'&branch_name',&balance);

SQL> insert into customer values('&customer_name','&customer_street','&customer_city');

SQL> insert into depositor values('&customer_name',&accno);

SQL> insert into loan values(&loan_number,'&branch_name',&amount);

SQL> insert into borrower values('&customer_name',&loan_number);

3.

SQL> select d.customer_name from depositor d,account a where a.accno=d.accno and a.branch_name='KRMarket' group by d.customer_name having count(*)>=2;

4.

SQL> select c.customer_name from customer c where exists(select branch_name from branch where branch_city='Bangalore') and not exists ((select branch_name from branch where branch_city='Bangalore') minus (select b.branch_name from branch b,account a,depositor d where d.customer_name=c.customer_name and d.accno=a.accno and a.branch_name=b.branch_name));

5.

SQL> delete from account where accno in (select a.accno from account

a,branch b where a.branch_name=b.branch_name and
b.branch_city='Bangalore');

WEEK 8

Consider the Cricket database given below. The primary keys are underlined. Assume relevant data types for attributes.

PLAYER (PId, Lname, Fname, Country, Yborn, Bplace)
MATCH (MatchId, Team1,Team2, Ground, Date, Winner)
BATTING (MatchId, Pid, Nruns, Fours, Sixes)
BOWLING (MatchId, Pid, Novers, Maidens, Nruns, Nwickets)

Create the above tables in SQL. Specify primary and foreign keys properly. Enter at least 5 tuples in each table with relevant data. Solve the following queries.

i. Display the sorted list of ground names where Australia has played as team1
ii. Find the match information of all matches in which Dhoni did batting.
iii. Find the names of players who did batting in match 2689

```
create table PLAYER (

Pid int,

Lname varchar(10),

Fname varchar(10),

Country varchar(10),

Yborn date, Bplace varchar(10),

primary key (Pid)

);

create table MATCHING (

MatchID int,

Team1 varchar(10),

Team2 varchar(10),

Ground varchar(10),

Date date, Winner varchar(10),

primary key(MatchID)

);

create table BATTING (

MatchID int,
```

Pid int,

Nruns int,

Fours int, Sixes int,

primary key(MatchID,Pid),foreign key(MatchID) references MATCHING(MatchID),foreign key(Pid) references PLAYER(Pid)

);

create table BOWLING (

MatchID int,

Pid int,

Novers int,

Maidens varchar(10),

Nruns int, Nwickets int,

primary key(MatchID,Pid),foreign key(MatchID) references MATCHING(MatchID),foreign key(Pid) references PLAYER(Pid)

);

//insert values

insert into PLAYER values("1,"aa","Dhoni","india","01-jan-88","chennai"); insert into MATCHING values("12,"Australia","D2","china","01-jan-2012","india"); insert into BATTING values("12,1,20,4,4); insert into BOWLING values("12,1,12,"cc",20,2);

//Display the sorted list of ground names where Australia has played as team1.

SELECT Ground

FROM MATCHING

WHERE Team1='Australia' ORDER BY Ground;

//Find the match information of all matches in which Dhoni did batting.

SELECT *

FROM (PLAYER natural join MATCHING) natural join BATTING WHERE Fname='Dhoni';

//Find the names of players who did batting in match 2689.

SELECT Fname,Lname

FROM (PLAYER natural join MATCHING) natural join BATTING WHERE MatchID=2686;

//8B)Consider the following Movie table with the following attributes -

Actor_name,Actor_id, Actor_birthdate , Dirctor_name,Director_id, Director_birthdate, film_title, year of production ,type (thriller, comedy, etc.)

Create 10 collections with data relevant to the following questions. Write and execute MongoDB queries:

i. List all the movies acted by John and Elly in the year 2012. ii. List only the name and type of the movie where Ram has acted sorted by movie names

```
db.createCollection("movie")
```

```
{ "ok" : 1 }
```

```
db.movie.insert([{act_n:"ram",act_id:13,act_bdate:"2/3/1997",dir_n:"williams",dir_id:101,dir_bdate:"12/9/1987",film:"battleship",year:2015,type:"thriller"}])
```

```
db.movie.insert([{act_n:"john",act_id:11,act_bdate:"1/2/1998",dir_n:"ram",dir_id:100,dir_bdate:"2/3/1997",film:"john wick",year:2012,type:"killer"}])
```

```
db.movie.insert([{act_n:"elly",act_id:12,act_bdate:"4/12/1998",dir_n:"ram",dir_id:100,dir_bdate:"2/3/1997",film:"aquaman",year:2012,type:"action"}])
```

```
db.movie.insert([{act_n:"ram",act_id:13,act_bdate:"2/3/1997",dir_n:"thomas",dir_id:103,dir_bdate:"12/3/1999",film:"xxx",year:2018,type:"action"}])
```

```
db.movie.insert([{act_n:"john",act_id:11,act_bdate:"1/2/1998",dir_n:"ram",dir_id:100,dir_bdate:"2/3/1997",film:"mr.bean",year:2018,type:"comedy"}])
```

```
//List all the movies acted by John and Elly in the year 2012. db.movie.find({$and : [{act_n:{$in : ["john","elly"]}},{year:2012}]},{film:1}).pretty()
```

```
{ "_id" : ObjectId("5c273e247fd87cec5944fabf"), "film" : "john wick" } { "_id" : ObjectId("5c273e2e7fd87cec5944fac0"), "film" : "aquaman" }
```

```
//ii. List only the name and type of the movie where Ram has acted sorted by movie names. db.movie.find({act_n:"ram"},{film:1,type:1}).sort({film:1}).pretty()
```

```
{ "_id" : ObjectId("5c273ddb7fd87cec5944fabd"), "film" : "battleship", "type" : "thriller" }
```

```
{ "_id" : ObjectId("5c273ef97fd87cec5944fac1"), "film" : "xxx", "type" : "action" }
```

Week 9

Consider the Aircraft database given below. The primary keys are underlined. Assume relevant data types for attributes.

AIRCRAFT (Aircraft ID, Aircraft_name, Cruising_range)
CERTIFIED (Emp ID, Aircraft ID)
EMPLOYEE (Emp ID, Ename, Salary)

Create the above tables in SQL. Specify primary and foreign keys properly. Enter at least 5 tuples in each table with relevant data. Solve the following queries.

i. Find the names of pilots certified for Boeing aircraft

ii. Arrange the Aircrafts with respect to the ascending order of distance. iii. Find the name of pilots who can operate flights with a range greater than 3000 miles but are not certified on any Boeing aircraft.

```
Create table aircraft(

aid varchar(9)
primary key, aname
varchar(10), crange
int

);


Create table
employees( eid
varchar(9) primary
key, ename
varchar(10), salary
int

);


Create table certified(

eid varchar(9)references employees(eid),

aid varchar(9)references aircraft(aid)

);

insert into aircraft values('B001','Boeing',4000);
insert into aircraft values('B002','Boeing',2500);
insert into aircraft
values('BB003','Blackbeard',6000); insert into
aircraft values('S004','Supermarine',8000); insert
into aircraft values('L005','Lockheed',2100);

insert into employees
values(1,'Johnny',40000); insert into
employees values(2,'Timmy',60000); insert
into employees values(3,'Lawrence',70000);
insert into employees values(4,'Zuzu',90000);
insert into employees values(5,'Matt',80000);

insert into certified values(1,'B001');
insert into certified values(1,'B002');
insert into certified values(3,'S004');
insert into certified values(4,'S004');
insert into certified values(5,'L005');
insert into certified values(2,'B002');
```

```
insert into certified
values(4,'BB003'); insert into
certified values(3,'BB003'); insert
into certified values(4,'L005');

//Find names of pilots certified to fly Boeing

SELECT DISTINCT E.ename

FROM employees E, certified C, aircraft A

WHERE E.eid = C.eid AND C.aid = A.aid AND A.aname='Boeing';

//Arrange flight no with respect to ascending order of distance

SELECT aid

FROM aircraft

ORDER BY crange ASC;

//Find the name of pilots who can operate flights with a range greater than 3000 miles but are not certified on any Boeing aircraft.

select distinct(ename) from employees E,certified C, aircraft A where A.crange > 3000 and C.aid NOT in(select aid from aircraft A where A.aname='Boeing') and E.eid = C.eid
```

Week 10

Consider the Supply-Parts database given below. The primary keys are underlined. Assume relevant data types for attributes.

SUPPLIER (Sid, Sname, Address)
PART (PID, Pname, Color)
SHIPMENT (Sid, PID, Cost)

Create the above tables in SQL. Specify primary and foreign keys properly. Enter at least 5 tuples in each table with relevant data. Solve the following queries.

i. Find the Sid's of suppliers who supply a green part
ii. For every supplier print the name of the supplier and the total number of parts that he/she supplies
iii. Update the part color supplied by supplier s3 to yellow

```
CREATE TABLE SUPPLIER

(

Sid int NOT NULL PRIMARY KEY,

Sname varchar(16) NOT NULL UNIQUE,

Address varchar(20) NOT NULL

);
```

```
CREATE TABLE PART
(
Pid int NOT NULL PRIMARY KEY,
Pname varchar(18) NOT NULL,
Color varchar(10) NOT NULL,
);
CREATE TABLE SHIPMENT
(
Sid int NOT NULL REFERENCES SUPPLIER,
Pid int NOT NULL REFERENCES PART,
Cost int NOT NULL,
PRIMARY KEY (Sid, Pid)
);
CREATE TABLE SHIPMENT
(
Sid int,
Pid int, Cost
int, primary
key(Sid,Pid),
foreign key(Sid) references SUPPLIER(Sid),
foreign key(Pid) references PART(Pid));

INSERT INTO SUPPLIER VALUES (1, 'Smith', 'London');
INSERT INTO SUPPLIER VALUES (2, 'Jones', 'Paris');
INSERT INTO SUPPLIER VALUES (3, 'Blake', 'Paris');
INSERT INTO SUPPLIER VALUES (4, 'Clark', 'London');
INSERT INTO SUPPLIER VALUES (5, 'Adams', 'Athens');

INSERT INTO PART VALUES (1, 'Nut', 'Red');
INSERT INTO PART VALUES (2, 'Bolt', 'Green');
INSERT INTO PART VALUES (3, 'Screw', 'Blue');
INSERT INTO PART VALUES (4, 'Screw', 'Red');
INSERT INTO PART VALUES (5, 'Cam', 'Blue');
```

```
INSERT INTO PART VALUES (6, 'Cog', 'Red');

INSERT INTO SHIPMENT VALUES (1, 1, 300);
INSERT INTO SHIPMENT VALUES (1, 2, 200);
INSERT INTO SHIPMENT VALUES (1, 3, 400);
INSERT INTO SHIPMENT VALUES (1, 4, 200);
INSERT INTO SHIPMENT VALUES (1, 5, 100);
INSERT INTO SHIPMENT VALUES (1, 6, 100);
INSERT INTO SHIPMENT VALUES (2, 1, 300);
INSERT INTO SHIPMENT VALUES (2, 2, 400);
INSERT INTO SHIPMENT VALUES (3, 2, 200);
INSERT INTO SHIPMENT VALUES (4, 2, 200);
INSERT INTO SHIPMENT VALUES (4, 4, 300);
INSERT INTO SHIPMENT VALUES (4, 5, 400);

//Find the Sids of suppliers who supply green part
SELECT Distinct S.Sid
FROM SUPPLIER S, PART P, SHIPMENT C
WHERE C.Sid=S.Sid AND P.PID=C.PID AND P.Color like 'Green';

//For every supplier print the name of the supplier and the total number of parts that he/she supplies.
SELECT S.Sname, COUNT(*) as PartCount
FROM SUPPLIER S, SHIPMENT C, PART P
WHERE C.Sid = S.Sid and P.PID = C.PID
GROUP BY S.Sname, S.Sid

//Update part color supplied by supplier S3 to Yellow
UPDATE PART
SET Color='Yellow'
WHERE PID IN (SELECT C.PID FROM  SUPPLIER S, SHIPMENT C WHERE C.Sid=S.Sid  and C.Sid=3);
```