1. A) Create the following relation for the student :-
Student(regno:string,name:string,class:string,b date:date,marks1:int,marks1:int, marks2:int,marks3:int)
Create the above tables by properly specifying the primary keys & foreign keys:
i. Enter at least five tuples of the above relation.
ii. Demonstrate the usage of following clauses for the above relation.
a. Where c. Having
b. Order By d. GroupBy
iii. Demonstrate the usage of following clauses for the above relation.
a. Sum c. Count e. Between
b. Avg d. Like f. Max & Min

iv. Demonstrate the rollback and commit command for the above relation.

b)Write PL/SQL program to demonstrate user defined exception handling.

```
create table student(
    regno varchar2(20) primary key,
    name varchar2(20),
    class varchar2(20),
    bdate date,
    marks1 number(3),
    marks2 number(3),
    marks3 number(3)
);
```

i. Enter at least five tuples of the above relation.

```
  insert into student values('001','jafar','BCA','12-
OCT-2004',45,54,34);
  insert into student
values('002','shaneef','BBA','12-nov-
2003',99,98,78);
  insert into student
values('005','humaid','BCA','10-dec-
2003',87,67,97);
  insert into student
values('009','mudasir','BCA','19-jan-
2004',98,89,97);
  insert into student
values('010','ruvaid','BCOM','10-feb-
2005',76,56,98);
```

ii. Demonstrate the usage of following clauses for the above relation.

a. Where c. Having

b. Order By d. GroupBy

```
 select * from student where class='BCOM';
 select * from student order by class desc;
 select class from student group by class;
 select regno,name,marks1 from student group
by regno,name,marks1
       having marks1 > 95;
```

iii. Demonstrate the usage of following clauses for the above relation.

a. Sum c. Count e. Between

b. Avg d. Like f. Max & Min

```
 select count('regno') as count from student ;
 select avg(marks1) as avg_marks1 from student
;
```

select sum(marks1) as sum_marks2 from student ;
select max(marks2) as sum_marks2 from student ;
select min(marks2) as sum_marks2 from student ;
select * from student where marks1 between 80 and 100;
select * from student where class like 'BCA';

iv. Demonstrate the rollback and commit command for the above relation.

commit;

insert into student values('011','raif','BCA','16-DEC-2005',95,53,23);

Select * from student;

```
rollback;

  Select * from student;
```

b)Write PL/SQL program to demonstrate user defined exception handling.

```
DECLARE
    invalid_salary EXCEPTION;
    salary NUMBER(10,2);
BEGIN
    salary := &salary;
    IF salary < 2000 THEN
        RAISE invalid_salary;
    END IF;
    DBMS_OUTPUT.PUT_LINE('Salary = ' ||
salary);
```

```
EXCEPTION
   WHEN invalid_salary THEN
      DBMS_OUTPUT.PUT_LINE('Error: Salary is too low. It must be greater than or equal to 2000');
   WHEN OTHERS THEN
      DBMS_OUTPUT.PUT_LINE('An unexpected error occurred: ' || SQLERRM);
END;
/
```

---

2) Consider the following database that maintain information about employees & Departments :-
Employee(empid:int,ename:string,age:int,salary:int,#deptno:int)
Department(deptno:int,dname:string,#manager-id:int)

Create the above tables by properly specifying the primary keys & foreign keys.

i. Enter at least 5 tuples for each relation.

ii. Display emp-id & emp name whose salary lies between 10,000 and 50,000.

iii. List emp name & salary for all the employee working for CS Dept.

iv. Display emp name & dept name for all the manager.

b)Write PL/SQL program to insert a new row (INSERT INTO command).

```
create table department(
   deptno int primary key,
   name varchar2(20),
   manager_id number(10)
   );
```

```
create table employee (
    empid int primary key,
    name varchar2(20),
    age int,
    salary int,
    deptno int references department
   );
```

i. Enter at least 5 tuples for each relation.

```
insert into department values(20,'CS',12);
insert into department values(15,'ACCOUNT',8);
insert into department values(1,'SALES',17);
insert into department values(6,'HR',20);
insert into department
values(7,'MARKETING',21);

insert into employee values
```

(121,'hasan',40,50000,20);
insert into employee values
(122,'mudasir',20,1000000,7);
insert into employee values
(123,'fahmaan',23,20,15);
insert into employee values
(124,'noman',18,500000,6);
insert into employee values
(125,'afreed',19,50,1);

ii. Display emp-id & emp name whose salary lies between 10,000 and 50,000.

select name,empid from employee where salary between 1 and 50;

iii. List emp name & salary for all the employee working for CS Dept.

```
select  name,salary from employee where
deptno=20;
```

iv. Display emp name & dept name for all the manager.

```
select name,dname from employee,department
where employee.deptno=department.deptno;
```

b)Write PL/SQL program to insert a new row (INSERT INTO command).

```
create table emp (
empno int primary key,
name varchar2(20),
salary int,
gender char(1)
);

insert into emp values (1,'hasan',2000,'m');
```

```
set serveroutput on;

declare
var varchar2(40):='i love india';
begin
   dbms_output.put_line(var);
insert into emp values (2,'kaif',4000,'m');
end;
/
```

---

3. Consider the following schema for Order Database :-

SALESMAN(Salesman_id,Name,City,Commission)

CUSTOMER(Customer_i, Cust_Name, City, Grade, Salesman_id)

ORDERS(Ord_No, Purchase_Amt, Ord_Date, #Customer_id,Salesman_id)

Create the above tables by properly specifying the primary keys & foreign keys.

Enter at least five tables for each relation.

Write SQL queries to

i. Count the customers with grades above Bangalore's average.

ii. Find the name and numbers of all salesmen who had more than one customer.

iii. List all salesmen and indicate those who have and don't have customers in their cities (Use UNION operation.)

iv. Create a view that finds the sales man who has the customer with the highest order of a day.

v. Demonstrate the DELETE operation by removing salesman with id 101.

All his orders must also be deleted.

b)Write PL/SQL program to demonstrate %ROWCOUNT attribute.

```
create table salesman
    (
    salesman_id int primary key,
    name varchar(20),
    city varchar(20),
    commission varchar(20)
    );

create table customer
   (
  customer_id int primary key,
  cust_name varchar(20),
  city varchar(20),
   grade int,
   salesman_id int,
   foreign key (salesman_id) references salesman
(salesman_id) on delete set null
   );

create table orders
(
ord_no int primary key,
purchase_amt decimal (10,2),
ord_date date,
customer_id int,
salesman_id int,
foreign key (customer_id) references customer
(customer_id) on delete cascade,
foreign key (salesman_id) references salesman
(salesman_id) on delete cascade
);
```

```
insert into salesman values
(001,'azhar','bhtkal','18%');
insert into salesman values
(002,'shaneef','manglore','20%');
insert into salesman values
(003,'kaif','kundapur','10%');
insert into salesman values
(004,'jafar','banglore','20%');
insert into salesman values
(005,'ruvaid','karwar','20%');


insert into customer values
(101,'raif','bhatkal',10,001);
insert into customer values
(102,'uzair','dehli',20,002);
```

```
insert into customer values
(103,'kasim','banglore',20,003);
insert into customer values
(104,'shaeef','goa',20,003);
insert into customer values
(105,'faizaan','goa',20,004);
insert into customer values
(106,'arshad','banglore',15,004);

insert into orders values (201,100000,'22-oct-
2022',101,001);
insert into orders values (202,10000,'31-mar-
2020',102,002);
insert into orders values (203,4321,'13-dec-
2021',103,002);
insert into orders values (204,97821,'17-jan-
2023',104,004);
```

insert into orders values (205,30000,'23-jan-2022',105,005);

i. Count the customers with grades above Bangalore's average.

```
 select grade,count (customer_id) as
count_of_customer
  from customer
 where grade >
    (select avg(grade)
      from customer
     where city='banglore'
 )
group by grade;
```

ii. Find the name and numbers of all salesmen who had more than one customer.

```
select name ,salesman_id
from salesman a
where  1< (select count(*)
 from customer c
where c.salesman_id=a.salesman_id
);
```

iii. List all salesmen and indicate those who have and don't have customers in their cities (Use UNION operation.)

```
SELECT s.salesman_id,
      s.name,
      c.cust_name,
      s.commission
FROM salesman s
JOIN customer c ON s.salesman_id =
c.salesman_id
UNION
```

```
SELECT s.salesman_id,
       s.name,
       'no_match' AS cust_name,
       s.commission
FROM salesman s
WHERE s.salesman_id NOT IN (SELECT
salesman_id FROM customer);
```

iv. Create a view that finds the sales man who has the customer with the highest order of a day.

```
CREATE VIEW vw_highest_order AS
SELECT o.ord_date, s.name, s.salesman_id
FROM orders o, salesman s
WHERE o.salesman_id = s.salesman_id
  AND o.purchase_amt = (
     SELECT MAX(purchase_amt)
     FROM orders
     WHERE ord_date = o.ord_date
  );
```

iv. Create a view that finds the sales man who has the customer with the highest order of a day.

v. Demonstrate the DELETE operation by removing salesman with id 101. All his orders must also be deleted.

```
DELETE FROM salesman
WHERE salesman_id = 4;
```

b)Write PL/SQL program to demonstrate %ROWCOUNT attribute.

```
create table emp (
empno int primary key,
name varchar2(20),
salary int,
gender char(1)
);

insert into emp values (1,'hasan',2000,'m');



set serveroutput on;
begin
   insert into emp values (3,'raju',6000,'m');
dbms_output.put_line('number of record
inserted:'||to_char(sql%rowcount));
end;
/
```

4.Consider the Insurances database given below. The primary keys are underlined and the datatypes are specified :-

PERSON(DRIVER- D#:string, name:string, address:string)
CAR (Regno: string, model:string,year:int)
ACCIDENT(report-number: int, date:date,location:string)
OWNS(#driver-id:string, #Regno: string)
PARTICIPATED (#driver-id:string, #Regno:string, #report-number: int,Damage amount:int)
Create the above tables by property specifying the primary keys and the foreign keys.
Enter at least five tables for each relation.
Write SQL queries to
i. Demonstrate how you
a. Update the damage amount for the car with a specific Reg. no in the accident with report number 1 to 3.
b. Add a new accident to the database.

ii. Find the total number of people who owned cars that were involved in accident since 2018.
iii. Find the total number of accidents in which cars belonging to a specific model were involved.

b)Write PL/SQL program demonstrate exception handling for the above query v

```
create table person(
driver_id varchar2(20) primary key,
d_name varchar2(20),
address varchar2(100)
);

create table car(
reg_no varchar2(20) primary key,
model varchar2(20),
year int
);

create table accident(
report_number int primary key,
r_date date,
location varchar2(40)
);

create table owns(
driver_id varchar2(20),
reg_no varchar2(20),
foreign key (driver_id) references person
(driver_id),
foreign key (reg_no) references car (reg_no)
);
```

```
create table participated(
driver_id varchar2(20),
reg_no varchar2(20),
report_number int,
damage_amount int,
foreign key (driver_id) references person
(driver_id),
foreign key (reg_no) references car (reg_no),
foreign key (report_number) references accident
(report_number)
);

insert into person values('d1','kaif','banglore');
insert into person values('d2','rahul','goa');
insert into person
values('d3','mudasir','manglore');
```

```
insert into person
values('d4','ruvaid','manglore');
insert into person values('d5','raid','bhatkal');

insert into car values('ka-47-01','thar',2020);
insert into car values('ka-47-02','omani',2000);
insert into car values('ka-47-03','swift',2023);
insert into car values('ka-47-04','Ford',2002);
insert into car values('ka-47-05','Tesla ',2024);

insert into accident values(1,'12-dec-
2023','sagar');
insert into accident values(2,'31-dec-
2021','mumbai');
insert into accident values(3,'12-jan-
2024','karwar');
insert into accident values(4,'17-dec-
2002','banglore');
```

```
insert into accident values(5,'15-feb-
2023','udupi');

insert into owns values('d1','ka-47-01');
insert into owns values('d2','ka-47-02');
insert into owns values('d3','ka-47-03');
insert into owns values('d4','ka-47-04');
insert into owns values('d5','ka-47-05');

insert into participated values('d1','ka-47-
01',1,10000);
insert into participated values('d2','ka-47-
02',2,10200);
insert into participated values('d3','ka-47-
03',3,20000);
insert into participated values('d4','ka-47-
04',4,9000);
```

insert into participated values('d5','ka-47-05',5,90000);

i. Demonstrate how you
a. Update the damage amount for the car with a specific Reg. no in the accident with report number 1 to 3.

```
update participated set damage_amount=25000
    where(reg_no='ka-47-03' and report_number
between 1 and 3);
```

b. Add a new accident to the database.

```
insert into accident values(6,'21-feb-
2023','bhatkal');
```

ii. Find the total number of people who owned cars that were involved in accident since 2018.

```
SELECT COUNT(driver_id)
      FROM accident, participated
      WHERE r_date BETWEEN '01-JAN-2023' AND
'21-DEC-2023'
      AND accident.report_number =
participated.report_number;
```

iii. Find the total number of accidents in which cars belonging to a specific model were involved.

```
SELECT COUNT(car.model)
    FROM accident, participated, car
     WHERE accident.report_number =
participated.report_number
       AND car.reg_no = participated.reg_no
       AND car.model='omani';
```

b)Write PL/SQL program demonstrate exception handling for the above query v

```
BEGIN
   DECLARE
      num NUMBER := 10;
      a NUMBER := 0;
      result NUMBER;

   BEGIN
      -- Attempt division
      result := num / a;
      DBMS_OUTPUT.PUT_LINE('Result: ' ||
result);
   EXCEPTION
      WHEN ZERO_DIVIDE THEN
         DBMS_OUTPUT.PUT_LINE('Error: Division
by zero occurred.');
   END;
```

```
END;
/
```

---

5. The following tables are maintained by a book 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)
Create the above tables by properly specifying the primary keys and the foreign keys.
Enter at least five tuples for each relation.

Write SQL queries to
i. Give the details of the authors who have 2 or more books in the catalog and the price of their total books are greater than the average price of the books in the catalog and the year of publication is after 2000.
ii. Find the author of the book, which has maximum sales.
iii. Demonstrate how you increase the price of books published by a specific publisher by 10%.

b)Write PL/SQL program illustrates how to create and call a function .

```
create table author(
author_id varchar2(20) primary key,
a_name varchar2(20),
a_city varchar2(20),
a_country varchar2(20)
);

create table publisher(
publisher_id varchar2(20) primary key,
p_name varchar2(20),
p_city varchar2(20),
p_country varchar2(20)
);

create table category(
category_id varchar2(20) primary key,
descripition varchar2(20)
);

create table catalog1 (
book_id varchar2(20) primary key,
tittle varchar2(50),
author_id varchar2(20),
publisher_id varchar2(20),
category_id varchar2(20),
year int,
price int,
foreign key (author_id) references author
(author_id),
foreign key (category_id) references category
(category_id)
);

create table order_details (
order_no varchar2(20) primary key,
book_id varchar2(20),
quantatiy number(10),
foreign key (book_id) references catalog1
(book_id)
);
```

```
insert into author values
('a1','shaneef','bhatkal','india');
insert into author values
('a2','nashil','moraco','africa');
insert into author values
('a3','jafar','udupi','india');
insert into author values
('a4','kaif','shirali','india');
insert into author values ('a5','raif','goa','india');

insert into publisher values
('p1','mohammed','banglore','india');
insert into publisher values ('p2','mohan','usman
nagar','pakisatan');
insert into publisher values
('p3','zaheer','gulmi','india');
```

```
insert into publisher values
('p4','umar','bhatkal','india');
insert into publisher values
('p5','raid','goa','india');

insert into category values ('c1','kids');
insert into category values ('c2','horror');
insert into category values ('c3','romantic');
insert into category values ('c4','story');
insert into category values ('c5','action');

insert into catalog1 values ('b1','The Magic
Tree','a1','p1','c1',2022,101);
insert into catalog1 values ('b2','A Day with the
Moon','a1','p2','c3',2002,102);
insert into catalog1 values ('b3','The Lost
Puppy','a2','p5','c2',2023,103);
```

```
insert into catalog1 values ('b4','The Secret
Garden','a4','p2','c5',2002,104);
insert into catalog1 values ('b5','Toms Big
Adventure','a4','p3','c3',2020,105);

insert into order_details values ('1','b1',110);
insert into order_details values ('2','b2',34);
insert into order_details values ('3','b3',1200);
insert into order_details values ('4','b4',234);
insert into order_details values ('5','b5',300);
```

i. Give the details of the authors who have 2 or more books in the catalog and the price of their total books are greater than the average price of the books in the catalog and the year of publication is after 2000.

```
SELECT *
```

```
FROM author a
WHERE a.author_id IN (
   SELECT catalog1.author_id
   FROM catalog1
   WHERE catalog1.year > 2000
   AND catalog1.price > (
      SELECT AVG(price)
      FROM catalog1
   )
   GROUP BY catalog1.author_id
   HAVING COUNT(catalog1.author_id) >= 2
);
```

ii. Find the author of the book, which has maximum sales.

```
SELECT author_id
   FROM catalog1, order_details
   WHERE order_details.book_id =
catalog1.book_id
     AND order_details.quantatiy = (
        SELECT MAX(quantatiy)
        FROM order_details
     );
```

iii. Demonstrate how you increase the price of books published by a specific publisher by 10%.

```
update catalog1
set price=price+(price*0.10)
where publisher_id in
(select publisher.publisher_id from publisher
where publisher.p_name='zaheer');
```

b)Write PL/SQL program illustrates how to create and call a function .

```
create or replace function add_number(num1 in
number,num2 in number)
return number
is
begin
   return num1+num2;
end add_number;
/

declare
 result number;
begin
   result:=add_number(10,20);
   dbms_output.put_line('total='||result);
end;
/
```

6. Consider the following database of student enrolment in courses and books adopted each course :-

STUDENT(regno: string, name:string, major: string, bdate: date)

COURSE (course: int, cname: string, dept: string)

ENROLL (#regno: string, course#:int, sem: int marks:int)

TEXTBOOK(book-ISBN: int,book-title: string, publisher: string, author:string)

BOOK_ADOPTION(course#:int, sem:int,book-ISBN#:int)

Create the above tables by properly specifying the primary keys and the foreign Keys

Enter at least five tuples for each relation.

Write SQL queries to

i. Demonstrate how you add a textbook to the database and make this book be adapted by some department.
ii. Produce list of textbooks (include Course, Book-ISBN, Book-title) in the alphabetical order for
courses offered by the 'SCIENCE' department that use more than two books.
iii. List any department that has its adopted books published by a specific publisher.

b)Write PL/SQL program to demonstrate user defined exception handling.

```
 create table student (
regno int primary key,
name varchar2(20),
major varchar2(20),
bdate date
);

create table course (
course_id varchar2(20) primary key,
cname varchar2(20),
dept varchar2(20)
);

create table enroll (
regno int,
course_id varchar2(20),
sem int,
marks int,
foreign key (regno) references student (regno),
foreign key (course_id) references course
(course_id)
);
```

```
create table textbook (
book_isbn varchar2(20) primary key,
book_tittle varchar2(20),
publisher varchar2(20),
author varchar2(20)
);

create table book_adoption (
course_id varchar2(20),
sem int,
book_isbn varchar2(20),
foreign key (course_id) references course
(course_id) ,
foreign key (book_isbn) references textbook
(book_isbn)
);
```

```
insert into student values ('1','azhar','cs','12-feb-
2004');
insert into student values
('2','shaneef','science','31-jan-2000');
insert into student values
('3','jafar','commerce','12-dec-2002');
insert into student values
('4','sibtain','physics','12-feb-2005');
insert into student values ('5','nashil','maths','12-
dec-2000');

insert into course values ('c1','bca','computer');
insert into course values ('c2','bsc','science');
insert into course values ('c3','bba','business');
insert into course values ('c4','b
com','accountancy');
insert into course values ('c5','m
com','accountancy');
```

```
insert into enroll values (1,'c1',6,91);
insert into enroll values (2,'c2',5,92);
insert into enroll values (3,'c3',4,95);
insert into enroll values (4,'c4',3,96);
insert into enroll values (5,'c5',2,98);

insert into textbook values
('b1','physics','hayaan','akeyas');
insert into textbook values
('b2','java','sehran','ismail');
insert into textbook values
('b3','chemistry','hayaan','akeyas');
insert into textbook values ('b4','python','sehran
','ismail');
insert into textbook values
('b5','biology','hayaan','akeyas');
```

```
insert into book_adoption values ('c1',6,'b1');
insert into book_adoption values ('c2',5,'b2');
insert into book_adoption values ('c1',4,'b3');
insert into book_adoption values ('c2',5,'b4');
insert into book_adoption values ('c1',4,'b5');
```

i. Demonstrate how you add a textbook to the database and make this book be adapted by some department.

```
insert into textbook values
('b6','c++','sehran','ismail');
insert into book_adoption values ('c2',6,'b2');
```

ii. Produce list of textbooks (include Course, Book-ISBN, Book-title) in the alphabetical order for courses offered by the 'SCIENCE' department that use more than two books.

```
SELECT course.course_id,
book_adoption.book_isbn, textbook.book_tittle
FROM course, book_adoption, textbook
WHERE
   course.course_id = book_adoption.course_id
   AND book_adoption.book_isbn =
textbook.book_isbn
   AND course.course_id IN (
      SELECT course.course_id
      FROM course, book_adoption
      WHERE
         course.course_id =
book_adoption.course_id
         AND dept = 'computer'
      GROUP BY course.course_id
      HAVING COUNT(book_adoption.book_isbn)
>= 2
   )
```

```
ORDER BY textbook.book_tittle;
```

iii. List any department that has its adopted books published by a specific publisher.

```
SELECT DISTINCT course.dept
FROM course,book_adoption,textbook
where
textbook.book_isbn=book_adoption.book_isbn
and book_adoption.course_id=course.course_id
and textbook.publisher = 'sehran';
```

b)Write PL/SQL program to demonstrate user defined exception handling.

```
DECLARE
   invalid_salary EXCEPTION;
```

```
    salary NUMBER(10,2);
BEGIN
    salary := &salary;
    IF salary < 2000 THEN
        RAISE invalid_salary;
    END IF;
    DBMS_OUTPUT.PUT_LINE('Salary = ' ||
salary);

EXCEPTION
    WHEN invalid_salary THEN
        DBMS_OUTPUT.PUT_LINE('Error: Salary is
too low. It must be greater than or equal to
2000');
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('An unexpected
error occurred: ' || SQLERRM);
END;
```

/