**Implementation:**

-- Delete the following table if they exist in the database previously

DROP TABLE ROOM;

DROP TABLE BILL;

DROP TABLE APPOINTMENT;

DROP TABLE PATIENT;

DROP TABLE DOCTOR;

-- ----------------------------creation of the TABLE DOCTOR--------------------------------

CREATE TABLE DOCTOR(

doctorId number(7) NOT NULL,

name varchar(30) NOT NULL,

designation varchar(25),

address varchar(35),

phoneNo varchar(20),

gender varchar(6),

birthDate date,

dept varchar(30)

);

ALTER TABLE DOCTOR RENAME COLUMN dept TO department;

ALTER TABLE DOCTOR DROP COLUMN department;

ALTER TABLE DOCTOR ADD department varchar(30);

--------------------------Adding CONSTRAINT into TABLE DOCTOR--------------------------

ALTER TABLE DOCTOR ADD CONSTRAINT DOCTOR\_PK PRIMARY KEY(doctorId);

----------------------------- creation of the TABLE PATIENT-------------------------------

CREATE TABLE PATIENT(

patientId number(5) NOT NULL,

name varchar(30) NOT NULL,

address varchar(35),

age number(3),

phoneNo varchar(20) UNIQUE,

gender varchar(6),

CONSTRAINT PATIENT\_PK PRIMARY KEY(patientId)

);

--------------------------------Modification using alter---------------------------------------

ALTER TABLE PATIENT MODIFY patientId number(5) check(patientId>10000);

------------------------Dropping and Adding CONSTRAINT into TABLE PATIENT---------------

ALTER TABLE PATIENT DROP CONSTRAINT PATIENT\_PK;

ALTER TABLE PATIENT ADD CONSTRAINT PATIENT\_PK PRIMARY KEY(patientId);

------------------------------ creation of the TABLE APPOINTMENT-----------------------------

CREATE TABLE APPOINTMENT(

doctorId number(7) NOT NULL,

patientId number(5) NOT NULL,

appointmentDate date

);

-------------------------------Adding CONSTRAINT into TABLE APPOINTMENT--------------------

ALTER TABLE APPOINTMENT ADD CONSTRAINT APPOINTMENT\_PK PRIMARY KEY(doctorId,patientId);

ALTER TABLE APPOINTMENT ADD CONSTRAINT APPOINTMENT\_FK1 FOREIGN KEY(doctorId) references DOCTOR(doctorId) ON DELETE CASCADE;

ALTER TABLE APPOINTMENT ADD CONSTRAINT APPOINTMENT\_FK2 FOREIGN KEY(patientId) references PATIENT(patientId) ON DELETE CASCADE;

-- --------------------------creation of the TABLE ROOM----------------------------------------

CREATE TABLE ROOM(

roomId number(4) NOT NULL,

patientId number(5) NOT NULL,

type varchar(20),

varietyWard varchar(20)

);

----------------------------Adding CONSTRAINT into TABLE ROOM--------------------------------

ALTER TABLE ROOM ADD CONSTRAINT ROOM\_PK PRIMARY KEY(roomId);

ALTER TABLE ROOM ADD CONSTRAINT ROOM\_FK FOREIGN KEY(patientId) references PATIENT(patientId) ON DELETE CASCADE;

-- -------------------------creation of the TABLE BILL---------------------------------------------

CREATE TABLE BILL(

billNo varchar(5) NOT NULL,

patientId number(5) NOT NULL,

doctorCharge number(8,2),

roomCharge number(8,2)

);

------------------------Adding CONSTRAINT into TABLE BILL-------------------------------------

ALTER TABLE BILL ADD CONSTRAINT BILL\_PK PRIMARY KEY(billNo);

ALTER TABLE BILL ADD CONSTRAINT BILL\_FK FOREIGN KEY(patientId) references PATIENT(patientId) ON DELETE CASCADE;

-------------------------Use of Disable and enable of constraint-----------------------------------

ALTER TABLE BILL DISABLE CONSTRAINT BILL\_PK;

ALTER TABLE BILL ENABLE CONSTRAINT BILL\_PK;

------------------------------Use of Subquery------------------------------

--------------The patient whose total bill is between 5000 and 7000------------------------

SELECT patientId,name FROM PATIENT WHERE patientId IN(SELECT patientId FROM BILL WHERE (roomCharge+doctorCharge) BETWEEN 5000 AND 7000);

--------------The patient whose total bill is not between 5000 and 7000------------------------

SELECT patientId,name FROM PATIENT WHERE patientId IN(SELECT patientId FROM BILL WHERE (roomCharge+doctorCharge) NOT BETWEEN 5000 AND 7000);

--------------The patient whose total bill is not between 5000 and 7000 using aliasing-----------

SELECT p.patientId,p.name FROM PATIENT p WHERE p.patientId IN(SELECT b.patientId FROM BILL b WHERE (b.roomCharge+b.doctorCharge)<5000 OR (b.roomCharge+b.doctorCharge)>7000);

----------------------------------Use of Join-------------------------------------

-----------------All the doctor's and patient's info information using UNION ALL-----------------

SELECT name,address,phoneNo,gender FROM DOCTOR UNION ALL SELECT name,address,phoneNo,gender FROM PATIENT;

---------------All the doctor's and patient's info information using and UNION-----------------

SELECT name,address,phoneNo,gender FROM DOCTOR UNION SELECT name,address,phoneNo,gender FROM PATIENT;

---------All the doctor's info information whose is not in patient's using INTERSECT------------

SELECT name,address,phoneNo,gender FROM DOCTOR INTERSECT SELECT name,address,phoneNo,gender FROM PATIENT;

--All the doctor's info information whose is not in patient's using INTERSECT

SELECT name,address,phoneNo,gender FROM DOCTOR MINUS SELECT name,address,phoneNo,gender FROM PATIENT;

-----------------All Appointment date of each Patient under the doctor name---------

SELECT p.name,d.name,a.appointmentDate AS AppDate FROM DOCTOR d,PATIENT p,APPOINTMENT a WHERE d.doctorId=a.doctorId AND a.patientId=p.patientId;

------------------Bill description of each patient by JOIN-------------------------------

SELECT p.patientId,b.patientId,p.name,b.billNo,b.doctorCharge,b.roomCharge FROM PATIENT p JOIN BILL b ON p.patientId=b.patientId;

--------------------Bill description of each patient by JOIN and USING------------------

SELECT patientId,p.name,b.billNo,b.doctorCharge,b.roomCharge FROM PATIENT p JOIN BILL b USING (patientId);

---------------------Bill description of each patient by NATURAL JOIN-------------------

SELECT patientId,p.name,b.billNo,b.doctorCharge,b.roomCharge FROM PATIENT p NATURAL JOIN BILL b;

----------------------CROSS JOIN between patient and bill table-----------------------

SELECT p.patientId,p.name,b.billNo,b.doctorCharge,b.roomCharge FROM PATIENT p CROSS JOIN BILL b;

----------------Bill description of each patient by INNER JOIN same as JOIN-------------

SELECT p.patientId,b.patientId,p.name,b.billNo,b.doctorCharge,b.roomCharge FROM PATIENT p INNER JOIN BILL b ON p.patientId=b.patientId;

-----------------Bill description of each patient by LEFT OUTER JOIN---------------------

SELECT p.patientId,b.patientId,b.billNo,b.doctorCharge,b.roomCharge FROM PATIENT p LEFT OUTER JOIN BILL b ON p.patientId=b.patientId;

------------------Bill description of each patient by RIGHT OUTER JOIN------------------

SELECT p.patientId,b.patientId,b.billNo,b.doctorCharge,b.roomCharge FROM PATIENT p RIGHT OUTER JOIN BILL b ON p.patientId=b.patientId;

---------------------Bill description of each patient by FULL OUTER JOIN-----------------

SELECT p.patientId,b.patientId,b.billNo,b.doctorCharge,b.roomCharge FROM PATIENT p FULL OUTER JOIN BILL b ON p.patientId=b.patientId;

----------------------The Room charge in Bill using SELF JOIN---------------------

SELECT b1.roomCharge,b2.roomCharge FROM BILL b1 JOIN BILL b2 ON b1.roomCharge=b2.roomCharge;

------------------------------------Use of PL/SQL---------------------------------

--Showing maximum bill including their disCountCharge using PL/SQL

SET SERVEROUTPUT ON

DECLARE

--Type OwnType is number(9,2);

totalCharge BILL.roomCharge%type;

maxBillNo BILL.billNo%type;

maxBillPatientId BILL.patientId%type;

disCountCharge number(8,2);

BEGIN

SELECT MAX(doctorCharge+roomCharge) INTO totalCharge FROM BILL;

SELECT billNo,patientId INTO maxBillNo,maxBillPatientId FROM BILL WHERE (doctorCharge+roomCharge) IN(totalCharge);

DBMS\_OUTPUT.PUT\_LINE('The maximum charge of a patient is : '||totalCharge||' with Patient billNo : '||maxBillNo||' and ID No : '||maxBillPatientId);

IF totalCharge<=3000 THEN

disCountCharge := totalCharge - totalCharge\*0.10;

ELSIF totalCharge<=10000 THEN

disCountCharge := totalCharge - totalCharge\*0.15;

ELSIF totalCharge<=50000 THEN

disCountCharge := totalCharge - totalCharge\*0.20;

ELSIF totalCharge<=100000 THEN

disCountCharge := totalCharge - totalCharge\*0.25;

ELSE

disCountCharge := totalCharge - totalCharge\*0.30;

END IF;

DBMS\_OUTPUT.PUT\_LINE('The disCountCharge charge of a patient is : '||disCountCharge);

END;

/

--showing room description of the patient whose room type is Single Bed, AC using ------CURSOR

SET SERVEROUTPUT ON

DECLARE

CURSOR roomCursor IS SELECT roomId,patientId,type,varietyWard FROM ROOM WHERE type='Single Bed, AC';

accessVar roomCursor%ROWTYPE;

rowCounting int;

BEGIN

OPEN roomCursor;

SELECT COUNT(\*) INTO rowCounting FROM ROOM WHERE type='Single Bed, AC';

DBMS\_OUTPUT.PUT\_LINE('roomId patientId type varietyWard');

DBMS\_OUTPUT.PUT\_LINE('-----------------------------------------------------');

LOOP

FETCH roomCursor INTO accessVar;

DBMS\_OUTPUT.PUT\_LINE(accessVar.roomId || ' ' || accessVar.patientId || ' ' || accessVar.type || ' ' || accessVar.varietyWard);

DBMS\_OUTPUT.PUT\_LINE('-----------------------------------------------------');

EXIT WHEN roomCursor%ROWCOUNT>rowCounting-1;

END LOOP;

CLOSE roomCursor;

END;

/

------------------Insertion into APPOINTMENT table using PROCEDURE-----------------------

CREATE OR REPLACE PROCEDURE InsertIntoAppointment(docId DOCTOR.doctorId%type,patId PATIENT.patientId%type,appoinDate APPOINTMENT.appointmentDate%type) IS

BEGIN

INSERT INTO APPOINTMENT VALUES(docId,patId,appoinDate);

commit;

END InsertIntoAppointment;

/

-------------------Calling the PROCEDURE for Inserting into APPOINTMENT-------------------

SET SERVEROUTPUT ON

BEGIN

InsertIntoAppointment(1207001,10001,'25-APR-15');

END;

/

---------------------Finding totalCharge for a PATIENT using FUNCTION---------------------

CREATE OR REPLACE FUNCTION TreatMentCharge(bNo BILL.billNo%type) RETURN NUMBER IS

totCharge BILL.roomCharge%type;

BEGIN

SELECT (doctorCharge+roomCharge) INTO totCharge FROM BILL WHERE billNo=bNo;

RETURN totCharge;

END TreatMentCharge;

/

------------------Calling the FUNCTION for Calculating totalCharge for PATIENT----------------

SET SERVEROUTPUT ON

DECLARE

id BILL.billNo%type;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('The total for Patient ID ' ||'A-212' ||' is : '||TreatMentCharge('A 212'));

END;

/

------------------Triggering for Insertion and Update for BIll TABLE-----------------------------

CREATE OR REPLACE TRIGGER BillChecking BEFORE INSERT OR UPDATE ON BILL FOR EACH ROW

DECLARE

minRoomCharge BILL.roomCharge%type := 500;

--minDoctorCharge BILL.doctorCharge%type := 2000;

BEGIN

IF :new.roomCharge<minRoomCharge THEN

RAISE\_APPLICATION\_ERROR(-20000,'Room Charge is too small');

END IF;

END;

/