

NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA

SURATHKAL, MANGALORE - 575 025

Course Code – CS254

Course Name – Database Systems Lab

Lab - 02

Date – February 2, 2022

Submitted To

Dr. Annappa B

Department of Computer Science and Engineering

National Institute of Technology Karnataka, Surathkal

Submitted By

Md Rakib Hasan

Roll – 201CS132

Department of Computer Science and Engineering

**1. Build a database of hospital management systems. The database should have the information about –**

**PATIENT (p\_id, r\_id, d\_id, p\_name, city, contact)**

**DOCTORS (d\_id, p\_id, name, salary, specification)**

**ROOM (r\_id, p\_id, room\_type)**

**TEST & DIAGNOSIS (p\_id, diagno, diag\_details)**

**Assume the data types. Insert at least five values for each table.**

CREATE DATABASE hospital;

USE hospital;

CREATE TABLE patient (

    p\_id VARCHAR(5) NOT NULL,

    r\_id VARCHAR(5) NOT NULL,

    d\_id VARCHAR(5) NOT NULL,

    p\_name VARCHAR(50),

    city VARCHAR(20),

    contact INT,

    PRIMARY KEY (p\_id));

CREATE TABLE doctors (

    d\_id VARCHAR(5) NOT NULL,

    p\_id VARCHAR(5) NOT NULL,

    name VARCHAR(50),

    salary INT,

    specification VARCHAR(50),

    PRIMARY KEY (d\_id));

CREATE TABLE room (

    r\_id VARCHAR(5) NOT NULL,

    p\_id VARCHAR(5) NOT NULL,

    room\_type VARCHAR(20),

    PRIMARY KEY (r\_id));

CREATE TABLE test\_diagnosis (

    p\_id VARCHAR(5) NOT NULL,

    diagno VARCHAR(5),

    diag\_details VARCHAR(50));

INSERT INTO doctors

    VALUES ('d1', 'p1', 'Doctor A', 1000, 'Specification A'),

    ('d2', 'p2', 'Doctor B', 2000, 'Specification B'),

    ('d3', 'p3', 'Doctor C', 3000, 'Specification C'),

    ('d4', 'p4', 'Doctor D', 4000, 'Specification D'),

    ('d5', 'p5', 'Doctor E', 5000, 'Specification E');

INSERT INTO patient

    VALUES ('p1', 'r1', 'd1', 'Patient A', 'City A', 01),

    ('p2', 'r2', 'd2', 'Patient B', 'City B', 02),

    ('p3', 'r3', 'd3', 'Patient C', 'City C', 03),

    ('p4', 'r4', 'd4', 'Patient D', 'City D', 04),

    ('p5', 'r5', 'd5', 'Patient E', 'City E', 05);

INSERT INTO room

    VALUES ('r1', 'p1', 'ROOM A'),

    ('r2', 'p2', 'ROOM B'),

    ('r3', 'p3', 'ROOM C'),

    ('r4', 'p4', 'ROOM D'),

    ('r5', 'p5', 'ROOM E');

INSERT INTO test\_diagnosis

    VALUES ('p1', 'd1', 'Details A'),

    ('p2', 'd2', 'Details B'),

    ('p3', 'd3', 'Details C'),

    ('p4', 'd4', 'Details D'),

    ('p5', 'd5', 'Details E');

ALTER TABLE patient

ADD FOREIGN KEY (r\_id) REFERENCES room(r\_id);

ALTER TABLE patient

ADD FOREIGN KEY (d\_id) REFERENCES doctors(d\_id);

ALTER TABLE doctors

ADD FOREIGN KEY (p\_id) REFERENCES patient(p\_id);

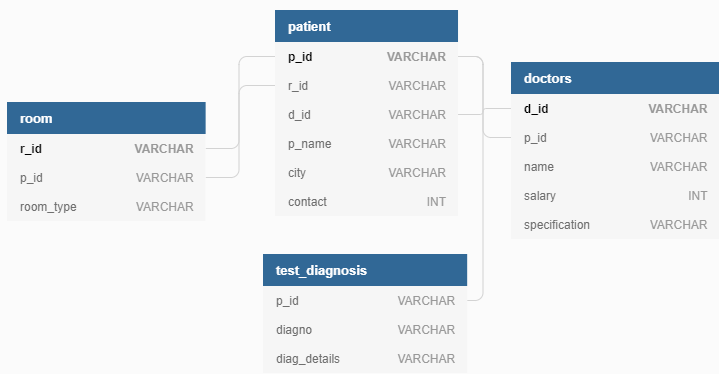
ALTER TABLE test\_diagnosis

ADD FOREIGN KEY (p\_id) REFERENCES patient(p\_id);

ALTER TABLE room

ADD FOREIGN KEY (p\_id) REFERENCES patient(p\_id);

**a. Draw a schema diagram for your table.**

****

**b. Use the above tables, attributes and demonstrate the working of primary key, foreign key, null value, table and column level constraints.**

**Primary key:** Primary key is the unique identifier of a table’s record. Primary key should be unique and not null.

**Foreign key:** Foreign key is used to create relation or connection between two tables. Also, the key which is foreign in one table, that key is used as primary key of another table.

**Null Value:** Null is used to represent the missing value.

**Table and Column level constraints:** Constraints are used to specify rules for the data. It can limit the type of data that can go into a table. If there is any violation between the constraints and the data action, this action is aborted. Table level constraints apply to the whole table whereas column level constraints only apply to a column. NOT NULL, DEFAULT, UNIQUE, PRIMARY and FOREIGN KEY are some of the example of mostly used constraints.

**c. Demonstrate the below violation of constraints in relational database.**

**Domain Constraint**

**Entity Integrity constraint**

**Key Constraints**

**Referential Integrity**

**Domain Constraint:** Domain constraints get violated because of inappropriate data type and also if data is not in the range of that domain. For example – If it is set that input value should be more then 10, and if someone enter 9 then domain constraint will be violated.

**Entity Integrity Constraint:** If we insert NULL in any primary key then entity integrity constraint gets violated.

**Key Constraints:** If we try to insert an already existing key in a table then key constraint gets violated.

**Referential Integrity:** If we insert a value in foreign key, which is not present in primary key then referential integrity gets violated.

**2. Build a Department database which consist of following information:**

**Student (Rollno, Name, Dob, Gender, Doa, Bcode);**

**Branch (Bcode, Bname, Dno);**

**Department (Dno, Dname);**

**Course (Ccode, Cname, Credits, Dno);**

**Branch\_Course (Bcode, Ccode, Semester);**

**Enrolls (Rollno, Ccode, Grade);**

**Write Queries for the following:**

CREATE DATABASE IF NOT EXISTS department\_info;

USE department\_info;

CREATE TABLE department (

    d\_no VARCHAR(5) NOT NULL,

    d\_name VARCHAR(50),

    PRIMARY KEY (d\_no));

CREATE TABLE branch (

    b\_code VARCHAR(5) NOT NULL,

    b\_name VARCHAR(50),

    d\_no VARCHAR(5) NOT NULL,

    PRIMARY KEY (b\_code),

    FOREIGN KEY (d\_no) REFERENCES department(d\_no));

CREATE TABLE course (

    c\_code VARCHAR(5) NOT NULL,

    c\_name VARCHAR(50),

    credits INT,

    d\_no VARCHAR(5) NOT NULL,

    PRIMARY KEY (c\_code),

    FOREIGN KEY (d\_no) REFERENCES department(d\_no));

CREATE TABLE student (

    rollno INT NOT NULL,

    name VARCHAR(50),

    dob DATE,

    gender CHAR(1),

    doa DATE,

    b\_code VARCHAR(5) NOT NULL,

    PRIMARY KEY (rollno),

    FOREIGN KEY (b\_code) REFERENCES branch(b\_code)

    );

CREATE TABLE branch\_course (

    b\_code VARCHAR(5) NOT NULL,

    c\_code VARCHAR(5) NOT NULL,

    semester INT,

    FOREIGN KEY (b\_code) REFERENCES branch(b\_code),

    FOREIGN KEY (c\_code) REFERENCES course(c\_code));

CREATE TABLE enrolls (

    rollno INT NOT NULL,

    c\_code VARCHAR(5) NOT NULL,

    grade INT,

    FOREIGN KEY (rollno) REFERENCES student(rollno),

    FOREIGN KEY (c\_code) REFERENCES course(c\_code));

INSERT INTO department

    VALUES ("D1", "CSE"),

    ("D2", "EEE"),

    ("D3", "IT"),

    ("D4", "ME"),

    ("D5", "IPE");

INSERT INTO branch

    VALUES ("B1", "B1\_CSE", "D1"),

    ("B2", "B2\_CSE", "D1"),

    ("B3", "B1\_EEE", "D2"),

    ("B4", "B2\_EEE", "D2"),

    ("B5", "B1\_IT", "D3"),

    ("B6", "B2\_IT", "D3"),

    ("B7", "B1\_ME", "D4"),

    ("B8", "B1\_IPE", "D5");

INSERT INTO course

    VALUES ("C1", "C1\_CSE", 4, "D1"),

    ("C2", "C2\_CSE", 4, "D1"),

    ("C3", "C1\_EEE", 2, "D2"),

    ("C4", "C1\_EEE", 3, "D2"),

    ("C5", "C1\_IT", 4, "D3"),

    ("C6", "C1\_IT", 4, "D3"),

    ("C7", "C1\_ME", 1, "D4"),

    ("C8", "C1\_IPE", 4, "D5");

INSERT INTO student

    VALUES (1, "S1", "2001-01-01", "M", "2005-01-01", "B1"),

    (2, "S2", "2000-01-01", "M", "2005-01-02", "B2"),

    (3, "S3", "2001-02-01", "F", "2005-01-01", "B3"),

    (4, "S4", "2001-03-01", "M", "2005-01-05", "B4"),

    (5, "S5", "2001-07-01", "M", "2005-01-01", "B5"),

    (6, "S6", "2001-02-05", "F", "2005-01-04", "B6"),

    (7, "S7", "2001-01-20", "M", "2005-01-06", "B7"),

    (8, "S8", "2001-05-01", "F", "2005-01-07", "B8"),

    (9, "S9", "2001-01-01", "M", "2005-01-01", "B1"),

    (10, "S10", "2001-02-01", "M", "2005-02-01", "B2"),

    (11, "S11", "2001-03-01", "F", "2005-01-11", "B3"),

    (12, "S12", "2001-04-01", "M", "2005-01-14", "B4"),

    (13, "S13", "2001-05-01", "M", "2005-01-13", "B5");

INSERT INTO branch\_course

    VALUES ("B1", "C2", 1),

    ("B2", "C2", 2),

    ("B3", "C3", 3),

    ("B4", "C4", 4),

    ("B5", "C5", 5),

    ("B6", "C6", 6),

    ("B7", "C7", 7),

    ("B8", "C8", 1),

    ("B2", "C1", 1),

    ("B2", "C8", 2),

    ("B3", "C4", 3),

    ("B1", "C4", 4),

    ("B5", "C6", 5),

    ("B6", "C5", 6),

    ("B8", "C7", 1),

    ("B7", "C8", 8);

INSERT INTO enrolls

    VALUES (1, "C1", 4),

    (2, "C2", 4),

    (3, "C3", 4),

    (4, "C4", 5),

    (5, "C5", 4),

    (6, "C6", 6),

    (7, "C7", 4),

    (8, "C8", 5),

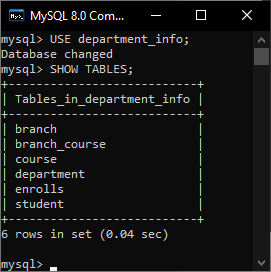
    (9, "C1", 8),

    (10, "C2", 9),

    (11, "C3", 4),

    (12, "C4", 7),

    (13, "C5", 6);

****

**a. Print the details of students who are from the same department.**

SELECT s.rollno,

        s.name,

        s.dob,

        s.gender,

        s.doa,

        d.d\_name

FROM student s

JOIN branch b

    USING (b\_code)

JOIN department d

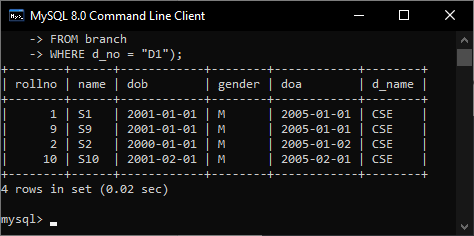
    USING (d\_no)

WHERE b\_code IN (

    SELECT b\_code

    FROM branch

    WHERE d\_no = "D1")

****

**b. Get the details of branches under a particular Department;**

SELECT d.d\_no as DEPT\_ID,

        d.d\_name as DEPT,

        b.b\_code as BRANCH\_CODE,

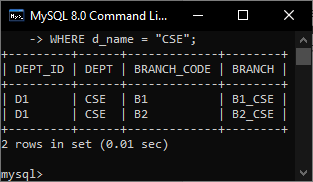
        b.b\_name as BRANCH

FROM department d

JOIN branch b

    USING (d\_no)

WHERE d\_name = "CSE"

****

**c. Print the names of courses offered in a particular department in a particular semester;**

SELECT c.c\_name, d.d\_name, bc.semester

FROM branch\_course bc

JOIN branch

    USING (b\_code)

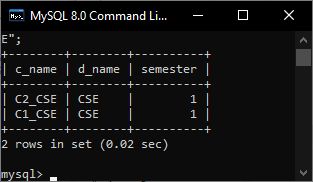
JOIN department d

    using (d\_no)

JOIN course c

    using (c\_code)

WHERE semester = 1 AND d\_name = "CSE"

****

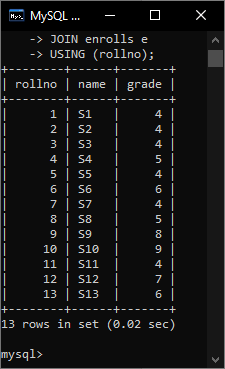
**d. Print the students roll no, Name and grades;**

SELECT s.rollno, s.name, e.grade

FROM student s

JOIN enrolls e

    USING (rollno)

****

**e. Print the details of students who are enrolled for different courses.**

SELECT s.rollno, s.name, s.gender, s.doa, c.c\_name, e.grade

FROM enrolls e

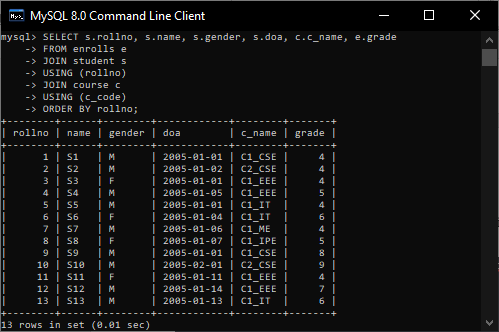
JOIN student s

    USING (rollno)

JOIN course c

    USING (c\_code)

ORDER BY rollno

****