Tab 1

**SQL TEST 1(dataset 1)**

**1.Exercise 1: List All Books and Their Authors**

**Exercise:** Show the name of each author together with the title of the book they wrote and the year in which that book was published.

**Solution:**

**select author.name,book.title,book.publish\_year from author inner join book on author.ID=book.ID;**

**Solution output:**

| **name** | **title** | **publish\_year** |
| --- | --- | --- |
| **Dennis Stokes** | **Faith Of Light** | **1995** |
| **Randolph Vasquez** | **Memory Of Hope** | **2000** |
| **Marcella Cole** | **Soulless girl** | **2008** |
| **Daniel Branson** | **Warrior Of Wind** | **2005** |
| **Lisa Mullins** | **Weak Heart** | **1980** |

**2.Exercise 2: List Authors and Books Published After 2005**

**Exercise:** Show the name of each author together with the title of the book they wrote and the year in which that book was published. Show only books published after 2005.

**Solution:**

select author.name,book.title,book.publish\_year from author inner join book on author.ID = book.ID where publish\_year > 2005;

**Solution output:**

| **name** | **title** | **publish\_year** |
| --- | --- | --- |
| **Marcella Cole** | **Soulless girl** | **2008** |

**3.Exercise 3: Show Books Adapted Within 4 Years and Rated Lower Than the Adaptation**

**Exercise:** For each book, show its title, adaptation title, adaptation year, and publication year.

Include only books with a rating lower than the rating of their corresponding adaptation. Additionally, show only those books for which an adaptation was released within four years of the book’s publication.

Rename the title column from the book table to book\_title and the title column from the adaptation table to adaptation\_title.

**Solution:**

**select book.title as book\_title, adaptation.title as adaptation\_title, book.publish\_year, adaptation.release\_year from book join adaptation on book.id = adaptation.book\_id where adaptation.release\_year - book.publish\_year <= 4 and book.rating < adaptation.rating;**

**Solution output:**

| **book\_title** | **adaptation\_title** | **publish\_year** | **release\_year** |
| --- | --- | --- | --- |
| **Memory Of Hope** | **Patrons And Bearers** | **2000** | **2004** |

**4.Exercise 4: Show All Books and Their Adaptations (If Any)**

**Exercise:** Show the title of each book together with the title of its adaptation and the date of the release. Show all books, regardless of whether they had adaptations.

**Solution:**

**select book.title,adaptation.title as title2,adaptation.release\_year from book left join adaptation on book.id = adaptation.book\_id;**

**Solution output:**

| **title** | **title 2** | **release\_year** |
| --- | --- | --- |
| **Soulless girl** | **Gone With The Wolves: The Beginning** | **2008** |
| **Weak Heart** | **Blacksmith With Silver** | **2014** |
| **Faith Of Light** | **Companions Of Tomorrow** | **2001** |
| **Memory Of Hope** | **Patrons And Bearers** | **2004** |
| **Warrior Of Wind** | **Homeless Warrior** | **2008** |

**5.Exercise 5: Show All Books and Their Movie Adaptations**

**Exercise:** Show all books with their movie adaptations. Select each book's title, the name of its publishing house, the title of its adaptation, and the type of the adaptation. Keep the books with no adaptations in the result.

**Solution:**

**select book.title,book.publishing\_house,adaptation.title as title2,adaptation.type from book left join adaptation on book.id = adaptation.book\_id where type = 'movie' or type is null;**

**Solution output:**

| **title** | **publishing\_house** | **title 2** | **type** |
| --- | --- | --- | --- |
| **Soulless girl** | **Golden Albatros** | **Gone With The Wolves: The Beginning** | **movie** |
| **Weak Heart** | **Diarmud Inc.** | **Blacksmith With Silver** | **movie** |
| **Faith Of Light** | **White Cloud Press** | **Companions Of Tomorrow** | **movie** |
| **Memory Of Hope** | **Rutis Enterprises** | **Patrons And Bearers** | **movie** |
| **Warrior Of Wind** | **Maverick** | **Homeless Warrior** | **movie** |

**6.Exercise 6: List All the Books and All the Authors**

**Exercise:** Display the title of each book along with the name of its author. Show all books, even those without an author. Show all authors, even those who haven't published a book yet. Use a FULL JOIN.

**Solution:**

**select book.title,author.name from book cross join author on book.author\_id = author.ID;**

**Solution output:**

| **title** | **name** |
| --- | --- |
| **Faith Of Light** | **Randolph Vasquez** |

**7. Case Study 1:** List All Books and Their Authors

\*\*Scenario\*\*: You have two tables: `books` and `authors`. You need to list all books along with their authors.

**Solution:**

**select book.title,author.name as author from book inner join author on book.ID=author.ID;**

**Solution output:**

| **title** | **author** |
| --- | --- |
| **Soulless girl** | **Marcella Cole** |
| **Weak Heart** | **Lisa Mullins** |
| **Faith Of Light** | **Dennis Stokes** |
| **Memory Of Hope** | **Randolph Vasquez** |
| **Warrior Of Wind** | **Daniel Branson** |

**8.Case Study 2:** List Authors and Books Published After 2005

\*\*Scenario\*\*: You need to list authors and the titles of books published after 2005.

**Solution:**

**select author.name,book.title from author inner join book on author.ID = book.ID where publish\_year > 2005;**

**Solution output:**

| **title** | **name** |
| --- | --- |
| **Marcella Cole** | **Soulless girl** |

**9.Case Study 3:** Show Books Adapted Within 4 Years and Rated Lower Than the Adaptation

\*\*Scenario\*\*: You have tables `books` and `adaptations`. You need to show books that were adapted within 4 years and rated lower than the adaptation.

**Solution:**

select book.title as book\_title , adaptation.title as adaptation\_title, book.rating as book\_rating, adaptation.rating as adaptation\_rating from book join adaptation on book.id = adaptation.book\_id where adaptation.release\_year - book.publish\_year <= 4 and book.rating < adaptation.rating;

**Solution output:**

| book\_title | adptation\_title | book\_rating | adaptation\_rating |
| --- | --- | --- | --- |
| Memory Of Hope | Patrons And Bearers | 2.7 | 3.2 |

10.**Case Study 4:** Show All Books and Their Movie Adaptations (If Any)

\*\*Scenario\*\*: You have tables `books` and `movies`. You need to show all books and their movie adaptations, if any.

**Solution:**

select book.title as book\_title,adaptation.title as adaptation\_movie from book left join adaptation on book.id = adaptation.book\_id where type = 'movie' or type is null;

**Solution output:**

| book\_title | adptation\_movie |
| --- | --- |
| Soulless girl | Gone With The Wolves: The Beginning |
| Weak Heart | Blacksmith With Silver |
| Faith Of Light | Companions Of Tomorrow |
| Memory Of Hope | Patrons And Bearers |
| Warrior Of Wind | Homeless Warrior |

Main sql query:

create database dataset;

use dataset;

create table if not exists author(

ID int not null,name char(20),birth\_year int,death\_year int ,primary key(ID));

insert into author values(1,"Marcella Cole",1983,null ),

(2,"Lisa Mullins",1891,1950),

(3,"Dennis Stokes",1935,1994),

(4,"Randolph Vasquez",1957,2004),

(5,"Daniel Branson",1965,1990);

create table book(

ID int,author\_id int ,title char(50),publish\_year int,publishing\_house char(20),rating float);

insert into book values(1,null,"Soulless girl",2008,"Golden Albatros",4.3),

(2,null,"Weak Heart",1980,"Diarmud Inc.",3.8),

(3,4,"Faith Of Light",1995,"White Cloud Press",4.3),

(4,null,"Memory Of Hope",2000,"Rutis Enterprises",2.7),

(5,6,"Warrior Of Wind",2005,"Maverick",4.8);

create table adaptation(

book\_id int,type char(20),title char(50),release\_year int,rating float);

insert into adaptation values(1,"movie","Gone With The Wolves: The Beginning",2008,3),

(3,"movie","Companions Of Tomorrow",2001,4.2),

(5,"movie","Homeless Warrior",2008,4),

(2,"movie","Blacksmith With Silver",2014,4.3),

(4,"movie","Patrons And Bearers",2004,3.2);

create table book\_review(

book\_id int,review char(150),author char(50));

insert into book\_review values (1,"An incredible book","Sylvia Jones"),

(1,"Great, although it has some flaws","Jessica Parker"),

(2,"Dennis Stokes takes the reader ride full of emotions","Thomas Green"),

(3,"Incredible craftsmanship of the author","Martin Freeman"),

(4,"Not the best book by this author","Jude Falth"),

(5,"Claudia Johnson at her best!","Joe Marqiz"),

(6,"I cannot recall more captivating plot","Alexander Durham");

select \* from book\_review;

select \* from adaptation;

select \* from author;

select \* from book;

/\*ex1 query\*/

select author.name,book.title,book.publish\_year from author inner join book on author.ID=book.ID;

/\*ex2 query\*/

select author.name,book.title,book.publish\_year from author inner join book on author.ID = book.ID where publish\_year > 2005;

/\*ex3 query\*/

select book.title as book\_title, adaptation.title as adaptation\_title, book.publish\_year, adaptation.release\_year

from book join adaptation on book.id = adaptation.book\_id where adaptation.release\_year - book.publish\_year <= 4

and book.rating < adaptation.rating;

/\*ex4 query\*/

select book.title,adaptation.title as title2,adaptation.release\_year from book left join adaptation

on book.id = adaptation.book\_id;

/\*ex5 query\*/

select book.title,book.publishing\_house,adaptation.title as title2,adaptation.type from book

left join adaptation on book.id = adaptation.book\_id where type = 'movie' or type is null;

/\* ex6 query\*/

select book.title,author.name from book cross join author on book.author\_id = author.ID;

/\* case 1 \*/

select book.title,author.name as author from book inner join author on book.ID=author.ID;

/\* case 2 \*/

select author.name,book.title from author inner join book on author.ID = book.ID where publish\_year > 2005;

/\* case 3 \*/

select book.title as book\_title , adaptation.title as adaptation\_title, book.rating as book\_rating, adaptation.rating as adaptation\_rating

from book join adaptation on book.id = adaptation.book\_id

where adaptation.release\_year - book.publish\_year <= 4 and book.rating < adaptation.rating;

/\* case 4 \*/

select book.title as book\_title,adaptation.title as adaptation\_movie from book left join adaptation on book.id = adaptation.book\_id where type = 'movie' or type is null;

11. **Case Study 5:** List All Products, Prices, Producers, and Departments

\*\*Scenario\*\*: You have tables `products`, `producers`, and `departments`. You need to list all products along with their prices, producers, and departments.

**Solution:**

select \* from case5;

**Solution output:**

| name | price | producer\_name | department\_name |
| --- | --- | --- | --- |
| Product 1 | 10.99 | Producer A | Department 1 |
| Product 2 | 15.99 | Producer B | Department 2 |
| Product 3 | 12.99 | Producer C | Department 3 |

**Main query:**

Use dataset;

create table if not exists case5(

name char(50),price float,producer\_name varchar(50),department\_name varchar(50));

insert into case5 values("Product 1 ",10.99,"Producer A","Department 1"),

("Product 2 ",15.99,"Producer B","Department 2"),

("Product 3 ",12.99,"Producer C","Department 3");

/\* case 5 \*/

select \* from case5;

**Sql test 2 (data set 3)**

**1.Exercise 1: List All Workers and Their Direct Supervisors.**

**Exercise:** Show all workers' names together with the names of their direct supervisors. Rename the columns apprentice\_name and master\_name, respectively. Consider only workers who have a supervisor (i.e. a master).

**Solution:**

**select apprentice.name as apprentice\_name, master.name as master\_name from workshop\_workers apprentice join workshop\_workers master on apprentice.master\_id = master.id;**

**Solution output:**

| **apprentice\_name** | **master\_name** |
| --- | --- |
| **Kate Brown** | **Mathew Conn** |
| **John Doe** | **Suzan Gregowitch** |

**Main query:**

use dataset;

create table if not exists workshop\_workers(

ID int not null,name char(50),specialization char(50),master\_id int,experience int,project\_id int);

insert into workshop\_workers values(1,"Mathew Conn","woodworking",NULL,20,1),

(2,"Kate Brown","woodworking",1,4,1),

(3,"John Doe","incrusting",5,3,1),

(4,"John Kowalsky","woodworking",7,2,3),

(5,"Suzan Gregowitch","incrusting",NULL,15,4);

select \* from workshop\_workers;

/\* ex 1`\*/

select apprentice.name as apprentice\_name, master.name as master\_name from workshop\_workers apprentice

join workshop\_workers master on apprentice.master\_id = master.id;

**Sql test 3**

**1.TASK 1: Age Constraint:**

Scenario : Ensure that the `age` column only accepts values between 18 and 65.

Expected Output :

create table mydata(Fullname char(30),age int,gender char(40),salary int,city char(30),pan char(30),check(age=18 and age>=65));

INSERT INTO mydata (Fullname, age, gender, salary, city, Pan)

VALUES ('John Doe', 70, 'Male', 20000, 'Mumbai', 'ABCDE1234F');

Error:

13:16:44 INSERT INTO mydata (Fullname, age, gender, salary, city, Pan) VALUES ('John Doe', 70, 'Male', 20000, 'Mumbai', 'ABCDE1234F') Error Code: 3819. Check constraint 'mydata\_chk\_1' is violated. 0.000 sec.

Error solved:

create table mydata(Fullname char(30),age int,gender char(40),salary int,city char(30),pan char(30),

check(age>=18 || age>=65));

INSERT INTO mydata (Fullname, age, gender, salary, city, Pan)

VALUES ("John Doe", 50, "Male", 20000, "Mumbai", "ABCDE1234F");

select \* from mydata;

| Full name | age | gender | salary | city | pan |
| --- | --- | --- | --- | --- | --- |
| John Doe | 50 | Male | 20000 | Mumbai | ABCDE1234F |

2.**TASK 2: Gender Constraint:**

Scenario : Ensure that the `gender` column only accepts values 'Male', 'Female', or 'Other'.

Expected Output :

create table mydata(Fullname char(30),age int,gender char(40),salary int,city char(30),pan char(30),check(gender=”male”);

INSERT INTO mydata (Fullname, age, gender, salary, city, Pan)

VALUES ("John Doe", 50, "Unknown", 20000, "Mumbai", "ABCDE1234F");

select \* from mydata;

Error:

13:54:19 INSERT INTO mydata (Fullname, age, gender, salary, city, Pan) VALUES ("John Doe", 50, "unknown", 20000, "Mumbai", "ABCDE1234F") Error Code: 3819. Check constraint 'mydata\_chk\_1' is violated. 0.000 sec

Error solved:

create table mydata(Fullname char(30),age int,gender char(40),salary int,city char(30),pan char(30),check(gender="male"));

INSERT INTO mydata (Fullname, age, gender, salary, city, Pan)

VALUES ("John Doe", 50, "male", 20000, "Mumbai", "ABCDE1234F");

select \* from mydata;

| Fullname | age | gender | salary | city | pan |
| --- | --- | --- | --- | --- | --- |
| John Doe | 50 | male | 20000 | Mumbai | ABCDE1234F |

3. **TASK 3: Salary Constraint**

Scenario : Ensure that the `salary` column only accepts values greater than 10000 (already present).

Expected Output :

create table mydata(Fullname char(30),age int,gender char(40),salary int,city char(30),pan char(30),check(salary<=10000));

INSERT INTO mydata (Fullname, age, gender, salary, city, Pan)

VALUES ("John Doe", 50, "male", 90000, "Mumbai", "ABCDE1234F");

select \* from mydata;

Error:

14:02:13 INSERT INTO mydata (Fullname, age, gender, salary, city, Pan) VALUES ("John Doe", 50, "male", 90000, "Mumbai", "ABCDE1234F") Error Code: 3819. Check constraint 'mydata\_chk\_1' is violated. 0.000 sec

Error solved:

create table mydata(Fullname char(30),age int,gender char(40),salary int,city char(30),pan char(30),check(salary<=10000));

INSERT INTO mydata (Fullname, age, gender, salary, city, Pan)

VALUES ("John Doe", 50, "male", 9000, "Mumbai", "ABCDE1234F");

select \* from mydata;

| Fullname | age | gender | salary | city | pan |
| --- | --- | --- | --- | --- | --- |
| John Doe | 50 | male | 9000 | Mumbai | ABCDE1234F |

4. **TASK 4: PAN Unique Constraint**

Scenario : Ensure that the `Pan` column only accepts unique values.

Expected Output :

create table mydata(Fullname char(30),age int,gender char(40),salary int,city char(30),pan char(30) unique);

INSERT INTO mydata (Fullname, age, gender, salary, city, Pan) VALUES ("John Doe", 50, "male", 9000, "Mumbai", "ABCDE1234F"), ('Robert Smith', 40, 'Male', 30000, 'Hyderabad', 'ABCDE1234F');

select \* from mydata;

Error:

INSERT INTO mydata (Fullname, age, gender, salary, city, Pan) VALUES ("John Doe", 50, "male", 9000, "Mumbai", "ABCDE1234F"), ('Robert Smith', 40, 'Male', 30000, 'Hyderabad', 'ABCDE1234F') Error Code: 1062. Duplicate entry 'ABCDE1234F' for key 'mydata.pan'.

Error solved:

create table mydata(Fullname char(30),age int,gender char(40),salary int,city char(30),pan char(30) unique);

INSERT INTO mydata (Fullname, age, gender, salary, city, Pan)VALUES ("John Doe", 50, "male", 9000, "Mumbai", "ABCDE1234F"), ('Robert Smith', 40, 'Male', 30000, 'Hyderabad', 'ABCRE1234F');

select \* from mydata;

| Fullname | age | gender | salary | city | pan |
| --- | --- | --- | --- | --- | --- |
| John Doe | 50 | male | 9000 | Mumbai | ABCDE1234F |
| Robert Smith | 40 | Male | 30000 | Hyderabad | ABCRE1234F |

5. **TASK 5: City Default Value**

Scenario : Ensure that the `city` column defaults to 'Chennai' if not provided.

Expected Output :

create table mydata(Fullname char(30),age int,gender char(40),salary int,city char(30) default'chennai',pan char(30));

INSERT INTO mydata (Fullname, age, gender, salary, city, Pan)

VALUES ("John Doe", 50, "male", 9000, "Chennai", "ABCDE1234F");

select \* from mydata;

| Fullname | age | gender | salary | city | pan |
| --- | --- | --- | --- | --- | --- |
| John Doe | 50 | male | 9000 | Chennai | ABCDE1234F |

6. **TASK 6: Email Unique Constraint**

Scenario: Ensure that the `email` column only accepts unique values.

Expected Output :

create table mydata(Fullname char(30),age int,gender char(40),salary int,city char(30) ,pan char(30),email char(50) unique);

INSERT INTO mydata (Fullname, age, gender, salary, city, Pan,email)

VALUES("JohnDoe",50,"male"9000,"Chennai","ABCDE1234F","john123@gmail.com"), ("Jane Smith", 32, "Female", 27000, "Mumbai", "FGHIJ5678K", "john123@gmail.com");select \* from mydata;

Error:

INSERT INTO mydata (Fullname, age, gender, salary, city, Pan,email) VALUES ("John Doe", 50,"male",9000,"Chennai","ABCDE1234F","john123@gmail.com"), ("JaneSmith",32,"Female",27000,"Mumbai","FGHIJ5678K","john123@gmail.com")Error Code: 1062. Duplicate entry 'john123@gmail.com' for key 'mydata.email’.

Error solved:

create table mydata(Fullname char(30),age int,gender char(40),salary int,city char(30) ,pan char(30),email char(50) unique);

INSERT INTO mydata (Fullname, age, gender, salary, city, Pan,email)

VALUES ("John Doe", 50, "male", 9000, "Chennai", "ABCDE1234F","john123@gmail.com"),

("Jane Smith", 32, "Female", 27000, "Mumbai", "FGHIJ5678K", "john12@gmail.com");

select \* from mydata;

| Fullname | age | gender | salary | city | pan | Email |
| --- | --- | --- | --- | --- | --- | --- |
| John Doe | 50 | male | 9000 | Chennai | ABCDE1234F | john123@gmail.com |
| Jane Smith | 32 | Female | 27000 | Mumbai | FGHIJ5678K | john12@gmail.com |

7. **TASK 7: Not Null Constraint on Fullname**

Scenario: Ensure that the `Fullname` column does not accept null values.

Expected Output :

create table mydata(Fullname char(30) not null,age int,gender char(40),salary int,city char(30) ,pan char(30));

INSERT INTO mydata ( age, gender, salary, city, Pan)

VALUES ( 50, "male", 9000, "Chennai", "ABCDE1234F");

select \* from mydata;

Error:

INSERT INTO mydata ( age, gender, salary, city, Pan) VALUES ( 50, "male", 9000, "Chennai", "ABCDE1234F") Error Code: 1364. Field 'Fullname' doesn't have a default value.

Error solved:

create table mydata(Fullname char(30) not null,age int,gender char(40),salary int,city char(30) ,pan char(30));

INSERT INTO mydata ( Fullname,age, gender, salary, city, Pan)

VALUES ( "john",50, "male", 9000, "Chennai", "ABCDE1234F");

select \* from mydata;

| Fullname | age | gender | salary | city | pan |
| --- | --- | --- | --- | --- | --- |
| john | 50 | male | 9000 | Chennai | ABCDE1234F |

8.**TASK 8: Default Value for Gender**

Scenario: Ensure that the `gender` column defaults to 'Not Specified' if not provided.

Expected Output :

create table mydata(Fullname char(30) ,age int,gender char(40)default'Female',salary int,city char(30) ,pan char(30));

INSERT INTO mydata ( Fullname,age,salary, city, Pan)

VALUES ( "sharme",50,9000, "Chennai", "ABCDE1234F");

select \* from mydata;

| Fullname | age | gender | salary | city | pan |
| --- | --- | --- | --- | --- | --- |
| sharme | 50 | Female | 9000 | Chennai | ABCDE1234F |

9.  **TASK 9: Mobile Number Constraint**

Scenario: Ensure that the `mobile` column only accepts 10-digit numbers.

Expected Output :

create table mydata(Fullname char(10),age int,gender char(40),salary int,city char(30) ,pan char(30),

mobile\_num varchar(10) check(mobile\_num regexp '[0-9]'));

INSERT INTO mydata ( Fullname,age,gender,salary, city, Pan,mobile\_num)

VALUES ( "sharme",50,"female",9000, "Chennai", "ABCDE1234F","2345678");

select \* from mydata;

Error:

INSERT INTO mydata ( Fullname,age,gender,salary, city, Pan,mobile\_num) VALUES ( "sharme",50,"female",9000, "Chennai", "ABCDE1234F","2345678") Error Code: 3819. Check constraint 'mydata\_chk\_1' is violated.

Error solved:

create table mydata(Fullname char(10),age int,gender char(40),salary int,city char(30) ,pan char(30),mobile\_num varchar(10) check(mobile\_num regexp '[0-9]'));

INSERT INTO mydata(Fullname,age,gender,salary,city,Pan,mobile\_num)VALUES ( "sharme",50,"female",9000, "Chennai", "ABCDE1234F","1234567890");select \* from mydata;

| Fullname | age | gender | salary | city | pan | mobile\_num |
| --- | --- | --- | --- | --- | --- | --- |
| sharme | 50 | female | 9000 | Chennai | ABCDE1234F | 1234567890 |

10.**TASK 10: Department Foreign Key Constraint**

Scenario: Ensure that the `department\_id` in `mydata` references `id` in `departments` table.

Expected Output :

create table dep(id int primary key, name char(50));

create table mydata(Fullname char(10),age int,gender char(40),salary int,city char(30) ,pan char(30),

dep\_id int,foreign key(dep\_id) references dep(id));

INSERT INTO mydata ( Fullname,age,gender,salary, city, Pan,dep\_id)

VALUES ( "sharme",50,"female",9000, "Chennai", "ABCDE1234F",99);

select \* from mydata;

Error:

INSERT INTO mydata ( Fullname,age,gender,salary, city, Pan,dep\_id) VALUES ( "sharme",50,"female",9000, "Chennai", "ABCDE1234F",99) Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails (`new`.`mydata`, CONSTRAINT `mydata\_ibfk\_1` FOREIGN KEY (`dep\_id`) REFERENCES `dep` (`id`)) .

11. **TASK 11: Positive Age Constraint**

*Scenario: Ensure that the `age` column only accepts positive values.*

Expected Output :

create table mydata(Fullname char(30) ,age int check (age>0),gender char(40),salary int,city char(30) ,pan char(30));

INSERT INTO mydata ( Fullname,age,salary, city, Pan)

VALUES ( "sharme",-50,9000, "Chennai", "ABCDE1234F");

select \* from mydata;

Error:

INSERT INTO mydata ( Fullname,age,salary, city, Pan) VALUES ( "sharme",-50,9000, "Chennai", "ABCDE1234F") Error Code: 3819. Check constraint 'mydata\_chk\_1' is violated.

Error solved:

create table mydata(Fullname char(30) ,age int check (age>0),gender char(40),salary int,city char(30) ,pan char(30));

INSERT INTO mydata ( Fullname,age,gender,salary, city, Pan)

VALUES ( "sharme",50,"female",9000, "Chennai", "ABCDE1234F");

select \* from mydata;

| Fullname | age | gender | salary | city | pan |
| --- | --- | --- | --- | --- | --- |
| sharme | 50 | female | 9000 | Chennai | ABCDE1234F |

12.  **TASK 12: Fullname Length Constraint**

Scenario: Ensure that the `Fullname` column only accepts names between 3 and 20 characters.

Expected Output :

create table mydata(Fullname char(20) check (char\_length(Fullname)>=3 and char\_length(Fullname)<=20) ,age int,gender char(40),salary int,city char(30) ,pan char(30));

INSERT INTO mydata ( Fullname,age,gender,salary, city, Pan)

VALUES ( "jo",50,"female",9000, "Chennai", "ABCDE1234F");

select \* from mydata;

Error:

INSERT INTO mydata ( Fullname,age,gender,salary, city, Pan) VALUES ( "jo",50,"female",9000, "Chennai", "ABCDE1234F") Error Code: 3819. Check constraint 'mydata\_chk\_1' is violated.

Error solved:

create table mydata(Fullname char(20) check (char\_length(Fullname)>=3 and char\_length(Fullname)<=20) ,age int,gender char(40),salary int,city char(30) ,pan char(30));

INSERT INTO mydata ( Fullname,age,gender,salary, city, Pan)

VALUES ( "joo",50,"female",9000, "Chennai", "ABCDE1234F");select \* from mydata;

| Fullname | age | gender | salary | city | pan |
| --- | --- | --- | --- | --- | --- |
| joo | 50 | female | 9000 | Chennai | ABCDE1234F |

13.  **TASK 13: City Not Null Constraint**

Scenario: Ensure that the `city` column does not accept null values.

Expected Output :

create table mydata(Fullname char(20) ,age int,gender char(40),salary int,city char(30) not null ,pan char(30));

INSERT INTO mydata ( Fullname,age,gender,salary, Pan)

VALUES ( "joo",50,"female",9000, "ABCDE1234F");

select \* from mydata;

Error:

INSERT INTO mydata ( Fullname,age,gender,salary, Pan) VALUES ( "joo",50,"female",9000, "ABCDE1234F") Error Code: 1364. Field 'city' doesn't have a default value.

Error solved:

create table mydata(Fullname char(20) ,age int,gender char(40),salary int,city char(30) not null ,pan char(30));

INSERT INTO mydata ( Fullname,age,gender,salary, city, Pan)

VALUES ( "joo",50,"female",9000, "Chennai", "ABCDE1234F");

select \* from mydata;

| Fullname | age | gender | salary | city | pan |
| --- | --- | --- | --- | --- | --- |
| joo | 50 | female | 9000 | Chennai | ABCDE1234F |

14. **TASK 14: Age Default Value**

Scenario: Ensure that the `age` column defaults to 25 if not provided.

Expected Output :

create table mydata(Fullname char(20) ,age int default 25,gender char(40),salary int,city char(30) ,pan char(30));

INSERT INTO mydata ( Fullname,gender,salary,city, Pan)

VALUES ( "joo","female",9000,"chennai", "ABCDE1234F");

select \* from mydata;

| Fullname | age | gender | salary | city | pan |
| --- | --- | --- | --- | --- | --- |
| joo | 25 | female | 9000 | chennai | ABCDE1234F |

15.**TASK 15: Salary Between Range Constraint**

Scenario: Ensure that the `salary` column only accepts values between 15000 and 50000.

Expected Output :

create table mydata(Fullname char(20) ,age int,gender char(40),salary int check (salary >=15000 and salary <=50000),city char(30) ,pan char(30));

INSERT INTO mydata ( Fullname,age,gender,salary,city, Pan)

VALUES ( "joo",20,"female",9000,"chennai", "ABCDE1234F");

select \* from mydata;

Error:

INSERT INTO mydata ( Fullname,age,gender,salary,city, Pan) VALUES ( "joo",20,"female",9000,"chennai", "ABCDE1234F") Error Code: 3819. Check constraint 'mydata\_chk\_1' is violated.

Error solved:

create table mydata(Fullname char(20) ,age int,gender char(40),salary int check (salary >=15000 and salary <=50000),city char(30) ,pan char(30));

INSERT INTO mydata ( Fullname,age,gender,salary,city, Pan)

VALUES ( "joo",20,"female",49999,"chennai", "ABCDE1234F");

select \* from mydata;

| Fullname | age | gender | salary | city | pan |
| --- | --- | --- | --- | --- | --- |
| joo | 20 | female | 49999 | chennai | ABCDE1234F |

16. **TASK 16: Country Default Value**

Scenario: Add a `country` column and ensure it defaults to 'India' if not provided.

Expected Output :

create table mydata(Fullname char(20) ,age int,gender char(40),salary int ,city char(30) ,pan char(30));

INSERT INTO mydata ( Fullname,age,gender,salary,city, Pan)

VALUES ( "joo",20,"female",49999,"chennai", "ABCDE1234F");

alter table mydata add column country varchar(20) default 'India';

select \* from mydata;

| Fullname | age | gender | salary | city | pan | country |
| --- | --- | --- | --- | --- | --- | --- |
| joo | 20 | female | 49999 | chennai | ABCDE1234F | India |

**Case Study 1 : Employee Management System**

Scenario:

You have been hired as a database administrator for a new company, and you need to create an Employee Management System to keep track of the employees, their departments, projects they are working on, and their attendance.

**create database if not exists EmployeeManagement;**

**use EmployeeManagement;**

**create table Departments( id int primary key auto\_increment,dep\_name varchar(50) not null);**

**create table Employees( id int primary key auto\_increment ,emp\_name varchar(50) not null,age int not null,**

**gender varchar(10) not null,salary int not null check(salary >30000),**

**dep\_id int,foreign key (dep\_id) references Departments (id));**

**create table Projects(id int primary key auto\_increment,pr\_name varchar(50) not null,**

**budget int not null check(budget >10000));**

**create table Employee\_Projects(employee\_id int ,foreign key (employee\_id) references Employees(id),**

**project\_id int,foreign key (project\_id) references Projects(id),primary key(project\_id,employee\_id));**

**create table Attendance (id int primary key auto\_increment,employee\_id int,**

**foreign key (employee\_id) references Employees(id),date date not null,status enum ('present','absent','on leave'));**

**insert into Departments (dep\_name)**

**values("HR"),("IT"),("finance"),("HR"),("IT");**

**insert into Employees(emp\_name,age,gender,salary,dep\_id)**

**values("sharime",22,"female",71000,2),**

**("pavi",23,"female",40000,1),**

**("sindhu",24,"female",31000,3),**

**("sangi",25,"female",42000,4),**

**("priya",21,"female",69000,5);**

**insert into Projects(pr\_name,budget)**

**values ("bank",50000),("school",15000),("webapp",60000),("appdev",70000),("ngo",45000);**

**insert into Employee\_Projects(employee\_id,project\_id)**

**values(1,2),(2,3),(3,4),(4,5),(5,1);**

**insert into Attendance(employee\_id,date,status)**

**values(2,'2024-03-01','present'),(3,'2024-03-01','absent'),(1,'2024-03-01','on leave'),**

**(4,'2024-03-01','absent'),(2,'2024-03-01','present');**

**select \* from Departments;**

**select\* from Employees;**

**select \* from Projects;**

**select \* from Employee\_projects;**

**select \* from Attendance;**

**1.#Retrieve all employees**

**select\* from Employees;**

| **id** | **emp\_name** | **age** | **gender** | **salary** | **dep\_id** |
| --- | --- | --- | --- | --- | --- |
| **1** | **sharime** | **22** | **female** | **71000** | **2** |
| **2** | **pavi** | **23** | **female** | **40000** | **1** |
| **3** | **sindhu** | **24** | **female** | **31000** | **3** |
| **4** | **sangi** | **25** | **female** | **42000** | **4** |
| **5** | **priya** | **21** | **female** | **69000** | **5** |

**2.#Retrieve all employees in the 'IT' department.**

**select Employees.emp\_name from Employees where id in (select id from Departments where dep\_name='IT');**

| **emp\_name** |
| --- |
| **pavi** |
| **priya** |

**3.# List all projects with a budget greater than 50000.**

**select pr\_name,budget from Projects where budget > 50000;**

| **pr\_name** | **budget** |
| --- | --- |
| **webapp** | **60000** |
| **appdev** | **70000** |

**4.#Show the names of employees and the projects they are working on.**

**select Employees.emp\_name,Projects.pr\_name from Employees join Projects on Employees.id = Projects.id;**

| **emp\_name** | **pr\_name** |
| --- | --- |
| **sharime** | **bank** |
| **pavi** | **school** |
| **sindhu** | **webapp** |
| **sangi** | **appdev** |
| **priya** | **ngo** |

**5.#Count the number of employees in each department.**

**select Departments.dep\_name,count(Employees.dep\_id) from Departments left join Employees on Departments.id=Employees.id**

**group by Departments.dep\_name;**

| **dep\_name** | **Count Employees.dep\_id** |
| --- | --- |
| **HR** | **2** |
| **IT** | **2** |
| **finance** | **1** |

**6.Find employees with a salary greater than 50000.**

**select Employees.emp\_name,Employees.salary from Employees where salary >50000;**

| emp\_name | salary |
| --- | --- |
| sharime | 71000 |
| priya | 69000 |

**7.Get attendance records for a specific employee.**

**select Attendance.date,Attendance.status from Attendance where Attendance.employee\_id=3;**

| date | status |
| --- | --- |
| 2024-03-01 | absent |

**8.Update Records: Increase the salary of employees in the 'HR' department by 10%.**

**update Employees join Departments on Employees.dep\_id=Departments.id set Employees.salary= Employees.salary \*1.10 where Departments.dep\_name='HR';**

| **id** | **emp\_name** | **age** | **gender** | **salary** | **dep\_id** |
| --- | --- | --- | --- | --- | --- |
| **1** | **sharime** | **22** | **female** | **71000** | **2** |
| **2** | **pavi** | **23** | **female** | **44000** | **1** |
| **3** | **sindhu** | **24** | **female** | **31000** | **3** |
| **4** | **sangi** | **25** | **female** | **46200** | **4** |
| **5** | **priya** | **21** | **female** | **69000** | **5** |

**9.Change the department of an employee.**

**update Employees set dep\_id = (select id from Departments where dep\_name = 'Finance') where id = 5;**

| **id** | **emp\_name** | **age** | **gender** | **salary** | **dep\_id** |
| --- | --- | --- | --- | --- | --- |
| **1** | **sharime** | **22** | **female** | **71000** | **2** |
| **2** | **pavi** | **23** | **female** | **44000** | **1** |
| **3** | **sindhu** | **24** | **female** | **31000** | **3** |
| **4** | **sangi** | **25** | **female** | **46200** | **4** |
| **5** | **priya** | **21** | **female** | **69000** | **3** |

**10. Delete Records:**

**#Delete a project that is completed.**

**delete from Projects where pr\_name = 'ngo';**

| **id** | **pr\_name** | **budget** |
| --- | --- | --- |
| **6** | **bank** | **50000** |
| **7** | **school** | **15000** |
| **8** | **webapp** | **60000** |
| **9** | **appdev** | **70000** |

**12. Constraints and Modifications:**

**# Add a unique constraint to the `email` column in `Employees` table.**

**alter table Employees add column email varchar(50) unique;**

| **id** | **emp\_name** | **age** | **gender** | **salary** | **dep\_id** | **email** |
| --- | --- | --- | --- | --- | --- | --- |
| **1** | **sharime** | **22** | **female** | **71000** | **2** | **null** |
| **2** | **pavi** | **23** | **female** | **40000** | **1** | **null** |
| **3** | **sindhu** | **24** | **female** | **31000** | **3** | **null** |
| **4** | **sangi** | **25** | **female** | **42000** | **4** | **null** |
| **5** | **priya** | **21** | **female** | **69000** | **5** | **null** |

**13.Ensure `age` is always greater than 18.**

**alter table Employees add check (Age>18);**

**14.Add a default value 'Unknown' to `gender` if not specified.**

**Alter table Employees modify gender varchar(20) default 'Unknown';**

**Case Study 2: Library Management System**

**Scenario:**

**You are tasked with creating a database to manage a library's inventory, members, and borrowing records.**

**CREATE DATABASE LibraryDB;**

**USE LibraryDB;**

**CREATE TABLE Books (**

**id INT PRIMARY KEY AUTO\_INCREMENT,**

**title VARCHAR(100) NOT NULL,**

**author VARCHAR(50) NOT NULL,**

**published\_year YEAR NOT NULL,**

**genre VARCHAR(30)**

**)AUTO\_INCREMENT =1;**

**CREATE TABLE Members (**

**id INT AUTO\_INCREMENT PRIMARY KEY,**

**name VARCHAR(50) NOT NULL,**

**membership\_date DATE NOT NULL**

**)AUTO\_INCREMENT =1;**

**CREATE TABLE Borrowing (**

**id INT AUTO\_INCREMENT PRIMARY KEY,**

**book\_id INT,**

**member\_id INT,**

**borrow\_date DATE NOT NULL,**

**return\_date DATE,**

**FOREIGN KEY (book\_id) REFERENCES Books(id),**

**FOREIGN KEY (member\_id) REFERENCES Members(id)**

**)AUTO\_INCREMENT =1;**

**INSERT INTO Books (title, author, published\_year, genre)**

**VALUES**

**('To Kill a Mockingbird', 'Harper Lee', 1960, 'Fiction'),**

**('1984', 'George Orwell', 1949, 'Dystopian'),**

**('The Great Gatsby', 'F. Scott Fitzgerald', 1925, 'Classic'),**

**('Moby-Dick', 'Herman Melville', 1990, 'Adventure'),**

**('Pride and Prejudice', 'Jane Austen', 1911, 'Romance');**

**INSERT INTO Members (name, membership\_date)**

**VALUES**

**('Sharime', '2023-01-15'),**

**('pavi', '2022-11-03'),**

**('sangi', '2024-03-22'),**

**('sindhu', '2020-06-10'),**

**('priya', '2021-09-28');**

**INSERT INTO Borrowing (book\_id, member\_id, borrow\_date, return\_date)**

**VALUES**

**(1, 1, '2024-08-01', '2024-08-15'),**

**(2, 2, '2024-09-05', '2024-09-20'),**

**(3, 3, '2024-10-10', NULL),**

**(4, 4, '2024-06-10', '2024-06-25'),**

**(5, 5, '2024-03-05', '2024-03-20');**

**1. Queries:**

**# List all books.**

**select \* from books;**

| **id** | **title** | **author** | **published\_year** | **genre** |
| --- | --- | --- | --- | --- |
| **1** | **To Kill a Mockingbird** | **Harper Lee** | **1960** | **Fiction** |
| **2** | **1984** | **George Orwell** | **1949** | **Dystopian** |
| **3** | **The Great Gatsby** | **F. Scott Fitzgerald** | **1925** | **Classic** |
| **4** | **Moby-Dick** | **Herman Melville** | **1990** | **Adventure** |
| **5** | **Pride and Prejudice** | **Jane Austen** | **1911** | **Romance** |

**2.#List all members.**

**select \* from members;**

| **id** | **name** | **membership\_date** |
| --- | --- | --- |
| **1** | **Sharime** | **2023-01-15** |
| **2** | **pavi** | **2022-11-03** |
| **3** | **sangi** | **2024-03-22** |
| **4** | **sindhu** | **2020-06-10** |
| **5** | **priya** | **2021-09-28** |

**3.#Show borrowing records with member names and book titles.**

**Select Borrowing.borrow\_date,Borrowing.return\_date,members.name,**

**books.title from Borrowing join members on Borrowing.id=members.id**

**JOIN books ON Borrowing.member\_id = books.id;**

| **borrow\_date** | **return\_date** | **name** | **title** |
| --- | --- | --- | --- |
| **2024-08-01** | **2024-08-15** | **Sharime** | **To Kill a Mockingbird** |
| **2024-09-05** | **2024-09-20** | **pavi** | **1984** |
| **2024-10-10** |  | **sangi** | **The Great Gatsby** |
| **2024-06-10** | **2024-06-25** | **sindhu** | **Moby-Dick** |
| **2024-03-05** | **2024-03-20** | **priya** | **Pride and Prejudice** |

**4.# Count the number of books borrowed by each member.**

**SELECT Members.name, COUNT(Borrowing.id) FROM Members LEFT JOIN Borrowing ON Members.id = Borrowing.member\_id**

**GROUP BY Members.id;**

| **name** | **COUNT(Borrowing.id)** |
| --- | --- |
| **pavi** | **1** |
| **priya** | **1** |
| **sangi** | **1** |
| **Sharime** | **1** |
| **sindhu** | **1** |

**Case Study 3: Student Management System**

**Scenario:**

**Create a database to manage students, their courses, and grades.**

**CREATE DATABASE StudentDB;**

**USE StudentDB;**

**CREATE TABLE Students (**

**id INT AUTO\_INCREMENT PRIMARY KEY,**

**name VARCHAR(50) NOT NULL,**

**age INT NOT NULL,**

**email VARCHAR(50) NOT NULL UNIQUE**

**)auto\_increment=1;**

**CREATE TABLE Courses (**

**id INT AUTO\_INCREMENT PRIMARY KEY,**

**name VARCHAR(50) NOT NULL**

**)auto\_increment=1;**

**CREATE TABLE Enrollments (**

**student\_id INT,**

**course\_id INT,**

**grade CHAR(1),**

**PRIMARY KEY (student\_id, course\_id),**

**FOREIGN KEY (student\_id) REFERENCES Students(id),**

**FOREIGN KEY (course\_id) REFERENCES Courses(id)**

**);**

**INSERT INTO Students (name, age, email) VALUES**

**('Sharime', 20, 'sharime@example.com'),**

**('pavi', 22, 'pavi@example.com'),**

**('sindhu', 19, 'sindhu@example.com');**

**INSERT INTO Courses (name) VALUES**

**('CSE'),**

**('EEE'),**

**('IT');**

**INSERT INTO Enrollments (student\_id, course\_id, grade) VALUES**

**(1, 1, 'A'),**

**(1, 2, 'B'),**

**(2, 1, 'B'),**

**(2, 3, 'A'),**

**(3, 2, 'C');**

**1.Queries:**

**List all students.**

**SELECT \* FROM Students;**

| **id** | **name** | **age** | **email** |
| --- | --- | --- | --- |
| **1** | **Sharime** | **20** | **sharime@example.com** |
| **2** | **pavi** | **22** | **pavi@example.com** |
| **3** | **sindhu** | **19** | **sindhu@example.com** |

**2.#List all courses.**

**SELECT \* FROM Courses;**

| **id** | **name** |
| --- | --- |
| **1** | **CSE** |
| **2** | **EEE** |
| **3** | **IT** |

**3.#**Show the names of students and the courses they are enrolled in.

**SELECT Students.name, Courses.name as courses\_nameFROM Students JOIN Enrollments ON Students.id = Enrollments.student\_id**

**JOIN Courses ON Enrollments.Course\_id = Courses.id;**

| **name** | **Courses\_name** |
| --- | --- |
| **Sharime** | **CSE** |
| **Sharime** | **EEE** |
| **pavi** | **CSE** |
| **pavi** | **IT** |
| **sindhu** | **EEE** |

**4.#Count the number of students in each course.**

**SELECT Courses.name as courses\_name,COUNT(Enrollments.student\_id) FROM Courses LEFT JOIN Enrollments ON Courses.id = Enrollments.course\_id GROUP BY Courses.id;**

| **courses\_name** | **COUNT(Enrollments.student\_id)** |
| --- | --- |
| **CSE** | **2** |
| **EEE** | **2** |
| **IT** | **1** |

**Case Study 4 : Retail Store Inventory System**

**Scenario:**

**Design a database to manage a retail store's inventory, suppliers, and sales.**

**create database RetailDB;**

**use RetailDB;**

**create table Products(id INT Primary Key Auto\_Increment,name VARCHAR(50) Not Null,price DECIMAL(10, 2) Not Null, Check**

**(price > 0),stock INT Not Null Check (Stock >= 0))auto\_increment=1;**

**CREATE TABLE Suppliers (id INT AUTO\_INCREMENT PRIMARY KEY,name VARCHAR(50) NOT NULL,contact VARCHAR(50))auto\_increment=1;**

**CREATE TABLE Sales (id INT AUTO\_INCREMENT PRIMARY KEY,product\_id INT,date DATE NOT NULL,quantity INT NOT NULL CHECK (quantity > 0),**

**total\_price DECIMAL(10, 2) NOT NULL,FOREIGN KEY (product\_id) REFERENCES Products(id))auto\_increment=1;**

**INSERT INTO Products (name, price, stock)**

**VALUES ('dariy milk', 10.00, 50),**

**('suger', 15.50, 5),**

**('pen', 20.00, 100),**

**('pencil', 5.00, 0);**

**INSERT INTO Suppliers (name, contact)**

**VALUES ('Supplier1', '758-767-3455'),**

**('Supplier2', '987-123-7865');**

**INSERT INTO Sales (product\_id, date, quantity, total\_price)**

**VALUES (1, '2024-01-01', 5, 50.00),**

**(2, '2024-01-02', 2, 31.00),**

**(3, '2024-01-03', 10, 200.00),**

**(1, '2024-01-04', 3, 30.00);**

**1.Queries:**

**# List all products.**

**SELECT \* FROM Products;**

| **id** | **name** | **stock** | **price** |
| --- | --- | --- | --- |
| **1** | **dariy milk** | **10.00** | **50** |
| **2** | **suger** | **15.50** | **5** |
| **3** | **pen** | **20.00** | **100** |
| **4** | **pencil** | **5.00** | **0** |

**2.#Show products with stock less than 10.**

**SELECT \* FROM Products WHERE stock < 10;**

| **Id** | **name** | **price** | **stock** |
| --- | --- | --- | --- |
| **2** | **suger** | **15.50** | **5** |
| **4** | **pencil** | **5.00** | **0** |

**3.#List sales with product names and total prices.**

**SELECT products.name AS product\_name, sales.date, sales.quantity, sales.total\_price FROM sales JOIN Products**

**ON sales.product\_id = products.id;**

| **product\_name** | **date** | **quantity** | **total\_price** |
| --- | --- | --- | --- |
| **dariy milk** | **2024-01-01** | **5** | **50.00** |
| **suger** | **2024-01-02** | **2** | **31.00** |
| **pen** | **2024-01-03** | **10** | **200.00** |
| **dariy milk** | **2024-01-04** | **3** | **30.00** |
| **dariy milk** | **2024-01-01** | **5** | **50.00** |

**4.#Calculate the total sales for each product.**

**SELECT products.name AS product\_name, SUM(Sales.total\_price) AS total\_sales FROM Sales JOIN Products ON Sales.product\_id = products.id GROUP BY products.name;**

| **product\_name** | **total\_sales** |
| --- | --- |
| **dariy milk** | **160.00** |
| **suger** | **62.00** |
| **pen** | **400.00** |

**Case Study 5: Hospital Management System**

**Scenario:**

**Create a database to manage patients, doctors, and appointments in a hospital.**

**CREATE DATABASE HospitalDB;**

**use HospitalDB;**

**CREATE TABLE Patients (**

**id INT AUTO\_INCREMENT PRIMARY KEY,**

**name VARCHAR(50) NOT NULL,**

**age INT NOT NULL,**

**gender VARCHAR(10),**

**contact VARCHAR(50)**

**)auto\_increment=1;**

**CREATE TABLE Doctors (**

**id INT AUTO\_INCREMENT PRIMARY KEY,**

**name VARCHAR(50) NOT NULL,**

**specialization VARCHAR(50) NOT NULL**

**)auto\_increment=1;**

**CREATE TABLE Appointments (**

**id INT AUTO\_INCREMENT PRIMARY KEY,**

**patient\_id INT,**

**doctor\_id INT,**

**appointment\_date DATE NOT NULL,**

**status ENUM('Scheduled', 'Completed', 'Cancelled'),**

**FOREIGN KEY (patient\_id) REFERENCES Patients(id),**

**FOREIGN KEY (doctor\_id) REFERENCES Doctors(id)**

**)auto\_increment=1;**

**INSERT INTO Patients (name, age, gender, contact) VALUES**

**('John ', 30, 'Male', '971-456-7890'),**

**('yuvarani', 40, 'Female', '987-764-3210'),**

**('kathir', 25, 'Male', '638-666-7777'),**

**('pavi', 35, 'Female', '944-999-0000');**

**INSERT INTO Doctors (name, specialization) VALUES**

**('Dr. Sharime', 'Cardiology'),**

**('Dr. sindhu', 'Neurology'),**

**('Dr. sangi', 'Orthopedics'),**

**('Dr. priya', 'Pediatrics');**

**INSERT INTO Appointments (patient\_id, doctor\_id, appointment\_date, status) VALUES**

**(1, 1, '2024-10-16', 'Scheduled'),**

**(2, 2, '2024-10-17', 'Completed'),**

**(3, 3, '2024-10-18', 'Cancelled'),**

**(4, 4, '2024-10-19', 'Scheduled'),**

**(1, 2, '2024-10-20', 'Completed');**

**1. Queries:**

**#List all patients.**

**SELECT \* FROM Patients;**

| **id** | **name** | **age** | **gender** | **contact** |
| --- | --- | --- | --- | --- |
| **1** | **John** | **30** | **Male** | **971-456-7890** |
| **2** | **yuvarani** | **40** | **Female** | **987-764-3210** |
| **3** | **kathir** | **25** | **Male** | **638-666-7777** |
| **4** | **pavi** | **35** | **Female** | **944-999-0000** |

**2.#List all doctors.**

**SELECT \* FROM Doctors;**

| **id** | **Name** | **specialization** |
| --- | --- | --- |
| **1** | **Dr. Sharime** | **Cardiology** |
| **2** | **Dr. sindhu** | **Neurology** |
| **3** | **Dr. sangi** | **Orthopedics** |
| **4** | **Dr. priya** | **Pediatrics** |

**3.#Show appointments with patient and doctor names.**

**SELECT Appointments.id AS appointment\_id,patients.name AS patient\_name,doctors.name AS**

**doctor\_name,Appointments.appointment\_date,**

**Appointments.status FROM Appointments JOIN Patients ON Appointments.patient\_id = patients.id JOIN Doctors**

**ON Appointments.doctor\_id = Doctors.id;**

| **appointment\_id** | **Patient\_name** | **doctor\_name** | **date** | **status** |
| --- | --- | --- | --- | --- |
| **6** | **John** | **Dr. Sharime** | **2024-10-16** | **Scheduled** |
| **10** | **John** | **Dr. sindhu** | **2024-10-20** | **Completed** |
| **7** | **yuvarani** | **Dr. sindhu** | **2024-10-17** | **Completed** |
| **8** | **kathir** | **Dr. sangi** | **2024-10-18** | **Cancelled** |
| **9** | **pavi** | **Dr. priya** | **2024-10-19** | **Scheduled** |

**4.#Count appointments per doctor.**

**SELECT Doctors.name AS doctor\_name,COUNT( Appointments.id) AS appointment\_count FROM Doctors**

**LEFT JOIN Appointments ON Doctors.id = Appointments.doctor\_id GROUP BY Doctors.id;**

| **doctor\_name** | **appointment\_count** |
| --- | --- |
| **Dr. Sharime** | **1** |
| **Dr. sindhu** | **2** |
| **Dr. sangi** | **1** |
| **Dr. priya** | **1** |

**Case Study 6: Online Learning Platform**

**Scenario:**

**Design a database for an online learning platform to manage courses, instructors, and student enrollments.**

**CREATE DATABASE LearningDB;**

**USE LearningDB;**

**CREATE TABLE Courses (**

**id INT AUTO\_INCREMENT PRIMARY KEY,**

**title VARCHAR(100) NOT NULL,**

**description TEXT**

**)auto\_increment=1;**

**CREATE TABLE Instructors (**

**id INT AUTO\_INCREMENT PRIMARY KEY,**

**name VARCHAR(50) NOT NULL,**

**expertise VARCHAR(50) NOT NULL**

**)auto\_increment=1;**

**CREATE TABLE Students (**

**id INT AUTO\_INCREMENT PRIMARY KEY,**

**name VARCHAR(50) NOT NULL,**

**email VARCHAR(50) NOT NULL UNIQUE**

**)auto\_increment=1;**

**CREATE TABLE Enrollments (**

**student\_id INT,**

**course\_id INT,**

**enrollment\_date DATE NOT NULL,**

**PRIMARY KEY (student\_id, course\_id),**

**FOREIGN KEY (student\_id) REFERENCES Students(id),**

**FOREIGN KEY (course\_id) REFERENCES Courses(id)**

**)auto\_increment=1;**

**INSERT INTO Courses (title, description) VALUES**

**('MySql', 'A beginner course in mysql concepts.'),**

**('Data Structures and Algorithms', 'Learn about data structures and algorithm design.'),**

**('Web Development', 'An introduction to front-end and back-end web development.');**

**INSERT INTO Instructors (name, expertise) VALUES**

**('yuva', 'Programming'),**

**('kathi', 'Algorithms'),**

**('pavi', 'Web Development');**

**INSERT INTO Students (name, email) VALUES**

**('sharime', 'sharme.doe@example.com'),**

**('sindhu', 'sindhu.smith@example.com'),**

**('sangee', 'sangee.lee@example.com');**

**INSERT INTO Enrollments (student\_id, course\_id, enrollment\_date) VALUES**

**(1, 1, '2024-07-01'),**

**(1, 2, '2024-09-01'),**

**(2, 3, '2024-10-10'),**

**(3, 1, '2024-11-15');**

**1. Queries:**

**# List all courses.**

**SELECT \* FROM Courses;**

| **id** | **title** | **description** |
| --- | --- | --- |
| **1** | **MySql** | **A beginner course in mysql concepts.** |
| **2** | **Data Structures and Algorithms** | **Learn about data structures and algorithm design.** |
| **3** | **Web Development** | **An introduction to front-end and back-end web development.** |

**2.# List all instructors.**

**SELECT \* FROM Instructors;**

| **id** | **instructor** | **expertise** |
| --- | --- | --- |
| **1** | **yuva** | **Programming** |
| **2** | **kathi** | **Algorithms** |
| **3** | **pavi** | **Web Development** |

**3.#Show enrollments with student names and course titles.**

**SELECT Students.name AS student\_name,Courses.title AS course\_title,Enrollments.enrollment\_date**

**FROM Enrollments JOIN Students ON Enrollments.student\_id = Students.id**

**JOIN Courses ON Enrollments.course\_id = Courses.id;**

| **student\_name** | **courses\_name** | **Enroolment\_date** |
| --- | --- | --- |
| **sharime** | **MySql** | **2024-07-01** |
| **sharime** | **Data Structures and Algorithms** | **2024-09-01** |
| **sindhu** | **Web Development** | **2024-10-10** |
| **sangee** | **MySql** | **2024-11-15** |

**4.#Count the number of students enrolled in each course.**

**SELECT Courses.title AS course\_title,COUNT(Enrollments.student\_id) AS student\_count**

**FROM Courses LEFT JOIN Enrollments ON Courses.id = Enrollments.course\_id**

**GROUP BY Courses.id;**

| **course\_title** | **student\_count** |
| --- | --- |
| **MySql** | **2** |
| **Data Structures and Algorithms** | **1** |
| **Web Development** | **1** |