**An SQL project based on Hospital Management System**

**Group Name: Runtime Terror**

**Group Members:**

* Anvi
* Aruna
* Atreyi
* Gauri
* Riddhi
* Vaibhavi

**Project Abstract:**

Structured Query Language (SQL) plays a critical role in managing hospital databases by providing an efficient and systematic way to store, retrieve, and manipulate healthcare-related data. In a hospital setting, maintaining patient records, doctor schedules, appointments, medical histories, and billing details is crucial for operational efficiency. SQL enables data integrity, security, and quick access to information through its powerful querying capabilities. It helps generate reports that aid in decision-making, such as identifying high-risk patients, tracking revenue, and managing resource allocation. By leveraging SQL, hospitals can ensure smooth workflow automation, minimize errors, and enhance patient care through real-time data management.

**Objective:**

You are tasked with designing and implementing a Hospital Database Management System using MySQL. The goal is to create a well-structured relational database that stores hospital-related data and allows efficient querying to generate reports and derive insights.

**Project Scope:**

1. **Database Creation:**

Design and create a relational database for hospital management using MySQL.

Define appropriate tables, relationships, primary keys, and foreign keys to ensure data integrity.

1. **Data Ingestion:**

Load data from the provided files into the created tables using SQL INSERT queries or bulk import techniques.

1. **Query Writing & Report Generation:**

Write SQL queries to answer specific business questions and generate reports for hospital operations and decision-making.

**Dataset & Tables:**

* The database includes all the tables in the form of CSV files.

**Tools & Technologies Used:**

* MYSQL

**SQL Queries :**

1. **Write a SQL query to identify the physicians who are the department heads.**

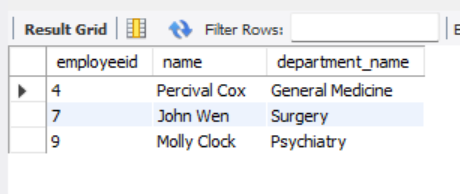
* **Query:**

SELECT p.employeeid, p.name, d.name AS department\_name

FROM physician p

JOIN department d ON p.employeeid = d.head;

* **Result :**

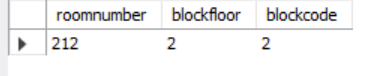
****

1. **Write a SQL query to locate the floor and block where room number 212 is located.**

* **Query :**

SELECT roomnumber, blockfloor, blockcode FROM room WHERE roomnumber = 212;

* **Result:**

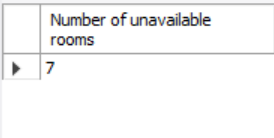


1. **Write a SQL query to count the number of unavailable rooms. Return count as "Number of unavailable rooms.**

* **Query :**

SELECT COUNT(\*) AS 'Number of unavailable rooms' FROM room WHERE unavailable='t';

* **Result :**

****

1. **Write a SQL query to identify the physician and the department with which he or she is affiliated.**

* **Query:**

SELECT

p.name AS physician\_name,

d.name AS department\_name

FROM

physician AS p

JOIN

affiliated\_with AS a ON p.employeeid = a.physician

JOIN

department AS d ON d.departmentid = a.department;

* **Result:**

****

1. **Write a SQL query to find those physicians who have received special training.**

* **Query:**

SELECT employeeid, p.name,pd.name

FROM physician as p,

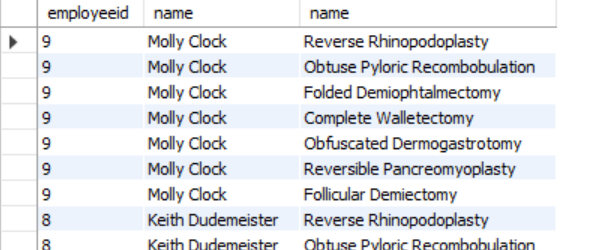
trained\_in as t,

proceduredata as pd;

where p.employeeid = t.physician,

t.treatment = d.code;

* **Result:**

****

1. **Write a SQL query to identify the patients and the number of physicians with whom they have scheduled appointments.**

* **Query:**

SELECT P1.ssn,P1.name,COUNT(DISTINCT A1.physician) as total\_appointments

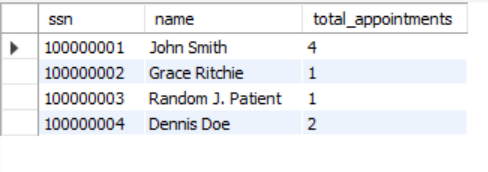
FROM patient as P1

LEFT JOIN appointment AS A1

ON P1.ssn = A1.patient

GROUP BY P1.ssn,P1.name;

* **Result:**

****

1. **Write a SQL query to count the number of unique patients who have been scheduled for examination room 'C'.**

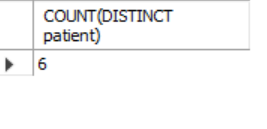
* **Query :**

SELECT COUNT(DISTINCT patient)

FROM appointment

WHERE examinationroom = 'C';

* **Result:**

****

1. **Write a SQL query to count the number of available rooms for each floor in each block. Sort the result-set on floor ID, ID of the block.**

**SELECT**

* **Query:**

blockfloor AS floor\_id,

blockcode AS block\_id,

COUNT(\*) AS available\_rooms

FROM

room

WHERE

unavailable = 'f' -- Only count available rooms

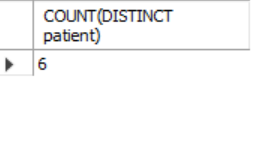
GROUP BY

blockfloor, blockcode

ORDER BY

blockfloor, blockcode;

* **Result:**

****

1. **Create a view to display the name of the patients, their block, floor, and room number where they are admitted.**

* **Query:**

CREATE VIEW PatientRoomDetails AS

SELECT

p.name AS patient\_name,

r.blockcode,

r.blockfloor,

r.roomnumber

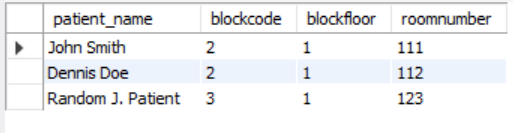
FROM patient p

JOIN stay s ON p.ssn = s.patient -- Linki patients with their stay details

JOIN room r ON s.room = r.roomnumber; -- Link stay details to rooms

SELECT \* FROM PatientRoomDetails;

* **Result:**

****

1. **Write a SQL query to find those patients who have undergone a procedure costing more than $5,000, as well as the name of the physician who has provided primary care, should be identified.**

* **Query:**

SELECT

pat.name AS patient\_name,

phy.name AS physician\_name,

prc.cost AS procedure\_cost

FROM undergoes u

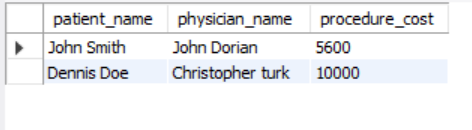
JOIN patient pat ON u.patient = pat.ssn

JOIN physician phy ON pat.pcp = phy.employeeid

JOIN procedure\_name prc ON u.procedure = prc.code

WHERE prc.cost > 5000;

* **Result:**

****

1. **Write a SQL query to identify those patients whose primary care is provided by a physician who is not the head of any department.**

* **Query:**

SELECT

pat.name AS patient\_name,

phy.name AS physician\_name

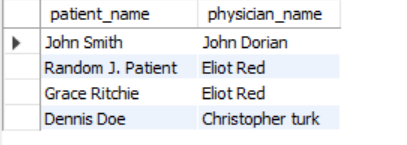
FROM patient pat

JOIN physician phy ON pat.pcp = phy.employeeid

LEFT JOIN department dept ON phy.employeeid = dept.head

WHERE dept.head IS NULL;

* **Result:**

****

1. **Retrieve the names of patients who have been prescribed at least one medication by a physician from the Psychiatry department using a subquery.**

* **Query:**

SELECT

P.ssn AS patient\_ssn,

P.name AS patient\_name

FROM patient AS P

WHERE P.ssn IN (

SELECT PR.patient

FROM prescribes AS PR

JOIN physician AS Ph ON PR.physician = Ph.employeeid

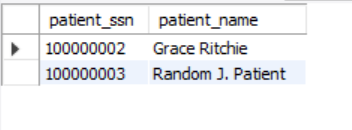
JOIN affiliated\_with AS AW ON Ph.employeeid = AW.physician

JOIN department AS D ON AW.department = D.departmentid

WHERE D.name = 'Psychiatry'

);

* **Result:**

****

1. **Create a trigger that prevents inserting a new appointment if the physician does not have a primary affiliation with a department.**

* **Query:**

**-- First remove any existing trigger**

DROP TRIGGER IF EXISTS prevent\_non\_primary\_affiliation;

DELIMITER //

CREATE TRIGGER prevent\_non\_primary\_affiliation

BEFORE INSERT ON appointment

FOR EACH ROW

BEGIN

DECLARE is\_primary VARCHAR(1);

**-- Get the primary affiliation status**

SELECT primaryaffiliation INTO is\_primary

FROM affiliated\_with

WHERE physician = NEW.physician

LIMIT 1;

**-- Block if no record exists or if not primary**

IF is\_primary IS NULL OR is\_primary != 't' THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = CONCAT('Physician ', NEW.physician, ' is not primarily affiliated');

END IF;

END//

DELIMITER ;

**INSERT STATEMENTS:**

**1. Valid insert:**

INSERT INTO appointment

(appointmentid, patient, prepnurse, physician, startdatetime, enddatetime, examinationroom)

VALUES

(2001, 3001, NULL, 1, '2023-12-01 09:00:00', '2023-12-01 09:30:00', 'Exam Room 1');

**2. Invalid insert:**

-- Physician 3 has primaryaffiliation = f - SHOULD FAIL

INSERT INTO appointment

(appointmentid, patient, prepnurse, physician, startdatetime, enddatetime, examinationroom)

VALUES

(2002, 3002, NULL, 3, '2023-12-01 10:00:00', '2023-12-01 10:30:00', 'Exam Room 2');

**-- Expected error: "Physician 3 is not primarily affiliated"**

* **Result:**

****

1. **Update the insurance ID of patients whose primary care physician (PCP) is 'John Dorian' to a new value '99999999'.**

* **Query:**

SET SQL\_SAFE\_UPDATES = 0;

-- allows to update and delete

UPDATE patient

SET insuranceid = '99999999'

WHERE pcp = (

SELECT employeeid

FROM physician

WHERE name = 'John Dorian'

);

SET SQL\_SAFE\_UPDATES = 1;

-- sets back to safe mode

optional-to display the update

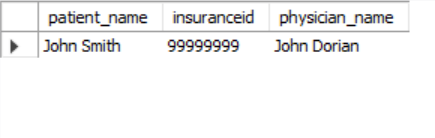
[[SELECT p.name AS patient\_name, p.insuranceid, ph.name AS physician\_name

FROM patient p

JOIN physician ph ON p.pcp = ph.employeeid

WHERE ph.name = 'John Dorian';]]

* **Result:**

****

1. **Retrieve each physician's name along with the number of appointments they have, and show the ranking of each physician based on the number of appointments in descending order.**

* **Query:**

SELECT

ph.name AS physician\_name,

COUNT(a.appointmentid) AS appointment\_count,

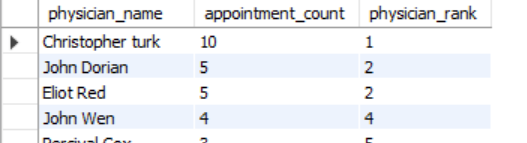
RANK() OVER (ORDER BY COUNT(a.appointmentid) DESC) AS physician\_rank

FROM physician ph

LEFT JOIN appointment a ON ph.employeeid = a.physician

GROUP BY ph.employeeid, ph.name;

* **Result:**

****

### 

### 

### 

### 

### 

### 

### 

### 

### **Valuable Insights for Hospital Data Analysis :**

#### **Physician & Department Analysis**

* Identify department heads and their respective departments.
* Analyze physician affiliations across departments to understand workload distribution and cross-department collaboration.

#### **Appointment Trends**

* Rank physicians based on the total number of appointments.
* Track how many different physicians each patient has interacted with to measure continuity of care.

#### **Room Utilization**

* Monitor room availability and usage by floor and block.
* Optimize space management by identifying underutilized or overbooked rooms.

#### **Treatment & Prescription Patterns**

* Identify patients who have undergone high-cost procedures.
* Analyze prescriptions originating from specific departments (e.g., Psychiatry) to detect patterns or anomalies.

#### **Data Validation Rules**

* Enforce data integrity by preventing appointments for physicians not assigned to a primary department.
* Ensure consistent and accurate department affiliation for every physician.

**Conclusion:**

This project demonstrates how structured hospital data can be transformed into meaningful insights to improve operational efficiency and patient care. By analyzing physician affiliations, appointment trends, room utilization, treatment costs, and enforcing critical validation rules, we’ve highlighted opportunities for better decision-making across departments.

Implementing automated checks—such as blocking appointments for non-primarily affiliated physicians—ensures data accuracy and integrity. Moreover, administrative capabilities like bulk insurance updates streamline routine processes, reducing manual workload and minimizing errors.

Overall, this data-driven approach not only enhances resource allocation and treatment transparency but also lays the groundwork for scalable improvements in healthcare management systems.