Healthcare Database Analytics

Improving Hospital Operations and Patient Care Using SQL

1. Project Overview

The **Healthcare Analytics** project focuses on designing a relational healthcare database and performing structured SQL analysis to extract insights related to patient demographics, hospital utilization, treatment costs, and operational performance. The project simulates a real-world hospital information system by integrating patient, hospital, admission, and treatment data.

The primary objective is to demonstrate how **SQL-based analytics** can support healthcare decision-making by improving operational efficiency, optimizing resource utilization, and enhancing patient care outcomes.

2. Database Design & Architecture

A relational database named **HealthcareDB** was created using SQL, consisting of the following core tables:

Tables Created

- **PATIENTS** – Stores patient demographics (age, gender, address)
- **HOSPITALS** – Stores hospital details (location, capacity)
- **ADMISSIONS** – Tracks hospital admissions, discharge dates, and admission reasons
- **TREATMENTS** – Records medical procedures, costs, and outcomes

Key Design Features

- Primary and foreign key constraints to maintain data integrity
- One-to-many relationships between:
    - Patients → Admissions
    - Hospitals → Admissions
    - Admissions → Treatments
- Proper data types for dates, costs, and categorical attributes

This schema supports scalable healthcare analytics and mimics real-world hospital databases.

3. Data Population

Sample data was inserted into all tables to simulate hospital operations, including:

- 10 patients with varied demographics
- 5 hospitals across different locations
- Multiple admissions with discharge tracking
- Treatments with procedure costs and outcomes

4. SQL Analytics Performed

4.1 Patient Demographics Analysis

- Calculated patient count and average age grouped by gender
- Helped identify demographic distribution of hospital patients

4.2 Hospital Utilization

- Identified hospitals with the highest number of admissions
- Highlighted potential capacity and resource demand issues

4.3 Treatment Cost Analysis

- Computed total treatment costs per hospital
- Identified high-revenue hospitals and cost-intensive care areas

4.4 Length of Stay Analysis

- Calculated average patient length of stay by hospital
- Identified extended hospital stays contributing to higher costs

4.5 Advanced Filtering & Business Queries

- Patients staying longer than 7 days
- Treatments performed frequently across hospitals
- Combined admission and treatment data for complete patient histories
- Admission reason analysis (e.g., surgery vs therapy)

5. Advanced SQL Techniques Used

Subqueries

- Identified hospitals with the highest average treatment cost

Views

- Created **HospitalPerformance** view to summarize:
    - Number of admissions
    - Average length of stay

- Total revenue per hospital

Window Functions

- RANK() to rank hospitals by total revenue
- DENSE_RANK() to rank treatments by frequency

These techniques demonstrate advanced SQL skills used in real-world analytics and reporting.

6. Code

```
-- Creating a database named HealthcareDB.

CREATE DATABASE HealthcareDB



-- creating tables within healthcaredb

USE HealthcareDB



CREATE TABLE PATIENTS(

PatientID INT AUTO_INCREMENT PRIMARY KEY,

FullName VARCHAR(20) NOT NULL,

Age INT,

Gender VARCHAR(20),

Address VARCHAR(200)

);

ALTER TABLE PATIENTS

CHANGE Gender Gender VARCHAR(20)

CREATE TABLE HOSPITALS(

HospitalID INT AUTO_INCREMENT PRIMARY KEY,

HospitalName VARCHAR(100) NOT NULL,

Location VARCHAR(255),
```

```
Capacity INT

)


CREATE TABLE ADMISSIONS(

AdmissionID INT AUTO_INCREMENT PRIMARY KEY,

PatientID INT ,

HospitalID INT ,

AdmissionDate DATE NOT NULL,

DischargeDate DATE ,

ReasonForAdmission VARCHAR(200),

CONSTRAINT fk_patients FOREIGN KEY (PatientID) REFERENCES PATIENTS(PatientID),

CONSTRAINT fk_hospitals FOREIGN KEY (HospitalID) REFERENCES
HOSPITALS(HospitalID)

)


CREATE TABLE TREATMENTS(

TreatmentID INT AUTO_INCREMENT PRIMARY KEY,

AdmissionID INT,

ProcedureName VARCHAR(200),

Cost DECIMAL,

Outcome VARCHAR(100),

FOREIGN KEY (AdmissionID) REFERENCES ADMISSIONS(AdmissionID)

)


-- Insert data into Patients table
```

```
INSERT INTO Patients (FullName, Age, Gender, Address) VALUES

('John Doe', 45, 'Male', '123 Elm Street'),

('Jane Smith', 34, 'Female', '456 Oak Avenue'),

('Sam Brown', 29, 'Male', '789 Pine Road'),

('Lisa White', 52, 'Female', '321 Maple Lane'),

('Tom Green', 67, 'Male', '654 Birch Blvd'),

('Alice Johnson', 40, 'Female', '987 Willow Court'),

('Robert Black', 60, 'Male', '564 Cypress Road'),

('Emily Davis', 25, 'Female', '321 Cedar Avenue'),

('Michael Scott', 50, 'Male', '742 Birch Lane'),

('Sarah Taylor', 33, 'Female', '159 Spruce Drive');


-- Insert data into Hospitals table

INSERT INTO Hospitals (HospitalName, Location, Capacity) VALUES

('General Hospital', 'New York', 500),

('City Clinic', 'Los Angeles', 200),

('Central Medical Center', 'Chicago', 300),

('Regional Health Facility', 'Houston', 150),

('Sunrise Hospital', 'Phoenix', 400);


-- Insert data into Admissions table

INSERT INTO Admissions (PatientID, HospitalID, AdmissionDate, DischargeDate,
ReasonForAdmission) VALUES

(1, 1, '2024-11-01', '2024-11-05', 'Surgery'),

(2, 2, '2024-11-03', '2024-11-08', 'Therapy'),
```

(3, 3, '2024-11-10', '2024-11-15', 'Accident'),

(4, 4, '2024-11-12', '2024-11-19', 'Routine Checkup'),

(5, 5, '2024-12-01', '2024-12-08', 'Infection'),

(6, 1, '2024-12-01', NULL, 'Surgery'),

(7, 2, '2024-12-02', '2024-12-05', 'Fracture Repair'),

(8, 3, '2024-12-03', NULL, 'Chronic Illness'),

(9, 4, '2024-12-03', '2024-12-18', 'Therapy'),

(10, 5, '2024-12-04', '2024-12-18', 'Infection');

-- Insert data into Treatments table

INSERT INTO Treatments (AdmissionID, ProcedureName, Cost, Outcome) VALUES

(1, 'Appendectomy', 1500.00, 'Successful'),

(2, 'Physical Therapy', 800.00, 'Ongoing'),

(3, 'Fracture Repair', 3000.00, 'Successful'),

(4, 'Blood Test', 200.00, 'Pending'),

(5, 'Antibiotics', 500.00, 'Improved'),

(6, 'Gallbladder Surgery', 4000.00, 'Successful'),

(7, 'X-Ray', 300.00, 'Successful'),

(8, 'Chemotherapy', 5000.00, 'Ongoing'),

(9, 'MRI Scan', 1200.00, 'Pending'),

(10, 'Diabetes Treatment', 700.00, 'Improved');

Healthcare Analytics Queries:

Patient Demographics: Retrieve the number of patients grouped by gender and calculate the average age of patients.

```
SELECT COUNT(PatientID) AS NumberofPatients, Gender, AVG(AGE) AS AverageAge

FROM PATIENTS

GROUP BY Gender;
```

Hospital Utilization: Identify hospitals with the highest number of admissions.

```
SELECT HospitalID, count(AdmissionID) AS NumberofAdmission

FROM ADMISSIONS

GROUP BY HospitalID;
```

Treatment Costs: Calculate the total cost of treatments provided at each hospital.

```
SELECT admissions.hospitalid,  sum(treatments.cost) AS totalCost

FROM TREATMENTS

join admissions

on admissions.admissionid = treatments.admissionid

GROUP BY admissions.HospitalID;
```

Length of Stay Analysis: Extract the average length of stay for patients grouped by hospital.

```
SELECT hospitalid, avg(datediff(dischargedate,admissiondate)) as avgLengthofStay

FROM admissions

GROUP BY hospitalid;
```

Advanced Filtering:

```
# List all patients who stayed longer than 7 days in any hospital.
SELECT hospitalid, datediff(dischargedate, admissiondate) as LengthofStay
FROM admissions
WHERE datediff(dischargedate, admissiondate) > 7;

# Identify treatments that have been performed more than 5 times across all hospitals
SELECT admissions.hospitalid, count(treatments.procedurename) as NoofProcedures
FROM treatments
JOIN admissions
ON treatments.admissionid = admissions.admissionid
GROUP BY admissions.hospitalid
HAVING count(treatments.procedurename) > 5;

# Combining Data:

# Combineadmission and treatment data to display complete patient histories.
SELECT *
FROM admissions
JOIN treatments
ON admissions.admissionid = treatments.admissionid;

# Combinelists of patients admitted for different reasons (e.g., surgery and therapy)
SELECT admissions.reasonforadmission , patients.fullname, patients.patientid
FROM admissions
```

```
JOIN patients

ON admissions.patientid = patients.patientid; # group by admissions.reasonforadmission


# Subqueries and Views:


# Useasubquery to find the hospital with the highest average treatment cost.

SELECT HOSPITALS.HOSPITALID, HOSPITALS.HOSPITALNAME, HOSPITALS.LOCATION,
COST AS AVERAGETREATMENTCOST

FROM ADMISSIONS

JOIN HOSPITALS

ON ADMISSIONS.HOSPITALID = HOSPITALS.HOSPITALID

JOIN TREATMENTS

ON ADMISSIONS.ADMISSIONID = TREATMENTS.ADMISSIONID

WHERE COST = (

                    SELECT AVG(COST) AS AVERAGECOST

                    FROM TREATMENTS

                GROUP BY ADMISSIONID

                    ORDER BY AVERAGECOST DESC LIMIT 1

                     );


# Create a view named HospitalPerformance to display the total number of admissions, average
length of stay, and total revenue generated for each hospital.


CREATE VIEW HOSPITALPERFORMANCE AS
```

```
SELECT HOSPITALS.HOSPITALID,
                        HOSPITALS.HOSPITALNAME,
                        HOSPITALS.LOCATION,
                        COUNT(ADMISSIONS.ADMISSIONID) AS NUMOFADMISSIONS,
                        AVG(DATEDIFF(DISCHARGEDATE, ADMISSIONDATE)) AS
AVELENGTHOFSTAY,
SUM(COST) AS TOTALREVENUE
FROM HOSPITALS
JOIN ADMISSIONS
        ON HOSPITALS.HOSPITALID =  ADMISSIONS.HOSPITALID
JOIN TREATMENTS
        ON TREATMENTS.ADMISSIONID = ADMISSIONS.ADMISSIONID
GROUP BY HOSPITALS.HOSPITALID;


DROP VIEW HOSPITALPERFORMANCE;
SELECT * FROM HOSPITALPERFORMANCE


# Window Functions:


# Use the RANK function to rank hospitals based on their total revenue.
SELECT HOSPITALNAME, TOTALREVENUE,
        RANK() OVER (ORDER BY TOTALREVENUE) AS RANKBYTOTALREVENUE
FROM HOSPITALPERFORMANCE


# Use DENSE_RANK to rank treatments based on their frequency
```

```
SELECT TREATMENTID, PROCEDURENAME, COUNT(PROCEDURENAME) AS
COUNTOFPROCEDURES,

                    DENSE_RANK() OVER (ORDER BY COUNT(PROCEDURENAME))
RANKBYFREQUENCYOFTREATMENTS

FROM TREATMENTS

GROUP BY TREATMENTID
```

7. Key Insights

- Certain hospitals generate significantly higher revenue due to treatment complexity and patient volume
- Longer patient stays directly increase operational costs and reduce room availability
- A small subset of treatments contributes disproportionately to total revenue
- Hospital performance varies widely, indicating opportunities for efficiency improvement

8. Business Impact

- Enables hospital administrators to identify high-cost operations
- Supports better capacity and resource planning
- Helps optimize treatment pricing and operational workflows
- Provides a foundation for healthcare dashboards and reporting systems

9. Conclusion

This project demonstrates how **SQL-driven healthcare analytics** can transform raw hospital data into actionable insights. By combining relational database design, business queries, views, and window functions, the analysis supports informed decision-making for hospital operations, financial planning, and patient care optimization.

The project showcases strong SQL fundamentals along with real-world analytical thinking applicable to healthcare, business intelligence, and data analyst roles.