**SDG Selection and Problem Definition**

For this project, I choose **SDG 3: Good Health**. Specifically, I will focus on the problem of **tracking and analyzing hospital patient data to improve healthcare outcomes**.

**Problem Definition**

The problem I aim to address is the lack of efficient data management systems in hospitals, leading to difficulties in tracking patient information, medical histories, and treatment outcomes. This can result in delayed diagnoses, inappropriate treatments, and poor patient care. By designing a database to store and analyze patient data, hospitals can improve healthcare outcomes, reduce costs, and enhance patient satisfaction.

**Database Design**

**ERD (Entity-Relationship Diagram)**

The ERD for this project includes the following entities:

* **Patients**
  + Patient ID (primary key)
  + Name
  + Date of Birth
  + Contact Information
* **Doctors**
  + Doctor ID (primary key)
  + Name
  + Specialty
  + Contact Information
* **Appointments**
  + Appointment ID (primary key)
  + Patient ID (foreign key)
  + Doctor ID (foreign key)
  + Appointment Date and Time
  + Diagnosis
  + Treatment
* **Medications**
  + Medication ID (primary key)
  + Medication Name
  + Dosage
  + Frequency
* **Prescriptions**
  + Prescription ID (primary key)
  + Patient ID (foreign key)
  + Doctor ID (foreign key)
  + Medication ID (foreign key)
  + Prescription Date
  + Expiration Date

**Schema**

Here is the SQL code to create the database schema:

sql

VerifyOpen In EditorEditCopy code

1CREATE TABLE Patients (

2 PatientID INT PRIMARY KEY,

3 Name VARCHAR(255) NOT NULL,

4 DateOfBirth DATE NOT NULL,

5 ContactInformation VARCHAR(255) NOT NULL

6);

7

8CREATE TABLE Doctors (

9 DoctorID INT PRIMARY KEY,

10 Name VARCHAR(255) NOT NULL,

11 Specialty VARCHAR(255) NOT NULL,

12 ContactInformation VARCHAR(255) NOT NULL

13);

14

15CREATE TABLE Appointments (

16 AppointmentID INT PRIMARY KEY,

17 PatientID INT NOT NULL,

18 DoctorID INT NOT NULL,

19 AppointmentDateTime DATETIME NOT NULL,

20 Diagnosis VARCHAR(255) NOT NULL,

21 Treatment VARCHAR(255) NOT NULL,

22 FOREIGN KEY (PatientID) REFERENCES Patients(PatientID),

23 FOREIGN KEY (DoctorID) REFERENCES Doctors(DoctorID)

24);

25

26CREATE TABLE Medications (

27 MedicationID INT PRIMARY KEY,

28 MedicationName VARCHAR(255) NOT NULL,

29 Dosage VARCHAR(255) NOT NULL,

30 Frequency VARCHAR(255) NOT NULL

31);

32

33CREATE TABLE Prescriptions (

34 PrescriptionID INT PRIMARY KEY,

35 PatientID INT NOT NULL,

36 DoctorID INT NOT NULL,

37 MedicationID INT NOT NULL,

38 PrescriptionDate DATE NOT NULL,

39 ExpirationDate DATE NOT NULL,

40 FOREIGN KEY (PatientID) REFERENCES Patients(PatientID),

41 FOREIGN KEY (DoctorID) REFERENCES Doctors(DoctorID),

42 FOREIGN KEY (MedicationID) REFERENCES Medications(MedicationID)

43);

**Sample Data**

Here is some sample data to populate the database:

sql

VerifyOpen In EditorEditCopy code

1INSERT INTO Patients (PatientID, Name, DateOfBirth, ContactInformation)

2VALUES

3 (1, 'John Doe', '1990-01-01', 'john.doe@example.com'),

4 (2, 'Jane Smith', '1995-06-01', 'jane.smith@example.com'),

5 (3, 'Bob Johnson', '1980-03-01', 'bob.johnson@example.com');

6

7INSERT INTO Doctors (DoctorID, Name, Specialty, ContactInformation)

8VALUES

9 (1, 'Dr. Maria Rodriguez', 'Cardiology', 'maria.rodriguez@example.com'),

10 (2, 'Dr. John Taylor', 'Oncology', 'john.taylor@example.com'),

11 (3, 'Dr. Emily Chen', 'Pediatrics', 'emily.chen@example.com');

12

13INSERT INTO Appointments (AppointmentID, PatientID, DoctorID, AppointmentDateTime, Diagnosis, Treatment)

14VALUES

15 (1, 1, 1, '2022-01-01 10:00:00', 'Hypertension', 'Medication and lifestyle changes'),

16 (2, 2, 2, '2022-01-15 14:00:00', 'Breast cancer', 'Surgery and chemotherapy'),

17 (3, 3, 3, '2022-02-01 11:00:00', 'Flu', 'Rest and medication');

18

19INSERT INTO Medications (MedicationID, MedicationName, Dosage, Frequency)

20VALUES

21 (1, 'Lisinopril', '10mg', 'Once daily'),

22 (2, 'Metform

Part 3: SQL Programming Data Retrieval: Write SQL queries to retrieve relevant data based on your problem definition. Data Analysis: Write SQL queries to analyze data and generate insights related to your SDG problem. Part 4: Data Analysis Using Excel Import Data: Import data from your database into Excel. Analysis: Analyze the data using pivot tables, charts, and other Excel tools. Dashboard: Create an interactive Excel dashboard to visualize key insights.

**Part 3: SQL Programming**

**Data Retrieval**

To retrieve relevant data based on the problem definition, we can write SQL queries that extract information about patients, their appointments, and prescribed medications. Here are some example queries:

1. **Retrieve all patients and their appointments:**

sql

VerifyOpen In EditorEditCopy code

1SELECT

2 p.Name AS PatientName,

3 a.AppointmentDateTime,

4 d.Name AS DoctorName,

5 a.Diagnosis,

6 a.Treatment

7FROM

8 Patients p

9JOIN

10 Appointments a ON p.PatientID = a.PatientID

11JOIN

12 Doctors d ON a.DoctorID = d.DoctorID;

1. **Retrieve all medications prescribed to a specific patient:**

sql

VerifyOpen In EditorEditCopy code

1SELECT

2 p.Name AS PatientName,

3 m.MedicationName,

4 pr.PrescriptionDate,

5 pr.ExpirationDate

6FROM

7 Prescriptions pr

8JOIN

9 Patients p ON pr.PatientID = p.PatientID

10JOIN

11 Medications m ON pr.MedicationID = m.MedicationID

12WHERE

13 p.PatientID = 1; -- Change the ID as needed

1. **Count the number of appointments per doctor:**

sql

VerifyOpen In EditorEditCopy code

1SELECT

2 d.Name AS DoctorName,

3 COUNT(a.AppointmentID) AS AppointmentCount

4FROM

5 Doctors d

6LEFT JOIN

7 Appointments a ON d.DoctorID = a.DoctorID

8GROUP BY

9 d.Name;

**Data Analysis**

To analyze the data and generate insights related to the SDG problem, we can use SQL queries that aggregate and summarize the data:

1. **Find the most common diagnosis among patients:**

sql

VerifyOpen In EditorEditCopy code

1SELECT

2 Diagnosis,

3 COUNT(\*) AS DiagnosisCount

4FROM

5 Appointments

6GROUP BY

7 Diagnosis

8ORDER BY

9 DiagnosisCount DESC;

1. **Analyze the average number of appointments per patient:**

sql

VerifyOpen In EditorEditCopy code

1SELECT

2 p.Name AS PatientName,

3 COUNT(a.AppointmentID) AS AppointmentCount

4FROM

5 Patients p

6LEFT JOIN

7 Appointments a ON p.PatientID = a.PatientID

8GROUP BY

9 p.Name;

1. **Identify the most frequently prescribed medication:**

sql

VerifyOpen In EditorEditCopy code

1SELECT

2 m.MedicationName,

3 COUNT(pr.PrescriptionID) AS PrescriptionCount

4FROM

5 Prescriptions pr

6JOIN

7 Medications m ON pr.MedicationID = m.MedicationID

8GROUP BY

9 m.MedicationName

10ORDER BY

11 PrescriptionCount DESC;

**Part 4: Data Analysis Using Excel**

**Import Data**

To import data from your database into Excel, you can follow these steps:

1. **Open Excel** and go to the **Data** tab.
2. Click on **Get Data** > **From Database** > **From SQL Server Database** (or the appropriate option for your database).
3. Enter your server and database information, and select the tables you want to import (e.g., Patients, Appointments, Medications, Prescriptions).

**Analysis**

Once the data is imported, you can analyze it using Excel tools:

1. **Pivot Tables**:
   * Create a pivot table to summarize data, such as the number of appointments per doctor or the most common diagnoses.
   * Drag fields into the Rows and Values areas to create meaningful summaries.
2. **Charts**:
   * Use charts to visualize data trends, such as the number of appointments over time or the distribution of diagnoses.
   * Select your data and choose **Insert** > **Charts** to create various types of charts (e.g., bar, line, pie).

**Dashboard**

To create an interactive Excel dashboard:

1. **Design the Layout**:
   * Plan the layout of your dashboard, including sections for key metrics, charts, and tables.
2. **Use Slicers**:
   * Add slicers to your pivot tables to allow users to filter data interactively (e.g., by doctor or diagnosis).
   * Select your pivot table, go to the **PivotTable Analyze** tab, and click on **Insert Slicer**.
3. **Combine Elements**:
   * Combine charts, tables, and slicers into a cohesive dashboard layout.
   * Format the dashboard for clarity and visual appeal.