**Healthcare Management System Project Assignment**

**Project Overview**

The goal of this project is to develop a Healthcare Management System (HMS) that utilizes a master ERP table for managing patient information.

**Objectives**

1. **Data Cleaning and Standardization:** Identify and correct data quality issues in a provided master dataset.
2. **Database Design:** Design an appropriate relational database schema based on the master table.
3. **SQL Programming:** Write SQL scripts to create the database tables and populate them with cleaned data.
4. **Advanced SQL Techniques:** Implement views, stored procedures, and triggers to enhance database functionality.
5. **Data Analysis:** Perform queries to extract meaningful insights from the data.

**Project Structure**

**1. ERP Master Patient Table Structure**

This table serves as the source of patient data containing a wide array of fields related to patient information, including personal details, medical history, and contact information.

 **Execute the SQL File**:  
Load the provided SQL file to create and populate the table in your database.

 **Data Cleaning**:  
Identify and address the issues in columns like city, state and so on

* **State and City Columns**
* **Phone Column**:  
  Standardize phone numbers to a uniform format
* **Duplicate Rows**:  
  Handle duplicate entries based on a unique combination of key fields (e.g., patient ID, name, date of birth).
* **Missing or Null Values**:  
  Handle null values appropriately:
  + For critical fields, attempt to infer missing data if possible.
  + Otherwise, replace with placeholders (e.g., "Unknown" or "N/A") or remove rows if data is insufficient.

**2. Tables to Create**

The following tables should be created to normalize the data from the ERP\_Master\_Patient table:

* **Patients Table:**
  + patient\_id
  + first\_name
  + last\_name
  + dob
  + gender
  + address
  + city
  + state
  + postal\_code
  + phone\_number
  + email
  + insurance\_provider
  + insurance\_policy\_number
  + blood\_type
  + allergies
  + medications
  + diagnosis
  + admission\_date
  + discharge\_date
  + emergency\_contact\_name
  + emergency\_contact\_phone
  + emergency\_contact\_relationship
  + insurance\_expiration\_date
  + blood\_pressure
  + heart\_rate
  + weight
  + height
  + temperature
* **Doctors Table:**
  + doctor\_id
  + first\_name
  + last\_name
  + specialty
  + phone\_number
  + email
* **Nurses Table:**
  + nurse\_id
  + first\_name
  + last\_name
  + phone\_number
  + email
* **Billing Table:**
  + billing\_id
  + patient\_id
  + total\_amount
  + amount\_paid
  + billing\_date
  + due\_date
  + payment\_status
  + insurance\_coverage
* **Insurance Table:**
  + insurance\_id
  + insurance\_provider
  + policy\_number
  + coverage\_start\_date
  + coverage\_end\_date
  + patient\_id
* **Rooms Table:**
  + room\_number
  + room\_type
  + capacity
  + current\_occupancy
  + is\_available

**3. Stored Procedures**

Students will create stored procedures to manage data within the database, including:

* **Load Data into Patients Table:** A stored procedure that populates the Patients table from the ERP\_Master\_Patient table, ensuring that duplicate records are not inserted.
* **Update Patient Information:** A stored procedure that allows updating patient details based on their unique patient ID.
* **Manage Billing Information:** A stored procedure to insert or update billing records, ensuring that related patient data is validated before making changes to the billing table.
* **Allocate Room to Patient:** A stored procedure that assigns a room to a patient upon admission, checking for available rooms and updating the Rooms table to reflect the current occupancy.
* **Record Patient Discharge:** A stored procedure that updates the patient’s discharge date and makes the room available again.
* **Process Payment:** A stored procedure to update billing records when a payment is made, checking that the payment does not exceed the total amount due.

**4. Views**

Students will create the following views to simplify data retrieval and facilitate patient, doctor, and room allocation:

* **View for Patient Summary:** A view that aggregates essential patient information for quick access.
* **View for Doctor Assignments:** A view that shows how many patients each doctor is responsible for, facilitating better understanding of patient loads.
* **View for Billing Summary:** A view that combines patient details with their billing information to provide a comprehensive overview of outstanding payments and billing history.
* **View for Room Availability:** A view that displays all rooms along with their current occupancy status, helping staff quickly find available rooms.
* **View for Patient Room Allocation:** A view that links patients with their assigned rooms, showing which patients are currently occupying which rooms and their admission dates.
* **View for Patient-Doctor Assignments:** A view that shows the relationship between patients and their assigned doctors, including the doctor’s specialty for quick reference.

**5. Data Integrity and Referential Integrity**

Data integrity is crucial in any database design, particularly in healthcare, where accuracy is vital. The following measures should be implemented:

* **Primary Keys:** Ensure that every table has a primary key to uniquely identify records.
* **Foreign Keys:** Use foreign keys to enforce referential integrity between related tables.
* **Check Constraints:** Implement check constraints to enforce valid values in specific columns.
* **Cascading Updates and Deletes:** Consider whether to implement cascading updates or deletes to maintain referential integrity when records are modified or removed.

**6. Additional Assignments**

* **Implement Window Functions:** Create a query that uses window functions to calculate the average length of stay for patients.
* **Implementing Triggers:** Design a trigger that logs updates made to patient records into an audit table to track changes.
* **Recursive Common Table Expressions (CTEs):** Use recursive CTEs to explore hierarchical data, such as family relationships based on emergency contacts stored in the Patients table.

**7. Deliverables**

* SQL scripts for data cleaning, table creation, and data manipulation.
* SQL scripts for the stored procedures, views, functions, and triggers.
* Screenshots or outputs from queries demonstrating the analysis performed.
* A final report summarizing the project, findings, and any challenges faced.

• Patients Table:

o patient\_id

o first\_name

o last\_name

o dob

o gender

o address

o city

o state

o postal\_code

o phone\_number

o email

o insurance\_provider

o insurance\_policy\_number

o blood\_type

o allergies

o medications

o diagnosis

o admission\_date

o discharge\_date

o emergency\_contact\_name

o emergency\_contact\_phone

o emergency\_contact\_relationship

o insurance\_expiration\_date

o blood\_pressure

o heart\_rate

o weight

o height

o temperature

• Doctors Table:

o doctor\_id

o first\_name

o last\_name

o specialty

o phone\_number

o email

• Nurses Table:

o nurse\_id

o first\_name

o last\_name

o phone\_number

o email

• Billing Table:

o billing\_id

o patient\_id

o total\_amount

o amount\_paid

o billing\_date

o due\_date

o payment\_status

o insurance\_coverage

• Insurance Table:

o insurance\_id

o insurance\_provider

o policy\_number

o coverage\_start\_date

o coverage\_end\_date

o patient\_id

• Rooms Table:

o room\_number

o room\_type

o capacity

o current\_occupancy

o is\_available

patient\_id. Data integrity is crucial in any database design, particularly in healthcare, where accuracy is vital. The following measures should be implemented:

• Primary Keys: Ensure that every table has a primary key to uniquely identify records.

• Foreign Keys: Use foreign keys to enforce referential integrity between related tables.

• Check Constraints: Implement check constraints to enforce valid values in specific columns.

• Cascading Updates and Deletes: Consider whether to implement cascading updates or deletes to maintain referential integrity when records are modified or removed.