**MSIS 618 – DATABASE MANAGEMENT**

**“HEALTHFLOW DATABASE SYSTEM”**

**Prepared by**

Aryan Jain

**Under the guidance of**

Prof. Kui Du

**Due Date:**

20th December 2023

A blue and white logo

Description automatically generated

Table of Contents

**Executive Summary 2**

**Project Description2,3**

Database Design: ER Diagram3,4,5

Normalization6,7

**Database Implementation……………………………………………………………………………**……………………………………………………………7,8

**Executive Summary**

In the dynamic landscape of healthcare, the efficient management of clinical data is paramount for delivering optimal patient care. Introducing the Health Flow System, a robust clinical database management project designed to revolutionize the handling of patient information, appointments, and treatment records across diverse healthcare institutions.

Project Objectives:

The core objectives of the Health Flow System encompass streamlining patient information management, enhancing the efficiency of appointment scheduling, and establishing a centralized repository for treatment records. This project addresses the universal challenges faced by healthcare organizations, including manual data handling, scheduling conflicts, and inaccuracies in clinical data.

Key Functionalities:

1. **Patient Information Management:**
   * Centralized storage and retrieval of patient records.
   * Dynamic updating of patient information for accurate medical histories.
2. **Appointment Scheduling:**
   * Seamless and efficient scheduling of patient appointments.
   * Automated reminders to reduce no-show rates and enhance patient engagement.
3. **Treatment Records:**
   * Comprehensive recording of treatment details, medications, and progress.
   * Integration with diagnostic systems for a holistic view of patient health.

Potential Impact:

The Health Flow System is poised to make a significant impact on healthcare workflows across diverse institutions. By introducing a cohesive and integrated approach to clinical data management, this system is expected to lead to improved patient outcomes, enhanced operational efficiency, and increased satisfaction among healthcare practitioners and administrative staff.

This executive summary provides a high-level overview of the Health Flow System project, highlighting its potential to transform the way clinical data is managed across a spectrum of healthcare institutions.

**Project Description**

Healthcare Environment:

The Health Flow System is designed to address the complex challenges faced by healthcare institutions in managing clinical data efficiently. Its application extends to a diverse range of healthcare environments, including hospitals, clinics, and medical practices. This project aims to create a versatile solution that can be tailored to meet the unique needs of healthcare organizations of varying scales.

Business Activity:

The focal point of the Health Flow System is the comprehensive management of clinical data to enhance the overall healthcare experience. The key business activities targeted by this system include:

1. **Patient Information Management**:
   * Centralizing and organizing patient records for quick and accurate access.
   * Enabling dynamic updates to patient information to maintain current and precise medical histories.
2. **Appointment Scheduling**:
   * Streamlining the scheduling process for patient appointments.
   * Implementing automated reminders to reduce no-show rates and improve overall appointment adherence.
3. **Treatment Records:**
   * Establishing a centralized repository for treatment records, including details on medications, procedures, and progress.
   * Integrating with diagnostic systems to provide healthcare practitioners with a holistic view of patient health.

Business Requirements:

The Health Flow System is tailored to meet the following business requirements within the healthcare environment:

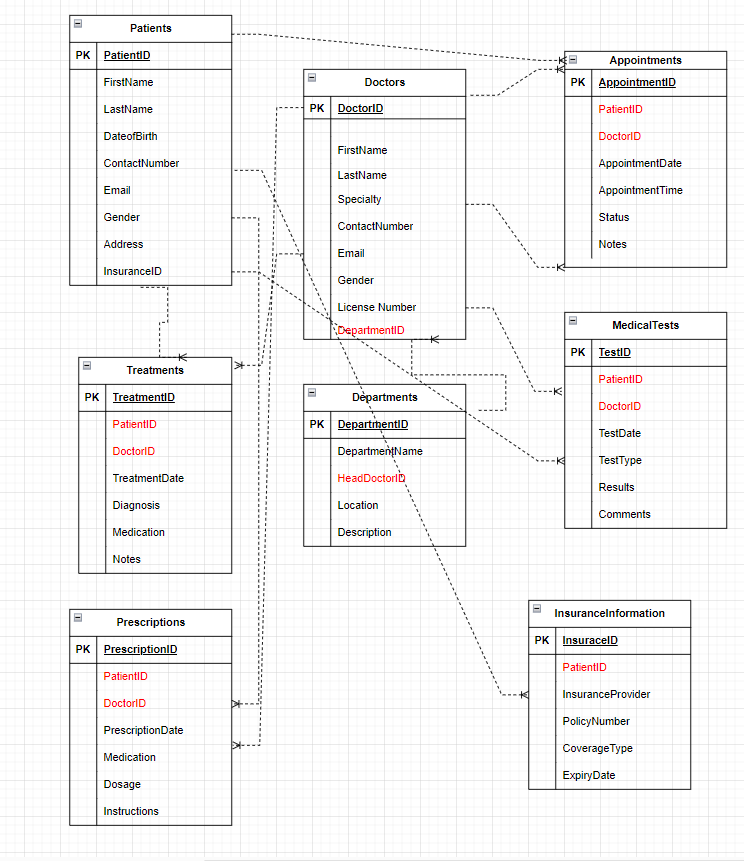
* Efficient Data Handling: Replace manual data handling processes with a streamlined and automated system.
* Appointment Efficiency: Enhance the efficiency of appointment scheduling and reduce scheduling conflicts.
* Data Accuracy: Ensure the accuracy and currency of patient information and treatment records.
* Holistic Patient View: Provide healthcare practitioners with a comprehensive view of patient health by integrating treatment and diagnostic data.
* User Accessibility: Design an intuitive system accessible to healthcare practitioners, administrative staff, and potentially patients for certain functionalities.

**Database Design: ER Diagram**

1. **Patients Entity:**
   * Attributes: PatientID (Primary Key), FirstName, LastName, DateOfBirth, ContactNumber, Email, Gender, Address, InsuranceID (Foreign Key, Highlighted in Red).
2. **Doctors Entity:**
   * Attributes: DoctorID (Primary Key), FirstName, LastName, Specialty, ContactNumber, Email, Gender, LicenseNumber, DepartmentID (Foreign Key, Highlighted in Red).
3. **Appointments Entity:**
   * Attributes: AppointmentID (Primary Key), PatientID (Foreign Key, Highlighted in Red), DoctorID (Foreign Key, Highlighted in Red), AppointmentDate, AppointmentTime, Status, Notes.
4. **Treatments Entity:**
   * Attributes: TreatmentID (Primary Key), PatientID (Foreign Key, Highlighted in Red), DoctorID (Foreign Key, Highlighted in Red), TreatmentDate, Diagnosis, Medication, Notes.
5. **Departments Entity:**
   * Attributes: DepartmentID (Primary Key), DepartmentName, HeadDoctorID (Foreign Key, Highlighted in Red), Location, Description.
6. **MedicalTests Entity:**
   * Attributes: TestID (Primary Key), PatientID (Foreign Key, Highlighted in Red), DoctorID (Foreign Key, Highlighted in Red), TestDate, TestType, Results, Comments.
7. **Prescriptions Entity:**
   * Attributes: PrescriptionID (Primary Key), PatientID (Foreign Key, Highlighted in Red), DoctorID (Foreign Key, Highlighted in Red), PrescriptionDate, Medication, Dosage, Instructions.
8. **InsuranceInformation Entity:**
   * Attributes: InsuranceID (Primary Key), PatientID (Foreign Key, Highlighted in Red), InsuranceProvider, PolicyNumber, CoverageType, ExpiryDate.

**Data Types, Constraints and Relationships:**

1. Patients Table:
   * Data Types: PatientID (INT), FirstName (VARCHAR), LastName (VARCHAR), DateOfBirth (DATE), ContactNumber (VARCHAR), Email (VARCHAR), Gender (CHAR), Address (TEXT), InsuranceID (INT)
   * Constraints: PatientID (Primary Key).
   * Relationships: None
2. Doctors Table:
   * Data Types: DoctorID (INT), FirstName (VARCHAR), LastName (VARCHAR), Specialty (VARCHAR), ContactNumber (VARCHAR), Email (VARCHAR), Gender (CHAR), LicenseNumber (VARCHAR), DepartmentID (INT)
   * Constraints: DoctorID (Primary Key), DepartmentID (Foreign Key), Other attributes with appropriate constraints.
   * Relationships: One-to-Many with Appointments (DoctorID to DoctorID)
3. Appointments Table:
   * Data Types: AppointmentID (INT), PatientID (INT, Foreign Key), DoctorID (INT, Foreign Key), AppointmentDate (DATE), AppointmentTime (TIME), Status (VARCHAR), Notes (TEXT)
   * Constraints: AppointmentID (Primary Key), PatientID (Foreign Key), DoctorID (Foreign Key).
   * Relationships: Many-to-One with Patients (PatientID to PatientID), Many-to-One with Doctors (DoctorID to DoctorID)
4. Treatments Table:
   * Data Types: TreatmentID (INT), PatientID (INT, Foreign Key), DoctorID (INT, Foreign Key), TreatmentDate (DATE), Diagnosis (TEXT), Medication (TEXT), Notes (TEXT)
   * Constraints: TreatmentID (Primary Key), PatientID (Foreign Key), DoctorID (Foreign Key).
   * Relationships: Many-to-One with Patients (PatientID to PatientID), Many-to-One with Doctors (DoctorID to DoctorID)
5. Departments Table:
   * Data Types: DepartmentID (INT), DepartmentName (VARCHAR), HeadDoctorID (INT, Foreign Key), Location (VARCHAR), Description (TEXT)
   * Constraints: DepartmentID (Primary Key), HeadDoctorID (Foreign Key).
   * Relationships: One-to-Many with Doctors (DepartmentID to DepartmentID)
6. MedicalTests Table:
   * Data Types: TestID (INT), PatientID (INT, Foreign Key), DoctorID (INT, Foreign Key), TestDate (DATE), TestType (VARCHAR), Results (TEXT), Comments (TEXT)
   * Constraints: TestID (Primary Key), PatientID (Foreign Key), DoctorID (Foreign Key).
   * Relationships: Many-to-One with Patients (PatientID to PatientID), Many-to-One with Doctors (DoctorID to DoctorID)
7. Prescriptions Table:
   * Data Types: PrescriptionID (INT), PatientID (INT, Foreign Key), DoctorID (INT, Foreign Key), PrescriptionDate (DATE), Medication (TEXT), Dosage (VARCHAR), Instructions (TEXT)
   * Constraints: PrescriptionID (Primary Key), PatientID (Foreign Key), DoctorID (Foreign Key).
   * Relationships: Many-to-One with Patients (PatientID to PatientID), Many-to-One with Doctors (DoctorID to DoctorID)
8. InsuranceInformation Table:
   * Data Types: InsuranceID (INT), PatientID (INT, Foreign Key), InsuranceProvider (VARCHAR), PolicyNumber (VARCHAR), CoverageType (VARCHAR), ExpiryDate (DATE)
   * Constraints: InsuranceID (Primary Key), PatientID (Foreign Key).
   * Relationships: Many-to-One with Patients (PatientID to PatientID)



**Normalization**

1. Patients Table:
   * Attributes: PatientID (PK), FirstName, LastName, DateOfBirth, ContactNumber, Email, Gender, Address, InsuranceID (FK)
   * Functional Dependencies: PatientID → FirstName, LastName, DateOfBirth, ContactNumber, Email, Gender, Address, InsuranceID
   * BCNF: Already in BCNF
2. Doctors Table:
   * Attributes: DoctorID (PK), FirstName, LastName, Specialty, ContactNumber, Email, Gender, LicenseNumber, DepartmentID (FK)
   * Functional Dependencies: DoctorID → FirstName, LastName, Specialty, ContactNumber, Email, Gender, LicenseNumber, DepartmentID
   * BCNF: Already in BCNF
3. Appointments Table:
   * Attributes: AppointmentID (PK), PatientID (FK), DoctorID (FK), AppointmentDate, AppointmentTime, Status, Notes
   * Functional Dependencies: AppointmentID → PatientID, DoctorID, AppointmentDate, AppointmentTime, Status, Notes
   * BCNF: Already in BCNF
4. Treatments Table:
   * Attributes: TreatmentID (PK), PatientID (FK), DoctorID (FK), TreatmentDate, Diagnosis, Medication, Notes
   * Functional Dependencies: TreatmentID → PatientID, DoctorID, TreatmentDate, Diagnosis, Medication, Notes
   * BCNF: Already in BCNF
5. Departments Table:
   * Attributes: DepartmentID (PK), DepartmentName, HeadDoctorID (FK), Location, Description
   * Functional Dependencies: DepartmentID → DepartmentName, HeadDoctorID, Location, Description
   * BCNF: Already in BCNF
6. MedicalTests Table:
   * Attributes: TestID (PK), PatientID (FK), DoctorID (FK), TestDate, TestType, Results, Comments
   * Functional Dependencies: TestID → PatientID, DoctorID, TestDate, TestType, Results, Comments
   * BCNF: Already in BCNF
7. Prescriptions Table:
   * Attributes: PrescriptionID (PK), PatientID (FK), DoctorID (FK), PrescriptionDate, Medication, Dosage, Instructions
   * Functional Dependencies: PrescriptionID → PatientID, DoctorID, PrescriptionDate, Medication, Dosage, Instructions
   * BCNF: Already in BCNF
8. InsuranceInformation Table:
   * Attributes: InsuranceID (PK), PatientID (FK), InsuranceProvider, PolicyNumber, CoverageType, ExpiryDate
   * Functional Dependencies: InsuranceID → PatientID, InsuranceProvider, PolicyNumber, CoverageType, ExpiryDate
   * BCNF: Already in BCNF

**Database Implementation**

**SQL DDL Statements**

I successfully implemented the database in MySQL using SQL Data Definition Language (DDL) statements. The DDL statements were carefully crafted to create tables reflecting the entities identified in the Entity-Relationship Diagram (ERD). Primary keys, foreign keys, and necessary constraints were defined to ensure the integrity of the data. The database structure was defined through precise SQL statements. Each table captures the relationships and attributes outlined in the ERD.

**Data Population**

Following table creation, the database was populated with sample data to validate its functionality. SQL statements were employed to insert records into each table, covering a diverse range of scenarios. Sample data was added to simulate real-world scenarios and ensure that the tables and relationships function as expected. This step involved populating tables with records reflecting various use cases.

This process has resulted in a functional database structure, and the inclusion of sample data facilitates thorough testing and validation.

**Queries**

**Query 1**: Retrieve a patient's medical history.

**Description**: This query retrieves a comprehensive medical history for a patient, including details of treatments, tests, and prescriptions.

**Value to Business Users:** This query enables healthcare providers to access a patient's complete medical history in one view. It aids in making informed decisions about ongoing and past treatments, tests, and prescribed medications.

**Query 2:** Get detailed information about patients along with their upcoming appointments.

**Description:** This query provides detailed information about patients, including their date of birth, along with upcoming appointments.

**Value to Business Users:** This query assists in managing patient appointments efficiently, allowing healthcare providers to plan and allocate resources effectively.

**Query 3:** Identify patients with expired insurance policies.

**Description:** This query identifies patients with insurance policies that have expired.

**Value to Business Users:** This query helps in ensuring that patients have valid insurance coverage, facilitating proper billing and financial management.

**Query 4:** Identify patients awaiting medical test results.

**Description:** This query identifies patients who are waiting for the results of medical tests.

**Value to Business Users:** This query aids healthcare providers in prioritizing and expediting the communication of test results to patients.

**Query 5:** Identify patients eligible for upcoming medical tests based on their medical history.

**Description:** This query identifies patients who are eligible for upcoming medical tests based on their medical history.

**Value to Business Users:** This query supports proactive healthcare management by identifying patients due for upcoming tests based on their medical history.

**Query 6:** Display a list of doctors along with their respective departments.

**Description:** This query provides a list of doctors along with the departments they are associated with.

**Value to Business Users:** This query facilitates efficient management by displaying the organizational structure of doctors within their respective departments.

**Query 7:** Categorize patients into age groups for demographic analysis.

**Description:** This query categorizes patients into age groups for demographic analysis.

**Value to Business Users:** This query supports demographic analysis, helping healthcare providers understand the age distribution of their patient population.

**Query 8:** Display each doctor's specialty along with the count of patients they are treating.

**Description:** This query shows each doctor's specialty along with the count of patients they are treating.

**Value to Business Users:** This query provides insights into the workload of each doctor, assisting in resource allocation and scheduling.