**Hospital Database Management System**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Team 10** | | | | |
| **Team Members** | Avinash Raikesh | Animesh Shukla | Shruti Manohar | Vandit Kothari |
| **NU Email ID** | [raikesh.a@northeastern.edu](mailto:raikesh.a@northeastern.edu) | [shukla.an@northeastern.edu](mailto:shukla.an@northeastern.edu) | [manohar.s@northeastern.edu](mailto:manohar.s@northeastern.edu) | [kothari.va@northeastern.edu](mailto:kothari.va@northeastern.edu) |

        15

**Table of Contents**

[**1.** **OVERVIEW** 2](#_Toc87465462)

[**2.** **PROBLEM STATEMENT** 3](#_Toc87465463)

[**3.** **OBJECTIVES** 3](#_Toc87465464)

[**4.** **IDEA OF PROJECT** 4](#_Toc87465465)

[**5.** **ENTITY RELATIONSHIP DIAGRAM** 5](#_Toc87465466)

[**6.** **BUSINESS RULES** 6](#_Toc87465467)

[**7.** **TABLES** 7](#_Toc87465468)

[**i.** **ADMINISTRATION Table:** 7](#_Toc87465469)

[**ii.** **DOCTOR Table:** 8](#_Toc87465470)

[**iii.** **PATIENT Table:** 9](#_Toc87465471)

[**iv.** **DIAGNOSIS Table:** 10](#_Toc87465472)

[**v.** **IN\_PATIENT Table:** 11](#_Toc87465473)

[**vi.** **OUT\_PATIENT Table:** 11](#_Toc87465474)

[**vii.** **BILLING Table:** 12](#_Toc87465475)

[**viii.** **ROOM Table:** 12](#_Toc87465476)

[**ix.** **APPOINTMENT Table:** 13](#_Toc87465477)

[**x.** **DEPARTMENT Table:** 13](#_Toc87465478)

[**8.** **SECURITY** 14](#_Toc87465479)

[**1.** **Admin Access:** 14](#_Toc87465480)

[**2.** **Doctor Access:** 14](#_Toc87465481)

[**3.** **Patient Access:** 14](#_Toc87465482)

# **OVERVIEW**

Healthcare is a field that demands efficiency when it comes to processing vast amounts of medical data and effectively treating patients. Within the last twenty years or so, many organizations have had to face the difficult task of transferring medical data from file processing systems into a robust relational database that can provide all parties with access to quick and reliable medical as well as financial information. Throughout the entire process, doctors have always played a fundamental role in assessing and prescribing medication, performing surgeries and monitoring patients. As per 2021, there are approximately 6000 hospitals and most of them use a proper hospital database management system. Yet, not all of those databases have been implemented properly and data security  is another big issue. At the same time, it is very tough to conclude how effective that management system is for the staff of 6000 hospitals.

In accordance with HIPAA privacy standards, medical personnel are protected from disclosing patient information to fellow medical personnel, which means hospitals handle highly sensitive data that requires strong file security. In today's healthcare setting, this is enough to justify why file processing systems are antiquated. There have been many successful transitions, but some have faced numerous obstacles which have led to difficulties for hospitals, medical practitioners, and other hospital staff.

Choosing an RDBMS that is efficient is crucial when it comes to healthcare because workflow congestion can mean life or death for patients. It is critical to manage patient data, to arrange patient-doctor schedules, and to account for financial concerns of the parties in the process. A database management system will accommodate security concerns and update problems adequately if it is set up in a way that is able to take care of these issues.

# **PROBLEM STATEMENT**

A hospital management database system manages all areas of a hospital, including medical, financial, and administrative functions. With its flexibility, power, and easy-to-use properties, it is designed and developed to meet the needs of hospitals, healthcare providers, and patients.

# **OBJECTIVES**

1. To develop an automated system that processes medical information and better patient health care quickly, securely, and efficiently.
2. To store patients' medical records, appointment schedules, staff directories, room assignment information, billing data, and other details.
3. To minimize hospital congestion which will allow them to get treated and cared for faster.
4. To minimize hospital operating costs.
5. To provide top management and better coordination between different departments.

# **IDEA OF PROJECT**

To develop a system that processes medical information quickly, securely, and efficiently, we will be building a relational database. Patients' medical records, appointment schedules, staff directories, room assignment information, billing data, and other details can be stored in the system. This system is easier to use, less time consuming, and more secure than usual file processing programs. In addition to improving healthcare management, the staff-patient relationship would also benefit from enhanced precision by automating the process. With a single secure platform, patients could schedule their appointments, access their billing statements, and manage their medical records. With this approach, you can minimize hospital congestion, which will allow you to get treated and cared for faster. As a result of

Through hospital management systems, doctors can also retrieve/update medical notes more quickly, ensuring the best diagnostic practices are being used.

It will comprise doctors, patients, diagnoses, billing information, and plans for inpatient rooms. Administration will be able to view billing statements, assign patients to rooms based on availability, confirm/modify/delete appointments, and update patient/staff details. A doctor will be able to diagnose patients based on symptoms. Medical information is both confidential and subject to HIPAA laws, so all users will simply not have equal access to it. Patients won't have the option to share their records with others, and neither will doctors.

To make our proposal useful, we are assuming our hospital is still using a file processing system. The data intake process is as follows. A medical form is filled out online by the patient in advance. Patient information and medical history (name, address, height, weight, etc.) would be captured on this form by the admin who would then enter this information into a system that would update if the patient's condition changes. If a patient has previously visited a hospital, this information would be updated in their system; otherwise, it would be created if it's their first visit. Appointments with doctors are scheduled ahead of time, following a phone call from the patient. An administrator will determine their availability (offline) and then schedule the appointment. As soon as an appointment takes place, the doctor provides a proper diagnosis to the patient, which is entered into the medical records, updating that patient's profile. A bill will be issued to the patient following the appointment, containing the service charges. The patient's room is assigned by an administrator after they have been admitted. The patient's status is changed from inpatient to outpatient by an administrator after the patient has been discharged. The bill is administered one day after the patient leaves the hospital.

# **ENTITY RELATIONSHIP DIAGRAM**

When it comes to designing a database, Entity Relationship (ER) diagrams play a big role. Based on the business idea, we have drafted a representation of the ER model containing tables with relations.

Graphical user interface, text

Description automatically generated with medium confidence

# **BUSINESS RULES**

With the help of this database management system, the hospital organizations can successfully handle the data of the doctors, admins, patients, billings, and many more entities. With its proper implementation:

1. **ADMIN** can schedule/cancel **APPOINTMENTs** between **PATIENT** and a **DOCTOR**.
2. A **DOCTOR** will be part of single **DEPARTMENT**.
3. **PATIENTs** will be able to book maximum of 3 **APPOINTMENTs** in a week.
4. **DOCTOR** can see a maximum of 10 **PATIENTs** in a day.
5. **PATIENTs** visiting the hospital for routine check-up are considered as **OUT-PATIENTs.**
6. Through previous **DIAGNOSIS**, **DOCTORs** can fetch **PATIENT's** medical history.
7. **DOCTOR** will provide a **DIAGNOSIS** to **PATIENT** during their **APPOINTMENT** and accordingly ask them to get hospitalized.
8. **DOCTOR and PATIENT** will be able to see their upcoming appointments.
9. **ADMIN** will be able to see department level upcoming appointments.
10. **ADMIN** will assign a "**Vacant**" **ROOM** to a **PATIENT** if they are admitted.
11. Total cost of a hospital visit for a **PATIENT** will be calculated by using **DOCTOR\_FEES, ROOM\_CHARGE, PRESCRIPTION\_CHARGE** and **INSURANCE\_PERCENT**.
12. While getting discharged, **PATIENTs** will be issued a **BILL** which will contain all details.

# **TABLES**

Based on the business, we have created an idea of how many tables this system will consist of. These tables will define the database through column name, data type, constraints, and the description of those entities.

## **ADMINISTRATION Table:**

This table will store the details of the administrative staff of the hospital. While some of the fields are optional, most of them should not be null.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** | **Description** |
| **ADMIN\_ID** | NUMBER(10) | Primary Key | Administrative ID |
| **FIRST\_NAME** | VARCHAR(30) | Not Null | First Name of Administrator |
| **MIDDLE\_NAME** | VARCHAR(30) |  | Middle Name of Administrator |
| **LAST\_NAME** | VARCHAR(30) | Not Null | Last Name of Administrator |
| **PHONE\_NO** | NUMBER(10) | Not Null | Administrator’s contact information |
| **EMAIL\_ID** | VARCHAR(50) | Unique, Not Null |
| **OFFICE\_NO** | NUMBER(3) | Not Null | Administrator’s Office number |

## **DOCTOR Table:**

The doctor table will store all the information of the doctors like name, contact details, years of experience, their departments, and many more.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** | **Description** |
| **DOCTOR\_ID** | NUMBER(10) | Primary Key | Doctor’s ID |
| **FIRST\_NAME** | VARCHAR(30) | Not Null | Doctor’s First Name |
| **MIDDLE\_NAME** | VARCHAR(30) |  | Doctor’s Middle Name |
| **LAST\_NAME** | VARCHAR(30) | Not Null | Doctor’s Last Name |
| **PAGER\_NO** | NUMER(10) | Not Null | Doctor’s contact information |
| **PHONE\_NO** | NUMBER(10) | Not Null |
| **EMAIL\_ID** | VARCHAR(50) | Unique, Not Null |
| **PRACTICE\_START\_YEAR** | NUMBER(4) | Not Null | The year from which doctor started practicing medicine as a licensed physician |
| **FEES** | NUMBER(6,2) | Not Null | Charges for doctor’s appointment/services (USD) |
| **DEPT\_ID** | VARCHAR(10) | Foreign Key (DEPARTMENT – DEPT\_ID),  Not Null | Medical Department under which doctor works |
| **OFFICE\_NO** | NUMBER(3) | Not Null | Doctor’s Office number |

## **PATIENT Table:**

This is going to be one of the most important tables of our project as it will consist all the details of the patients.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** | **Description** |
| **PATIENT\_ID** | NUMBER(12) | Primary Key | Patient’s ID |
| **FIRST\_NAME** | VARCHAR(30) | Not Null | Patient’s First Name |
| **MIDDLE\_NAME** | VARCHAR(30) |  | Patient’s Middle Name |
| **LAST\_NAME** | VARCHAR(30) | Not Null | Patient’s Last Name |
| **DOB** | DATE | Not Null | Patient’s Date of Birth |
| **WEIGHT** | NUMBER(4) |  | Patient’s weight (lbs) |
| **HEIGHT** | NUMBER(4) |  | Patient’s height (cm) |
| **GENDER** | VARCHAR(2) | Not Null | Patient’s gender (M/F/NA) |
| **STREET\_NO** | NUMBER(5) |  | Patient’s Address |
| **STREET\_NAME** | VARCHAR(100) |  |
| **CITY** | VARCHAR(30) |  |
| **STATE** | CHAR(2) |  |
| **ZIP** | NUMBER(5) |  |
| **PHONE\_NO** | NUMBER(10) | Not Null | Patient’s contact Information |
| **EMAIL\_ID** | VARCHAR(50) | Null |
| **PRIM\_CARE\_DOC\_ID** | NUMBER(10) | Foreign Key (DOCTOR – DOCTOR\_ID),  Nullable | Patient’s primary care doctor |

## **DIAGNOSIS Table:**

The diagnosis table will have the data of the patients based on their treatment.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** | **Description** |
| **DIAGNOSIS\_ID** | NUMBER(10) | Primary Key | Patient’s diagnosis ID |
| **APPOINTMENT\_ID** | NUMBER(10) | Foreign Key (APPOINTMENT – APPOINTMENT \_ID),  Not Null | Appointment on which diagnosis is made. |
| **DIAGNOSIS\_DATE** | DATE | Not Null | Date on which patient was diagnosed |
| **DIAGNOSIS\_CATEGORY** | VARCHAR(100) | Not Null | Description of Medical ailments |
| **DIAGNOSIS\_DETAILS** | VARCHAR(2000) | Not Null | Details of diagnosis and recommendation moving forward |

## **IN\_PATIENT Table:**

The In\_patient table will only store the details of those who are currently admitted to the hospital.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** | **Description** |
| **IN\_PATIENT\_ID** | NUMBER(12) | Foreign Key (PATIENT – PATIENT\_ID),  Not Null | Patient’s ID who is admitted |
| **ROOM\_ID** | NUMBER(3) | Foreign Key (ROOM – ROOM\_ID),  Not Null | Room number where patient is admitted. |
| **ADMIT\_DATE** | DATE | Not Null | Date on which patient is admitted to the hospital |
| **DISCHARGE\_DATE** | DATE | Null | Date on which patient is discharged from the hospital |

## **OUT\_PATIENT Table:**

Unlike the In\_patient table, Out\_patient table will only store the data of those patients who are discharged from the hospital as well as those who are visiting for the consultation with the doctor.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** | **Description** |
| **OUT\_PATIENT\_ID** | NUMBER(12) | Foreign Key (PATIENT – PATIENT\_ID),  Not Null | Patient’s ID who is only consulted and not admitted |
| **LAST\_CONSULT\_DATE** | DATE | Not Null | Date on which patient last consulted with a doctor |

## **BILLING Table:**

This table holds a special place as it will store the financial or billing details along with room charges, doctor fees, many more attributes.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** | **Description** |
| **BILL\_ID** | NUMBER(10) | Primary Key | Bill ID |
| **BILL\_DATE** | DATE | Not Null | Date on which patient’s bill is sent |
| **PATIENT\_ID** | NUMBER(12) | Foreign Key (PATIENT – PATIENT\_ID),  Not Null | Patient’s ID |
| **PRESCRIPTION\_CHARGE** | NUMBER(6,2) | Not Null | Total charges for patient’s prescribed medicines |
| **INSURANCE\_PERCENT** | NUMBER(3) | Not Null | Percentage of total billing charges covered by patient’s insurance |
| **PAID** | CHAR(1) | Not Null | ‘Y’ – Paid  ‘N’ – Not Paid |

## **ROOM Table:**

The following table will contain the information about the rooms within the hospital that are dedicated to patients. It will show the availability of the rooms, room number, and its charges

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** | **Description** |
| **ROOM\_ID** | NUMBER(3) | Primary Key | Room number where patient is admitted. |
| **ROOM\_STATUS** | VARCHAR(10) | Not Null  Check | Room’s status i.e., ‘Vacant’ or ‘Occupied’ |
| **ROOM\_CHARGE** | NUMBER(6,2) | Not Null | Room charges per night |

## **APPOINTMENT Table:**

Appointment table will contain details like doctor id, patient id, and appointment id and it will be another building block for joining multiple tables.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** | **Description** |
| **APPOINTMENT\_ID** | NUMBER(10) | Primary Key | ID for scheduled appointment |
| **DOCTOR\_ID** | NUMBER(10) | Foreign Key (DOCTOR – DOCTOR\_ID),  Not Null | Doctor’s ID with whom appointment is booked |
| **PATIENT\_ID** | NUMBER(12) | Foreign Key (PATIENT – PATIENT\_ID),  Not Null | Patient’s ID for whom appointment is booked |
| **ADMIN\_ID** | NUMBER(10) | Foreign Key (ADMINISTRATION – ADMIN\_ID),  Not Null | Administrator’s ID who booked the appointment for patient |
| **APPOINTMENT\_DATE** | DATE | Not Null | Date and time of scheduled appointment |
| **APPOINTMENT\_STATUS** | VARCHAR(10) | Not Null  Check | Appointment status i.e., ‘Planned’, ‘In-Progress’, ‘Done’ or ‘Cancelled’ |

## **DEPARTMENT Table:**

In the end, the department table will list the details of all types of departments within that hospital.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** | **Description** |
| **DEPT\_ID** | NUMBER(4) | Primary Key | Medical department’s ID |
| **DEPT\_NAME** | VARCHAR(100) | Not Null | Medical Department Name |

# **SECURITY**

There are mainly three types of user access defined in our Hospital DBMS:

## **Admin Access:**

* 1. Schedule/cancel appointments.
  2. Create a new patient/doctor profile.
  3. View and update patient/doctor profile.
  4. Cannot access patient’s medical records (diagnosis details)

## **Doctor Access:**

* 1. View scheduled appointments.
  2. View patient’s medical history and create new medical records (diagnosis).
  3. Mark them as consulting patients.
  4. Admit Patients.
  5. Discharge Patients.
  6. Generate Bills.

## **Patient Access:**

* 1. View diagnosis details and medical history.
  2. View bills.