নিচে একটি হাসপাতাল ব্যবস্থাপনা সিস্টেমের ডেটাবেস ডিজাইন, টেবিলের এ্যাট্রিবিউট এবং তাদের মধ্যে সম্পর্কের বিস্তারিত বর্ণনা দেওয়া হল:

### ১. **Administration (প্রশাসন) টেবিল**

**এ্যাট্রিবিউট:**
- **AdminID**: প্রশাসকের ইউনিক আইডি (প্রাইমারি কি)
- **FirstName**: প্রথম নাম
- **LastName**: শেষ নাম
- **Phone**: ফোন নম্বর
- **Email**: ইমেল
- **Address**: ঠিকানা
- **Role**: ভূমিকা (যেমন: Admin, HR, Manager)

**বর্ণনা:**
এই টেবিল প্রশাসনিক ব্যক্তিদের তথ্য সংরক্ষণ করে।

### ২. **Users (ব্যবহারকারী) টেবিল**

**এ্যাট্রিবিউট:**
- **UserID**: ব্যবহারকারীর ইউনিক আইডি (প্রাইমারি কি)
- **UserName**: ব্যবহারকারীর নাম
- **Password**: পাসওয়ার্ড (হ্যাশ করা)
- **Role**: ভূমিকা (যেমন: Doctor, Nurse, Receptionist)
- **FirstName**: প্রথম নাম
- **LastName**: শেষ নাম
- **Phone**: ফোন নম্বর
- **Email**: ইমেল
- **Address**: ঠিকানা

**বর্ণনা:**
ব্যবহারকারীর ব্যক্তিগত এবং লগইন তথ্য সংরক্ষণ করে।

### ৩. **Roles (ভূমিকা) টেবিল**

**এ্যাট্রিবিউট:**
- **RoleID**: ভূমিকার ইউনিক আইডি (প্রাইমারি কি)
- **RoleName**: ভূমিকার নাম (যেমন: Doctor, Nurse)
- **Description**: ভূমিকার বর্ণনা

**বর্ণনা:**
ব্যবহারকারীর ভূমিকার তথ্য সংরক্ষণ করে।

### 8. **Permissions (অনুমতি) টেবিল**

**এ্যাট্রিবিউট:**
- **PermissionID**: অনুমতির ইউনিক আইডি (প্রাইমারি কি)
- **RoleID**: ভূমিকার আইডি (ফরেন কি)
- **PermissionName**: অনুমতির নাম (যেমন: View Patient Records, Update Billing)

**বর্ণনা:**
ভূমিকার অধীনে অনুমতিগুলি সংরক্ষণ করে।

### ৫. **UserRoles (ব্যবহারকারীর ভূমিকা) টেবিল**

**এ্যাট্রিবিউট:**
- **UserRoleID**: ব্যবহারকারীর ভূমিকার ইউনিক আইডি (প্রাইমারি কি)
- **UserID**: ব্যবহারকারীর আইডি (ফরেন কি)
- **RoleID**: ভূমিকার আইডি (ফরেন কি)
- **AdminID**: প্রশাসকের আইডি (ফরেন কি, যদি প্রযোজ্য)

**বর্ণনা:**
ব্যবহারকারীর ভূমিকা এবং প্রশাসকের সাথে সম্পর্ক নির্দেশ করে।

### ৬. **Patients (রোগী) টেবিল**

**এ্যাট্রিবিউট:**
- **PatientID**: রোগীর ইউনিক আইডি (প্রাইমারি কি)
- **FirstName**: প্রথম নাম
- **LastName**: শেষ নাম

- **DateOfBirth**: জন্ম তারিখ
- **Gender**: লিঙ্গ
- **Address**: ঠিকানা
- **Phone**: ফোন নম্বর
- **Email**: ইমেল
- **EmergencyContactName**: জরুরি যোগাযোগের নাম
- **EmergencyContactPhone**: জরুরি যোগাযোগের ফোন নম্বর
- **InsuranceDetails**: বীমার বিবরণ

**বর্ণনা:**
রোগীদের বেসিক তথ্য সংরক্ষণ করে।

### ৭. **Doctors (ডাক্তার) টেবিল**

**এ্যাট্রিবিউট:**
- **DoctorID**: ডাক্তারের ইউনিক আইডি (প্রাইমারি কি)
- **FirstName**: প্রথম নাম
- **LastName**: শেষ নাম
- **Specialization**: বিশেষজ্ঞতা
- **Phone**: ফোন নম্বর
- **Email**: ইমেল
- **Address**: ঠিকানা
- **JoiningDate**: যোগদানের তারিখ
- **Salary**: বেতন

**বর্ণনা:**
ডাক্তারের ব্যক্তিগত এবং কর্ম সংক্রান্ত তথ্য সংরক্ষণ করে।

### ৮. **Staff (স্টাফ) টেবিল**

**এ্যাট্রিবিউট:**
- **StaffID**: স্টাফের ইউনিক আইডি (প্রাইমারি কি)
- **FirstName**: প্রথম নাম
- **LastName**: শেষ নাম
- **Position**: পদবী

- **Phone**: ফোন নম্বর
- **Email**: ইমেল
- **Address**: ঠিকানা
- **Salary**: বেতন
- **JoiningDate**: যোগদানের তারিখ

**বর্ণনা:**
স্টাফদের ব্যক্তিগত এবং কর্ম সংক্রান্ত তথ্য সংরক্ষণ করে।

### ৯. **Appointments (নিযুক্তি) টেবিল**

**এ্যাট্রিবিউট:**
- **AppointmentID**: নিয়োগের ইউনিক আইডি (প্রাইমারি কি)
- **PatientID**: রোগীর আইডি (ফরেন কি)
- **DoctorID**: ডাক্তারের আইডি (ফরেন কি)
- **AppointmentDate**: নিয়োগের তারিখ
- **AppointmentTime**: নিয়োগের সময়
- **ReasonForVisit**: পরিদর্শনের কারণ
- **Status**: অবস্থান (যেমন: Scheduled, Completed, Canceled)

**বর্ণনা:**
রোগী এবং ডাক্তার মধ্যে নিয়োগের তথ্য সংরক্ষণ করে।

### ১০. **MedicalRecords (চিকিৎসা রেকর্ড) টেবিল**

**এ্যাট্রিবিউট:**
- **RecordID**: রেকর্ডের ইউনিক আইডি (প্রাইমারি কি)
- **PatientID**: রোগীর আইডি (ফরেন কি)
- **DoctorID**: ডাক্তারের আইডি (ফরেন কি)
- **Diagnosis**: নির্ণয়
- **Treatment**: চিকিৎসা
- **Prescription**: প্রেসক্রিপশন
- **DateOfRecord**: রেকর্ডের তারিখ

**বর্ণনা:**

রোগীর চিকিৎসা রেকর্ড সংরক্ষণ করে।

### ১১. **Billing (বিলিং) টেবিল**

**এ্যাট্রিবিউট:**
- **BillID**: বিলের ইউনিক আইডি (প্রাইমারি কি)
- **PatientID**: রোগীর আইডি (ফরেন কি)
- **BillDate**: বিলের তারিখ
- **TotalAmount**: মোট পরিমাণ
- **PaidAmount**: পরিশোধিত পরিমাণ
- **DueAmount**: বাকি পরিমাণ
- **PaymentMethod**: পরিশোধের পদ্ধতি
- **PaymentStatus**: পরিশোধের অবস্থা (যেমন: Paid, Unpaid)

**বর্ণনা:**
রোগীর বিলিং তথ্য সংরক্ষণ করে।

### ১২. **Inventory (ইনভেন্টরি) টেবিল**

**এ্যাট্রিবিউট:**
- **InventoryID**: ইনভেন্টরির ইউনিক আইডি (প্রাইমারি কি)
- **ItemName**: আইটেমের নাম
- **Quantity**: পরিমাণ
- **Supplier**: সরবরাহকারী
- **PurchaseDate**: ক্রয়ের তারিখ
- **ExpiryDate**: মেয়াদ শেষের তারিখ
- **Cost**: খরচ

**বর্ণনা:**
হাসপাতালের স্টক এবং ইনভেন্টরি সম্পর্কিত তথ্য সংরক্ষণ করে।

### **সম্পর্ক**

1. **Users এবং UserRoles:**
   - `UserID` ফিল্ড `UserRoles` টেবিলে ফরেন কি হিসেবে ব্যবহৃত হয়।

2. **Roles এবং Permissions:**
   - `RoleID` ফিল্ড `Permissions` টেবিলে ফরেন কি হিসেবে ব্যবহৃত হয়।

3. **Patients এবং Appointments:**
   - `PatientID` ফিল্ড `Appointments` টেবিলে ফরেন কি হিসেবে ব্যবহৃত হয়।

4. **Doctors এবং Appointments:**
   - `DoctorID` ফিল্ড `Appointments` টেবিলে ফরেন কি হিসেবে ব্যবহৃত হয়।

5. **Patients এবং MedicalRecords:**
   - `PatientID` ফিল্ড `MedicalRecords` টেবিলে ফরেন কি হিসেবে ব্যবহৃত হয়।

6. **Doctors এবং MedicalRecords:**
   - `DoctorID` ফিল্ড `MedicalRecords` টেবিলে ফরেন কি হিসেবে ব্যবহৃত হয়।

7. **Patients এবং Billing:**
   - `PatientID` ফিল্ড `Billing` টেবিলে ফরেন কি হিসেবে ব্যবহৃত হয়।

### **সারাংশ**

- **Administration** টেবিল প্রশাসনিক তথ্য সংরক্ষণ করে।
- **Users** টেবিল ব্যবহারকারীর লগইন এবং ব্যক্তিগত তথ্য সংরক্ষণ করে।
- **Roles** এবং **Permissions** টেবিল ব্যবস্থাপনার ভূমিকা এবং অনুমতিগুলি সংরক্ষণ করে।
- **UserRoles** টেবিল ব্যবহারকারীর ভূমিকা সম্পর্কিত তথ্য ধারণ করে।
- **Patients**, **Doctors**, **Staff**, **Appointments**, **MedicalRecords**, **Billing**, এবং **Inventory

** টেবিল হাসপাতালের বিভিন্ন কার্যক্রম সম্পর্কিত তথ্য সংরক্ষণ করে।

এটি হাসপাতাল ব্যবস্থাপনা সিস্টেমের টেবিল এবং তাদের সম্পর্কের বিস্তারিত বর্ণনা। যদি আপনার আরও কিছু প্রয়োজন হয়, দয়া করে জানান!

SQL কোড দিয়ে স্টোরড প্রোসিডিউরের মধ্যে বিভিন্ন ধরনের কার্যকলাপ করা সম্ভব। স্টোরড প্রোসিডিউর হলো একটি পূর্বনির্ধারিত SQL কোড ব্লক যা ডাটাবেসে সংরক্ষিত থাকে এবং পুনরায় ব্যবহৃত হতে পারে। এটি ডাটাবেস অপারেশনগুলোকে অটোমেট করার জন্য এবং পারফরম্যান্স উন্নত করার জন্য ব্যবহৃত হয়।

### স্টোরড প্রোসিডিউর তৈরির উদাহরণ:

#### ১. **লগইন প্রোসিডিউর**:
```sql
CREATE PROCEDURE VerifyLogin
   @UserName VARCHAR(255),
   @Password VARCHAR(255)
AS
BEGIN
   DECLARE @StoredPasswordHash VARBINARY(64);
   DECLARE @StoredSalt VARBINARY(16);
   DECLARE @PasswordHash VARBINARY(64);

   -- ইউজারনেম এর উপর ভিত্তি করে স্টোরড পাসওয়ার্ড হ্যাশ এবং সল্ট রিট্রিভ করা
   SELECT @StoredPasswordHash = PasswordHash, @StoredSalt = Salt
   FROM SignUp
   WHERE UserName = @UserName;

   -- পাসওয়ার্ড হ্যাশ তৈরি করা
   SELECT @PasswordHash = HASHBYTES('SHA2_256', @Password + CAST(@StoredSalt AS
VARCHAR(16)));

   -- পাসওয়ার্ড যাচাই করা
   IF @PasswordHash = @StoredPasswordHash
   BEGIN
      -- লগইন সফল হলে লগইন টেবিলে লগইন তথ্য ইনসার্ট করা
      INSERT INTO Login (UserName, LoginTime, IPAddress, DeviceDetails, Location)
      VALUES (@UserName, GETDATE(), '192.168.0.1', 'Mozilla/5.0', 'Dhaka');
```

```
        SELECT 'Login successful' AS Message;
    END
    ELSE
    BEGIN
        SELECT 'Invalid username or password' AS Message;
    END
END;
```

### ব্যাখ্যা:

- **`CREATE PROCEDURE`**: নতুন স্টোরড প্রোসিডিউর তৈরি করতে ব্যবহার করা হয়।
- **`@UserName` ও `@Password`**: ইনপুট প্যারামিটার যা প্রোসিডিউরের জন্য প্রদান করা হয়।
- **`DECLARE`**: স্থানীয় ভেরিয়েবল ঘোষণা করে।
- **`SELECT`**: ব্যবহারকারীর পাসওয়ার্ড হ্যাশ এবং সল্ট রিট্রিভ করার জন্য ব্যবহৃত হয়।
- **`HASHBYTES`**: পাসওয়ার্ড হ্যাশ করতে ব্যবহৃত।
- **`IF`**: পাসওয়ার্ড যাচাই করা হয় এবং সফল হলে লগইন তথ্য `Login` টেবিলে ইনসার্ট করা হয়।
- **`INSERT INTO`**: লগইন তথ্য `Login` টেবিলে ইনসার্ট করা হয়।
- **`SELECT`**: লগইন ফলাফল ফিরিয়ে দেয়।

#### ২. **নতুন ব্যবহারকারী নিবন্ধন প্রোসিডিউর**:

```sql
CREATE PROCEDURE RegisterUser
    @UserName VARCHAR(255),
    @Password VARCHAR(255),
    @Email VARCHAR(255),
    @PhoneNumber VARCHAR(15),
    @Role VARCHAR(50)
AS
BEGIN
    DECLARE @Salt VARBINARY(16);
    DECLARE @PasswordHash VARBINARY(64);

    -- এলোমেলো সল্ট তৈরি করা
```

```
    SET @Salt = CAST(CRYPT_GEN_RANDOM(16) AS VARBINARY(16));

    -- পাসওয়ার্ড হ্যাশ তৈরি করা
    SET @PasswordHash = HASHBYTES('SHA2_256', @Password + CAST(@Salt AS
VARCHAR(16)));

    -- নতুন ব্যবহারকারী নিবন্ধন করা
    INSERT INTO SignUp (UserName, PasswordHash, Salt, Email, PhoneNumber, Role)
    VALUES (@UserName, @PasswordHash, @Salt, @Email, @PhoneNumber, @Role);
END;
```

### ব্যাখ্যা:

- **`DECLARE`**: সল্ট এবং পাসওয়ার্ড হ্যাশের জন্য স্থানীয় ভেরিয়েবল ঘোষণা করে।
- **`SET @Salt`**: এলোমেলো সল্ট তৈরি করে।
- **`SET @PasswordHash`**: পাসওয়ার্ড হ্যাশ তৈরি করে।
- **`INSERT INTO`**: নতুন ব্যবহারকারী নিবন্ধন করে `SignUp` টেবিলে।

### স্টোরড প্রোসিডিউর কল করার উদাহরণ:

#### লগইন প্রোসিডিউর কল:

```sql
EXEC VerifyLogin @UserName = 'exampleUserName', @Password = 'examplePassword';
```

#### নিবন্ধন প্রোসিডিউর কল:

```sql
EXEC RegisterUser @UserName = 'newUser', @Password = 'newPassword', @Email =
'newUser@example.com', @PhoneNumber = '1234567890', @Role = 'User';
```

**উল্লেখযোগ্য কিছু বিষয়:**

- **পরীক্ষা করুন**: স্টোরড প্রোসিডিউর ব্যবহারের আগে সমস্ত প্যারামিটার এবং কুয়ারি সঠিক কিনা তা পরীক্ষা করা উচিত।
- **নিরাপত্তা**: ব্যবহারকারীর পাসওয়ার্ড এবং সল্টের নিরাপত্তা নিশ্চিত করতে স্টোরড প্রোসিডিউরের কোডে ভালভাবে কোড করা প্রয়োজন।

এভাবে, স্টোরড প্রোসিডিউর ব্যবহার করে বিভিন্ন ডাটাবেস অপারেশন কার্যকরভাবে করা যায়।

---

নিচে হাসপাতাল ব্যবস্থাপনা সিস্টেমের জন্য সাইন আপ এবং সাইন ইন টেবিল এবং তাদের এ্যাট্রিবিউটগুলির বিস্তারিত বর্ণনা দেওয়া হলো:

### **SignUp (সাইন আপ) টেবিল:**

**এ্যাট্রিবিউট:**
- **SignUpID**: ইউনিক সাইন আপ আইডি (প্রাইমারি কি)
- **UserName**: ব্যবহারকারীর নাম (ইউনিক)
- **Password**: পাসওয়ার্ড (হ্যাশ করা)
- **FirstName**: প্রথম নাম
- **LastName**: শেষ নাম
- **Email**: ইমেল (ইউনিক)
- **Phone**: ফোন নম্বর
- **Address**: ঠিকানা
- **Role**: ভূমিকা (যেমন: Admin, Doctor, Nurse)
- **SignUpDate**: সাইন আপের তারিখ

**কোড:**
```sql
CREATE TABLE SignUp (
    SignUpID INT PRIMARY KEY IDENTITY(1,1),
    UserName VARCHAR(255) NOT NULL UNIQUE,
    Password VARCHAR(255) NOT NULL,
    FirstName VARCHAR(255) NOT NULL,
    LastName VARCHAR(255) NOT NULL,
    Email VARCHAR(255) NOT NULL UNIQUE,
```

```
    Phone VARCHAR(20),
    Address VARCHAR(255),
    Role VARCHAR(50) NOT NULL,
    SignUpDate DATETIME DEFAULT GETDATE()
);
```

### **SignIn (সাইন ইন) টেবিল:**

**এ্যাট্রিবিউট:**
- **SignInID**: ইউনিক সাইন ইন আইডি (প্রাইমারি কি)
- **UserName**: ব্যবহারকারীর নাম (ফরেন কি)
- **SignInTime**: সাইন ইন সময়
- **SignOutTime**: সাইন আউট সময়

**কোড:**
```sql
CREATE TABLE SignIn (
    SignInID INT PRIMARY KEY IDENTITY(1,1),
    UserName VARCHAR(255) NOT NULL,
    SignInTime DATETIME DEFAULT GETDATE(),
    SignOutTime DATETIME NULL,
    FOREIGN KEY (UserName) REFERENCES SignUp(UserName)
);
```

### **Sample Insert Statements:**

#### **SignUp টেবিল:**
```sql
INSERT INTO SignUp (UserName, Password, FirstName, LastName, Email, Phone, Address, Role)
VALUES ('johndoe', 'hashed_password', 'John', 'Doe', 'john.doe@example.com', '1234567890', '123
Main St', 'Doctor');
```

#### **SignIn টেবিল:**
```sql
INSERT INTO SignIn (UserName, SignInTime)
VALUES ('johndoe', GETDATE());
```

### **Password Checking এবং Error Message Throw:**

সাইন আপ বা সাইন ইন করার সময় পাসওয়ার্ড চেক করা এবং ভুল তথ্য প্রদান করলে এরর মেসেজ থ্রো করার জন্য আমরা স্টোরড প্রসিডিউর ব্যবহার করতে পারি।

### **Check Password Stored Procedure:**
```sql
CREATE PROCEDURE CheckPassword
  @UserName VARCHAR(255),
  @Password VARCHAR(255)
AS
BEGIN
  DECLARE @StoredPassword VARCHAR(255);

  SELECT @StoredPassword = Password
  FROM SignUp
  WHERE UserName = @UserName;

  IF @StoredPassword IS NULL
  BEGIN
    RAISERROR('User does not exist.', 16, 1);
    RETURN;
  END

  IF @StoredPassword != @Password
  BEGIN
    RAISERROR('Incorrect password.', 16, 1);
    RETURN;
  END
```

```
    PRINT 'Login successful';
END;
```

### **Sign Up Stored Procedure:**
```sql
CREATE PROCEDURE SignUpUser
    @UserName VARCHAR(255),
    @Password VARCHAR(255),
    @FirstName VARCHAR(255),
    @LastName VARCHAR(255),
    @Email VARCHAR(255),
    @Phone VARCHAR(20),
    @Address VARCHAR(255),
    @Role VARCHAR(50)
AS
BEGIN
    IF EXISTS (SELECT 1 FROM SignUp WHERE UserName = @UserName)
    BEGIN
        RAISERROR('UserName already exists.', 16, 1);
        RETURN;
    END

    IF EXISTS (SELECT 1 FROM SignUp WHERE Email = @Email)
    BEGIN
        RAISERROR('Email already exists.', 16, 1);
        RETURN;
    END

    INSERT INTO SignUp (UserName, Password, FirstName, LastName, Email, Phone, Address,
Role, SignUpDate)
    VALUES (@UserName, @Password, @FirstName, @LastName, @Email, @Phone, @Address,
@Role, GETDATE());
```

```
    PRINT 'User signed up successfully';
END;
```

এই ডিজাইন এবং কোডগুলি দিয়ে, আপনি সাইন আপ এবং সাইন ইন কার্যকরভাবে পরিচালনা করতে পারবেন, পাসওয়ার্ড চেক করতে পারবেন এবং প্রয়োজনীয় ক্ষেত্রে এরর মেসেজ থ্রো করতে পারবেন।

### হাসপাতাল ব্যবস্থাপনা সিস্টেমে সাইনআপ, সাইনইন এবং সম্পর্কিত টেবিল

#### ১. **Signup (সাইনআপ) টেবিল**

**এ্যাট্রিবিউট:**
- **SignupID**: সাইনআপের ইউনিক আইডি (প্রাইমারি কি)
- **UserID**: ব্যবহারকারীর আইডি (ফরেন কি, ব্যবহারকারীর তথ্যের সাথে সম্পর্কিত)
- **SignupDate**: সাইনআপের তারিখ
- **Username**: ব্যবহারকারীর নাম (অন্য একটি নাম, যদি ভিন্ন থাকে)
- **PasswordHash**: পাসওয়ার্ডের হ্যাশ (গোপনীয়তা নিশ্চিত করার জন্য)
- **Email**: ইমেল ঠিকানা
- **Phone**: ফোন নম্বর
- **Address**: ঠিকানা
- **VerificationStatus**: যাচাইকরণের অবস্থা (যেমন: Verified, Not Verified)

**বর্ণনা:**
সাইনআপ টেবিল ব্যবহারকারীদের সাইনআপের সময় সম্পর্কিত তথ্য সংরক্ষণ করে।

#### ২. **Users (ব্যবহারকারী) টেবিল**

**এ্যাট্রিবিউট:**
- **UserID**: ব্যবহারকারীর ইউনিক আইডি (প্রাইমারি কি)
- **UserName**: ব্যবহারকারীর নাম
- **Password**: পাসওয়ার্ড (হ্যাশ করা)
- **Role**: ভূমিকা (যেমন: Doctor, Nurse, Receptionist)
- **FirstName**: প্রথম নাম
- **LastName**: শেষ নাম
- **Phone**: ফোন নম্বর

- **Email**: ইমেল
- **Address**: ঠিকানা

**বর্ণনা:**
ব্যবহারকারীর লগইন এবং ব্যক্তিগত তথ্য সংরক্ষণ করে।

#### ৩. **Roles (ভূমিকা) টেবিল**

**এ্যাট্রিবিউট:**
- **RoleID**: ভূমিকার ইউনিক আইডি (প্রাইমারি কি)
- **RoleName**: ভূমিকার নাম (যেমন: Doctor, Nurse)
- **Description**: ভূমিকার বর্ণনা

**বর্ণনা:**
ব্যবহারকারীর ভূমিকার তথ্য সংরক্ষণ করে।

#### ৪. **Permissions (অনুমতি) টেবিল**

**এ্যাট্রিবিউট:**
- **PermissionID**: অনুমতির ইউনিক আইডি (প্রাইমারি কি)
- **RoleID**: ভূমিকার আইডি (ফরেন কি)
- **PermissionName**: অনুমতির নাম (যেমন: View Patient Records, Update Billing)

**বর্ণনা:**
ভূমিকার অধীনে অনুমতিগুলি সংরক্ষণ করে।

#### ৫. **UserRoles (ব্যবহারকারীর ভূমিকা) টেবিল**

**এ্যাট্রিবিউট:**
- **UserRoleID**: ব্যবহারকারীর ভূমিকার ইউনিক আইডি (প্রাইমারি কি)
- **UserID**: ব্যবহারকারীর আইডি (ফরেন কি)
- **RoleID**: ভূমিকার আইডি (ফরেন কি)
- **AdminID**: প্রশাসকের আইডি (ফরেন কি, যদি প্রযোজ্য)

**বর্ণনা:**

ব্যবহারকারীর ভূমিকা এবং প্রশাসকের সাথে সম্পর্ক নির্দেশ করে।

#### ৬. **Patients (রোগী) টেবিল**

**এ্যাট্রিবিউট:**
- **PatientID**: রোগীর ইউনিক আইডি (প্রাইমারি কি)
- **FirstName**: প্রথম নাম
- **LastName**: শেষ নাম
- **DateOfBirth**: জন্ম তারিখ
- **Gender**: লিঙ্গ
- **Address**: ঠিকানা
- **Phone**: ফোন নম্বর
- **Email**: ইমেল
- **EmergencyContactName**: জরুরি যোগাযোগের নাম
- **EmergencyContactPhone**: জরুরি যোগাযোগের ফোন নম্বর
- **InsuranceDetails**: বীমার বিবরণ

**বর্ণনা:**
রোগীদের বেসিক তথ্য সংরক্ষণ করে।

#### ৭. **Doctors (ডাক্তার) টেবিল**

**এ্যাট্রিবিউট:**
- **DoctorID**: ডাক্তারের ইউনিক আইডি (প্রাইমারি কি)
- **FirstName**: প্রথম নাম
- **LastName**: শেষ নাম
- **Specialization**: বিশেষজ্ঞতা
- **Phone**: ফোন নম্বর
- **Email**: ইমেল
- **Address**: ঠিকানা
- **JoiningDate**: যোগদানের তারিখ
- **Salary**: বেতন

**বর্ণনা:**
ডাক্তারের ব্যক্তিগত এবং কর্ম সংক্রান্ত তথ্য সংরক্ষণ করে।

#### ৮. **Staff (স্টাফ) টেবিল**

**এ্যাট্রিবিউট:**
- **StaffID**: স্টাফের ইউনিক আইডি (প্রাইমারি কি)
- **FirstName**: প্রথম নাম
- **LastName**: শেষ নাম
- **Position**: পদবী
- **Phone**: ফোন নম্বর
- **Email**: ইমেল
- **Address**: ঠিকানা
- **Salary**: বেতন
- **JoiningDate**: যোগদানের তারিখ

**বর্ণনা:**
স্টাফদের ব্যক্তিগত এবং কর্ম সংক্রান্ত তথ্য সংরক্ষণ করে।

#### ৯. **Appointments (নিয়োগ) টেবিল**

**এ্যাট্রিবিউট:**
- **AppointmentID**: নিয়োগের ইউনিক আইডি (প্রাইমারি কি)
- **PatientID**: রোগীর আইডি (ফরেন কি)
- **DoctorID**: ডাক্তারের আইডি (ফরেন কি)
- **AppointmentDate**: নিয়োগের তারিখ
- **AppointmentTime**: নিয়োগের সময়
- **ReasonForVisit**: পরিদর্শনের কারণ
- **Status**: অবস্থান (যেমন: Scheduled, Completed, Canceled)

**বর্ণনা:**
রোগী এবং ডাক্তার মধ্যে নিয়োগের তথ্য সংরক্ষণ করে।

#### ১০. **MedicalRecords (চিকিৎসা রেকর্ড) টেবিল**

**এ্যাট্রিবিউট:**
- **RecordID**: রেকর্ডের ইউনিক আইডি (প্রাইমারি কি)

- **PatientID**: রোগীর আইডি (ফরেন কি)
- **DoctorID**: ডাক্তারের আইডি (ফরেন কি)
- **Diagnosis**: নির্ণয়
- **Treatment**: চিকিৎসা
- **Prescription**: প্রেসক্রিপশন
- **DateOfRecord**: রেকর্ডের তারিখ

**বর্ণনা:**
রোগীর চিকিৎসা রেকর্ড সংরক্ষণ করে।

#### ১১. **Billing (বিলিং) টেবিল**

**এ্যাট্রিবিউট:**
- **BillID**: বিলের ইউনিক আইডি (প্রাইমারি কি)
- **PatientID**: রোগীর আইডি (ফরেন কি)
- **BillDate**: বিলের তারিখ
- **TotalAmount**: মোট পরিমাণ
- **PaidAmount**: পরিশোধিত পরিমাণ
- **DueAmount**: বাকি পরিমাণ
- **PaymentMethod**: পরিশোধের পদ্ধতি
- **PaymentStatus**: পরিশোধের অবস্থা (যেমন: Paid, Unpaid)

**বর্ণনা:**
রোগীর বিলিং তথ্য সংরক্ষণ করে।

#### ১২. **Inventory (ইনভেন্টরি) টেবিল**

**এ্যাট্রিবিউট:**
- **InventoryID**: ইনভেন্টরির ইউনিক আইডি (প্রাইমারি কি)
- **ItemName**: আইটেমের নাম
- **Quantity**: পরিমাণ
- **Supplier**: সরবরাহকারী
- **PurchaseDate**: ক্রয়ের তারিখ
- **ExpiryDate**: মেয়াদ শেষের তারিখ
- **Cost**: খরচ

**বর্ণনা:**
হাসপাতালের স্টক এবং ইনভেন্টরি সম্পর্কিত তথ্য সংরক্ষণ করে।

### **সম্পর্ক**

- **Signup** টেবিলের **UserID** ফিল্ড **Users** টেবিলের সাথে সম্পর্কিত।
- **Users** টেবিলের **UserID** ফিল্ড **UserRoles** টেবিলে ফরেন কি হিসেবে ব্যবহৃত হয়।
- **Roles** টেবিলের **RoleID** ফিল্ড **Permissions** টেবিলে ফরেন কি হিসেবে ব্যবহৃত হয়।
- **Patients** টেবিলের **PatientID** ফিল্ড **Appointments**, **MedicalRecords**, এবং **Billing** টেবিলে ফরেন কি হিসেবে ব্যবহৃত হয়।
- **Doctors** টেবিলের **DoctorID** ফিল্ড **Appointments** এবং **MedicalRecords** টেবিলে ফরেন কি হিসেবে ব্যবহৃত হয়।
- **UserRoles** টেবিলের **AdminID** প্রশাসনিক ভূমিকা নির্দেশ করে, যা **Administration** টেবিলের সাথে সম্পর্কিত।

এই তথ্য আপনাকে সাইনআপ এবং সাইনইন প্রক্রিয়া সহ হাসপাতাল ব্যবস্থাপনা সিস্টেমের ডেটাবেস ডিজাইন বুঝতে সাহায্য করবে। যদি আরও কিছু জানার প্রয়োজন হয় বা সাহায্য দরকার হয়, আমাকে জানান!

---

### টেবিল ভিত্তিক ট্রিগার এবং স্টোর প্রসিডিউর ব্যবহারের স্থান নির্ধারণ

#### **ট্রিগার (Triggers):**

**১. Patients (রোগী) টেবিল:**
- **Trigger Name**: `TRG_AFTER_INSERT_PATIENT`
- **কার্যকরী সময়**: INSERT
- **বর্ণনা**: রোগীর রেকর্ড যুক্ত হলে একটি অটোম্যাটিক ট্রিগার চালু হবে যা একটি স্বাগত ইমেল পাঠাবে বা রোগীর তথ্য লগ করবে।

**২. Appointments (নিয়োগ) টেবিল:**
- **Trigger Name**: `TRG_BEFORE_INSERT_APPOINTMENT`
- **কার্যকরী সময়**: BEFORE INSERT
- **বর্ণনা**: নতুন নিয়োগের আগে চেক করবে যদি একই সময়ে অন্য কোনো নিয়োগ থাকে, তাহলে তা অবহিত করবে।

**৩. Billing (বিলিং) টেবিল:**

- **Trigger Name**: `TRG_AFTER_UPDATE_BILL`
- **কার্যকরী সময়**: AFTER UPDATE
- **বর্ণনা**: যখন বিলিং তথ্য আপডেট করা হয়, তখন রোগীকে বা অ্যাডমিনকে একটি নোটিফিকেশন পাঠানো হবে।

#### **স্টোর প্রসিডিউর (Stored Procedures):**

**১. Users (ব্যবহারকারী) টেবিল:**
- **Stored Procedure Name**: `SP_ADD_NEW_USER`
- **বর্ণনা**: নতুন ব্যবহারকারী যুক্ত করার জন্য একটি স্টোর প্রসিডিউর। এটি ব্যবহারকারীর তথ্য গ্রহণ করবে এবং সঠিক টেবিলগুলোতে ইনসার্ট করবে।

**২. MedicalRecords (চিকিৎসা রেকর্ড) টেবিল:**
- **Stored Procedure Name**: `SP_UPDATE_MEDICAL_RECORD`
- **বর্ণনা**: চিকিৎসা রেকর্ড আপডেট করার জন্য একটি স্টোর প্রসিডিউর। এটি রোগীর রেকর্ড গ্রহণ করে এবং আপডেট করবে।

**৩. Inventory (ইনভেন্টরি) টেবিল:**
- **Stored Procedure Name**: `SP_MANAGE_INVENTORY`
- **বর্ণনা**: ইনভেন্টরি পরিচালনা করার জন্য একটি স্টোর প্রসিডিউর। এটি নতুন আইটেম যুক্ত করা, বিদ্যমান আইটেম আপডেট করা এবং আইটেম মুছে ফেলার কাজ করবে।

**৪. Appointments (নিয়োগ) টেবিল:**
- **Stored Procedure Name**: `SP_SCHEDULE_APPOINTMENT`
- **বর্ণনা**: নতুন নিয়োগ নির্ধারণের জন্য একটি স্টোর প্রসিডিউর। এটি রোগী এবং ডাক্তারের আইডি গ্রহণ করে এবং নতুন নিয়োগ সংরক্ষণ করবে।

### **ট্রিগার এবং স্টোর প্রসিডিউর ব্যবহারের কারণ:**

**ট্রিগার:**
- ট্রিগার এমন কাজগুলি পরিচালনা করতে সহায়ক যা নির্দিষ্ট সময়ে স্বয়ংক্রিয়ভাবে সম্পন্ন করতে হবে।
- নির্দিষ্ট ইভেন্ট (যেমন: INSERT, UPDATE, DELETE) এর পর বা আগে নির্দিষ্ট কাজগুলি সম্পাদন করা।

**স্টোর প্রসিডিউর:**
- স্টোর প্রসিডিউর কমপ্লেক্স কাজগুলোকে ম্যানেজ করার জন্য ব্যবহৃত হয়।
- মাল্টিপল টেবিল বা কাজকে একত্রিত করার জন্য স্টোর প্রসিডিউর উপকারী।

- কোড পুনরায় ব্যবহার এবং রক্ষণাবেক্ষণ সহজ করে।

এইভাবে ট্রিগার এবং স্টোর প্রসিডিউর হাসপাতাল ব্যবস্থাপনা সিস্টেমে ব্যবহৃত হবে। যদি আরও কিছু জানতে বা সাহায্য করতে হয়, জানাবেন।

---

নিচে হাসপাতাল ব্যবস্থাপনা সিস্টেমের টেবিলগুলির জন্য সম্পূর্ণ SQL কোড সহ ট্রিগার এবং স্টোর প্রসিডিউরের কোড দেওয়া হলো:

### **টেবিল তৈরির কোড:**

```sql
CREATE TABLE Administration (
    AdminID INT PRIMARY KEY,
    FirstName VARCHAR(255) NOT NULL,
    LastName VARCHAR(255) NOT NULL,
    Phone VARCHAR(20),
    Email VARCHAR(255),
    Address VARCHAR(255),
    Role VARCHAR(50) NOT NULL
);

CREATE TABLE Users (
    UserID INT PRIMARY KEY,
    UserName VARCHAR(255) NOT NULL,
    Password VARCHAR(255) NOT NULL,
    Role VARCHAR(50) NOT NULL,
    FirstName VARCHAR(255),
    LastName VARCHAR(255),
    Phone VARCHAR(20),
    Email VARCHAR(255),
    Address VARCHAR(255)
);

CREATE TABLE Roles (
    RoleID INT PRIMARY KEY,
```

```
    RoleName VARCHAR(50) NOT NULL,
    Description VARCHAR(255)
);

CREATE TABLE Permissions (
    PermissionID INT PRIMARY KEY,
    RoleID INT,
    PermissionName VARCHAR(255) NOT NULL,
    FOREIGN KEY (RoleID) REFERENCES Roles(RoleID)
);

CREATE TABLE UserRoles (
    UserRoleID INT PRIMARY KEY,
    UserID INT,
    RoleID INT,
    AdminID INT,
    FOREIGN KEY (UserID) REFERENCES Users(UserID),
    FOREIGN KEY (RoleID) REFERENCES Roles(RoleID),
    FOREIGN KEY (AdminID) REFERENCES Administration(AdminID)
);

CREATE TABLE Patients (
    PatientID INT PRIMARY KEY,
    FirstName VARCHAR(255),
    LastName VARCHAR(255),
    DateOfBirth DATE,
    Gender VARCHAR(10),
    Address VARCHAR(255),
    Phone VARCHAR(20),
    Email VARCHAR(255),
    EmergencyContactName VARCHAR(255),
    EmergencyContactPhone VARCHAR(20),
    InsuranceDetails VARCHAR(255)
);
```

```
CREATE TABLE Doctors (
    DoctorID INT PRIMARY KEY,
    FirstName VARCHAR(255),
    LastName VARCHAR(255),
    Specialization VARCHAR(255),
    Phone VARCHAR(20),
    Email VARCHAR(255),
    Address VARCHAR(255),
    JoiningDate DATE,
    Salary DECIMAL(10, 2)
);

CREATE TABLE Staff (
    StaffID INT PRIMARY KEY,
    FirstName VARCHAR(255),
    LastName VARCHAR(255),
    Position VARCHAR(255),
    Phone VARCHAR(20),
    Email VARCHAR(255),
    Address VARCHAR(255),
    Salary DECIMAL(10, 2),
    JoiningDate DATE
);

CREATE TABLE Appointments (
    AppointmentID INT PRIMARY KEY,
    PatientID INT,
    DoctorID INT,
    AppointmentDate DATE,
    AppointmentTime TIME,
    ReasonForVisit VARCHAR(255),
    Status VARCHAR(50),
    FOREIGN KEY (PatientID) REFERENCES Patients(PatientID),
    FOREIGN KEY (DoctorID) REFERENCES Doctors(DoctorID)
);
```

```
CREATE TABLE MedicalRecords (
    RecordID INT PRIMARY KEY,
    PatientID INT,
    DoctorID INT,
    Diagnosis VARCHAR(255),
    Treatment VARCHAR(255),
    Prescription VARCHAR(255),
    DateOfRecord DATE,
    FOREIGN KEY (PatientID) REFERENCES Patients(PatientID),
    FOREIGN KEY (DoctorID) REFERENCES Doctors(DoctorID)
);

CREATE TABLE Billing (
    BillID INT PRIMARY KEY,
    PatientID INT,
    BillDate DATE,
    TotalAmount DECIMAL(10, 2),
    PaidAmount DECIMAL(10, 2),
    DueAmount DECIMAL(10, 2),
    PaymentMethod VARCHAR(50),
    PaymentStatus VARCHAR(50),
    FOREIGN KEY (PatientID) REFERENCES Patients(PatientID)
);

CREATE TABLE Inventory (
    InventoryID INT PRIMARY KEY,
    ItemName VARCHAR(255),
    Quantity INT,
    Supplier VARCHAR(255),
    PurchaseDate DATE,
    ExpiryDate DATE,
    Cost DECIMAL(10, 2)
);
```

### **ট্রিগার কোড:**

**Patients (রোগী) টেবিলের জন্য ট্রিগার:**

```sql
CREATE TRIGGER TRG_AFTER_INSERT_PATIENT
ON Patients
AFTER INSERT
AS
BEGIN
   DECLARE @PatientID INT;
   SELECT @PatientID = inserted.PatientID FROM inserted;

   -- Example: Log the patient insertion or send a welcome email
   PRINT 'New patient added with ID: ' + CAST(@PatientID AS VARCHAR);
END;
```

**Appointments (নিয়োগ) টেবিলের জন্য ট্রিগার:**

```sql
CREATE TRIGGER TRG_BEFORE_INSERT_APPOINTMENT
ON Appointments
INSTEAD OF INSERT
AS
BEGIN
   DECLARE @AppointmentDate DATE;
   DECLARE @AppointmentTime TIME;
   DECLARE @DoctorID INT;

   SELECT @AppointmentDate = inserted.AppointmentDate,
        @AppointmentTime = inserted.AppointmentTime,
        @DoctorID = inserted.DoctorID
   FROM inserted;
```

```
    IF EXISTS (
        SELECT 1
        FROM Appointments
        WHERE AppointmentDate = @AppointmentDate
          AND AppointmentTime = @AppointmentTime
          AND DoctorID = @DoctorID
    )
    BEGIN
        RAISERROR ('The doctor already has an appointment at this time.', 16, 1);
        ROLLBACK TRANSACTION;
    END
    ELSE
    BEGIN
        INSERT INTO Appointments (PatientID, DoctorID, AppointmentDate, AppointmentTime,
ReasonForVisit, Status)
        SELECT PatientID, DoctorID, AppointmentDate, AppointmentTime, ReasonForVisit, Status
        FROM inserted;
    END;
END;
```

**Billing (বিলিং) টেবিলের জন্য ট্রিগার:**

```sql
CREATE TRIGGER TRG_AFTER_UPDATE_BILL
ON Billing
AFTER UPDATE
AS
BEGIN
    DECLARE @BillID INT;
    DECLARE @PatientID INT;
    DECLARE @TotalAmount DECIMAL(10, 2);
    DECLARE @PaidAmount DECIMAL(10, 2);
    DECLARE @DueAmount DECIMAL(10, 2);
```

```
    SELECT @BillID = inserted.BillID,
           @PatientID = inserted.PatientID,
           @TotalAmount = inserted.TotalAmount,
           @PaidAmount = inserted.PaidAmount,
           @DueAmount = inserted.DueAmount
    FROM inserted;

    -- Example: Notify the patient or admin about the billing update
    PRINT 'Billing updated for patient ID: ' + CAST(@PatientID AS VARCHAR);
END;
```

### **স্টোর প্রসিডিউর কোড:**

**Users (ব্যবহারকারী) টেবিলের জন্য স্টোর প্রসিডিউর:**

```sql
CREATE PROCEDURE SP_ADD_NEW_USER
    @UserName VARCHAR(255),
    @Password VARCHAR(255),
    @Role VARCHAR(50),
    @FirstName VARCHAR(255),
    @LastName VARCHAR(255),
    @Phone VARCHAR(20),
    @Email VARCHAR(255),
    @Address VARCHAR(255)
AS
BEGIN
    INSERT INTO Users (UserName, Password, Role, FirstName, LastName, Phone, Email,
Address)
    VALUES (@UserName, @Password, @Role, @FirstName, @LastName, @Phone, @Email,
@Address);
END;
```

**MedicalRecords (চিকিৎসা রেকর্ড) টেবিলের জন্য স্টোর প্রসিডিউর:**

```sql
CREATE PROCEDURE SP_UPDATE_MEDICAL_RECORD
    @RecordID INT,
    @Diagnosis VARCHAR(255),
    @Treatment VARCHAR(255),
    @Prescription VARCHAR(255),
    @DateOfRecord DATE
AS
BEGIN
    UPDATE MedicalRecords
    SET Diagnosis = @Diagnosis,
        Treatment = @Treatment,
        Prescription = @Prescription,
        DateOfRecord = @DateOfRecord
    WHERE RecordID = @RecordID;
END;
```

**Inventory (ইনভেন্টরি) টেবিলের জন্য স্টোর প্রসিডিউর:**

```sql
CREATE PROCEDURE SP_MANAGE_INVENTORY
    @Action VARCHAR(10),
    @InventoryID INT = NULL,
    @ItemName VARCHAR(255) = NULL,
    @Quantity INT = NULL,
    @Supplier VARCHAR(255) = NULL,
    @PurchaseDate DATE = NULL,
    @ExpiryDate DATE = NULL,
    @Cost DECIMAL(10, 2) = NULL
AS
BEGIN
```

```
    IF @Action = 'ADD'
    BEGIN
        INSERT INTO Inventory (ItemName, Quantity, Supplier, PurchaseDate, ExpiryDate, Cost)
        VALUES (@ItemName, @Quantity, @Supplier, @PurchaseDate, @ExpiryDate, @Cost);
    END
    ELSE IF @Action = 'UPDATE'
    BEGIN
        UPDATE Inventory
        SET ItemName = @ItemName,
            Quantity = @Quantity,
            Supplier = @Supplier,
            PurchaseDate = @PurchaseDate,
            ExpiryDate = @ExpiryDate,
            Cost = @Cost
        WHERE InventoryID = @InventoryID;
    END
    ELSE IF @Action = 'DELETE'
    BEGIN
        DELETE FROM Inventory WHERE InventoryID = @InventoryID;
    END
END;
```

**Appointments (নিয়োগ) টেবিলের জন্য স্টোর প্রসিডিউর:**

```sql
CREATE PROCEDURE SP_SCHEDULE_APPOINTMENT
    @PatientID INT,
    @DoctorID INT,
    @AppointmentDate DATE,
    @AppointmentTime TIME,
    @ReasonForVisit VARCHAR(255),
    @Status VARCHAR(50)
AS
BEGIN
```

```
    IF EXISTS (
        SELECT 1
        FROM Appointments
        WHERE AppointmentDate = @AppointmentDate
          AND AppointmentTime = @AppointmentTime
          AND DoctorID = @DoctorID
    )
    BEGIN
        RAISERROR ('The doctor already has an appointment at

 this time.', 16, 1);
    END
    ELSE
    BEGIN
        INSERT INTO Appointments (PatientID, DoctorID, AppointmentDate, AppointmentTime,
ReasonForVisit, Status)
        VALUES (@PatientID, @DoctorID, @AppointmentDate, @AppointmentTime,
@ReasonForVisit, @Status);
    END;
END;
```

এই কোডগুলির মাধ্যমে আপনার হাসপাতাল ব্যবস্থাপনা সিস্টেমের টেবিলগুলি, ট্রিগার এবং স্টোর প্রসিডিউর সম্পূর্ণ করা হয়েছে। যদি আপনার আরও কিছু প্রয়োজন হয়, দয়া করে জানান!

---

নিচে হাসপাতাল ব্যবস্থাপনা সিস্টেমের টেবিল, ট্রিগার এবং স্টোর প্রসিডিউর সম্পূর্ণ করার জন্য পাসওয়ার্ড এবং অন্যান্য চেক কন্ডিশনসহ কোড দেওয়া হলো:

### **টেবিল তৈরির কোড:**

```sql
CREATE TABLE Administration (
    AdminID INT PRIMARY KEY,
    FirstName VARCHAR(255) NOT NULL,
```

```
    LastName VARCHAR(255) NOT NULL,
    Phone VARCHAR(20),
    Email VARCHAR(255),
    Address VARCHAR(255),
    Role VARCHAR(50) NOT NULL
);

CREATE TABLE Users (
    UserID INT PRIMARY KEY,
    UserName VARCHAR(255) NOT NULL UNIQUE,
    Password VARCHAR(255) NOT NULL,
    Role VARCHAR(50) NOT NULL,
    FirstName VARCHAR(255),
    LastName VARCHAR(255),
    Phone VARCHAR(20),
    Email VARCHAR(255) UNIQUE,
    Address VARCHAR(255)
);

CREATE TABLE Roles (
    RoleID INT PRIMARY KEY,
    RoleName VARCHAR(50) NOT NULL,
    Description VARCHAR(255)
);

CREATE TABLE Permissions (
    PermissionID INT PRIMARY KEY,
    RoleID INT,
    PermissionName VARCHAR(255) NOT NULL,
    FOREIGN KEY (RoleID) REFERENCES Roles(RoleID)
);

CREATE TABLE UserRoles (
    UserRoleID INT PRIMARY KEY,
    UserID INT,
```

```
    RoleID INT,
    AdminID INT,
    FOREIGN KEY (UserID) REFERENCES Users(UserID),
    FOREIGN KEY (RoleID) REFERENCES Roles(RoleID),
    FOREIGN KEY (AdminID) REFERENCES Administration(AdminID)
);

CREATE TABLE Patients (
    PatientID INT PRIMARY KEY,
    FirstName VARCHAR(255),
    LastName VARCHAR(255),
    DateOfBirth DATE,
    Gender VARCHAR(10),
    Address VARCHAR(255),
    Phone VARCHAR(20),
    Email VARCHAR(255),
    EmergencyContactName VARCHAR(255),
    EmergencyContactPhone VARCHAR(20),
    InsuranceDetails VARCHAR(255)
);

CREATE TABLE Doctors (
    DoctorID INT PRIMARY KEY,
    FirstName VARCHAR(255),
    LastName VARCHAR(255),
    Specialization VARCHAR(255),
    Phone VARCHAR(20),
    Email VARCHAR(255),
    Address VARCHAR(255),
    JoiningDate DATE,
    Salary DECIMAL(10, 2)
);

CREATE TABLE Staff (
    StaffID INT PRIMARY KEY,
```

```
    FirstName VARCHAR(255),
    LastName VARCHAR(255),
    Position VARCHAR(255),
    Phone VARCHAR(20),
    Email VARCHAR(255),
    Address VARCHAR(255),
    Salary DECIMAL(10, 2),
    JoiningDate DATE
);

CREATE TABLE Appointments (
    AppointmentID INT PRIMARY KEY,
    PatientID INT,
    DoctorID INT,
    AppointmentDate DATE,
    AppointmentTime TIME,
    ReasonForVisit VARCHAR(255),
    Status VARCHAR(50),
    FOREIGN KEY (PatientID) REFERENCES Patients(PatientID),
    FOREIGN KEY (DoctorID) REFERENCES Doctors(DoctorID)
);

CREATE TABLE MedicalRecords (
    RecordID INT PRIMARY KEY,
    PatientID INT,
    DoctorID INT,
    Diagnosis VARCHAR(255),
    Treatment VARCHAR(255),
    Prescription VARCHAR(255),
    DateOfRecord DATE,
    FOREIGN KEY (PatientID) REFERENCES Patients(PatientID),
    FOREIGN KEY (DoctorID) REFERENCES Doctors(DoctorID)
);

CREATE TABLE Billing (
```

```
    BillID INT PRIMARY KEY,
    PatientID INT,
    BillDate DATE,
    TotalAmount DECIMAL(10, 2),
    PaidAmount DECIMAL(10, 2),
    DueAmount DECIMAL(10, 2),
    PaymentMethod VARCHAR(50),
    PaymentStatus VARCHAR(50),
    FOREIGN KEY (PatientID) REFERENCES Patients(PatientID)
);

CREATE TABLE Inventory (
    InventoryID INT PRIMARY KEY,
    ItemName VARCHAR(255),
    Quantity INT,
    Supplier VARCHAR(255),
    PurchaseDate DATE,
    ExpiryDate DATE,
    Cost DECIMAL(10, 2)
);
```

### **ট্রিগার কোড:**

**Patients (রোগী) টেবিলের জন্য ট্রিগার:**

```sql
CREATE TRIGGER TRG_AFTER_INSERT_PATIENT
ON Patients
AFTER INSERT
AS
BEGIN
    DECLARE @PatientID INT;
    SELECT @PatientID = inserted.PatientID FROM inserted;
```

```
    -- Example: Log the patient insertion or send a welcome email
    PRINT 'New patient added with ID: ' + CAST(@PatientID AS VARCHAR);
END;
```

**Appointments (নিয়োগ) টেবিলের জন্য ট্রিগার:**

```sql
CREATE TRIGGER TRG_BEFORE_INSERT_APPOINTMENT
ON Appointments
INSTEAD OF INSERT
AS
BEGIN
    DECLARE @AppointmentDate DATE;
    DECLARE @AppointmentTime TIME;
    DECLARE @DoctorID INT;

    SELECT @AppointmentDate = inserted.AppointmentDate,
           @AppointmentTime = inserted.AppointmentTime,
           @DoctorID = inserted.DoctorID
    FROM inserted;

    IF EXISTS (
        SELECT 1
        FROM Appointments
        WHERE AppointmentDate = @AppointmentDate
          AND AppointmentTime = @AppointmentTime
          AND DoctorID = @DoctorID
    )
    BEGIN
        RAISERROR ('The doctor already has an appointment at this time.', 16, 1);
        ROLLBACK TRANSACTION;
    END
    ELSE
    BEGIN
```

```
    INSERT INTO Appointments (PatientID, DoctorID, AppointmentDate, AppointmentTime,
ReasonForVisit, Status)
    SELECT PatientID, DoctorID, AppointmentDate, AppointmentTime, ReasonForVisit, Status
    FROM inserted;
  END;
END;
```

**Billing (বিলিং) টেবিলের জন্য ট্রিগার:**

```sql
CREATE TRIGGER TRG_AFTER_UPDATE_BILL
ON Billing
AFTER UPDATE
AS
BEGIN
  DECLARE @BillID INT;
  DECLARE @PatientID INT;
  DECLARE @TotalAmount DECIMAL(10, 2);
  DECLARE @PaidAmount DECIMAL(10, 2);
  DECLARE @DueAmount DECIMAL(10, 2);

  SELECT @BillID = inserted.BillID,
      @PatientID = inserted.PatientID,
      @TotalAmount = inserted.TotalAmount,
      @PaidAmount = inserted.PaidAmount,
      @DueAmount = inserted.DueAmount
  FROM inserted;

  -- Example: Notify the patient or admin about the billing update
  PRINT 'Billing updated for patient ID: ' + CAST(@PatientID AS VARCHAR);
END;
```

### **স্টোর প্রসিডিউর কোড:**

**Users (ব্যবহারকারী) টেবিলের জন্য স্টোর প্রসিডিউর:**

```sql
CREATE PROCEDURE SP_ADD_NEW_USER
    @UserName VARCHAR(255),
    @Password VARCHAR(255),
    @Role VARCHAR(50),
    @FirstName VARCHAR(255),
    @LastName VARCHAR(255),
    @Phone VARCHAR(20),
    @Email VARCHAR(255),
    @Address VARCHAR(255)
AS
BEGIN
    IF EXISTS (SELECT 1 FROM Users WHERE UserName = @UserName)
    BEGIN
        RAISERROR ('UserName already exists.', 16, 1);
        RETURN;
    END

    IF EXISTS (SELECT 1 FROM Users WHERE Email = @Email)
    BEGIN
        RAISERROR ('Email already exists.', 16, 1);
        RETURN;
    END

    IF LEN(@Password) < 8 OR @Password NOT LIKE '%[0-9]%' OR @Password NOT LIKE
'%[A-Za-z]%' OR @Password NOT LIKE '%[@#$%^&*()]%'
    BEGIN
        RAISERROR ('Password must be at least 8 characters long and contain at least one number,
one letter, and one special character.', 16, 1);
        RETURN;
    END
```

```
    INSERT INTO Users (UserName, Password, Role, FirstName, LastName, Phone, Email,
Address)
    VALUES (@UserName, @Password, @Role, @FirstName, @LastName, @Phone, @Email,
@Address);
END;
```

**MedicalRecords (চিকিৎসা রেকর্ড) টেবিলের জন্য স্টোর প্রসিডিউর:**

```sql
CREATE PROCEDURE SP_UPDATE_MEDICAL_RECORD
    @RecordID INT,
    @Diagnosis VARCHAR(255),
    @Treatment VARCHAR(255),
    @Prescription VARCHAR(255),
    @DateOfRecord DATE
AS
BEGIN
    UPDATE MedicalRecords
    SET Diagnosis = @Diagnosis,
        Treatment = @Treatment,
        Prescription = @Prescription,
        DateOfRecord = @DateOfRecord
    WHERE RecordID = @RecordID;
END;
```

**Inventory (ইনভেন্টরি) টেবিলের জন্য স্টোর প্রসিডিউর:**

```sql
CREATE PROCEDURE SP_MANAGE_INVENTORY
    @Action VARCHAR(10),
    @InventoryID INT = NULL,
    @ItemName VARCHAR(255) = NULL,
    @Quantity INT = NULL,
```

```
    @Supplier VARCHAR(255) = NULL,
    @PurchaseDate DATE = NULL,
    @ExpiryDate DATE = NULL,
    @Cost DECIMAL(10, 2) = NULL
AS
BEGIN
    IF @Action = 'ADD'
    BEGIN
        INSERT INTO Inventory (ItemName, Quantity, Supplier, PurchaseDate, ExpiryDate, Cost)
        VALUES (@ItemName, @Quantity, @Supplier, @PurchaseDate, @ExpiryDate, @Cost);
    END
    ELSE IF @Action = 'UPDATE'
    BEGIN
        UPDATE Inventory
        SET ItemName = @ItemName,
            Quantity = @Quantity,
            Supplier = @Supplier,
            PurchaseDate = @PurchaseDate,
            ExpiryDate = @ExpiryDate,



            Cost = @Cost
        WHERE InventoryID = @InventoryID;
    END
    ELSE IF @Action = 'DELETE'
    BEGIN
        DELETE FROM Inventory WHERE InventoryID = @InventoryID;
    END
END;
```

এই কোডগুলির মাধ্যমে আপনার হাসপাতাল ব্যবস্থাপনা সিস্টেমের টেবিলগুলি, ট্রিগার এবং স্টোর প্রসিডিউর সম্পূর্ণ করা হয়েছে। যদি আপনার আরও কিছু প্রয়োজন হয়, দয়া করে জানান!

---

নিচে হাসপাতাল ব্যবস্থাপনা সিস্টেমের প্রতিটি টেবিলের জন্য ইনসার্ট স্টেটমেন্ট কোডগুলি দেওয়া হলো:

### **Administration (প্রশাসন) টেবিল:**
```sql
INSERT INTO Administration (AdminID, FirstName, LastName, Phone, Email, Address, Role)
VALUES (1, 'John', 'Doe', '1234567890', 'john.doe@example.com', '123 Main St', 'Admin');
```

### **Users (ব্যবহারকারী) টেবিল:**
```sql
INSERT INTO Users (UserID, UserName, Password, Role, FirstName, LastName, Phone, Email,
Address)
VALUES (1, 'johndoe', 'P@ssw0rd', 'Doctor', 'John', 'Doe', '1234567890', 'john.doe@example.com',
'123 Main St');
```

### **Roles (ভূমিকা) টেবিল:**
```sql
INSERT INTO Roles (RoleID, RoleName, Description)
VALUES (1, 'Doctor', 'Medical Doctor');
```

### **Permissions (অনুমতি) টেবিল:**
```sql
INSERT INTO Permissions (PermissionID, RoleID, PermissionName)
VALUES (1, 1, 'View Patient Records');
```

### **UserRoles (ব্যবহারকারীর ভূমিকা) টেবিল:**
```sql
INSERT INTO UserRoles (UserRoleID, UserID, RoleID, AdminID)
VALUES (1, 1, 1, 1);
```

### **Patients (রোগী) টেবিল:**

```sql
INSERT INTO Patients (PatientID, FirstName, LastName, DateOfBirth, Gender, Address, Phone, Email, EmergencyContactName, EmergencyContactPhone, InsuranceDetails)
VALUES (1, 'Jane', 'Doe', '1990-01-01', 'Female', '456 Elm St', '0987654321', 'jane.doe@example.com', 'John Doe', '1234567890', 'Ins123');
```

### **Doctors (ডাক্তার) টেবিল:**
```sql
INSERT INTO Doctors (DoctorID, FirstName, LastName, Specialization, Phone, Email, Address, JoiningDate, Salary)
VALUES (1, 'John', 'Doe', 'Cardiology', '1234567890', 'john.doe@example.com', '123 Main St', '2020-01-01', 100000);
```

### **Staff (স্টাফ) টেবিল:**
```sql
INSERT INTO Staff (StaffID, FirstName, LastName, Position, Phone, Email, Address, Salary, JoiningDate)
VALUES (1, 'Mary', 'Smith', 'Nurse', '1234567890', 'mary.smith@example.com', '789 Pine St', 50000, '2019-01-01');
```

### **Appointments (নিযুক্তি) টেবিল:**
```sql
INSERT INTO Appointments (AppointmentID, PatientID, DoctorID, AppointmentDate, AppointmentTime, ReasonForVisit, Status)
VALUES (1, 1, 1, '2024-08-01', '09:00:00', 'Regular Checkup', 'Scheduled');
```

### **MedicalRecords (চিকিৎসা রেকর্ড) টেবিল:**
```sql
INSERT INTO MedicalRecords (RecordID, PatientID, DoctorID, Diagnosis, Treatment, Prescription, DateOfRecord)
VALUES (1, 1, 1, 'Hypertension', 'Medication', 'DrugA 50mg', '2024-08-01');
```

```

### **Billing (বিলিং) টেবিল:**
```sql
INSERT INTO Billing (BillID, PatientID, BillDate, TotalAmount, PaidAmount, DueAmount, PaymentMethod, PaymentStatus)
VALUES (1, 1, '2024-08-01', 200.00, 200.00, 0.00, 'Credit Card', 'Paid');
```

### **Inventory (ইনভেন্টরি) টেবিল:**
```sql
INSERT INTO Inventory (InventoryID, ItemName, Quantity, Supplier, PurchaseDate, ExpiryDate, Cost)
VALUES (1, 'Bandages', 100, 'SupplierA', '2024-01-01', '2025-01-01', 50.00);
```

এই ইনসার্ট স্টেটমেন্টগুলি হাসপাতাল ব্যবস্থাপনা সিস্টেমের প্রতিটি টেবিলে নমুনা ডেটা ইনসার্ট করবে। যদি আরও কিছু প্রয়োজন হয়, দয়া করে জানান!
```