Clinic ER

Abdelrahman Tageldin

AA2200780

Introduction:

This a database system is used to manage a clinic that has medical staff, patients and appointments.

Each entity stores important information like medical ID and SSN to be able to uniquely identify each individual and who attends a specific appointment.

An app has been made to be used by the front desk of a clinc.

The app currently is made using C# and Microsoft SQL server and the connection is made using SQL Server Connection String. The app has 3 section one section to run analytics and queries, section 2 to insert, edit and remove information from the database and section to book appointments.

Requirement specifications

1. **Medical Staff Management**
   * The system must store data about medical staff members, including their Social Security Number (SSN), name, phone number, address, date of birth (DOB), and gender.
   * The system must provide different categories for medical staff, such as doctors and nurses, with separate tables.
2. **Doctor and Nurse Management**
   * The system must maintain information about doctors and nurses, including a unique identifier (ID) and their corresponding SSN (linked to the medical staff table).
   * Doctors should have specializations, stored in a separate table and linked to the doctors.
3. **Patient Management**
   * The system must manage patient data, including a unique identifier (Patient ID), name, SSN, DOB, and gender.
   * Relationships between patients and other entities such as appointments, prescriptions, and medical records should be established.
4. **Appointment Management**
   * The system must support the scheduling and management of appointments.
   * Appointments should include an Appointment ID, Patient ID, Doctor ID, and attendance status (e.g., present, absent).
   * Each appointment is uniquely identified by both appointment id and patient id together.as it depends on on the patient.
5. **Prescription Management**
   * The system must support prescription management, linking prescriptions to appointments and patients.
   * Prescription data should include a Prescription ID and links to the Appointment ID and Patient ID.
   * Drugs prescribed should be recorded in a separate table linked to prescriptions.
6. **Medical Records Management**
   * The system must manage medical records, including a unique identifier (Record ID), diagnosis information, and the associated Patient ID.

ER Diagram

A diagram of a company

Description automatically generated

Relation schema:

A diagram of a company

Description automatically generated

BCNF:

The prescriptions table was decomposed into two tables: prescriptions and prescription\_Drugs

Complex operations:

1.check how many visits a patient has booked with a doctor with a specific specialization.

2. display the total number of prescriptions issued by each doctor and list each drug prescribed

3. update the attendance status of an appointment

Aggregate functions:

1. list all appointments for a specific doctor
2. find the maximum number of appointments attended by a patient
3. count the number of appointments made by each doctor and display the doctor's name and specialization

NON-Exist query: Query to find patients who have never had an appointment with any doctor

A screenshot of a computer

Description automatically generated

SQL full code:

CREATE TABLE Medical\_staff

(

SSN VARCHAR(11) NOT NULL,

Name VARCHAR(100) NOT NULL,

Phone\_Number VARCHAR(15) NOT NULL,

Address VARCHAR(255) NOT NULL,

DOB DATE NOT NULL,

Gender CHAR(1) NOT NULL,

PRIMARY KEY (SSN)

);

CREATE TABLE Doctor

(

Doctor\_ID INT NOT NULL,

SSN VARCHAR(11) NOT NULL,

PRIMARY KEY (Doctor\_ID),

FOREIGN KEY (SSN) REFERENCES Medical\_staff(SSN)

);

CREATE TABLE Nurse

(

Nurse\_ID INT NOT NULL,

SSN VARCHAR(11) NOT NULL,

PRIMARY KEY (Nurse\_ID),

FOREIGN KEY (SSN) REFERENCES Medical\_staff(SSN)

);

CREATE TABLE Patients

(

Name VARCHAR(100) NOT NULL,

SSN VARCHAR(11) NOT NULL,

Patient\_ID INT NOT NULL,

DOB DATE NOT NULL,

Gender CHAR(1) NOT NULL,

PRIMARY KEY (Patient\_ID),

UNIQUE (SSN)

);

CREATE TABLE Appointments

(

Attendacne\_status VARCHAR(20) NOT NULL,

Appointment\_ID INT NOT NULL,

Patient\_ID INT NOT NULL,

Doctor\_ID INT NOT NULL,

PRIMARY KEY (Appointment\_ID, Patient\_ID),

FOREIGN KEY (Patient\_ID) REFERENCES Patients(Patient\_ID),

FOREIGN KEY (Doctor\_ID) REFERENCES Doctor(Doctor\_ID)

);

CREATE TABLE Prescription

(

Prescription\_ID INT NOT NULL,

Appointment\_ID INT NOT NULL,

Patient\_ID INT NOT NULL,

PRIMARY KEY (Prescription\_ID),

FOREIGN KEY (Appointment\_ID, Patient\_ID) REFERENCES Appointments(Appointment\_ID, Patient\_ID)

);

CREATE TABLE Assists

(

Appointment\_ID INT NOT NULL,

Patient\_ID INT NOT NULL,

Nurse\_ID INT NOT NULL,

PRIMARY KEY (Appointment\_ID, Patient\_ID),

FOREIGN KEY (Appointment\_ID, Patient\_ID) REFERENCES Appointments(Appointment\_ID, Patient\_ID),

FOREIGN KEY (Nurse\_ID) REFERENCES Nurse(Nurse\_ID)

);

CREATE TABLE Doctor\_specialization

(

Doctor\_specialization VARCHAR(100) NOT NULL,

Doctor\_ID INT NOT NULL,

PRIMARY KEY (Doctor\_specialization),

FOREIGN KEY (Doctor\_ID) REFERENCES Doctor(Doctor\_ID)

);

CREATE TABLE Prescription\_Drugs

(

Drugs VARCHAR(100) NOT NULL,

Prescription\_ID INT NOT NULL,

PRIMARY KEY (Drugs, Prescription\_ID),

FOREIGN KEY (Prescription\_ID) REFERENCES Prescription(Prescription\_ID)

);

CREATE TABLE Medical\_records

(

Record\_ID INT NOT NULL,

Diagoses VARCHAR(255) NOT NULL,

Patient\_ID INT NOT NULL,

PRIMARY KEY (Record\_ID),

FOREIGN KEY (Patient\_ID) REFERENCES Patients(Patient\_ID)

);

-- MedicalStaff table

INSERT INTO Medical\_Staff (SSN,Name, Phone\_Number, Address, DOB,Gender)

VALUES

('123-45-6789', 'Dr. John Doe', '123-456-7890', '123 Main St', '1975-05-15', 'M'),

('234-56-7890', 'Dr. Jane Smith', '123-456-7891', '456 Elm St', '1980-06-20', 'F'),

('345-67-8901', 'Nurse Anne Lee', '123-456-7892', '789 Pine St', '1985-07-25', 'F'),

('456-78-9012', 'Nurse Mark White', '123-456-7893', '321 Oak St', '1990-08-30', 'M'),

('567-89-0123', 'Dr. Sarah Green', '123-456-7894', '654 Maple St', '1972-09-15', 'F'),

('678-90-1234', 'Nurse David Black', '123-456-7895', '987 Birch St', '1988-10-05', 'M'),

('789-01-2345', 'Dr. Emily Brown', '123-456-7896', '112 Cedar St', '1976-11-10', 'F'),

('890-12-3456', 'Nurse Laura Adams', '123-456-7897', '345 Ash St', '1991-12-15', 'F'),

('901-23-4567', 'Dr. Michael Johnson', '123-456-7898', '678 Walnut St', '1973-01-20', 'M'),

('012-34-5678', 'Nurse Sophia King', '123-456-7899', '789 Willow St', '1992-02-25', 'F');

-- Doctors table

INSERT INTO Doctor (Doctor\_ID, SSN)

VALUES

(1, '123-45-6789'),

(2, '234-56-7890'),

(3, '567-89-0123'),

(4, '789-01-2345'),

(5, '901-23-4567');

-- Nurses table

INSERT INTO Nurse (Nurse\_ID, SSN)

VALUES

(1, '345-67-8901'),

(2, '456-78-9012'),

(3, '678-90-1234'),

(4, '890-12-3456'),

(5, '012-34-5678');

-- Patients table

INSERT INTO Patients (Patient\_ID,Name, SSN, DOB,Gender)

VALUES

(1, 'Alice Jones', '111-22-3333', '1995-03-15', 'F'),

(2, 'Bob Williams', '222-33-4444', '1990-07-20', 'M'),

(3, 'Carol Martinez', '333-44-5555', '1985-11-25', 'F'),

(4, 'Daniel Lee', '444-55-6666', '1979-06-10', 'M'),

(5, 'Emily Green', '555-66-7777', '1993-08-05', 'F'),

(6, 'Frank Harris', '666-77-8888', '1982-09-30', 'M'),

(7, 'Grace Young', '777-88-9999', '1997-01-15', 'F'),

(8, 'Henry Wilson', '888-99-0000', '1988-04-25', 'M'),

(9, 'Ivy Robinson', '999-00-1111', '1991-05-10', 'F'),

(10, 'Jack Campbell', '000-11-2222', '1980-02-18', 'M'),

(11, 'Abdelrahman Tageldin', '512-21-6452', '2003-9-11', 'M');

-- Appointments table

INSERT INTO Appointments (Appointment\_ID, Patient\_ID, Doctor\_ID, Attendacne\_status)

VALUES

(1, 1, 1, 'Present'),

(2, 2, 2, 'Absent'),

(3, 3, 3, 'Present'),

(4, 4, 4, 'Absent'),

(5, 5, 5, 'Present'),

(6, 6, 2, 'Absent'),

(7, 7, 3, 'Present'),

(8, 8, 4, 'Absent'),

(9, 9, 5, 'Present'),

(10, 10, 1, 'Absent'),

(11, 1, 2, 'Present'),

(12, 2, 3, 'Absent'),

(13, 3, 4, 'Present'),

(14, 4, 5, 'Absent'),

(15, 5, 1, 'Present'),

(16, 6, 2, 'Absent'),

(17, 7, 3, 'Present'),

(18, 8, 4, 'Absent'),

(19, 9, 5, 'Present'),

(20, 10, 1, 'Absent');

-- Prescriptions table

INSERT INTO Prescription (Prescription\_ID, Appointment\_ID, Patient\_ID)

VALUES

(1, 1, 1),

(2, 2, 2),

(3, 3, 3),

(4, 4, 4),

(5, 5, 5),

(6, 6, 6),

(7, 7, 7),

(8, 8, 8),

(9, 9, 9),

(10, 10, 10),

(11, 11, 1),

(12, 12, 2),

(13, 13, 3),

(14, 14, 4),

(15, 15, 5),

(16, 16, 6),

(17, 17, 7),

(18, 18, 8),

(19, 19, 9),

(20, 20, 10);

-- Assists table

INSERT INTO Assists (Appointment\_ID, Patient\_ID, Nurse\_ID)

VALUES

(1, 1, 1),

(2, 2, 2),

(3, 3, 3),

(4, 4, 4),

(5, 5, 5),

(6, 6, 2),

(7, 7, 3),

(8, 8, 4),

(9, 9, 5),

(10, 10, 1),

(11, 1, 2),

(12, 2, 3),

(13, 3, 4),

(14, 4, 5),

(15, 5, 1),

(16, 6, 2),

(17, 7, 3),

(18, 8, 4),

(19, 9, 5),

(20, 10, 1);

-- DoctorSpecializations table

INSERT INTO Doctor\_specialization (Doctor\_ID, Doctor\_Specialization)

VALUES

(1, 'Cardiology'),

(2, 'Neurology'),

(3, 'Orthopedics'),

(4, 'Pediatrics'),

(5, 'Dermatology');

-- PrescriptionDrugs table

INSERT INTO Prescription\_Drugs (Drugs, Prescription\_ID)

VALUES

('Aspirin', 1),

('Metformin', 2),

('Amoxicillin', 3),

('Lisinopril', 4),

('Ibuprofen', 5),

('Acetaminophen', 6),

('Omeprazole', 7),

('Atorvastatin', 8),

('Loratadine', 9),

('Sertraline', 10),

('Albuterol', 11),

('Levothyroxine', 12),

('Ciprofloxacin', 13),

('Loperamide', 14),

('Cetirizine', 15),

('Sertraline', 16),

('Melatonin', 17),

('Gluten-Free Supplements', 18),

('Latanoprost', 19),

('Lithium', 20);

-- MedicalRecords table

INSERT INTO Medical\_Records (Record\_ID, Diagoses, Patient\_ID)

VALUES

(1, 'Hypertension', 1),

(2, 'Diabetes', 2),

(3, 'Bacterial Infection', 3),

(4, 'Hypertension', 4),

(5, 'Allergy', 5),

(6, 'Migraine', 6),

(7, 'Asthma', 7),

(8, 'Arthritis', 8),

(9, 'Hay Fever', 9),

(10, 'Depression', 10),

(11, 'Pneumonia', 1),

(12, 'Hypothyroidism', 2),

(13, 'Bronchitis', 3),

(14, 'Anxiety', 4),

(15, 'Fibromyalgia', 5),

(16, 'Sinusitis', 6),

(17, 'Conjunctivitis', 7),

(18, 'Hemorrhoids', 8),

(19, 'Vertigo', 9),

(20, 'Epilepsy', 10);

-- complex operations query 1

-- This query counts the number of visits a specific patient (patient ID = 2)

-- has booked with doctors who have a specific specialization ('Cardiology').

SELECT COUNT(\*) AS Num\_Visits

FROM Appointments AS A

-- Join the Appointments table with the Doctor table

JOIN Doctor AS D ON A.Doctor\_ID = D.Doctor\_ID

-- Join the Doctor table with the Doctor\_Specialization table

JOIN Doctor\_specialization AS DS ON D.Doctor\_ID = DS.Doctor\_ID

WHERE

-- Filter by the patient ID (in this case, 2)

A.Patient\_ID = 2

-- Filter by the doctor's specialization (in this case, 'Cardiology')

AND DS.Doctor\_specialization = 'Cardiology';

--Complex opeation 2 Query to display the total number of prescriptions issued by each doctor and list each drug prescribed

SELECT d.Doctor\_ID AS Doctor\_ID, ms.Name AS Doctor\_Name, pd.Drugs AS Drug, COUNT(p.Prescription\_ID) AS Total\_Prescriptions

FROM Prescription AS p

JOIN Appointments AS a ON p.Appointment\_ID = a.Appointment\_ID AND p.Patient\_ID = a.Patient\_ID

JOIN Doctor AS d ON a.Doctor\_ID = d.Doctor\_ID

JOIN Medical\_staff AS ms ON d.SSN = ms.SSN

JOIN Prescription\_Drugs AS pd ON p.Prescription\_ID = pd.Prescription\_ID

GROUP BY d.Doctor\_ID, ms.Name, pd.Drugs;

-- Complex opertaion 3 update the attendance status of an appointment

UPDATE Appointments

SET Attendacne\_status = 'Absent' -- Replace with the new attendance status you want to set

WHERE Appointment\_ID = 2 -- Replace with the ID of the appointment you want to update

AND Patient\_ID = 2; -- Replace with the ID of the patient for the appointment

--aggregite function 1 Query to list all appointments for a specific doctor

SELECT a.Appointment\_ID AS Appointment\_ID,

p.Name AS Patient\_Name,

a.Attendacne\_status AS Attendance\_Status

FROM Appointments AS a

JOIN Patients AS p ON a.Patient\_ID = p.Patient\_ID

WHERE a.Doctor\_ID = 2 -- Replace with the specific doctor's ID

ORDER BY a.Appointment\_ID;

--aggregete Function 2 to find the maximum number of appointments attended by a patient

SELECT p.Name AS Patient\_Name, p.Patient\_ID AS Patient\_ID, MAX(a.Appointment\_Count) AS Max\_Appointments\_Attended

FROM (

SELECT Patient\_ID, COUNT(\*) AS Appointment\_Count

FROM Appointments

WHERE Attendacne\_status = 'Present' -- Filter to include only attended appointments

GROUP BY Patient\_ID

) AS a

JOIN Patients AS p ON a.Patient\_ID = p.Patient\_ID

GROUP BY p.Name, p.Patient\_ID;

-- Aggregite function 3 to count the number of appointments made by each doctor and display the doctor's name and specialization

SELECT ms.Name AS Doctor\_Name, ds.Doctor\_specialization AS Specialization, COUNT(a.Appointment\_ID) AS Appointment\_Count

FROM Appointments AS a

JOIN Doctor AS d ON a.Doctor\_ID = d.Doctor\_ID

JOIN Medical\_staff AS ms ON d.SSN = ms.SSN

JOIN Doctor\_specialization AS ds ON d.Doctor\_ID = ds.Doctor\_ID

GROUP BY ms.Name, ds.Doctor\_specialization;

--NON Exist Query to find patients who have never had an appointment with any doctor

SELECT p.Name AS Patient\_Name, p.Patient\_ID AS Patient\_ID

FROM Patients AS p

WHERE NOT EXISTS (

SELECT 1

FROM Appointments AS a

WHERE a.Patient\_ID = p.Patient\_ID

);

Tables Data:

Appointments

A screenshot of a computer

Description automatically generated

Medical staff

A table with numbers and names

Description automatically generated

Doctors

A screenshot of a computer

Description automatically generated

Doctor\_spezcialtion

A screenshot of a medical form

Description automatically generated

Nurses

A screenshot of a computer

Description automatically generated

Assists

A white sheet with numbers

Description automatically generated

Medical\_records

A screenshot of a computer screen

Description automatically generated

Patients

A screenshot of a computer

Description automatically generated

Prescriptions

A screenshot of a white sheet with numbers

Description automatically generated

Prescriptions\_Drugs

A screenshot of a computer

Description automatically generated

Measures to handle errors:

Try and catch:

* The try block contains the code that could potentially throw an exception.
* The catch block catches any exceptions that occur within the try block and handles them.

Message Box Display:

* + Inside the catch block, a Message Box is used to display the error message to the user.
  + This ensures that if an error occurs during the insertion of appointment data into the database, the user is informed about the error.

SQL Injection protections is ensured using A parameterized query is a query in which placeholders are used for parameters and the parameter values are supplied at execution time.