create database Поликлиника

use Поликлиника

-- Создание таблицы "Пациенты"

CREATE TABLE Пациенты (

id\_пациента INT PRIMARY KEY IDENTITY(1,1),

ФИО VARCHAR(100) NOT NULL,

Дата\_рождения DATE,

Адрес VARCHAR(100),

Телефон VARCHAR(15),

Email VARCHAR(50)

);

-- Создание таблицы "Врачи"

CREATE TABLE Врачи (

id\_врача INT PRIMARY KEY IDENTITY(1,1),

ФИО VARCHAR(100) NOT NULL,

Специальность VARCHAR(100) NOT NULL,

Квалификация VARCHAR(50),

Кабинет VARCHAR(10),

Телефон VARCHAR(15),

Email VARCHAR(50)

);

-- Создание таблицы "Поликлинические карты"

CREATE TABLE Поликлинические\_карты (

id\_карты INT PRIMARY KEY IDENTITY(1,1),

id\_пациента INT,

id\_врача INT,

Дата\_поступления DATE,

Диагноз TEXT,

FOREIGN KEY (id\_пациента) REFERENCES Пациенты (id\_пациента),

FOREIGN KEY (id\_врача) REFERENCES Врачи (id\_врача)

);

-- Создание таблицы "Приемы"

CREATE TABLE Приемы (

id\_приема INT PRIMARY KEY IDENTITY(1,1),

id\_пациента INT,

id\_врача INT,

Дата DATE,

Время TIME,

Описание TEXT,

FOREIGN KEY (id\_пациента) REFERENCES Пациенты (id\_пациента),

FOREIGN KEY (id\_врача) REFERENCES Врачи (id\_врача)

);

-- Вставка данных в таблицу "Пациенты"

INSERT INTO Пациенты (ФИО, Дата\_рождения, Адрес, Телефон, Email)

VALUES

('Иванов Иван Иванович', '1990-05-15', 'ул. Центральная 10', '1234567890', 'ivanov@mail.com'),

('Петрова Анна Сергеевна', '1985-08-22', 'ул. Солнечная 5', '9876543210', 'petrova@mail.com'),

('Смирнов Сергей Викторович', '1978-11-30', 'ул. Пушкина 15', '5555555555', 'smirnov@mail.com');

-- Вставка данных в таблицу "Врачи"

INSERT INTO Врачи (ФИО, Специальность, Квалификация, Кабинет, Телефон, Email)

VALUES

('Козлова Елена Александровна', 'Стоматолог', 'Высшая', 'Кабинет 20', '1111111111', 'kozlova@mail.com'),

('Сидоров Игорь Петрович', 'Офтальмолог', 'Высшая', 'Кабинет 30', '2222222222', 'sidorov@mail.com'),

('Васильева Мария Николаевна', 'Педиатр', 'Высшая', 'Кабинет 10', '3333333333', 'vasilieva@mail.com');

-- Вставка данных в таблицу "Поликлинические\_карты"

INSERT INTO Поликлинические\_карты (id\_пациента, id\_врача, Дата\_поступления, Диагноз)

VALUES

(1, 1, '2022-01-10', 'ОРЗ'),

(2, 2, '2022-02-05', 'Кариес'),

(3, 3, '2022-03-15', 'Ангина');

-- Вставка данных в таблицу "Приемы"

INSERT INTO Приемы (id\_пациента, id\_врача, Дата, Время, Описание)

VALUES

(1, 1, '2022-01-15', '10:00:00', 'Осмотр и назначение лечения'),

(2, 2, '2022-02-10', '14:30:00', 'Плановый осмотр'),

(3, 3, '2022-03-20', '09:45:00', 'Прививка');

-- Представление "Список врачей"

CREATE VIEW Список\_врачей

AS

SELECT id\_врача, ФИО, Специальность, Квалификация, Кабинет, Телефон, Email

FROM Врачи;

-- Представление "Список поликлинических карт"

CREATE VIEW Список\_поликлинических\_карт

AS

SELECT Поликлинические\_карты.id\_пациента, Пациенты.ФИО AS ФИО\_пациента, Поликлинические\_карты.id\_врача, Врачи.ФИО AS ФИО\_врача, Дата\_поступления, Диагноз

FROM Поликлинические\_карты

JOIN Пациенты ON Поликлинические\_карты.id\_пациента = Пациенты.id\_пациента

JOIN Врачи ON Поликлинические\_карты.id\_врача = Врачи.id\_врача;

-- Представление "Список приемов"

CREATE VIEW Список\_приемов

AS

SELECT Приемы.id\_пациента, Пациенты.ФИО AS ФИО\_пациента, Приемы.id\_врача, Врачи.ФИО AS ФИО\_врача, Дата, Время, Описание

FROM Приемы

JOIN Пациенты ON Приемы.id\_пациента = Пациенты.id\_пациента

JOIN Врачи ON Приемы.id\_врача = Врачи.id\_врача;

-- Хранимая процедура "Добавить пациента"

CREATE PROCEDURE Добавить\_пациента

@ФИО VARCHAR(100),

@Дата\_рождения DATE,

@Адрес VARCHAR(100),

@Телефон VARCHAR(15),

@Email VARCHAR(50)

AS

BEGIN

INSERT INTO Пациенты (ФИО, Дата\_рождения, Адрес, Телефон, Email)

VALUES (@ФИО, @Дата\_рождения, @Адрес, @Телефон, @Email)

END

-- Хранимая процедура "Обновить диагноз в поликлинической карте"

CREATE PROCEDURE Обновить\_диагноз\_в\_поликлинической\_карте

@id\_пациента INT,

@id\_врача INT,

@Диагноз VARCHAR(100)

AS

BEGIN

UPDATE Поликлинические\_карты

SET Диагноз = @Диагноз

WHERE id\_пациента = @id\_пациента AND id\_врача = @id\_врача

END

-- Хранимая процедура "Удалить пациента"

CREATE PROCEDURE Удалить\_пациента

@id\_пациента INT

AS

BEGIN

DELETE FROM Пациенты WHERE id\_пациента = @id\_пациента

END

-- Триггер для проверки возраста пациента при добавлении нового приема

CREATE TRIGGER Проверить\_Возраст\_Пациента

ON Приемы

AFTER INSERT

AS

BEGIN

DECLARE @id\_пациента INT

DECLARE @Дата\_приема DATE

SELECT @id\_пациента = id\_пациента, @Дата\_приема = Дата FROM inserted

DECLARE @Возраст INT

SELECT @Возраст = DATEDIFF(YEAR, Дата\_рождения, @Дата\_приема) FROM Пациенты WHERE id\_пациента = @id\_пациента

IF @Возраст < 18

BEGIN

RAISERROR ('Пациент моложе 18 лет. Прием запрещен.', 16, 1)

ROLLBACK TRANSACTION

END

END

--Триггер для автоматического удаления записей о приемах при удалении пациента:

CREATE TRIGGER Удалить\_Приемы\_При\_Удалении\_Пациента

ON Пациенты

AFTER DELETE

AS

BEGIN

DELETE FROM Приемы

WHERE id\_пациента IN (SELECT id\_пациента FROM deleted)

END

--выводит текст при добавлении нового пациента

CREATE TRIGGER Вывод\_Сообщения

ON Пациенты

AFTER INSERT

AS

BEGIN

PRINT 'Новая запись была добавлена в таблицу.'

END

CREATE FUNCTION ПолучитьКоличествоПациентовПоСпециальности(@специальность VARCHAR(100))

RETURNS INT

AS

BEGIN

DECLARE @количество INT;

SELECT @количество = COUNT(DISTINCT Пациенты.id\_пациента)

FROM Пациенты

INNER JOIN Поликлинические\_карты ON Пациенты.id\_пациента = Поликлинические\_карты.id\_пациента

INNER JOIN Врачи ON Поликлинические\_карты.id\_врача = Врачи.id\_врача

WHERE Врачи.Специальность = @специальность;

RETURN @количество;

END;

CREATE FUNCTION ПолучитьПациентовПоВозрасту(@возрастОт INT, @возрастДо INT)

RETURNS TABLE

AS

RETURN (

SELECT Пациенты.\*

FROM Пациенты

WHERE DATEDIFF(YEAR, Пациенты.Дата\_рождения, GETDATE()) BETWEEN @возрастОт AND @возрастДо

);

CREATE FUNCTION ПолучитьКоличествоПриемовВрача(@id\_врача INT)

RETURNS INT

AS

BEGIN

DECLARE @количество INT;

SELECT @количество = COUNT(id\_приема)

FROM Приемы

WHERE id\_врача = @id\_врача;

RETURN @количество;

END;

1. **Какие современные инструментальные средства проектирования схемы базы данных используют?**

Существует несколько современных инструментальных средств для проектирования схемы базы данных. Вот некоторые из них:

1. ER-диаграммы: ER-диаграммы (диаграммы сущность-связь) являются одним из наиболее распространенных инструментов для моделирования баз данных. Они позволяют визуально представить сущности (таблицы), их атрибуты и связи между ними.

2. UML-диаграммы: Unified Modeling Language (UML) - это стандартный язык моделирования, который также может быть использован для проектирования баз данных. UML-диаграммы классов и диаграммы объектов могут быть применены для описания сущностей и их связей в базе данных.

3. CASE-средства: CASE (Computer-Aided Software Engineering) - это программные средства, которые предоставляют набор инструментов для анализа, проектирования и разработки систем, включая базы данных. Они обычно предлагают функциональности ER-моделирования, автоматической генерации кода, документирования и других задач, связанных с проектированием баз данных.

4. Программное обеспечение для моделирования данных: Существуют специализированные программные инструменты для моделирования данных, такие как MySQL Workbench, Microsoft Visio, Oracle SQL Developer Data Modeler и другие. Они предоставляют возможности для создания ER-диаграмм, генерации сценариев создания таблиц, управления связями и других аспектов проектирования баз данных.

5. Онлайн-сервисы: В последнее время стали популярны онлайн-сервисы, которые позволяют создавать и моделировать базы данных через веб-интерфейс. Примеры таких сервисов включают dbdiagram.io, draw.io и Lucidchart.