-- Создание базы данных "Детский сад"

CREATE DATABASE ДетскийСад;

GO

-- Использование базы данных "Детский сад"

USE ДетскийСад;

GO

-- Создание таблицы "Руководитель"

CREATE TABLE Руководитель (

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

Имя VARCHAR(100) NOT NULL,

Фамилия VARCHAR(100) NOT NULL,

Должность VARCHAR(50) NOT NULL

);

GO

-- Создание таблицы "Группа"

CREATE TABLE Группа (

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

Название VARCHAR(100) NOT NULL,

Возрастная\_категория VARCHAR(50) NOT NULL,

id\_руководителя INT,

FOREIGN KEY (id\_руководителя) REFERENCES Руководитель(id\_руководителя)

);

GO

-- Создание таблицы "Родитель"

CREATE TABLE Родитель (

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

Имя VARCHAR(100) NOT NULL,

Фамилия VARCHAR(100) NOT NULL,

Телефон VARCHAR(20) NOT NULL,

Email VARCHAR(100) NOT NULL

);

GO

-- Создание таблицы "Ребенок"

CREATE TABLE Ребенок (

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

Имя VARCHAR(100) NOT NULL,

Фамилия VARCHAR(100) NOT NULL,

Дата\_рождения DATE NOT NULL,

Пол VARCHAR(10) NOT NULL,

id\_группы INT,

id\_родителя INT,

FOREIGN KEY (id\_группы) REFERENCES Группа(id\_группы),

FOREIGN KEY (id\_родителя) REFERENCES Родитель(id\_родителя)

);

GO

-- Создание таблицы "Расписание"

CREATE TABLE Расписание (

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

id\_группы INT,

День\_недели VARCHAR(20) NOT NULL,

Время\_начала TIME NOT NULL,

Время\_окончания TIME NOT NULL,

FOREIGN KEY (id\_группы) REFERENCES Группа(id\_группы)

);

GO

-- Заполнение таблицы "Руководитель"

INSERT INTO Руководитель (Имя, Фамилия, Должность)

VALUES ('Иван', 'Петров', 'Директор'),

('Анна', 'Сидорова', 'Заведующая'),

('Мария', 'Иванова', 'Заведующая помощник');

-- Заполнение таблицы "Группа"

INSERT INTO Группа (Название, Возрастная\_категория, id\_руководителя)

VALUES ('Малыши', '1-2 года', 1),

('Старшая группа', '3-4 года', 2),

('Подготовительная группа', '5-6 лет', 2),

('Средняя группа', '2-3 года', 3),

('Старшие дети', '4-5 лет', 1);

-- Заполнение таблицы "Родитель"

INSERT INTO Родитель (Имя, Фамилия, Телефон, Email)

VALUES ('Алексей', 'Иванов', '123456789', 'alex@example.com'),

('Елена', 'Смирнова', '987654321', 'elena@example.com'),

('Дмитрий', 'Петров', '555555555', 'dmitry@example.com'),

('Ольга', 'Сидорова', '777777777', 'olga@example.com');

-- Заполнение таблицы "Ребенок"

INSERT INTO Ребенок (Имя, Фамилия, Дата\_рождения, Пол, id\_группы, id\_родителя)

VALUES ('Алиса', 'Иванова', '2019-02-15', 'Женский', 1, 1),

('Максим', 'Смирнов', '2017-09-10', 'Мужской', 2, 2),

('София', 'Петрова', '2016-05-20', 'Женский', 3, 1),

('Артем', 'Сидоров', '2018-11-02', 'Мужской', 4, 3),

('Елизавета', 'Иванова', '2017-07-12', 'Женский', 2, 4),

('Иван', 'Петров', '2020-03-28', 'Мужской', 5, 2);

-- Заполнение таблицы "Расписание"

INSERT INTO Расписание (id\_группы, День\_недели, Время\_начала, Время\_окончания)

VALUES (1, 'Понедельник', '09:00:00', '11:00:00'),

(2, 'Вторник', '10:00:00', '12:00:00'),

(3, 'Среда', '11:00:00', '13:00:00'),

(4, 'Четверг', '09:30:00', '11:30:00'),

(5, 'Пятница', '10:30:00', '12:30:00');

CREATE VIEW СписокДетейИРодителей AS

SELECT Ребенок.Имя AS Имя\_ребенка, Ребенок.Фамилия AS Фамилия\_ребенка,

Родитель.Имя AS Имя\_родителя, Родитель.Фамилия AS Фамилия\_родителя

FROM Ребенок

JOIN Родитель ON Ребенок.id\_родителя = Родитель.id\_родителя;

select \* from СписокДетейИРодителей

CREATE VIEW РасписаниеГруппы AS

SELECT Группа.Название AS Название\_группы, Расписание.День\_недели,

Расписание.Время\_начала, Расписание.Время\_окончания

FROM Группа

JOIN Расписание ON Группа.id\_группы = Расписание.id\_группы;

select \* from РасписаниеГруппы

CREATE VIEW РуководителиИГруппы AS

SELECT Руководитель.Имя, Руководитель.Фамилия, Группа.Название AS Название\_группы

FROM Руководитель

JOIN Группа ON Руководитель.id\_руководителя = Группа.id\_руководителя;

select \* from РуководителиИГруппы

CREATE PROCEDURE ДобавитьРебенка

@Имя VARCHAR(100),

@Фамилия VARCHAR(100),

@Дата\_рождения DATE,

@Пол VARCHAR(10),

@id\_группы INT,

@id\_родителя INT

AS

BEGIN

INSERT INTO Ребенок (Имя, Фамилия, Дата\_рождения, Пол, id\_группы, id\_родителя)

VALUES (@Имя, @Фамилия, @Дата\_рождения, @Пол, @id\_группы, @id\_родителя);

END;

CREATE PROCEDURE ОбновитьИнформациюОРебенке

@id\_ребенка INT,

@Имя VARCHAR(100),

@Фамилия VARCHAR(100),

@Дата\_рождения DATE,

@Пол VARCHAR(10),

@id\_группы INT,

@id\_родителя INT

AS

BEGIN

UPDATE Ребенок

SET Имя = @Имя, Фамилия = @Фамилия, Дата\_рождения = @Дата\_рождения,

Пол = @Пол, id\_группы = @id\_группы, id\_родителя = @id\_родителя

WHERE id\_ребенка = @id\_ребенка;

END;

CREATE PROCEDURE УдалитьРебенка

@id\_ребенка INT

AS

BEGIN

DELETE FROM Ребенок

WHERE id\_ребенка = @id\_ребенка;

END;

CREATE TRIGGER UpdateAgeCategory

ON Ребенок

AFTER UPDATE

AS

BEGIN

UPDATE Группа

SET Возрастная\_категория = (

SELECT Возрастная\_категория

FROM inserted

WHERE inserted.id\_группы = Группа.id\_группы

)

WHERE id\_группы IN (

SELECT id\_группы

FROM inserted

);

END;

-- Добавление столбца "Количество\_детей" в таблицу "Группа"

ALTER TABLE Группа

ADD Количество\_детей INT;

-- Создание триггера "UpdateChildCount" для автоматического обновления количества детей

CREATE TRIGGER UpdateChildCount

ON Ребенок

AFTER INSERT, UPDATE, DELETE

AS

BEGIN

-- Обновление количества детей в таблице "Группа"

UPDATE Группа

SET Количество\_детей = (

SELECT COUNT(\*) FROM Ребенок WHERE Ребенок.id\_группы = Группа.id\_группы

)

WHERE EXISTS (

SELECT 1 FROM inserted WHERE inserted.id\_группы = Группа.id\_группы

) OR EXISTS (

SELECT 1 FROM deleted WHERE deleted.id\_группы = Группа.id\_группы

);

END;

CREATE TRIGGER CascadeDeleteScheduleTrigger

ON Группа

AFTER DELETE

AS

BEGIN

DELETE FROM Расписание

WHERE id\_группы = (SELECT id\_группы FROM deleted);

END;

CREATE TRIGGER NewGroupTrigger

ON Группа

AFTER INSERT

AS

BEGIN

PRINT 'Новая группа добавлена!';

END;

CREATE FUNCTION ПолучитьСписокДетейВГруппе(@id\_группы INT)

RETURNS TABLE

AS

RETURN

(

SELECT Имя, Фамилия, Дата\_рождения

FROM Ребенок

WHERE id\_группы = @id\_группы

);

CREATE FUNCTION ПолучитьСписокРодителейДетей(@id\_группы INT)

RETURNS TABLE

AS

RETURN

(

SELECT Родитель.Имя, Родитель.Фамилия, Родитель.Телефон, Родитель.Email

FROM Родитель

INNER JOIN Ребенок ON Родитель.id\_родителя = Ребенок.id\_родителя

WHERE Ребенок.id\_группы = @id\_группы

);

CREATE FUNCTION ПолучитьРасписаниеГруппы(@id\_группы INT)

RETURNS TABLE

AS

RETURN

(

SELECT День\_недели, Время\_начала, Время\_окончания

FROM Расписание

WHERE id\_группы = @id\_группы

);

1. **Аутентификация и авторизация пользователей. Назначение серверных ролей и ролей баз данных. Авторизация пользователей при получении доступа к ресурсам**.

Аутентификация и авторизация пользователей являются важными механизмами обеспечения безопасности и контроля доступа к ресурсам в системе.

Аутентификация представляет собой процесс проверки подлинности пользователя, то есть установления его идентичности. Обычно это включает в себя предоставление учетных данных (например, логина и пароля) и их проверку на соответствие записям в системе. Целью аутентификации является убедиться, что пользователь является тем, за кого он себя представляет.

Авторизация, с другой стороны, определяет права доступа пользователя к определенным ресурсам или функциональности системы. После успешной аутентификации система определяет, какие привилегии и ограничения присваиваются пользователю. Авторизация гарантирует, что пользователь имеет соответствующие разрешения для выполнения определенных действий в системе.

В контексте баз данных, серверные роли и роли баз данных являются механизмами управления авторизацией и разграничения доступа к данным.

Серверные роли представляют группы пользователей с общими привилегиями на уровне сервера. Например, роль "sysadmin" обладает полными правами администратора сервера, а роль "db\_datareader" имеет право на чтение данных во всех базах данных на сервере.

Роли баз данных, с другой стороны, определяют уровень доступа пользователей к определенным базам данных и их объектам (таблицы, представления, хранимые процедуры и т.д.). Роли баз данных могут предоставлять разрешения на выполнение операций чтения, записи, изменения схемы и т.д.