CREATE TABLE Рецепты (

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

Название\_рецепта VARCHAR(100) NOT NULL,

Описание\_рецепта VARCHAR(MAX) NOT NULL

);

CREATE TABLE Ингредиенты (

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

Название\_ингредиента VARCHAR(100) NOT NULL,

Единица\_измерения VARCHAR(50) NOT NULL

);

CREATE TABLE Рецепты\_Ингредиенты (

id\_рецепта INT,

id\_ингредиента INT,

PRIMARY KEY (id\_рецепта, id\_ингредиента), --для того чтобы пары значений не повторялись

FOREIGN KEY (id\_рецепта) REFERENCES Рецепты (id\_рецепта),

FOREIGN KEY (id\_ингредиента) REFERENCES Ингредиенты (id\_ингредиента)

);

CREATE TABLE Повары (

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

Имя VARCHAR(50) NOT NULL,

Фамилия VARCHAR(50) NOT NULL,

Отчество VARCHAR(50) NOT NULL,

Возраст INT,

Зарплата INT,

id\_рецепта INT,

FOREIGN KEY (id\_рецепта) REFERENCES Рецепты (id\_рецепта)

);

CREATE TABLE Отзывы (

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

id\_рецепта INT,

Отзыв VARCHAR(MAX) NOT NULL,

Оценка INT NOT NULL,

FOREIGN KEY (id\_рецепта) REFERENCES Рецепты (id\_рецепта)

);

-- Заполнение таблицы Рецепты

INSERT INTO Рецепты (Название\_рецепта, Описание\_рецепта)

VALUES

('Борщ', 'Традиционный суп'),

('Пицца', 'Популярное итальянское блюдо'),

('Салат "Цезарь"', 'Классический салат с куриной грудкой и соусом Цезарь');

-- Заполнение таблицы Ингредиенты

INSERT INTO Ингредиенты (Название\_ингредиента, Единица\_измерения)

VALUES

('Картофель', 'г'),

('Свекла', 'г'),

('Морковь', 'г'),

('Лук', 'шт'),

('Чеснок', 'зубчик'),

('Мука', 'г'),

('Дрожжи', 'г'),

('Помидоры', 'шт'),

('Сыр', 'г'),

('Куриная грудка', 'г'),

('Салатный лист', 'г'),

('Сухарики', 'г'),

('Соус Цезарь', 'г');

-- Заполнение таблицы Рецепты\_Ингредиенты

INSERT INTO Рецепты\_Ингредиенты (id\_рецепта, id\_ингредиента)

VALUES

(1, 1), (1, 2), (1, 3), (1, 4), (1, 5),

(2, 6), (2, 7), (2, 8),

(3, 9), (3, 10), (3, 11), (3, 12), (3, 13);

-- Заполнение таблицы Повары

INSERT INTO Повары (Имя, Фамилия, Отчество, Возраст, Зарплата, id\_рецепта)

VALUES

('Иван', 'Иванов', 'Иванович', 35, 50000, 1),

('Петр', 'Петров', 'Петрович', 40, 60000, 2),

('Анна', 'Сидорова', 'Петровна', 28, 45000, 3);

-- Заполнение таблицы Отзывы

INSERT INTO Отзывы (id\_рецепта, Отзыв, Оценка)

VALUES

(1, 'Очень вкусный борщ!', 5),

(2, 'Отличная пицца, рекомендую!', 4),

(3, 'Салат "Цезарь" получился просто великолепным!', 5);

--Представление 1:

CREATE VIEW Рецепты\_Ингредиенты\_Детали AS

SELECT Рецепты.id\_рецепта, Рецепты.Название\_рецепта, Рецепты.Описание\_рецепта, Ингредиенты.id\_ингредиента, Ингредиенты.Название\_ингредиента, Ингредиенты.Единица\_измерения

FROM Рецепты

INNER JOIN Рецепты\_Ингредиенты ON Рецепты.id\_рецепта = Рецепты\_Ингредиенты.id\_рецепта

INNER JOIN Ингредиенты ON Рецепты\_Ингредиенты.id\_ингредиента = Ингредиенты.id\_ингредиента;

--Представление 2:

CREATE VIEW Рецепты\_Повары\_Детали AS

SELECT Рецепты.id\_рецепта, Рецепты.Название\_рецепта, Рецепты.Описание\_рецепта, Повары.id\_повара, Повары.Имя, Повары.Фамилия, Повары.Отчество, Повары.Возраст, Повары.Зарплата

FROM Рецепты

INNER JOIN Повары ON Рецепты.id\_рецепта = Повары.id\_рецепта;

--Представление 3

CREATE VIEW Рецепты\_Отзывы\_Детали AS

SELECT Рецепты.id\_рецепта, Рецепты.Название\_рецепта, Отзывы.id\_отзыва, Отзывы.Отзыв, Отзывы.Оценка

FROM Рецепты

INNER JOIN Отзывы ON Рецепты.id\_рецепта = Отзывы.id\_рецепта;

--Триггер 1:

--Триггер для таблицы "Рецепты", который автоматически устанавливает текущую дату в поле "Дата\_создания" при добавлении нового рецепта:

CREATE TRIGGER Триггер\_Рецепты\_Дата\_создания

ON Рецепты

AFTER INSERT

AS

BEGIN

UPDATE Рецепты

SET Дата\_создания = GETDATE()

WHERE id\_рецепта IN (SELECT id\_рецепта FROM inserted)

END;

--Триггер 2

-- Триггер для таблицы "Рецепты\_Ингредиенты", который запрещает удаление ингредиента, если он используется в каком-либо рецепте:

CREATE TRIGGER Триггер\_Рецепты\_Ингредиенты\_Удаление

ON Рецепты\_Ингредиенты

INSTEAD OF DELETE

AS

BEGIN

IF EXISTS (SELECT 1 FROM deleted d INNER JOIN Рецепты\_Ингредиенты ri ON d.id\_ингредиента = ri.id\_ингредиента)

BEGIN

RAISERROR ('Ингредиент используется в рецепте. Удаление невозможно.', 16, 1)

END

ELSE

BEGIN

DELETE ri

FROM Рецепты\_Ингредиенты ri

INNER JOIN deleted d ON ri.id\_ингредиента = d.id\_ингредиента

END

END;

--Триггер 3

--Триггер для таблицы "Отзывы", который обновляет среднюю оценку рецепта при добавлении или удалении отзыва:

CREATE TRIGGER Триггер\_Отзывы\_Обновление\_Оценки

ON Отзывы

AFTER INSERT, DELETE

AS

BEGIN

UPDATE Рецепты

SET Средняя\_оценка = (

SELECT AVG(Оценка)

FROM Отзывы

WHERE id\_рецепта = Рецепты.id\_рецепта

)

WHERE id\_рецепта IN (

SELECT id\_рецепта

FROM inserted

UNION

SELECT id\_рецепта

FROM deleted

)

END;

--Простые триггеры:

--Триггер 1:

CREATE TRIGGER Триггер\_Рецепты\_Добавление

ON Рецепты

AFTER INSERT

AS

BEGIN

PRINT 'Новый рецепт добавлен.'

END;

--Триггер 2

CREATE TRIGGER Триггер\_Ингредиенты\_Изменение\_Названия

ON Ингредиенты

AFTER UPDATE

AS

BEGIN

PRINT 'Название ингредиента было изменено.'

END;

--Триггер 3

CREATE TRIGGER Триггер\_Повары\_Удаление

ON Повары

AFTER DELETE

AS

BEGIN

PRINT 'Повар был удален из базы данных.'

END;

CREATE FUNCTION ПолучитьКоличествоИнгредиентовВРецепте(@id\_рецепта INT)

RETURNS INT

AS

BEGIN

DECLARE @количество INT;

SELECT @количество = COUNT(\*)

FROM Рецепты\_Ингредиенты

WHERE id\_рецепта = @id\_рецепта;

RETURN @количество;

END;

CREATE FUNCTION ПолучитьСреднийРейтингРецепта(@id\_рецепта INT)

RETURNS DECIMAL(10, 2)

AS

BEGIN

DECLARE @среднийРейтинг DECIMAL(10, 2);

SELECT @среднийРейтинг = AVG(Отзывы.Оценка)

FROM Отзывы

WHERE Отзывы.id\_рецепта = @id\_рецепта;

RETURN @среднийРейтинг;

END;

CREATE FUNCTION ПолучитьРецептыПоПовару(@id\_повара INT)

RETURNS TABLE

AS

RETURN

(

SELECT Рецепты.id\_рецепта, Рецепты.Название\_рецепта, Рецепты.Описание\_рецепта

FROM Рецепты

INNER JOIN Повары ON Рецепты.id\_рецепта = Повары.id\_рецепта

WHERE Повары.id\_повара = @id\_повара

);

--Процедура 1

CREATE PROCEDURE ДобавитьРецепт

@Название VARCHAR(100),

@Описание VARCHAR(MAX)

AS

BEGIN

INSERT INTO Рецепты (Название\_рецепта, Описание\_рецепта)

VALUES (@Название, @Описание)

END

--Процедура 2

CREATE PROCEDURE ДобавитьИнгредиент

@Название VARCHAR(100),

@Единица\_измерения VARCHAR(50)

AS

BEGIN

INSERT INTO Ингредиенты (Название\_ингредиента, Единица\_измерения)

VALUES (@Название, @Единица\_измерения)

END

--Процедура 3

CREATE PROCEDURE ДобавитьОтзыв

@id\_рецепта INT,

@Отзыв VARCHAR(MAX),

@Оценка INT

AS

BEGIN

INSERT INTO Отзывы (id\_рецепта, Отзыв, Оценка)

VALUES (@id\_рецепта, @Отзыв, @Оценка)

END

1. **Определите основные положения теории баз данных, хранилищ данных, баз знаний.**

База данных (БД) - организованная коллекция данных, которая хранится и управляется с использованием специальных программных средств.

Структура данных - описание способа организации данных в базе данных, включая определение таблиц, полей, связей и ограничений целостности.

Модель данных - абстрактное представление данных и их взаимосвязей в базе данных. Примеры моделей данных включают иерархическую, сетевую, реляционную и объектно-ориентированную модели.

Запросы - способы извлечения и модификации данных в базе данных. Запросы обычно формулируются с использованием языков запросов, таких как SQL.

Нормализация - процесс организации данных в базе данных с целью устранения избыточности и обеспечения эффективности работы с данными.

Хранилище данных — это специально организованная база данных, которая используется для агрегации, хранения и анализа больших объемов данных из различных источников. Основные положения хранилища данных включают:

1. Интеграция данных

2. Очистка данных

3. Исторические данные

База знаний — это система, которая хранит факты, правила, схемы и другую информацию, которая используется для поддержки принятия решений и решения проблем в определенной предметной области. Основные положения базы знаний включают:

1. Знания и факты

2. Принятие решений

3. Обновление и распределение знаний