РОСЖЕЛДОР
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Ростовский государственный университет путей сообщения»
(ФГБОУ ВО РГУПС)

Зырянкина К.Э.

Часть 1

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ЛАБОРАТОРНЫХ РАБОТ И САМОСТОЯТЕЛЬНОЙ РАБОТЫ СТУДЕНТОВ ПО ДИСЦИПЛИНЕ

ОП.04 «ОСНОВЫ ПРОЕКТИРОВАНИЯ БАЗ ДАННЫХ». ЧАСТЬ 1

для специальности
09.02.09 Веб-разработка

Ростов-на-Дону
2025

СОДЕРЖАНИЕ

Введение

Методические указания к лабораторным работам и по выполнению самостоятельной работы студентов составлены в соответствии с ФГОС СПО и рабочей программой профессионального модуля ОП.04 «Основы проектирования баз данных», которые являются частью программы подготовки специалистов среднего звена специальности 09.02.09 Веб-разработка.

Рабочей программой дисциплины ОП.04 «Основы проектирования баз данных» предусмотрено на выполнение лабораторных работ – 72 часов и самостоятельной работы студентов – 12 часов.

При выполнении лабораторных работ и при организации самостоятельной работы студентов используются активные и интерактивные формы обучения - просмотр и обсуждение учебных видеофильмов, групповая дискуссия, лекция - консультация, моделирование производственных процессов и ситуаций, обсуждение в группах, тренинг, кейс-метод, защита практических и лабораторных работ и другие.

Цель методических рекомендаций - оказание методической помощи студентам в выполнении лабораторных работ и в организации их самостоятельной работы по изучению учебного материала, для расширения, углубления и закрепления знаний и умений, а также формирования профессиональных (ПК) компетенций.

Код и содержание компетенции	Уметь	Знать
ОК 01 - Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам	распознавать задачу и/или проблему в профессиональном и/или социальном контексте, анализировать и выделять её составные части; определять этапы решения	актуальный профессиональный и социальный контекст, в котором приходится работать и жить; структура плана для решения задач, алгоритмы выполнения работ в профессиональной и смежных областях; основные источники информации и ресурсы для решения задач и/или проблем в профессиональном и/или социальном контексте; методы работы в профессиональной и смежных сферах; порядок оценки результатов решения задач

		профессиональной деятельности
ОК 02 – Осуществлять поиск, анализ и интерпретацию информации, необходимой для выполнения задач профессиональной деятельности	определять задачи для поиска информации, планировать процесс поиска, выбирать необходимые источники информации; выделять наиболее значимое в перечне информации, структурировать получаемую информацию, оформлять результаты поиска; оценивать практическую значимость результатов поиска; применять средства информационных технологий для решения профессиональных задач; использовать современное программное обеспечение в профессиональной деятельности; использовать различные цифровые средства для решения профессиональных задач	Номенклатуру информационных источников, применяемых в профессиональной деятельности; формат оформления результатов поиска информации; программное обеспечение в профессиональной деятельности, в том числе цифровые средства; приемы структурирования информации; современные средства и устройства информатизации, порядок их применения; программное обеспечение в профессиональной деятельности, в том числе цифровые средства;
ОК 04 - Работать в коллективе и команде, эффективно взаимодействовать с коллегами, руководством, клиентами	организовывать работу коллектива и команды; взаимодействовать с коллегами, руководством, клиентами в ходе профессиональной деятельности	психологические основы деятельности коллектива; психологические особенности личности
ОК 09 - Пользоваться профессиональной документацией на государственном и иностранном языках	понимать общий смысл четко произнесенных высказываний на известные темы (профессиональные и бытовые), понимать	правила построения простых и сложных предложений на профессиональные темы основные общеупотребительные глаголы (бытовая и

	тексты на базовые профессиональные темы	профессиональная лексика)
ПК 2.1 - Устанавливать прикладное программное обеспечение и модулей информационных ресурсов, включая их настройку	соблюдать процедуру установки прикладного программного обеспечения в соответствии с документацией; идентифицировать инциденты, возникающие при установке программного обеспечения, и принимать решение по изменению процедуры установки; пользоваться нормативно-технической документацией в области программного обеспечения; производить настройку параметров веб-сервера; устанавливать систему управления базами данных (СУБД);	принципы устройства и функционирования информационных ресурсов; принципы устройства и функционирования программных средств и платформ для разработки веб-ресурсов;
ПК 2.2 - Проводить работы по резервному копированию и развертыванию резервной копии информационных ресурсов	выполнять регламентные процедуры по резервированию данных; устанавливать прикладное программное обеспечение для резервирования информационных ресурсов.	основы управления изменениями; основы резервного развертывания и резервного копирования информационных ресурсов; общие основы решения практических задач по созданию резервных копий; возможности ИР.
ПК 2.3 - Настраивать права пользователей в соответствии с функциональными задачами (ролями) и на основании информации о	пользоваться нормативно-технической документацией в области программного обеспечения; идентифицировать права пользователей в	принципы устройства и функционирования информационных ресурсов; современные стандарты взаимодействия компонентов распределенных приложений; возможности ИР.

поведенческих факторах.	зависимости от функционала информационного ресурса; регламентировать уровни прав и ролей пользователей информационных ресурсов; применять регламентные процедуры управления правами доступа пользователей информационных ресурсов.	

1 ПЛАН РАСПРЕДЕЛЕНИЯ УЧЕБНОЙ НАГРУЗКИ

Объем дисциплины в академических часах с указанием количества академических часов, выделенных на контактную работу обучающихся с преподавателем (по видам учебных занятий) и на самостоятельную работу обучающихся

Вид учебной работы	**Всего часов**	**Число часов в семестре**	
		3	**4**
Объем образовательной программы учебной дисциплины	**144**	**72**	**72**
в том числе:			
Лекции (теоретическое обучение)	52	32	20
Практические занятия			
Лабораторные работы	72	32	40
Самостоятельная работа	12	6	6
Промежуточная аттестация (в форме зачета в 3 семестре и экзамена в 4 семестре)	**8**	**2**	**6**

Содержание дисциплины

Семестр №3

Наименование лекционных занятий	**Трудоемкость аудиторной работы, часы**
Раздел № 1 Основные понятия баз данных	
1.1 Основные понятия теории БД.	2
1.2 Технологии работы с БД.	2
1.3 Архитектура БД.	2
Раздел № 2 Взаимосвязи в моделях и реляционный подход к построению моделей	
2.1 Логическая и физическая независимость данных.	2
2.2 Типы моделей данных.	2
2.3 Реляционная модель данных.	2
2.4 Реляционная алгебра.	2
Раздел № 3 Этапы проектирования баз данных	
3.1 Основные этапы проектирования БД.	2
3.2 Концептуальное проектирование БД.	2

Наименование лекционных занятий	Трудоемкость аудиторной работы, часы
3.3 Нормализация БД.	2
Раздел № 4 Проектирование структур баз данных	
4.1 Средства проектирования структур БД. Классификация СУБД.	2
4.2 Основные характеристики и возможности СУБД MySQL.	2
4.3 Типы данных базы данных MySQL.	2
4.4 Объекты MySQL. Понятия: таблицы, запросы, формы, отчеты, макросы, модули	2
4.5 Назначение, виды и типы запросов. Мастер запроса. Конструктор запроса.	2
4.6 Организация интерфейса с пользователем. Основные требования к разработке пользовательского интерфейса.	2

Семестр № 4

Наименование лекционных занятий	Трудоемкость аудиторной работы, часы
Раздел № 5	
5.1 Основные понятия языка SQL. Синтаксис операторов, типы данных.	2
5.2 Создание, модификация и удаление таблиц. Операторы манипулирования данными.	2
5.3 Организация запросов на выборку данных при помощи языка SQL.	2
5.4 Организация запросов на выборку данных при помощи языка SQL из нескольких таблиц.	2
5.5 Подзапросы.	2
5.6 Сортировка и группировка данных в SQL.	2

Наименование лекционных занятий	Трудоемкость аудиторной работы, часы
5.7 Предикаты, используемые в запросах.	2
5.8 Реализация операций реляционной алгебры средствами языка SQL.	2
5.9 Транзакции.	2
5.10 Работа с представлениями, хранимыми процедурами и функциями.	2

Лабораторные работы

Семестр №3

Наименование (тематика) лабораторных работ, семинаров	Трудоемкость аудиторной работы, часы
Раздел № 1	
1.1 Анализ предметной области и выделение сущностей.	2
1.2 Установка и настройка СУБД.	2
Раздел № 2	
2.1 Обеспечение логической и физической независимости данных на примере рефакторинга базы данных.	2
2.2 Сравнительное моделирование: иерархическая, сетевая и реляционная модели данных.	2
2.3 Функциональный анализ предметной области.	1
2.4 Атрибутивный анализ предметной области	1
2.5 Моделирование ER-диаграмм.	2
Раздел № 3	
3.1 Особенности реляционной модели и проектирование баз данных.	4
3.2 Преобразование реляционной БД в сущности и связи.	2
3.3 Проектирование реляционной БД.	2

Наименование (тематика) лабораторных работ, семинаров	Трудоемкость аудиторной работы, часы
3.4 Проектирование реляционной БД. Нормализация таблиц	2
Раздел № 4	
4.1 Создание проекта БД. Создание БД. Редактирование и модификация таблиц.	2
4.2 Задание ключей. Создание основных объектов БД.	2
4.3 Задание значений и ограничений поля. Проверка введенного в поле значения. Отображение данных числового типа и типа дата	2
4.4 Редактирование, добавление и удаление записей в таблице. Применение логических условий к записям. Открытие, редактирование и пополнение табличного файла.	2
4.5 Создание ключевых полей. Задание индексов. Установление и удаление связей между таблицами.	2

Семестр №4

Наименование (тематика) лабораторных работ, семинаров	Трудоемкость аудиторной работы, часы
Раздел № 5	
5.1. Знакомство с языком SQL. Работа с консольным клиентом и графическим интерфейсом.	2
5.2 Создание структуры БД: операторы DDL (CREATE, ALTER, DROP).	2
5.3 Манипулирование данными: операторы INSERT, UPDATE, DELETE.	2
5.4 Простые запросы на выборку (SELECT). Фильтрация строк (WHERE).	2

Наименование (тематика) лабораторных работ, семинаров	Трудоемкость аудиторной работы, часы
5.5 Работа с функциями и выражениями в SELECT. Сортировка (ORDER BY).	2
5.6 Соединение таблиц: INNER JOIN, табличные псевдонимы (алиасы).	2
5.7 Внешние соединения: LEFT/RIGHT JOIN. Перекрестное соединение (CROSS JOIN).	2
5.8 Агрегирование данных: GROUP BY и агрегатные функции (COUNT, SUM, AVG).	2
5.9 Фильтрация групп. Условие HAVING. Сравнение WHERE и HAVING.	2
5.10 Простые подзапросы в условиях WHERE и HAVING.	2
5.11 Сложные подзапросы: коррелированные подзапросы, подзапросы в FROM.	2
5.12 Предикаты IN, EXISTS, ANY/SOME, ALL. Операция UNION.	2
5.13 Предикаты LIKE для поиска по шаблону. Работа с NULL (IS NULL, IS NOT NULL).	2
5.14 Реализация операций реляционной алгебры (проекция, выборка, соединение, деление) на SQL.	2
5.15 Создание и использование представлений (VIEW).	2
5.16 Написание простых хранимых процедур и функций. Использование переменных.	2
5.17 Управление транзакциями. BEGIN, COMMIT, ROLLBACK. Обработка ошибок.	2
5.18 Создание и анализ отчетов на основе SQL-запросов. Экспорт результатов.	2
5.19 Оптимизация запросов: создание индексов, использование EXPLAIN для анализа плана выполнения.	2

Наименование (тематика) лабораторных работ, семинаров	Трудоемкость аудиторной работы, часы
5.20. Итоговый комплексный проект: разработка полноценной базы данных и набора запросов по индивидуальному заданию.	2

Самостоятельное изучение учебного материала (самоподготовка)

Номер раздела данной дисциплины	Наименование тем, вопросов, вынесенных для самостоятельного изучения	Трудоемкость внеаудиторной работы, часы
Семестр № 3		
1	Установка СУБД MySQL.	1
3	Нормализация реляционной БД.	1
4	Проектирование БД по индивидуальному заданию.	4
Семестр № 4		
5	Разработка таблиц с помощью языка SQL по индивидуальному заданию.	4
5	Разработка запросов и отчетов с помощью языка SQL по индивидуальному заданию.	2

Лабораторная работа №1 «Анализ предметной области и выделение сущностей»

Теоретические сведения

Процесс проектирования базы данных начинается с анализа предметной области – выделенной части реального мира, информацию о которой необходимо систематизировать и хранить. Целью этого анализа является создание структурированного представления данных, которое ляжет в основу будущей базы данных.

Ключевыми конструктивными элементами для такого представления являются сущности – значимые объекты или понятия внутри предметной области, о которых необходимо накапливать сведения (например, «Студент», «Дисциплина» или «Зачётная книжка»).

Каждая сущность характеризуется набором атрибутов – конкретных свойств или параметров, описывающих её (для сущности «Студент» это могут быть: ФИО, дата рождения, номер зачётной книжки).

Для однозначной идентификации каждого экземпляра (конкретной записи) сущности среди других используется первичный ключ – специальный атрибут или их комбинация, гарантирующая уникальность.

Объекты в предметной области не существуют изолированно, они взаимодействуют друг с другом. Это взаимодействие моделируется с помощью связей – установленных ассоциаций между двумя или более сущностями. Для визуализации всей выделенной структуры – сущностей, их атрибутов и взаимосвязей – строится концептуальная модель данных (ER-модель). Это графическая схема, абстрагированная от деталей конкретной системы управления базами данных (СУБД).

Типы связей между сущностями.

Связи между сущностями классифицируются по количеству связанных экземпляров с каждой стороны:

Один-к-одному (1:1): Один экземпляр сущности A связан не более чем с одним экземпляром сущности B, и наоборот. Например, «Студент» – «Зачётная книжка» (у одного студента одна книжка, одна книжка принадлежит одному студенту).

Один-ко-многим (1:M): Одному экземпляру сущности A может соответствовать несколько экземпляров сущности B, но каждый экземпляр B связан только с одним A. Например, «Группа» – «Студент» (в одной группе много студентов, но каждый студент учится только в одной группе).

Многие-ко-многим (M:N): Одному экземпляру сущности A может соответствовать несколько экземпляров B, и наоборот. Например, «Студент» – «Дисциплина» (один студент изучает много дисциплин, одну дисциплину изучают много студентов).

Этапы анализа предметной области.

Анализ предметной области и построение концептуальной модели – это последовательный процесс, который можно разделить на следующие ключевые этапы:

1. Изучение и описание предметной области: детальный сбор и фиксация требований, правил и бизнес-процессов, характерных для рассматриваемой сферы.

2. Выделение сущностей: идентификация основных объектов или понятий, информацию о которых необходимо хранить в базе данных.

3. Определение атрибутов для каждой сущности: установление перечня свойств, которые полностью описывают каждую выделенную сущность, включая назначение первичного ключа.

4. Установление связей между сущностями: выявление и формальное описание взаимоотношений между сущностями с определением их типа (1:1, 1:M, M:M).

5. Построение ER-диаграммы: графическая визуализация результатов анализа с использованием стандартных нотаций (например, «воронья лапка» – Crow's Foot).

Описание предметной области для выполнения лабораторной работы

В колледже ведется учет учебного процесса. Каждый студент учится в определенной группе. Группа имеет куратора из числа преподавателей. Преподаватели ведут занятия по различным дисциплинам. Для каждой дисциплины составляется учебный план, включающий количество лекционных, практических и лабораторных часов. Студенты посещают занятия и получают оценки по дисциплинам. В семестре может быть несколько видов контроля: зачет, экзамен, дифференцированный зачет. Результаты контроля фиксируются в ведомости. Также ведется учет успеваемости студентов (посещаемость, текущие оценки).

Детализация:

Студент: ФИО, дата рождения, пол, номер зачетной книжки, дата поступления, статус (бюджет/контракт).

Группа: номер группы, специальность, год набора, куратор.

Преподаватель: ФИО, ученая степень, должность, кафедра, стаж работы.

Дисциплина: название, кафедра-разработчик, общее количество часов, семестр преподавания.

Учебный план: дисциплина, тип занятия (лекция/практика/лабораторная), количество часов, семестр.

Занятие: дата, номер пары, аудитория, преподаватель, дисциплина, группа.

Оценка: студент, дисциплина, вид контроля (зачет/экзамен), дата, оценка.

Последовательность выполнения лабораторной работы

1. Изучить описание предметной области.
2. Выделить основные бизнес-процессы.
3. Определить ключевых пользователей системы и их потребности в информации.
4. Составить список сущностей, которые должны храниться в базе данных для поддержки описанных бизнес-процессов.
5. Для каждой сущности определить атрибуты.
6. Установить связи между сущностями и определить их тип.
7. Построить концептуальную модель данных.

Лабораторная работа №2 «Установка и настройка СУБД»

Теоретические сведения

СУБД (Система управления базами данных) – комплекс программных средств, предназначенных для создания, ведения и совместного использования базы данных многими пользователями. MySQL – одна из самых популярных реляционных СУБД с открытым исходным кодом, широко используемая в веб-разработке.

Microsoft SQL Server Express – это бесплатная, полнофункциональная редакция SQL Server, предназначенная для обучения, разработки и небольших приложений. Она основана на том же ядре, что и коммерческие версии SQL Server, но имеет определенные ограничения по объему используемой памяти, количеству процессоров и размеру баз данных (до 10 ГБ на базу данных).

Компоненты SQL Server Express.

Типичная установка включает:

- Ядро СУБД SQL Server – основной сервис, выполняющий обработку запросов, транзакций и управление данными.
- SQL Server Management Studio (SSMS) – интегрированная среда для управления инфраструктурой SQL Server, включая графические инструменты для написания запросов, администрирования и настройки.
- SQL Server Configuration Manager – инструмент для управления сетевыми настройками и сервисами SQL Server.
- Командная утилита SQLCMD – консольное приложение для выполнения SQL-скриптов и администрирования.

Архитектура и основные понятия.

SQL Server работает по клиент-серверной архитектуре и использует следующие ключевые концепции:

Экземпляр (Instance) – отдельная установка SQL Server. На одном компьютере может быть несколько именованных экземпляров.

База данных – логический контейнер для объектов данных (таблиц, представлений, процедур). Системные базы данных (master, model, msdb, tempdb) создаются автоматически.

Схема (Schema) – пространство имен внутри базы данных, группирующее объекты и управляющее правами доступа.

Учетная запись (Login) и Пользователь (User): Login – учетная запись на уровне сервера для аутентификации; User – учетная запись внутри конкретной базы данных, связанная с Login.

Аутентификация.

SQL Server поддерживает два основных режима аутентификации:

Аутентификация Windows – использует учетные записи Windows (Active Directory или локальные). Рекомендуется для корпоративных сред.

Смешанный режим – поддерживает как аутентификацию Windows, так и собственную аутентификацию SQL Server (логин и пароль). Необходим для доступа с клиентов, не входящих в домен Windows.

Порядок установки СУБД.

Этап 1: Подготовка и скачивание.

1. Перейдите на страницу загрузки SQL Server Express: https://www.microsoft.com/ru-ru/sql-server/sql-server-downloads

2. Скачайте установщик SQL Server Express (рис. 1).

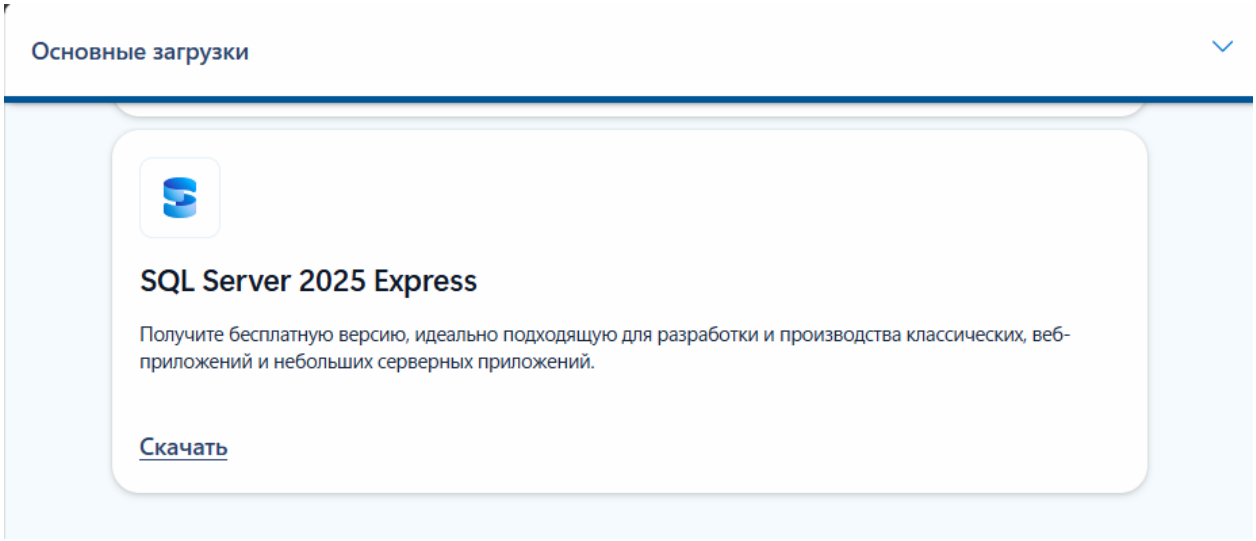


Рис.1 Установочный файл SQL Server Express на официальном сайте

2.1 Выберите вариант «Базовый» (Basic) для простой установки с настройками по умолчанию (рис. 2).

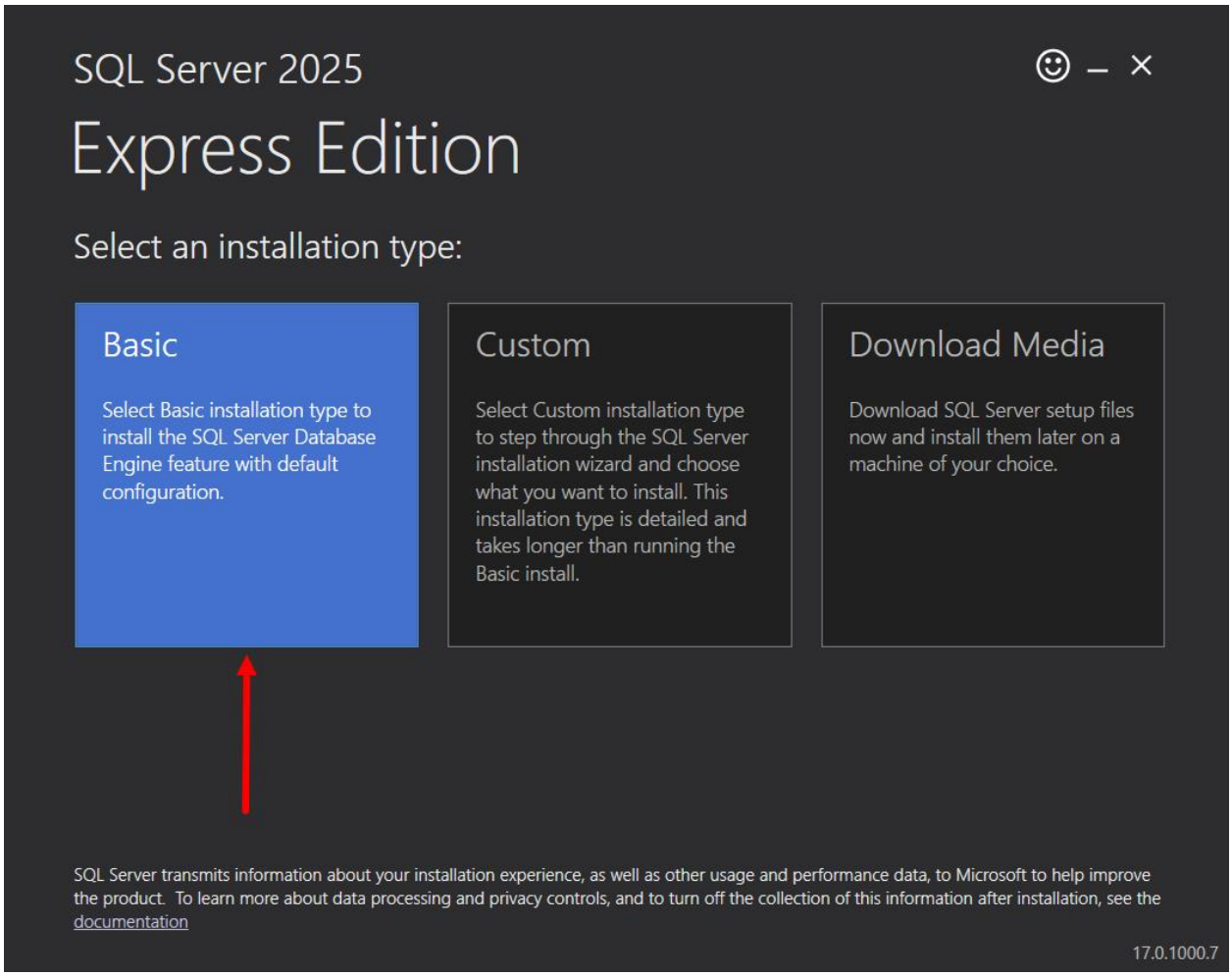


Рис. 2 Выбор базового варианта установки

2.2 Нужно выбрать место для установки SQL Server. Нажмите кнопку «Просмотр», если хотите установить по другому пути, а не по умолчанию. После выбора нажмите кнопку установки.

3. Скачайте SQL Server Management Studio (SSMS) отдельно с официального сайта Microsoft: https://learn.microsoft.com/ru-ru/ssms/install/install. Либо можно нажать на кнопку скачивания во время установки SQL Server Express (рис. 3)

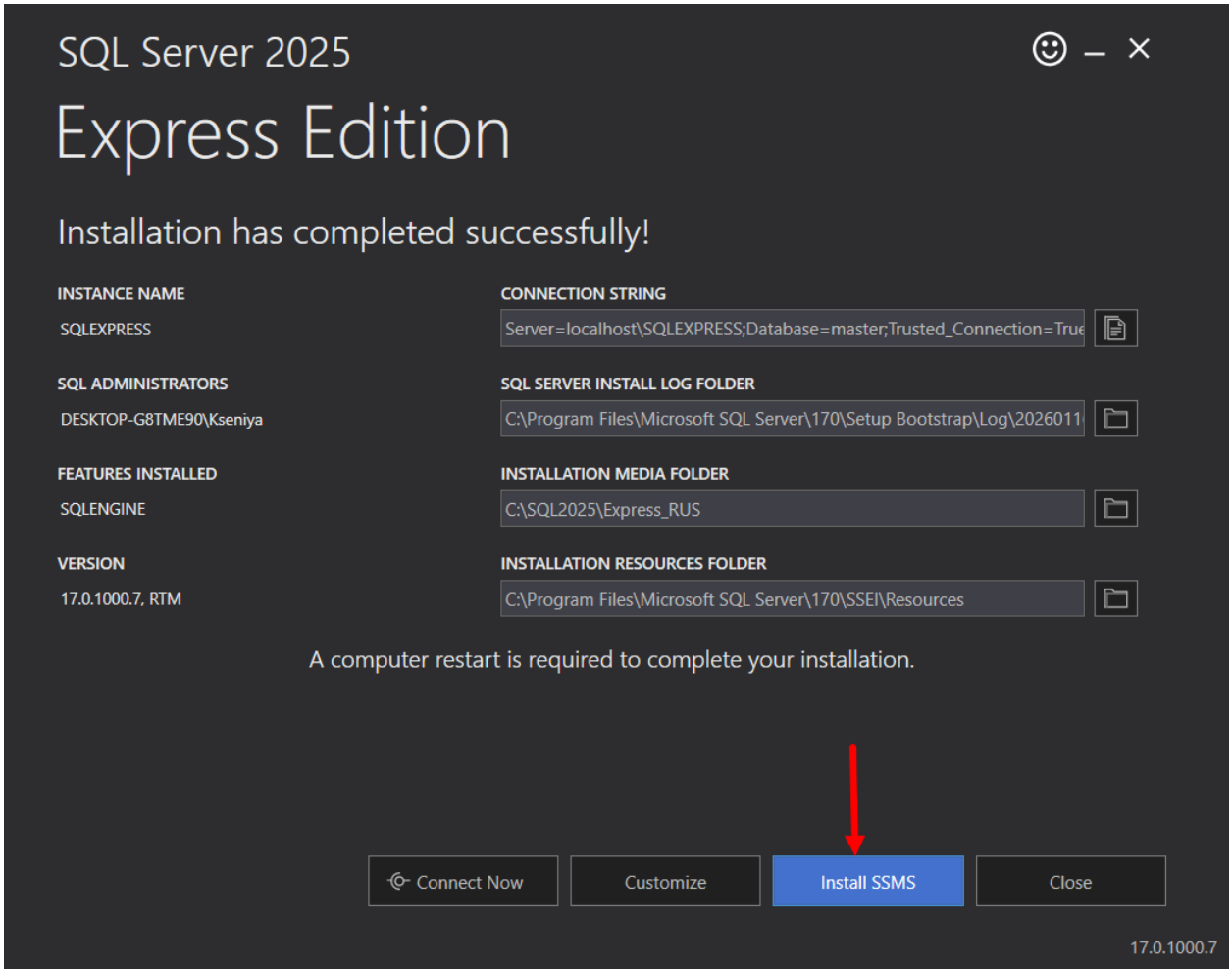


Рис. 3 Способ открытия скачивания SSMS

Загрузчик устанавливает последнюю версию установщика Visual Studio. Установщик – это отдельная программа, которая предоставляет все необходимое для установки и настройки SSMS.

После установки установщика Visual Studio его можно использовать для настройки установки SQL Server Management Studio, выбрав различные параметры на вкладках установщика.

3.1 Выберите требуемую рабочую нагрузку в установщике Visual Studio (рис. 4). Данный шаг не является обязательным, его можно пропустить.

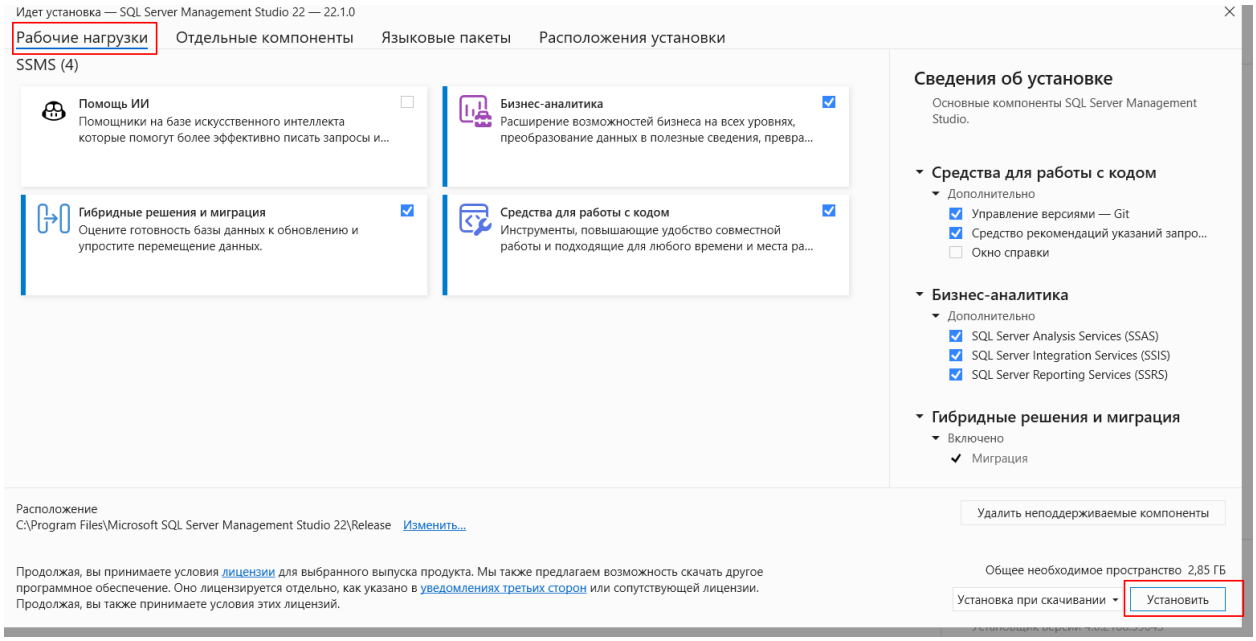


Рис. 4 Выбор требуемой рабочей нагрузки

Просмотрите сводки рабочей нагрузки, чтобы решить, какая рабочая нагрузка поддерживает необходимые функции. Например, выберите гибридную и миграционную рабочую нагрузку для оценки вашей готовности к обновлению и переноса данных.

3.2 Выбрав нужные рабочие нагрузки, выберите «Установить». Отображаются экраны состояния, показывающие ход установки SSMS (рис. 5).

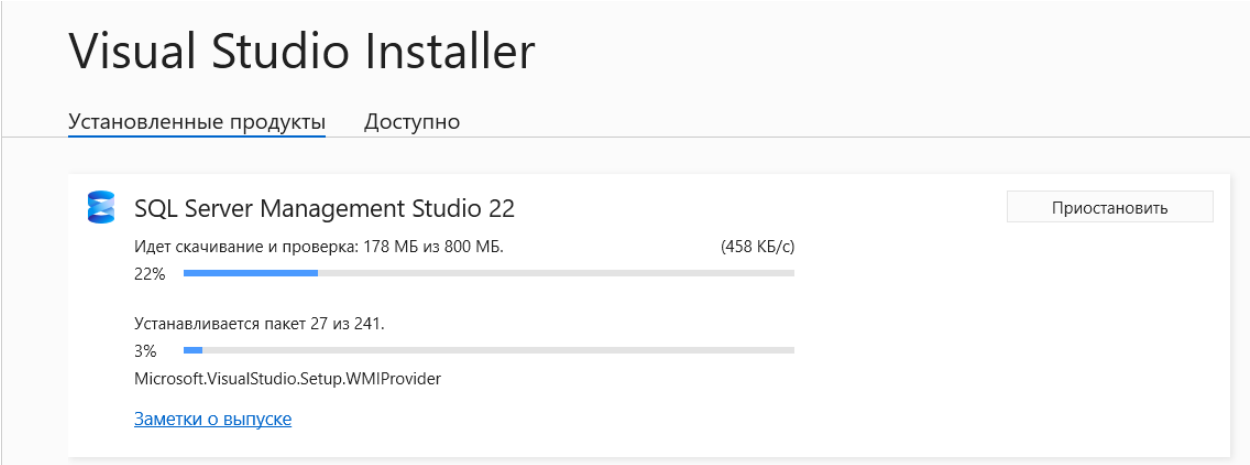


Рис. 5 Процесс установки SSMS

3.3 После успешной установки появится соответствующее окно (рис. 6). Перезапустите компьютер.

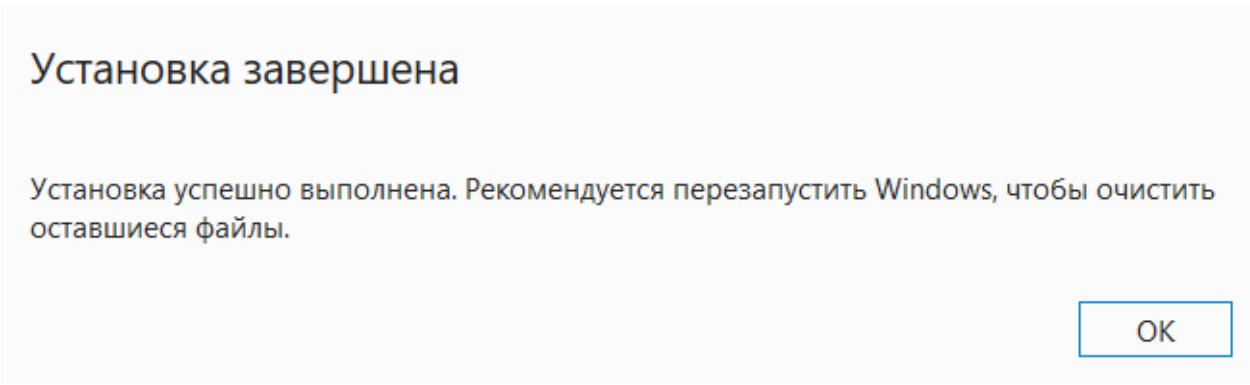


Рис. 6 Успешная установка

4. Запустите SSMS.

4.1 При первом запуске система предложит пройти авторизацию. Вы можете создать учетную запись, либо пропустить и сделать это позже самостоятельно (рис. 7).

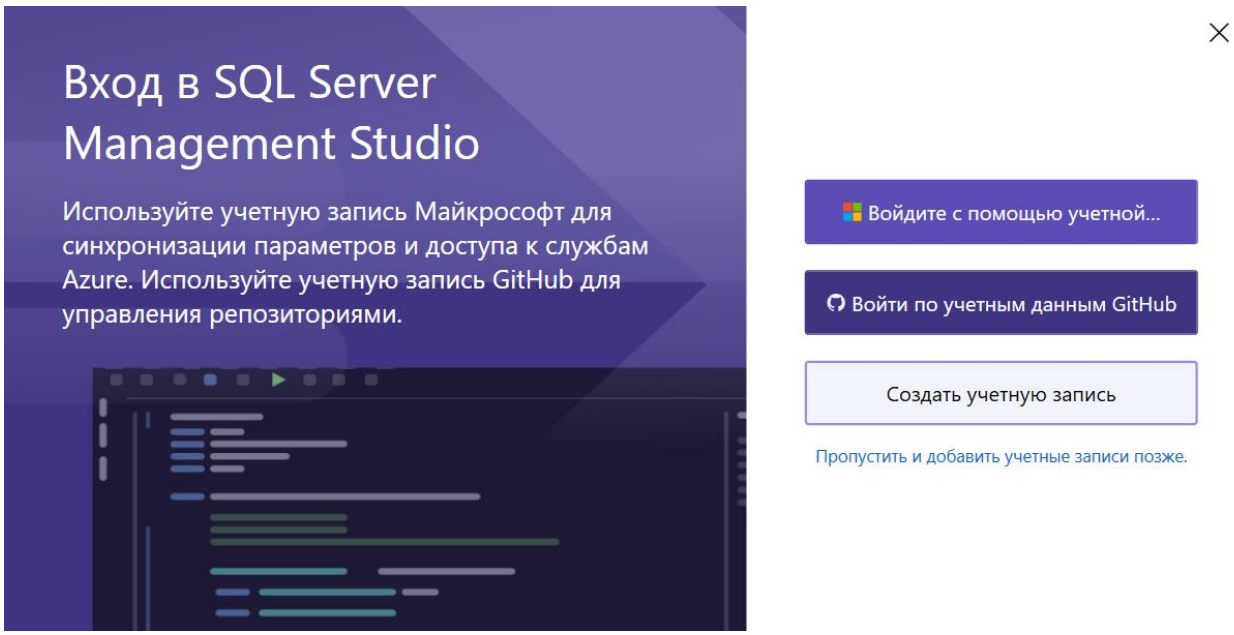


Рис. 7 Первый запуск SSMS

4.1 В появившемся диалоговом окне подключения укажите:

Имя сервера: localhost\SQLEXPRESS или .\SQLEXPRESS (точка означает локальный компьютер). Если использовался экземпляр по умолчанию, имя будет просто localhost.

Проверка подлинности: выберите «Проверка подлинности Windows» для входа под своей учетной записью администратора.

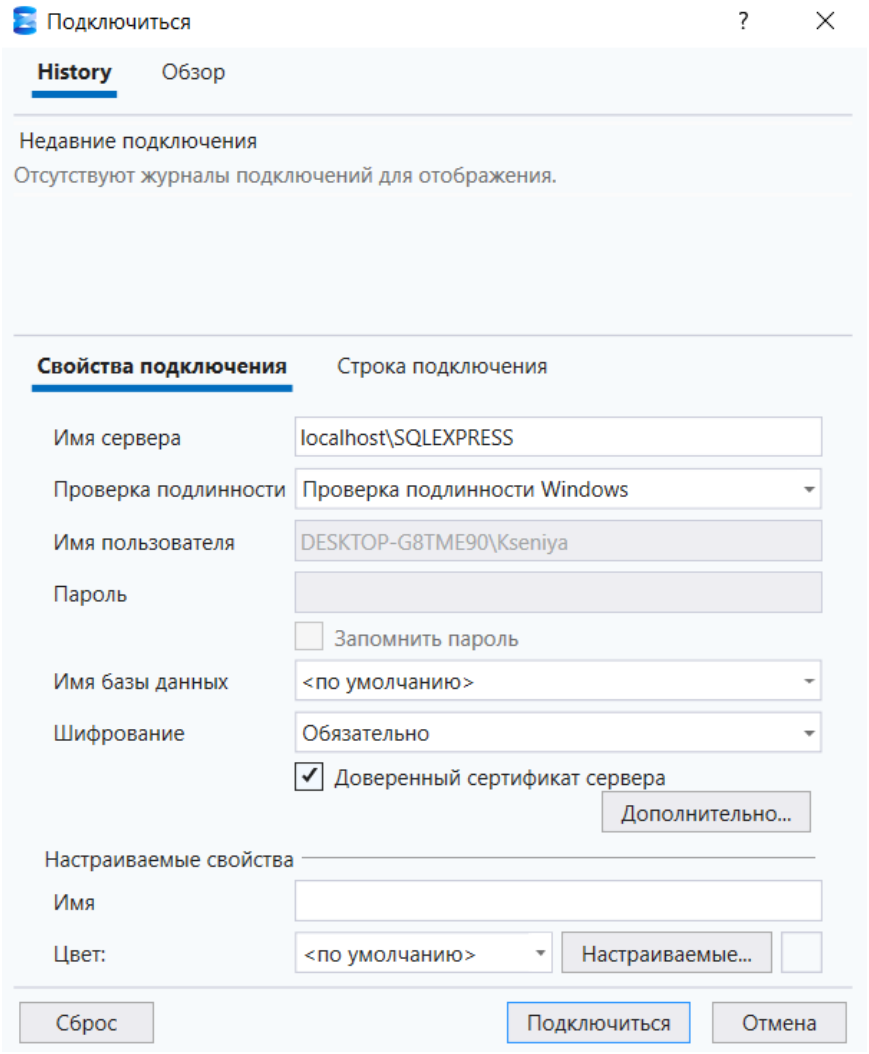


Рис. 8 Свойства подключения

4.2 Нажмите «Подключиться». Должно открыться главное окно SSMS с обозревателем объектов (Object Explorer) (рис. 9).

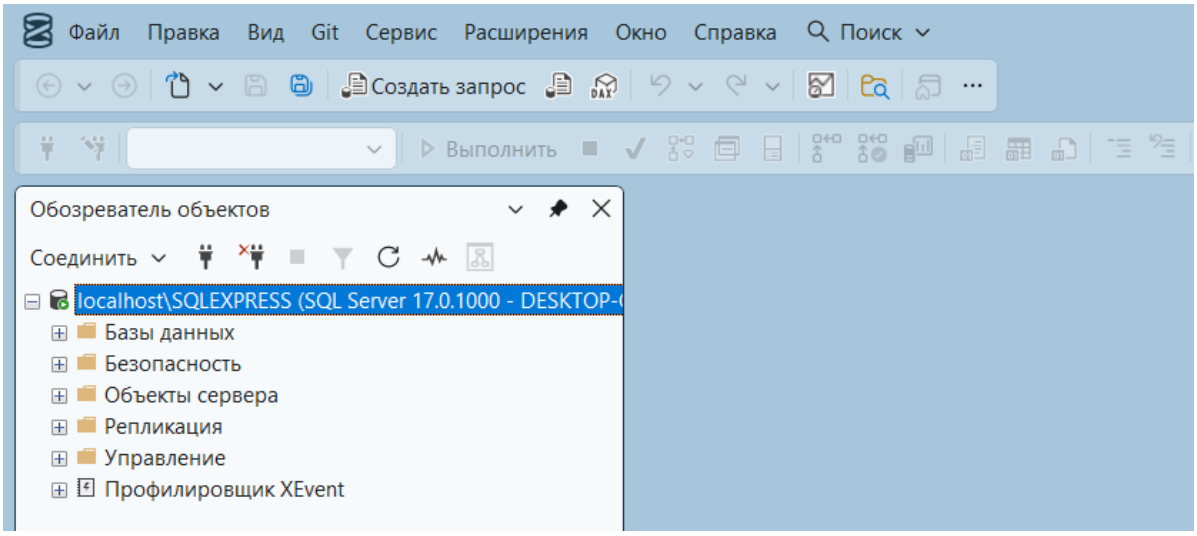


Рис. 9 Обозреватель объектов

4.3 Проверьте системные базы данных: В обозревателе объектов разверните узел «Базы данных» -> «Системные базы данных». Вы должны увидеть базы master, model, msdb, tempdb (рис. 10).

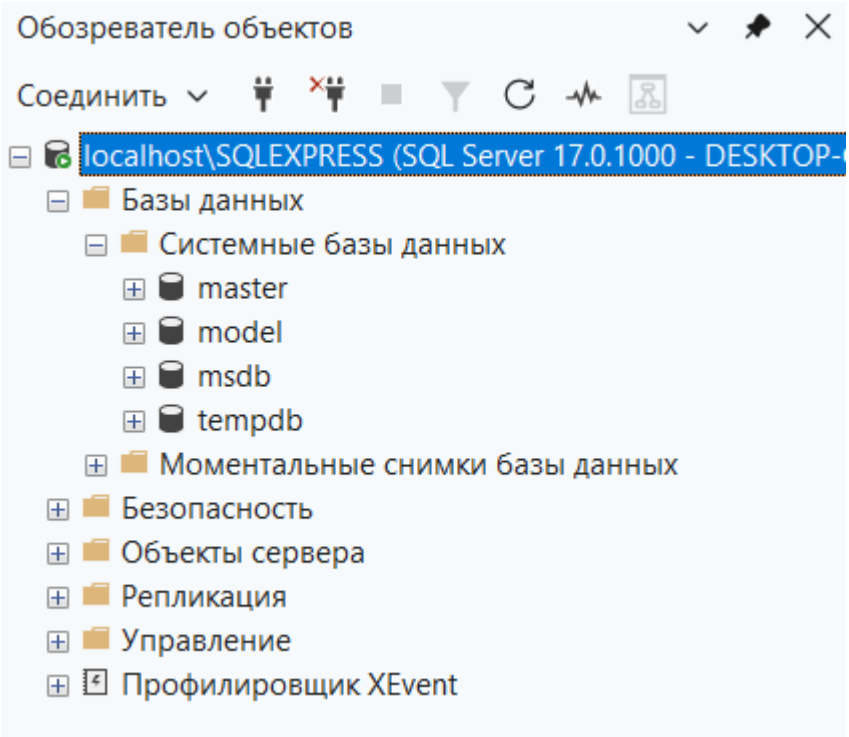


Рис. 10 Системные базы данных

5. Создайте новую базу данных:

5.1 Щелкните правой кнопкой мыши на папке «Базы данных» в обозревателе объектов. Выберите «Создать базу данных...» (рис. 11)

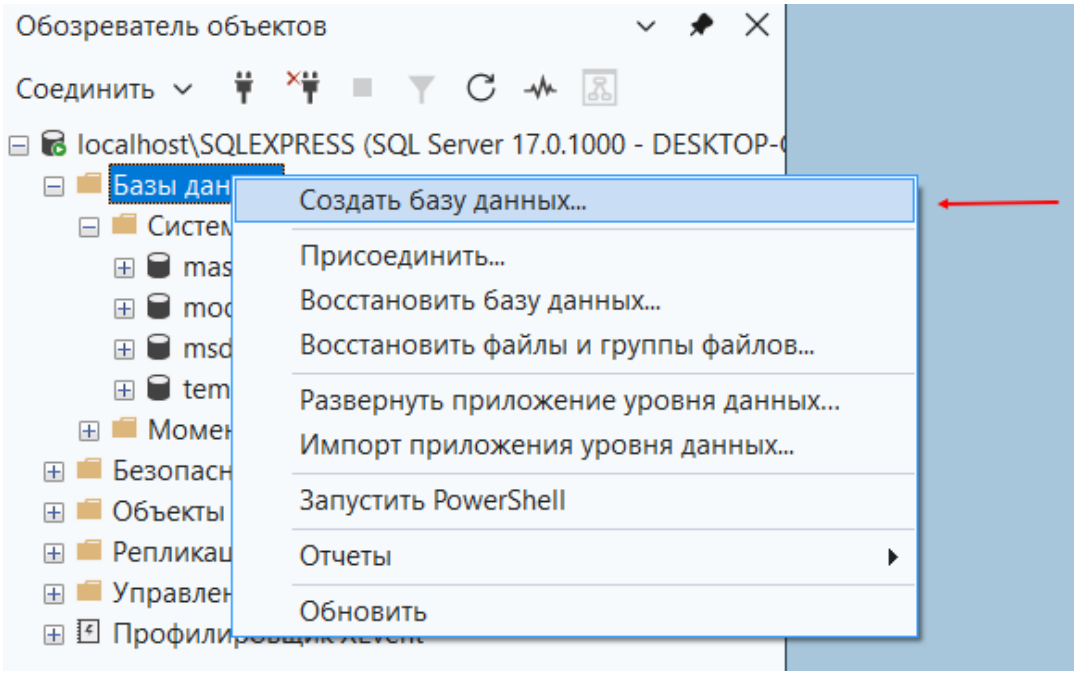


Рис. 11 Создание новой БД

5.2 Введите имя базы: college_db. Оставьте остальные параметры по умолчанию и нажмите «ОК» (рис. 12).

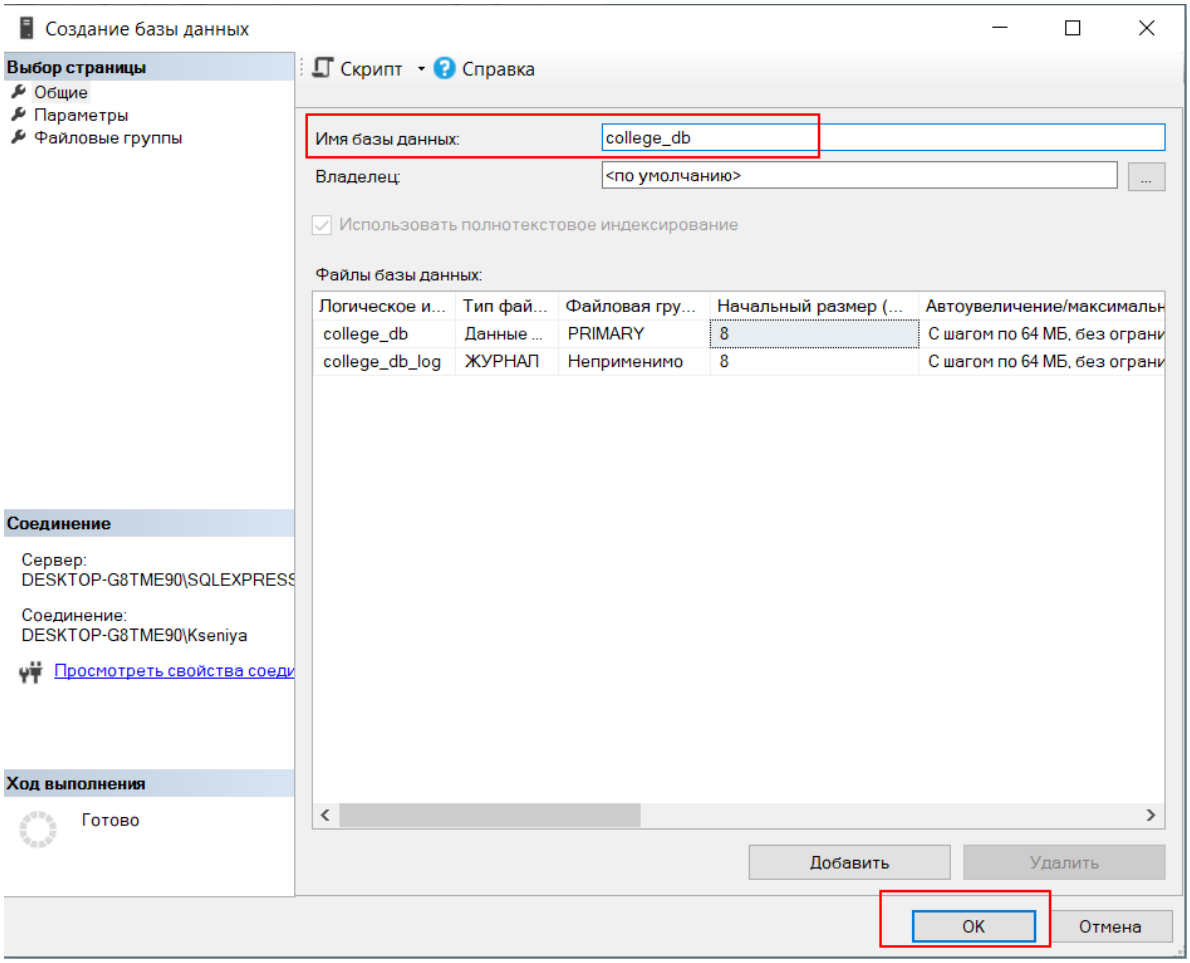


Рис. 13 Параметры новой БД

6. Создайте логин и пользователя:

6.1 В обозревателе объектов разверните узел «Безопасность» -> «Имена входа». Щелкните правой кнопкой -> «Создать имя входа...» (рис. 14).

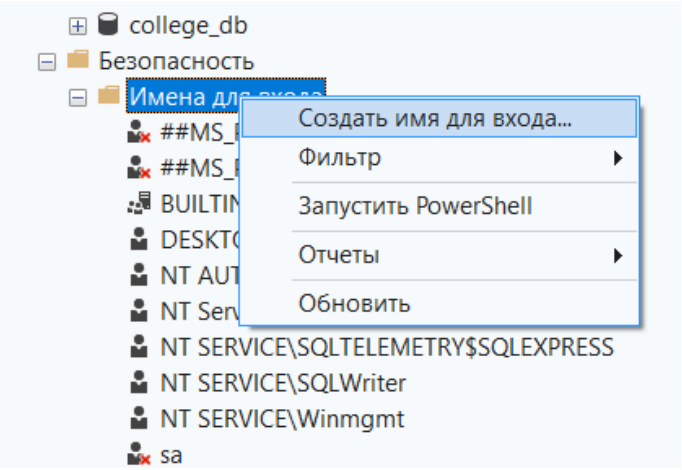


Рис. 14 Развертывание узла «Безопасность»

6.2 Введите имя входа: college_admin. Выберите «Проверка подлинности SQL Server» и задайте пароль (например, admin123) (рис. 15).

Важно: снимите галочку «Задать срок окончания действия пароля».

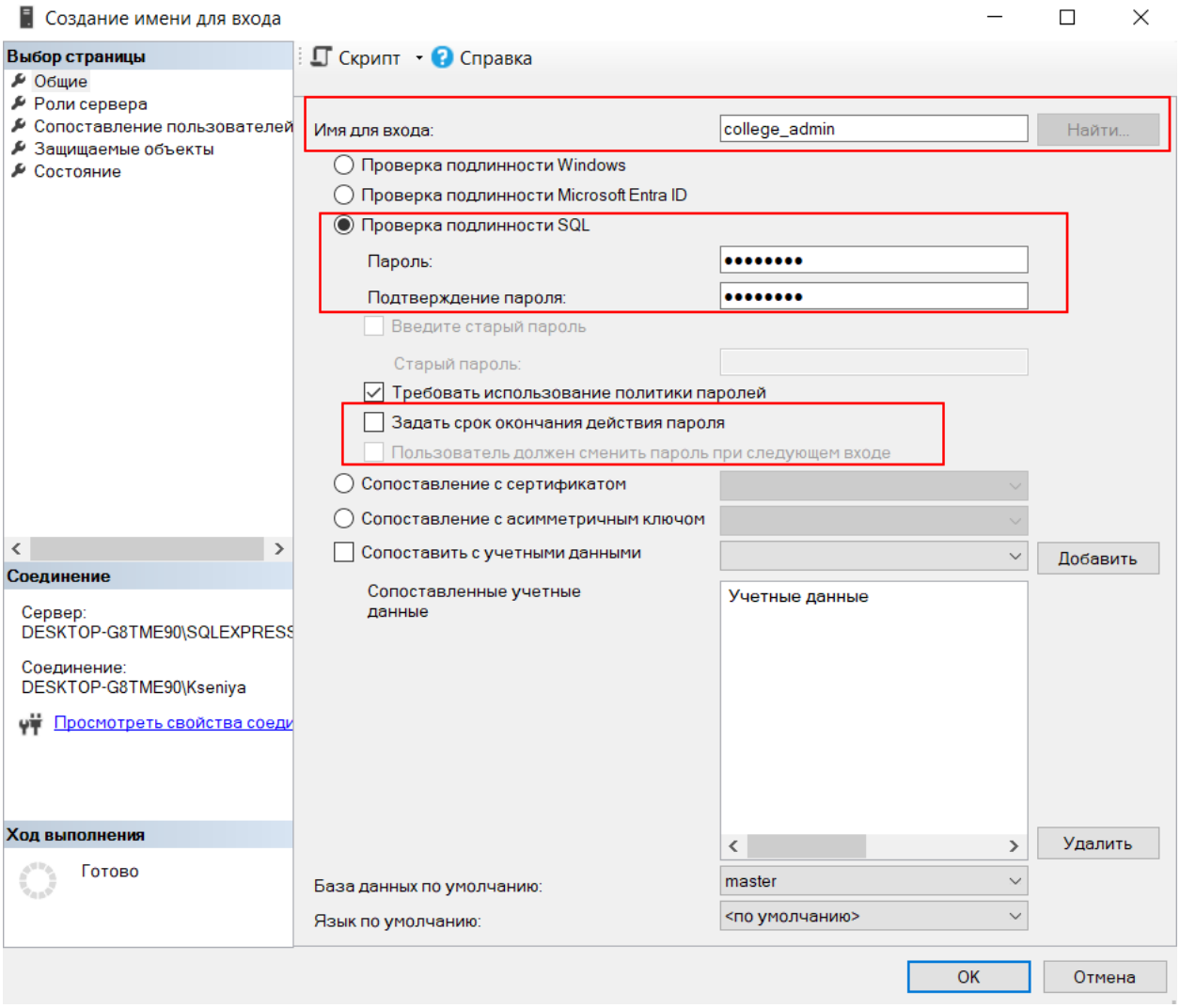


Рис. 15 Создание имени для входа

6.3 На вкладке «Сопоставление пользователей» в секции «Пользователи, сопоставленные с этим именем входа» отметьте галочкой базу данных college_db (рис. 16). Роль пользователя в этой базе данных автоматически станет db_owner (если нет, то установите роль самостоятельно). Нажмите «ОК».

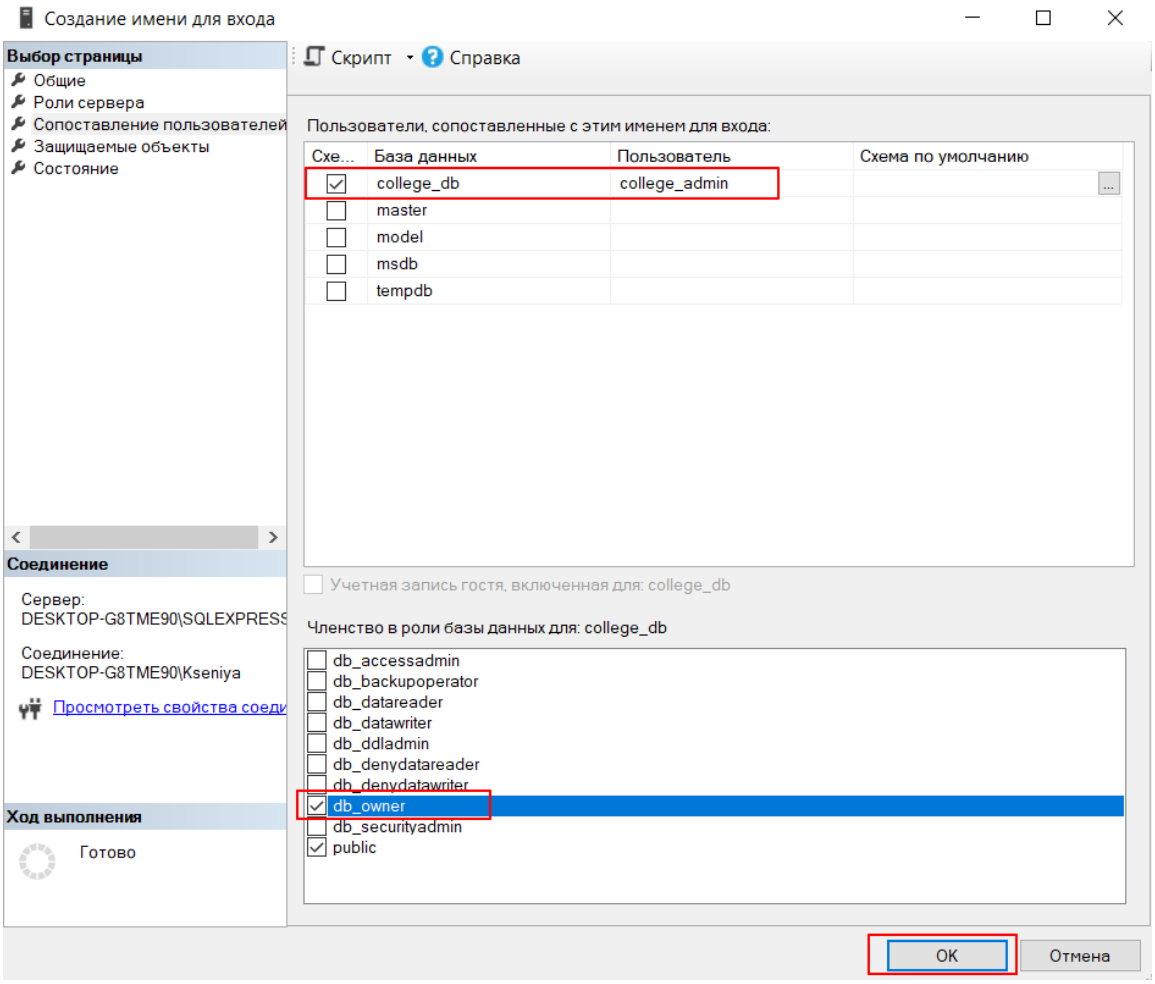


Рис. 16 Сопоставление пользователей

7. Проверьте подключение под новым пользователем:

7.1 В главном меню SSMS выберите «Файл» -> «Подключить к обозревателю объектов» (рис. 17).

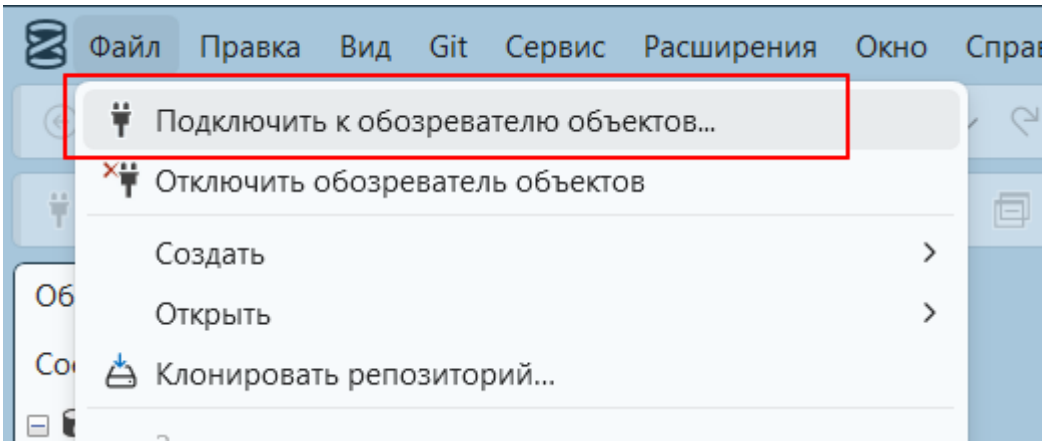


Рис. 17 Подключение к обозревателю объектов

7.2 В качестве имени сервера укажите localhost\SQLEXPRESS. Выберите «Проверка подлинности SQL Server». Введите имя входа college_admin и его пароль. Подключитесь (рис. 18).

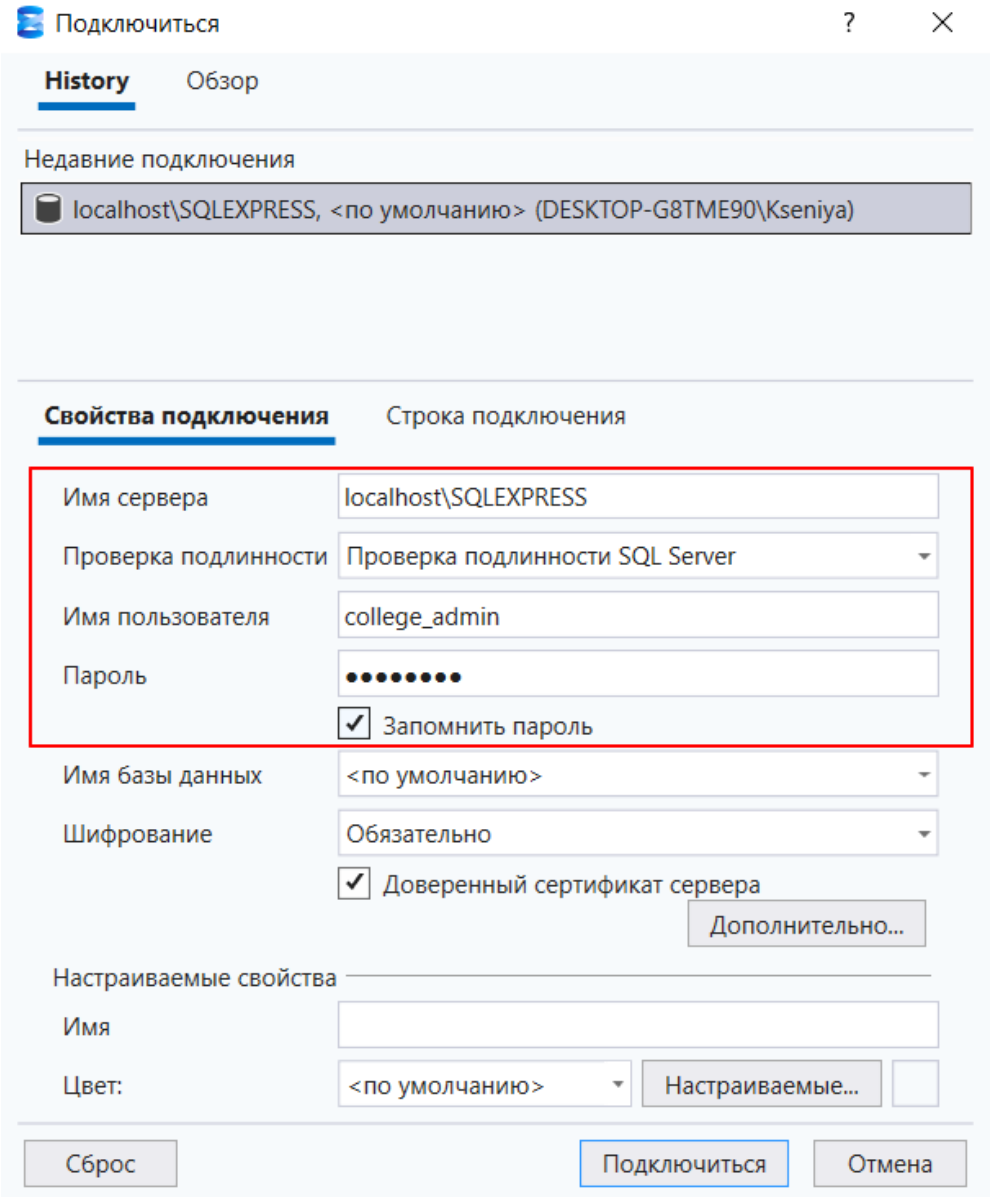


Рис. 18 Подключение к обозревателю объектов

Примечание: если Вы при подключении столкнулись с ошибкой 18456, проверьте в первую очередь корректность введенных значений логина и

пароля. Если логин и пароль введены корректно, тогда выполните следующие действия:

1. Подключитесь через Windows-аутентификацию.
2. Щелкните правой кнопкой по серверу → «Свойства»
3. Перейдите на вкладку «Безопасность». Выберите: «Режим проверки подлинности SQL Server и Windows» (рис. 19). Нажмите OK

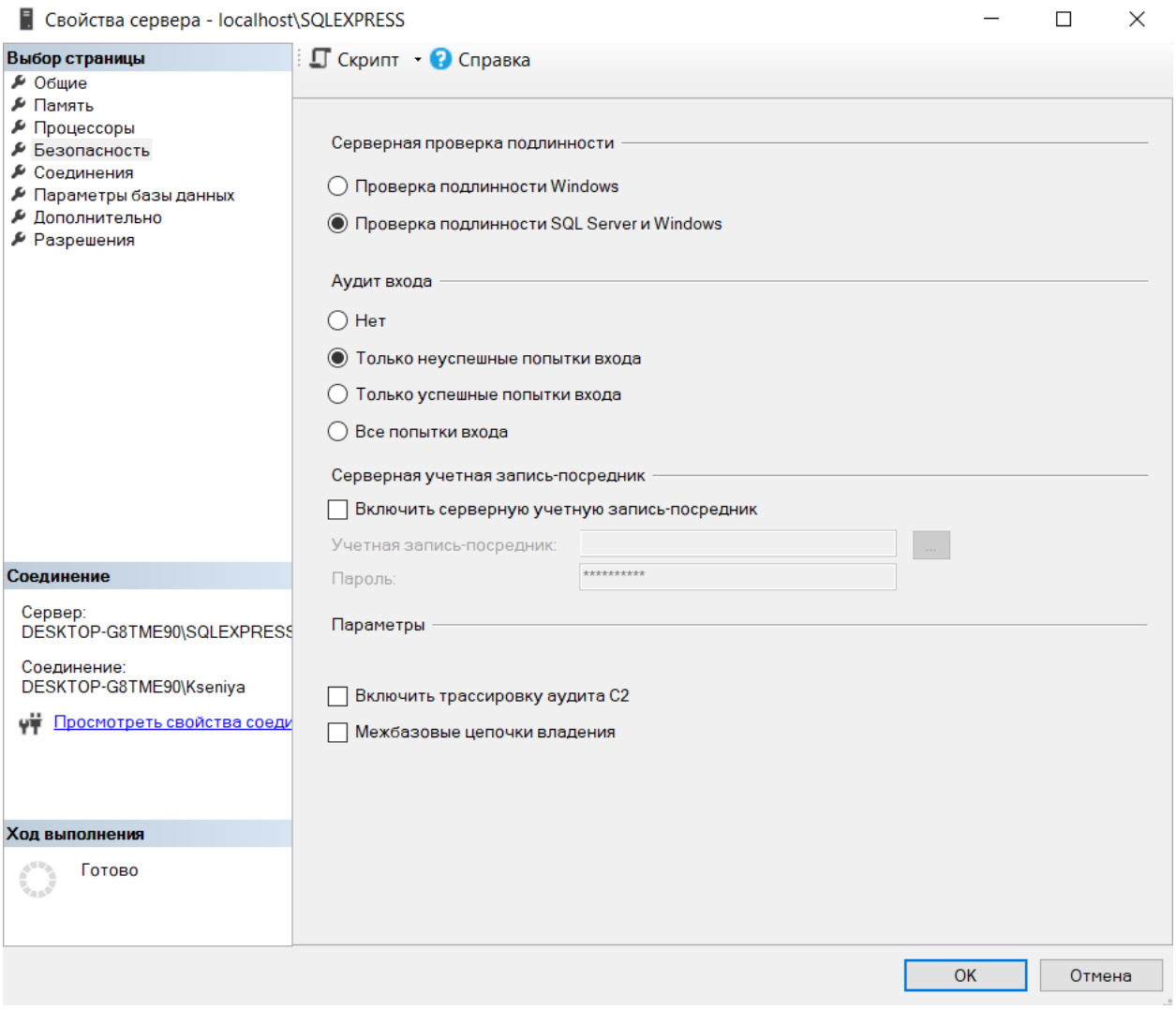


Рис. 19 Настройка смешанного режима подключения

4. Перезапустите SQL Server:

– Откройте SQL Server Configuration Manager
– Найдите «SQL Server (SQLEXPRESS)»
– Щелкните правой кнопкой → «Перезапустить» (рис. 20).

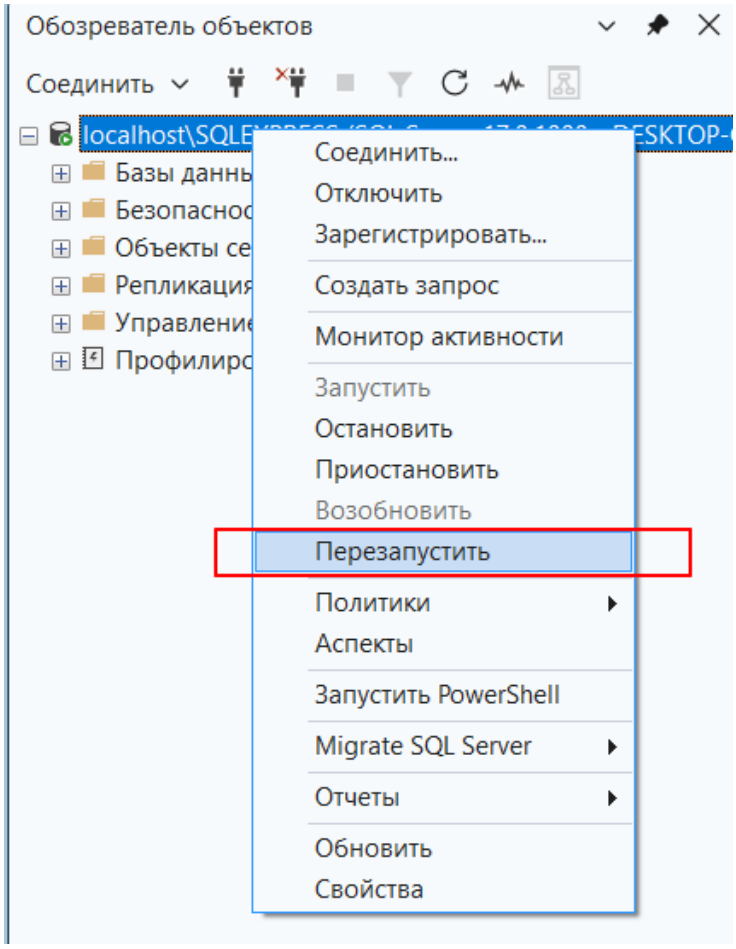


Рис. 20 Перезапуск сервера

Проверить, какой режим аутентификации установлен можно с помощью следующего запроса:

```
SELECT
 SERVERPROPERTY('IsIntegratedSecurityOnly') as 'Только_Windows',
 SERVERPROPERTY('IsMixedMode') as 'Смешанный_режим';
```

Результаты:

IsIntegratedSecurityOnly = 1 - ТОЛЬКО Windows-аутентификация

IsMixedMode = 1 - Включен смешанный режим

Последовательность выполнения лабораторной работы

1. Скачать установочный пакет SQL Server Express и SSMS.
2. Установить экземпляр SQL Server Express с настройками по умолчанию.
3. Установить SQL Server Management Studio (SSMS).
4. Выполнить первоначальную настройку сервера (режим аутентификации, учетные записи).
5. Подключиться к серверу с помощью SSMS.
6. Создать учетные записи и осуществить подключение через одну из них.

Лабораторная работа №3 «Обеспечение логической и физической независимости данных на примере рефакторинга базы данных»

Теоретические сведения

Независимость данных – фундаментальное свойство систем управления базами данных, позволяющее вносить изменения в организацию данных без необходимости модификации прикладных программ.

Логическая независимость данных.

Логическая независимость – способность изменять концептуальную схему (логическую структуру) базы данных без необходимости изменения внешних схем и прикладных программ.

Примеры логических изменений:

- Добавление новых таблиц.
- Добавление новых атрибутов в существующие таблицы.
- Изменение связей между таблицами.
- Изменение ограничений целостности.

Физическая независимость данных.

Физическая независимость – способность изменять внутреннюю схему (физическую организацию) базы данных без необходимости изменения концептуальной схемы и прикладных программ.

Примеры физических изменений:

- Изменение способа хранения данных (файловая система, секционирование).
- Добавление или удаление индексов.
- Изменение типов данных (с сохранением семантики).
- Оптимизация размещения данных на диске.

Для обеспечения физической независимости используют:

- Индексы для ускорения доступа.
- Кэширование часто запрашиваемых данных.

– Оптимизацию запросов.

Уровни архитектуры ANSI/SPARC.

Внешний уровень – множество представлений данных, каждое из которых предназначено для конкретного пользователя или приложения.

Концептуальный уровень – обобщенное представление всей базы данных, включающее все сущности, атрибуты и связи.

Внутренний уровень – описание физического хранения данных на носителях.

Описание предметной области для выполнения лабораторной работы

Рассмотрим базу данных «Учебный центр», которая хранит информацию о курсах, преподавателях, студентах и их успеваемости. Исходная структура упрощена для демонстрации принципов независимости данных.

База данных состоит из 4 таблиц:

1. Таблица Courses (Курсы):

course_id – уникальный идентификатор курса

course_name – название курса

description – описание курса

price – стоимость курса

duration_hours – продолжительность в часах

start_date – дата начала курса

end_date – дата окончания курса

2. Таблица Students (Студенты):

student_id – уникальный идентификатор студента

full_name – полное имя студента

birth_date – дата рождения

email – электронная почта

phone – номер телефона

registration_date – дата регистрации

3. Таблица Enrollments (Зачисления):

enrollment_id – уникальный идентификатор зачисления

student_id – ссылка на студента

course_id – ссылка на курс

enrollment_date – дата зачисления

status – статус зачисления (active, completed, cancelled)

4. Таблица Grades (Оценки):

grade_id – уникальный идентификатор оценки

enrollment_id – ссылка на зачисление

exam_date – дата экзамена

grade – оценка

comments – комментарии преподавателя

Проблемы в исходной структуре

1. Отсутствие категоризации курсов – все курсы хранятся в одной таблице без разделения по типам.
2. Нет истории изменений – при изменении статуса зачисления старая информация теряется.
3. Неоптимальные типы данных – некоторые столбцы используют VARCHAR вместо NVARCHAR.
4. Отсутствие индексов – поиск и соединение таблиц может быть медленным.

Последовательность выполнения лабораторной работы

1. Проанализировать логический рефакторинг:
   - Выделение категорий курсов в отдельную таблицу.
   - Добавление истории изменений статусов зачислений.
2. Проанализировать физический рефакторинг:
   - Создание индексов для оптимизации производительности.
   - Изменение типов данных без потери информации.

Лабораторная работа №4 «Сравнительное моделирование: иерархическая, сетевая и реляционная модели данных»

Теоретические сведения

Развитие моделей данных отражает историю попыток эффективно организовать информацию в компьютерах. Каждая новая модель появлялась как ответ на ограничения предыдущей.

Сетевая модель.

Концепция: развитие иерархической модели, снявшее ограничение «один родитель». Данные представляются в виде графа, где узел (запись) может быть связан с несколькими другими узлами. Связи определяются как «наборы» (sets).

Аналогия из жизни: маршруты общественного транспорта (одна остановка связана с несколькими маршрутами), социальные сети (один пользователь имеет множество связей).

Ключевой принцип: Поддержка отношений «ногие-ко-многим» (M:M) через специальные конструкции.

Иерархическая модель.

Концепция: данные организованы в виде дерева с корневым элементом. Каждый элемент (узел) имеет одного «родителя» (кроме корневого) и может иметь несколько «детей». Связи жёстко фиксированы в структуре.

Аналогия из жизни: структура папок в проводнике Windows, генеалогическое древо, организационная структура компании.

Ключевой принцип: отношение «один-ко-многим» (1:M), реализованное через физическое указание в записи на её «детей».

Связи объектов в сетевых и иерархических моделях данных.

Модель данных, как правило, включает несколько разных объектов. Между объектами модели данных устанавливаются связи. Совокупность взаимосвязанных конкретных объектов модели для некоторой предметной области образует базу данных.

Связи между любыми двумя объектами модели определяются групповыми отношениями между их экземплярами.

Групповое отношение (набор) – это строго иерархическое отношение между записями двух типов: главной записью набора и подчиненными записи набора.

Способы организации групповых отношений. Отличия в разных СУБД может иметь и организация связей (групповых отношений) объектов сетевой модели – цепи, логические связи (аналоги набора). При этом могут отличаться и способы реализации групповых отношений, например, с помощью адресных указателей или по ключу связи (атрибуту связи) – одинаковому в связанных объектах (записях, файлах) модели.

Связи в иерархических и сетевых моделях часто реализуются физически и при большом числе объектов преимущества организации базы данных могут резко снижаться из-за трудностей поддержания связей, которые становятся громоздкими.

В иерархических моделях непосредственный доступ по ключу, как правило, возможен только к объекту самого высокого уровня, который не подчинен другим объектам. К другим объектам доступ осуществляется по связям от объекта на вершине модели. В сетевых моделях непосредственный доступ по ключу может обеспечиваться к любому объекту независимо от его уровня в модели. Возможен также доступ по связям от любой точки доступа.

Структура объекта (записи, файла) в сетевых моделях чаще бывает линейной и реже имеет иерархическую структуру. Объект линейной структуры состоит только из простых и ключевых атрибутов. Структуры данных более низкого уровня также могут иметь свою специфику и названия. Например, атрибут – аналог элемента данных. Структура объекта (записи, сегмента) в иерархических моделях может быть иерархической (рис.21.) или линейной.

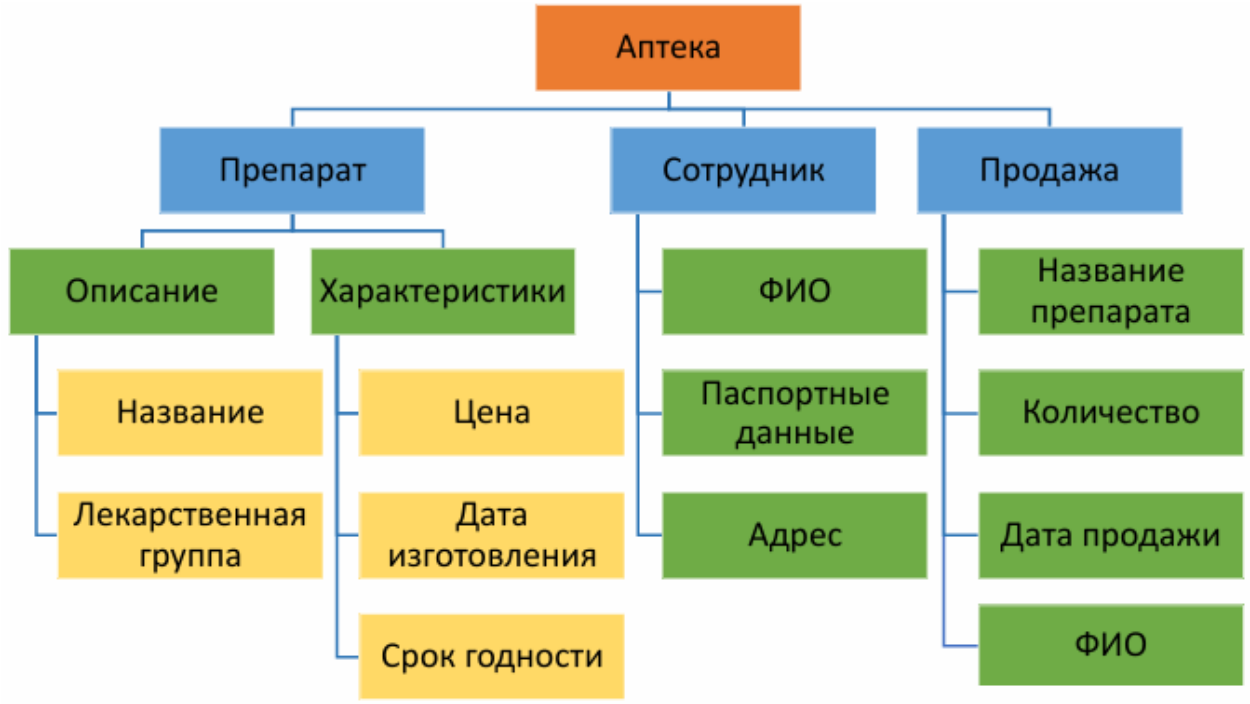


Рис. 21 Пример иерархической модели

В строго иерархических моделях, как правило, любой объект (запись, сегмент) может подчиняться только одному объекту вышестоящего уровня. В сетевых моделях – любой объект (запись, файл) может быть подчинен нескольким объектам. Отличие в топологии иерархической и сетевой модели схематически иллюстрирует рис.22.

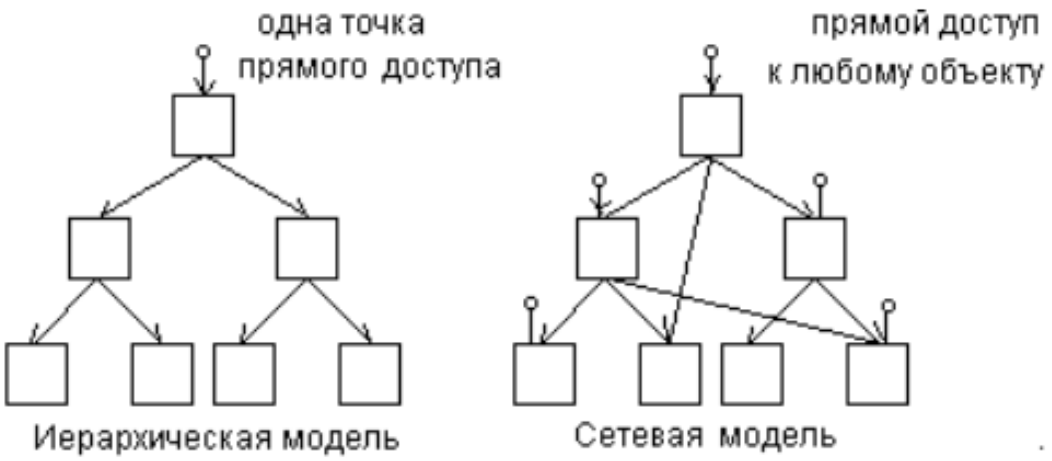


Рис. 22 Иллюстрация особенностей моделей по топологии и доступу

Сетевая модель позволяет создавать связи не только от предка к потомку, но и дополнительные связи между «соседними» предками или «потомками». Это позволяет описывать больше бизнес-процессов организации, чем при использовании иерархической модели.

Например, в предметной области «Аптека» в сетевой модели, представленной на рис. 23, можно учесть следующие процессы предметной области без дополнительного дублирования данных:

- какой сотрудник осуществил продажу того-или иного препарата (связь между множествами «Сотрудник» и «Препарат»);
- какой препарат был продан (связь между множествами «Препарат» и «Продажа»).

Примечание. В сетевой модели учет этих фактов тоже был осуществлен, но с использованием элементов/узлов «Название препарата» и «ФИО» на уровне «Продажа», которые также присутствуют на уровнях «Препарат» и «Сотрудник». То есть, в сетевой модели для более точного описания предметной области необходимо дублировать одну и ту же информацию в базе данных, что приводит к ее избыточности и может

привести к несогласованности данных и нарушению целостности. Иерархическая модель лишена подобных недостатков.

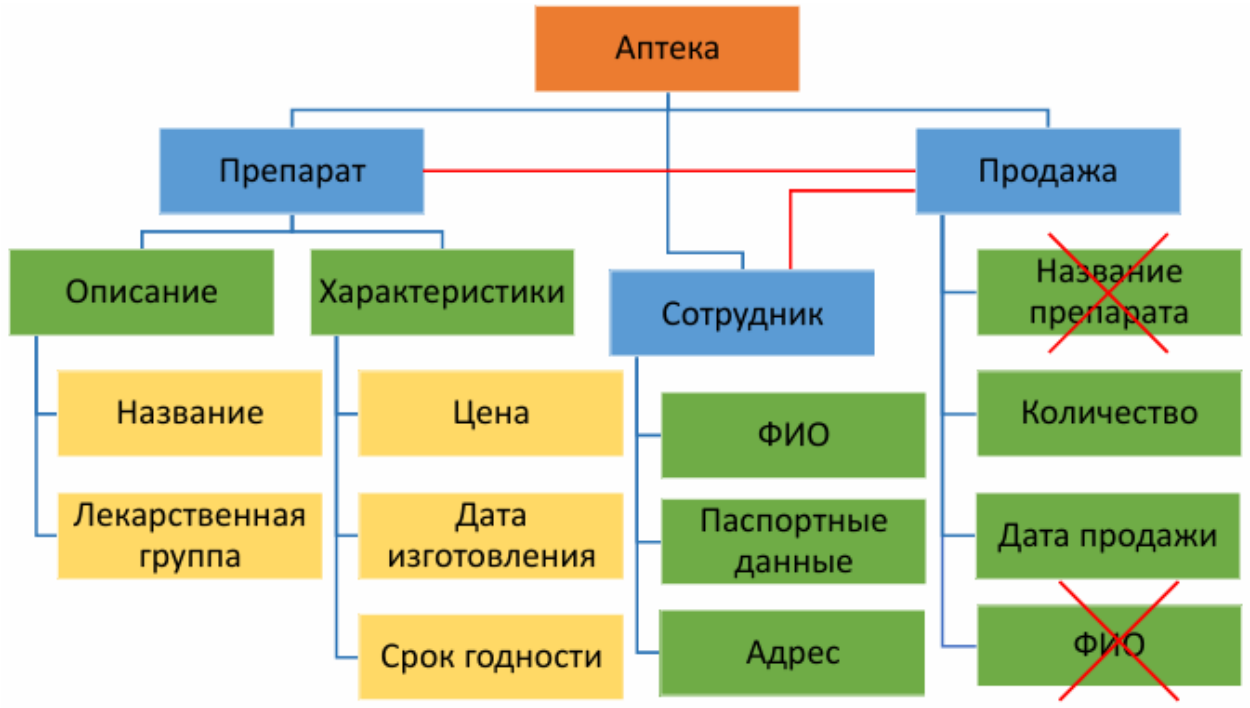


Рис. 23 Пример сетевой модели

Реляционная модель.

Концепция: данные организованы в виде таблиц (отношений). Строки (кортежи) содержат данные, а столбцы (атрибуты) определяют тип данных. Связи между таблицами устанавливаются не физически, а логически, через совпадающие значения ключей.

Аналогия из жизни: электронная таблица Excel, каталог в библиотеке.

Ключевой принцип: декларативный доступ к данным через язык SQL (Structured Query Language). Пользователь описывает что ему нужно, а не как это получить.

Реляционная модель данных является совокупностью простейших двумерных реляционных таблиц-отношений. Связи между двумя логически связанными таблицами в реляционной модели устанавливаются по равенству значений одинаковых атрибутов таблиц-отношений (рис. 24). Таблица-

отношение является универсальным объектом реляционных моделей. Это обеспечивает возможность унификации обработки данных в различных СУБД, поддерживающих реляционную модель. Операции обработки реляционных моделей основаны на использовании универсального аппарата алгебры отношений и реляционного исчисления.

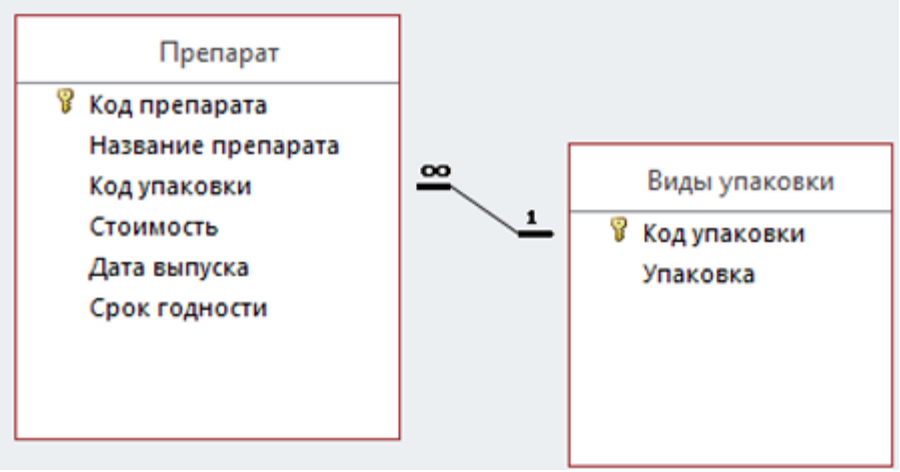


Рис. 24 Фрагмент схемы базы данных «Аптека»

Преимущества реляционных моделей. К достоинствам реляционной модели относятся простота представления данных реляционной модели благодаря табличной форме и минимальная избыточность данных при нормализации таблиц-отношений. В реляционных моделях обеспечивается независимость приложений пользователя от данных, допускающая включение или удаление отношений, изменение атрибутного состава отношений. В отличие от иерархических и сетевых, реляционные базы данных не требуют описания схемы данных и ее генерации, т.е. не требуется настройка СУБД на конкретную структуру БД. Универсальность процедур обработки данных является основой типовых средств в различных реляционных СУБД.

В табл. 1 представлено сравнение моделей на концептуальном уровне.

Таблица 1 – Сравнение моделей на концептуальном уровне

Аспект	Иерархическая	Сетевая	Реляционная
Основа	Дерево	Граф	Таблица (отношение)
Связи	Встроены в структуру записей (физические указатели)	Определены как отдельные «наборы связей»	Реализованы логически через значения атрибутов (ключи)
Гибкость	Низкая. Изменение структуры сложно.	Средняя. Сложнее изменить, чем иерархическую, но связи гибче.	Высокая. Можно легко добавлять таблицы и связи.
Сложность для программиста	Высокая. Требуется знание физической структуры для навигации.	Очень высокая. Сложные алгоритмы обхода графа.	Низкая. Работа через высокоуровневый SQL.
Целостность данных	Жёстко соблюдается структурой.	Зависит от корректности определения наборов связей.	Обеспечивается декларативными ограничениями (PRIMARY KEY, FOREIGN KEY).
Независимость данных	Очень низкая. Приложение тесно связано со структурой.	Низкая.	Высокая. Физическое хранение скрыто от приложения.

Структуры данных реляционной модели.

Структуры данных реляционной модели являются простыми и удобными для восприятия любого пользователя. К ним относятся простейшая двумерная реляционная таблица и ее поименованные столбцы (атрибуты), определяющие элементарное данное.

Реляционная таблица является основным типом структуры данных (объектом) реляционной модели. Структура этой таблицы определяется совокупностью столбцов (полей), для которых определен тип данного.

Основной логической единицей обработки (поиск, выборка, сортировка, вычисления) в реляционной БД является строка таблицы (запись).

Основные свойства реляционной таблицы:

- не может быть двух одинаковых строк;
- в каждой строке содержится по одному значению каждого атрибута.

Столбец соответствует некоторому элементу данных – простому атрибуту, который является простейшей структурой данных. В таблице не могут быть определены множественные элементы, группа или повторяющаяся группа, как в рассмотренных выше иерархических или сетевых моделях. Таким образом, таблица имеет чисто линейную структуру. Имя каждого столбца (атрибута) должно быть уникальным в структуре таблицы, т.е. имена не могут повторяться в одной таблице. Общее число строк не ограничено.

Первичный ключ (ПК) – это один или несколько атрибутов, однозначно идентифицирующих строку. Если первичный ключ состоит из одного атрибута, он называется простым, если из нескольких – составным первичным ключом. По значению первичного ключа может быть найден единственный экземпляр строки.

Вторичный ключ (ВК), в отличие от первичного, – это такой атрибут, значение которого может повторяться в нескольких записях таблицы, т.е. он не является уникальным. По значению вторичного ключа отыскивается несколько строк с одинаковым значением этого ключа. Атрибуты, входящие в состав первичного ключа, являются вторичными ключами (рис. 25).

Рис. 25 Пример внешнего ключа базы данных «Аптека»

Описание предметной области для выполнения лабораторной работы

Рассмотрим студию звукозаписи, которая:

1. Работает с Музыкантами.
2. Музыканты объединяются в Группы.
3. Группы записывают Альбомы в Студиях.
4. В записи одного альбома может участвовать несколько Звукорежиссёров.
5. Каждый альбом состоит из нескольких Треков (песен).
6. Один и тот же Трек может войти в разные сборники (Сборники).

Ключевые связи:

1. Группа состоит из нескольких Музыкантов. Музыкант может играть в нескольких Группах.
2. Альбом записывается одной Группой, но в одной Студии может быть записано много Альбомов.

3. Над одним Альбомом работают несколько Звукорежиссёров, и один Звукорежиссёр работает над многими Альбомами.

4. Альбом содержит несколько Треков. Один Трек принадлежит одному Альбому.

5. Трек может входить в несколько Сборников, и один Сборник содержит много Треков.

Последовательность выполнения лабораторной работы

1. Смоделировать иерархическую структуру.
   - Выбрать корневой элемент иерархии.
   - Построить схему дерева, отобразив все сущности.
   - Выявить и описать проблемы представления сложных связей.
2. Смоделировать сетевую структуру.
   - Определить типы записей.
   - Определить типы связей (наборы) между ними.
   - Нарисовать схему в виде графа.
3. Смоделировать реляционную структуру.
   - Определить таблицы, их атрибуты и ключи.
   - Нарисовать простую ER-диаграмму, отобразив связи между таблицами.
4. Провести сравнительный анализ.
   - Составить итоговую сравнительную таблицу.
   - Сформулировать вывод о применимости каждой модели к данной задаче.

Лабораторная работа №5 «Функциональный анализ предметной области»

Теоретические сведения

Проектирование базы данных – это процесс создания структуры данных, которая будет эффективно и безошибочно хранить информацию. Он состоит из ключевых этапов:

1. Сбор и анализ требований: что должна делать система?
2. Концептуальное проектирование: создание модели данных, не зависящей от конкретной СУБД. Результат – ER-диаграмма (Entity-Relationship, сущность-связь).
3. Логическое проектирование: преобразование концептуальной модели в схему данных для конкретного типа БД (реляционная, документная и т.д.). Определение таблиц, столбцов, типов данных, первичных и внешних ключей.
4. Физическое проектирование: оптимизация для конкретной СУБД (индексы, разделение данных и т.д.).

Функциональный анализ – это ядро первого и второго этапов. Мы отвечаем на вопросы: «Что?» и «Как связано?».

Ключевые компоненты концептуальной модели.

1. Сущность (Entity).

Это значимый объект или событие в предметной области, экземпляры которого можно отличить друг от друга.

Критерий: о сущности можно сказать: «О ней мы храним информацию в базе данных».

Примеры для ПО «Поликлиника»: Пациент, Врач, Прием, Диагноз, Лекарство.

Важно: сущность – это не конкретный человек «Иванов», а тип объектов «Пациент». Конкретный «Иванов» – это экземпляр сущности.

2. Атрибут (Attribute).

Это элементарная характеристика (свойство, поле), описывающая сущность.

Классификация атрибутов:

- Простой (атомарный): не разлагается на более мелкие части (например, Дата_рождения, Фамилия).
- Составной: может быть разделен на части (например, Адрес = Город + Улица + Дом + Квартира). На концептуальном уровне его можно указывать как единое целое.
- Однозначный: имеет одно значение для экземпляра (например, Серия_паспорта).
- Многозначный: может иметь несколько значений для одного экземпляра (например, Номер_телефона у пациента). В реляционной модели это недопустимо! Такой атрибут нужно выносить в отдельную сущность (например, КонтактныйТелефон).
- Вычисляемый (производный): его значение можно получить из других атрибутов (например, Возраст вычисляется из Даты_рождения и текущей даты). На концептуальной модели его обычно не указывают.
- Первичный ключ (Primary Key, PK): атрибут или минимальный набор атрибутов, однозначно идентифицирующий каждый экземпляр сущности. Значение PK не может повторяться и не может быть NULL (пустым).

Примеры первичного ключа: Номер_паспорта (для Пациента), Код_врача (для Врача), Номер_приема (для Приема).

Как выбрать? Ищем уникальный идентификатор (документ, код) или создаем искусственный (ID_Пациента).

3. Связь (Relationship).

Это логическое отношение между двумя или более сущностями. Связь имеет имя (обычно глагол) и тип (кардинальность) (табл. 2).

Таблица 2 – Типы связей

Тип связи	**Обозначение**	**Суть**	**Жизненный пример**
Один к одному	1:1	Экземпляр А связан не более чем с одним экземпляром Б, и наоборот.	Пациент – История_болезни. У пациента одна уникальная история, и история принадлежит одному пациенту.
Один ко многим	1:М	Один экземпляр А связан с несколькими экземплярами Б. Каждый экземпляр Б связан только с одним А.	Врач – Прием. Один врач проводит много приемов, но каждый прием ведет один конкретный врач.
Многие ко многим	М:М	Один экземпляр А связан с несколькими Б, и один экземпляр Б связан с несколькими А.	Пациент – Аллергия. У пациента может быть много аллергий (на пыльцу, на арахис). И одна аллергия (например, на арахис) может быть у многих пациентов.

Важное правило: связь «Многие ко многим» (М:М) на этапе логического проектирования обязательно устраняется путем создания промежуточной (ассоциативной) сущности. Для примера выше: Пациент –(1:М)–> Диагноз_Аллергия <–(М:1)– Вид_Аллергии.

Алгоритм выполнения функционального анализа.

1. Выписать все существительные из описания предметной области.

2. Отфильтровать их: отбросить синонимы, названия атрибутов (например, «фамилия» – это атрибут, а не сущность) и незначительные объекты.

3. Для каждой сущности:

– Составить список всех ее свойств (атрибутов).

– Выделить первичные ключи.

4. Проанализировать глаголы в описании. Они часто указывают на связи.

5. Для каждой пары связанных сущностей определить тип связи (1:1, 1:M, M:M), задавая контрольные вопросы: «Сколько А может быть связано с одним Б?» и «Сколько Б может быть связано с одним А?».

6. Проверить целостность: все ли сущности связаны? Все ли ключевые атрибуты уникальны?

Последовательность выполнения лабораторной работы

1. Изучите описание предметной области согласно индивидуальному варианту.

2. Выделите ключевые объекты (сущности), о которых идет речь.

3. Для каждой сущности определите:

– Список атрибутов.

– Первичный ключ.

4. Определите связи между выделенными сущностями.

Варианты индивидуальных заданий

Вариант 1. Автосервис.

Описание: система учета заказов на ремонт автомобилей, учета запчастей и клиентов.

Сущности:

Клиент: ФИО, Номер_телефона, Адрес_электронной_почты, Адрес.

Автомобиль: Гос_номер, Марка, Модель, Год_выпуска, VIN, Владелец.

Заказ-наряд: Номер_заказа, Дата_приема, Проблема_со_слов_клиента, Автомобиль, Статус (принят/в работе/готов).

Работа: Название_работы, Норма_часов, Стоимость_нормочаса.

Запчасть: Название, Производитель, Цена, Количество_на_складе.

ВыполненнаяРабота: Номер_Заказа, Код_Работы, Фактическое_время.

ИспользованнаяЗапчасть: Номер_Заказа, Запь, Количество.

Вариант 2. Онлайн-доставка еды.

Описание: агрегатор ресторанов. Клиенты выбирают блюда из меню разных ресторанов, оформляют заказ, который доставляется курьером.

Сущности:

Клиент: Номер_телефона, Имя, Адрес_доставки, История_заказов.

Ресторан: Название, Кухня, Адрес, Время_работы, Рейтинг.

Блюдо: Название, Описание, Цена, Категория, Ресторан.

Курьер: ФИО, Номер_телефона, Транспорт, Статус.

Заказ: Номер_заказа, Клиент, Курьер, Дата_время_заказа, Адрес_доставки, Статус, Сумма.

ПозицияЗаказа: Заказ, Блюдо, Количество, Цена_на_момент_заказа.

Вариант 3. Учет оборудования в колледже.

Описание: инвентаризация компьютерной техники и проекторов в аудиториях и лабораториях.

Сущности:

Аудитория: Номер_аудитории, Тип, Ответственный_преподаватель, Площадь.

ТипОборудования: Название, Характеристики.

ЕдиницаОборудования: Инвентарный_номер, Тип, Модель, Серийный_номер, Дата_поступления, Статус.

Размещение: Оборудование, Аудитория, Дата_установки, Дата_перемещения.

Ремонт: Оборудование, Дата_неисправности, Описание_проблемы, Дата_устранения, Стоимость.

Вариант 4. Система тестирования знаний.

Описание: платформа для создания тестов по дисциплинам, их прохождения студентами и анализа результатов.

Сущности:

Преподаватель: ФИО, Должность, Кафедра.

Дисциплина: Название.

Тест: Название_теста, Дисциплина, Создатель, Дата_создания, Макс_балл.

Вопрос: Текст_вопроса, Тип, Балл_за_правильный_ответ, Тест.

ВариантОтвета: Вопрос, Текст_варианта, Флаг_правильности.

Студент: Номер_зачетки, ФИО, Группа.

ПопыткаСдачи: Студент, Тест, Дата_время_начала, Дата_время_окончания, Набранный_балл.

ОтветСтудента: Попытка, Вопрос, Данный_ответ.

Вариант 5. Ветеринарная клиника.

Описание: учет пациентов (животных), их владельцев, визитов, диагнозов и назначений.

Сущности:

Владелец: ФИО, Адрес, Номер_телефона.

Питомец: Кличка, Вид, Порода, Дата_рождения, Владелец.

Врач: ФИО, Специализация, Лицензия, График_работы.

Визит: Питомец, Врач, Дата_время, Жалобы, Предварительный_диагноз.

Диагноз: Название, Описание.

Назначение: Визит, Диагноз, Лечение, Рекомендации.

Процедура: Название, Стоимость.

ПроведеннаяПроцедура: Визит, Процедура.

Вариант 6. Бронирование гостиничных номеров.

Описание: система управления бронированиями для небольшой гостиницы. Учитываются гости, типы и конкретные номера, а также факты их бронирования с указанием сроков.

Сущности:

Гость: Серия_номер_паспорта, ФИО, Номер_телефона, Гражданство.

Тип_номера: Название, Вместимость_чел, Площадь, Базовая_цена_за_сутки, Описание.

Номер: Номер_комнаты, Этаж, Тип_номера, Статус.

Бронирование: Номер_брони, Гость, Номер_комнаты, Дата_заезда, Дата_выезда, Дата_создания_брони, Статус, Предоплата.

Вариант 7. Запись в парикмахерскую/салон красоты.

Описание: система онлайн-записи для салона. Клиенты записываются к конкретным мастерам на определенные услуги в выбранное время.

Сущности:

Клиент: Номер_телефона, Имя, Предпочтения.

Мастер: ФИО, Специализация, График_работы, Стаж.

Услуга: Название, Длительность_мин, Цена, Мастер.

Расписание_записей: Клиент, Услуга, Дата, Время_начала, Время_окончания, Статус.

Вариант 8. Прокат спортинвентаря.

Описание: пункт сезонного проката. Учет инвентаря, клиентов, выдачи и возврата снаряжения с залогом.

Сущности:

Инвентарь: Инвентарный_номер, Тип, Модель, Размер/Рост, Состояние, Стоимость_проката_в_день.

Клиент: Серия_номер_паспорта, ФИО, Номер_телефона.

Договор_проката: Номер_договора, Клиент, Дата_выдачи, Планируемая_дата_возврата.

Позиция_проката: Договор_проката, Инвентарь, Залоговая_сумма, Фактическая_дата_возврата, Отметка_о_повреждении.

Вариант 9. Учет посещаемости студентов.

Описание: электронный журнал для фиксации присутствия студентов на занятиях.

Сущности:

Группа: Название, Специальность, Год_поступления.

Студент: Номер_зачетной_книжки, ФИО, Группа.

Дисциплина: Название, Преподаватель, Часы.

Занятие: Дисциплина, Дата, Номер_пары, Тема.

Посещаемость: Занятие, Студент, Статус.

Вариант 10. Агентство недвижимости (аренда).

Описание: учет объектов аренды, их владельцев, арендаторов и заключенных договоров.

Сущности:

Объект_недвижимости: Адрес, Тип, Площадь_кв_м, Количество_комнат, Месячная_арендная_плата, Владелец.

Владелец: ФИО, Паспортные_данные, Контактный_телефон, Доля_в_праве.

Арендатор: ФИО, Паспортные_данные, Место_работы, Контактный_телефон.

Договор_аренды: Номер_договора, Объект_недвижимости, Арендатор, Дата_начала, Дата_окончания, Статус, Комиссия_агентства.

Вариант 11. Фитнес-центр.

Описание: система учета клиентов, их абонементов, тренеров и групповых занятий.

Сущности:

Клиент: Номер_карты, ФИО, Дата_рождения, Номер_телефона, Адрес_электронной_почты.

Абонемент: Номер_абонемента, Клиент, Тип, Дата_начала, Дата_окончания, Статус.

Тренер: ФИО, Специализация, График_работы, Стаж.

Групповое_занятие: Название, Время_проведения, Зал, Максимальное_количество_участников, Тренер.

Запись_на_занятие: Клиент, Групповое_занятие, Дата_занятия.

Вариант 12. Пункт приема вторсырья.

Описание: учет сданного сырья, клиентов, расчетов за приемку.

Сущности:

Клиент: ФИО, ИНН/Паспорт, Расчетный_счет.

Вид_вторсырья: Название, Единица_измерения, Цена_за_единицу.

Акт_приемки: Номер_акта, Клиент, Дата_приемки, Сотрудник_приемщик, Общая_сумма.

Позиция_в_акте: Акт_приемки, Вид_вторсырья, Количество, Сумма_за_позицию.

Вариант 13. Кинотеатр.

Описание: учет расписания сеансов, продажи билетов, фильмов и кинозалов.

Сущности:

Фильм: Название, Длительность_мин, Жанр, Возрастное_ограничение, Режиссер, Страна.

Зал: Номер_зала, Тип, Количество_рядов, Количество_мест_в_ряду.

Сеанс: Фильм, Зал, Дата_и_время_начала, Цена_билета.

Билет: Уникальный_номер_билета, Сеанс, Ряд, Место, Статус.

Вариант 14. Библиотека.

Описание: система учета книжного фонда с учетом отдельных экземпляров, читателей и истории выдач.

Сущности:

Книга: ISBN_или_Шифр, Название, Автор, Издательство, Год_издания.

Экземпляр_книги: Инвентарный_номер, Книга, Дата_поступления, Состояние, Место_хранения.

Читатель: Номер_читательского_билета, ФИО, Группа_или_должность, Адрес, Телефон.

Выдача: Экземпляр_книги, Читатель, Дата_выдачи, Плановый_срок_возврата, Фактическая_дата_возврата, Штраф_за_просрочку.

Вариант 15. Рекрутинговое агентство.

Описание: база данных для подбора персонала: соискатели, вакансии от компаний, история собеседований.

Сущности:

Соискатель: ФИО, Профессия, Опыт_работы_лет, Резюме, Желаемая_зарплата.

Компания-заказчик: Название, Сфера_деятельности, Контактное_лицо.

Вакансия: Должность, Требования, Предлагаемая_зарплата, Статус, Компания.

Собеседование: Соискатель, Вакансия, Дата_время, Формат, Результат.

Вариант 16. Организация конференции.

Описание: регистрация участников, формирование секций и расписания докладов.

Сущности:

Участник: ФИО, Организация, Должность, Адрес_электронной_почты, Статус.

Секция: Название_секции, Время_начала, Время_окончания, Аудитория, Модератор.

Доклад: Название, Аннотация, Участник, Секция, Порядковый_номер_в_секции.

Регистрация: Участник, Секция.

Вариант 17. Музейный учет экспонатов.

Описание: инвентаризация музейных предметов, их перемещение по залам, учет реставраций.

Сущности:

Экспонат: Инвентарный_номер, Название, Эпоха/Период, Материал, Техника, Автор, Дата_поступления_в_фонд.

Зал: Название_зала, Тематическая_экспозиция, Этаж.

Место_хранения: Экспонат, Зал, Дата_размещения, Дата_изъятия, Витрина.

Реставрация: Экспонат, Дата_начала, Дата_окончания, Реставратор, Описание_работ, Стоимость.

Вариант 18. Заказ такси.

Описание: система приема заказов такси с диспетчером, учетом водителей, автомобилей и автоматическим расчетом стоимости.

Сущности:

Клиент: Номер_телефона, Имя, Рейтинг_клиента.

Водитель: ФИО, Лицензия, Марка_модель_авто, Гос_номер, Текущий_статус, Рейтинг.

Тариф: Название, Стоимость_посадки, Стоимость_за_км, Стоимость_за_мин_простоя.

Заказ: Номер_заказа, Клиент, Водитель, Тариф, Адрес_подачи, Адрес_назначения, Время_подачи, Время_начала_поездки, Время_окончания, Фактическое_расстояние, Итоговая_стоимость, Статус.

Вариант 19. Планировщик проектов.

Описание: управление задачами в рамках проектов, назначение ответственных сотрудников.

Сущности:

Проект: Название_проекта, Описание, Дата_начала_план, Дата_окончания_план, Статус.

Задача: Название_задачи, Описание, Статус, Приоритет, Дата_начала_факт, Дата_окончания_факт, Проект.

Сотрудник: Табельный_номер, ФИО, Должность, Отдел.

Исполнитель: Задача, Сотрудник, Роль.

Вариант 20. Служба доставки посылок.

Описание: отслеживание пути посылки через сортировочные центры.

Сущности:

Отделение/Склад: Индекс_отделения, Адрес, Тип.

Посылка: Трек_номер, Отправитель, Получатель, Вес, Объявленная_ценность, Текущий_статус, Отделение_отправления, Отделение_назначения.

Событие_маршрута: Посылка, Отделение, Дата_время, Тип_события.

Вариант 21. Аптека.

Описание: учет лекарств, поставок от производителей, продаж и остатков на складе.

Сущности:

Лекарство: МНН, Торговое_название, Форма_выпуска, Дозировка, Производитель.

Поставщик: Название_компании, ИНН, Контактное_лицо, Телефон.

Поставка: Номер_накладной, Поставщик, Дата_поставки, Сумма_накладной.

Позиция_поставки: Поставка, Лекарство, Количество, Цена_закупки, Срок_годности_партии.

Продажа: Номер_чека, Дата_продажи, Кассир.

Позиция_чека: Продажа, Лекарство, Количество, Цена_продажи.

Вариант 22. Фермерское хозяйство.

Описание: планирование и учет сельскохозяйственных работ на полях.

Сущности:

Поле: Номер_поля, Площадь_га, Культура_текущая, Тип_почвы.

Работник: ФИО, Должность, Квалификация.

Техника: Инвентарный_номер, Тип, Модель, Статус.

Вид_работ: Название, Норма_времени_на_га.

Факт_выполнения_работ: Поле, Вид_работ, Работник, Техника, Дата, Затраченное_время, Объем_выполнено_га.

Вариант 23. Клуб по интересам.

Описание: учет членов клуба, абонементов, расписания занятий и посещений.

Сущности:

Участник_клуба: ФИО, Дата_рождения, Контактный_телефон, Уровень_подготовки.

Абонемент: Номер_абонемента, Участник_клуба, Тип, Дата_начала, Дата_окончания, Остаток_занятий.

Тренер/Преподаватель: ФИО, Стиль/Направление, Стаж.

Занятие_по_расписанию: Название, Тренер, День_недели, Время_начала, Продолжительность.

Посещение: Участник_клуба, Занятие_по_расписанию, Дата, Списано_с_абонемента.

Вариант 24. Прокат игровых приставок и игр.

Описание: учет парка консолей, игровых дисков, клиентов и фактов аренды комплектов.

Сущности:

Игровая_консоль: Серийный_номер_консоли, Модель, Комплектация, Состояние.

Игра: Название, Жанр, Возрастной_рейтинг, Платформа.

Диск_с_игрой: Уникальный_код_диска, Игра, Состояние.

Клиент: Паспортные_данные, ФИО, Телефон, Залоговая_способность.

Аренда_комплекта: Клиент, Консоль, Дата_выдачи, Плановый_срок_возврата.

Игры_в_аренде: Аренда_комплекта, Диск_с_игрой.

Вариант 25. Онлайн-курс.

Описание: система для размещения курсов, регистрации студентов и отслеживания прогресса.

Сущности:

Курс: Название, Автор, Краткое_описание, Цена, Статус.

Модуль: Название_модуля, Порядковый_номер, Курс.

Урок: Название_урока, Тип, Ссылка_на_материал, Длительность, Модуль.

Студент: Адрес_электронной_почты, Имя, Дата_регистрации.

Запись_на_курс: Студент, Курс, Дата_записи, Статус_доступа.

Прогресс_изучения: Запись_на_курс, Урок, Статус, Дата_последнего_действия.

Вариант 26. Спортивная лига.

Описание: учет команд, игроков, календаря матчей и их результатов.

Сущности:

Команда: Название, Город, Год_основания, Тренер.

Игрок: ФИО, Дата_рождения, Амплуа, Команда.

Стадион: Название, Адрес, Вместимость.

Матч: Команда_хозяев, Команда_гостей, Стадион, Дата_и_время, Статус.

Результат_матча: Матч, Счет_хозяева, Счет_гости, Исход.

Вариант 27. Организация мероприятий.

Описание: планирование мероприятий, подбор исполнителей и расчет смет.

Сущности:

Мероприятие: Название, Тип, Дата, Клиент, Бюджет.

Клиент: ФИО/Название_компании, Контактный_телефон, Адрес_электронной_почты.

Исполнитель: Название_компании, Тип_услуг, Контактное_лицо, Рейтинг.

Смета: Мероприятие, Исполнитель, Услуга, Количество, Цена_за_единицу, Общая_стоимость.

Вариант 28. Учет личных финансов.

Описание: приложение для учета доходов и расходов по категориям и счетам.

Сущности:

Счет: Название_счета, Валюта, Текущий_баланс.

Категория: Название, Тип, Родительская_категория.

Транзакция: Дата, Сумма, Валюта, Счет, Категория, Контрагент, Комментарий, Тип_операции.

Вариант 29. Агрегатор репетиторов.

Описание: площадка для поиска репетиторов, регистрации учеников и отзывов.

Сущности:

Репетитор: ФИО, Предмет, Образование, Опыт_работы_лет, Ставка_в_час, Рейтинг.

Ученик: ФИО_родителя, ФИО_ученика, Класс, Предмет, Цель_занятий.

Заявка_ученика: Ученик, Репетитор, Дата_заявки, Статус.

Отзыв: Репетитор, Ученик, Дата, Текст_отзыва, Оценка.

Вариант 30. Система заявок в IT-отдел.

Описание: система регистрации и обработки заявок от сотрудников компании на IT-обслуживание.

Сущности:

Сотрудник_компании: Табельный_номер, ФИО, Отдел, Должность, Контактный_телефон, Адрес_электронной_почты.

IT-специалист: ФИО, Должность, Уровень_поддержки.

Тип_проблемы: Название, Стандартное_время_реакции.

Заявка: Номер_заявки, Сотрудник_компании, Тип_проблемы, Дата_создания, Подробное_описание, Срочность, Статус, IT-специалист, Решение_комментарий.

Лабораторная работа №6 «Атрибутивный анализ предметной области»

Теоретические сведения

Атрибутивный анализ – это углубление и детализация результатов функционального анализа. На этом этапе мы подробно описываем каждую характеристику (атрибут) выделенных ранее сущностей, подготавливая данные для перехода к логической модели (созданию таблиц).

Ключевые понятия:

Атрибут (Поле, Столбец) – минимальная единица информации, которая характеризует объект (сущность). Пример: для сущности Студент атрибутами являются Фамилия, Имя, Дата_рождения.

Домен – множество всех допустимых значений атрибута. Фактически, это тип данных и возможные ограничения.

Пример домена для атрибута Оценка: целое число от 2 до 5.

Пример домена для атрибута Электронная_почта: текстовые строки, содержащие символ '@'.

Характеристики атрибута (необходимо определить для каждого):

1. Тип данных: основная классификация хранимой информации.
   - Числовой (целые, дробные): для количественных данных (Цена, Количество, Возраст).
   - Строковый (Текстовый): для имен, названий, описаний (ФИО, Адрес, Название_книги).
   - Логический (Да/Нет, Истина/Ложь): для флагов и состояний (Оплачено, Активен, Есть_в_наличии).
   - Дата/Время: для хранения временных отметок (Дата_рождения, Дата_заказа).
   - Прочие (Денежный, Двоичный для фото).
2. Размер/Формат: уточнение к типу данных.
   - Для чисел: точность (например, 10,2 – 10 цифр, 2 после запятой).

– Для строк: максимальная длина (например, 100 символов).
– Для дат: формат (ДД.ММ.ГГГГ).

3. Обязательность (NULL/NOT NULL): может ли поле быть пустым при создании записи?
   – NOT NULL – поле обязательно для заполнения (например, Фамилия).
   – NULL – поле может быть пустым (например, Отчество, Дополнительный_телефон).
4. Уникальность: должны ли значения в этом атрибуте повторяться в разных записях?
   – Уникальный – значения не могут повторяться (например, Паспортные_данные).
   – Неуникальный – повторения допустимы (например, Город_проживания).

Ключи сущности:

Первичный ключ (Primary Key, PK) – атрибут или минимальная комбинация атрибутов, однозначно и уникально идентифицирующая каждый экземпляр (строку) сущности.

Требования: уникальность, непустое значение (NOT NULL), стабильность (редко меняется).

Виды:

– Естественный (натуральный): уже существует в предметной области (Паспорт, ISBN, Инвентарный_номер).
– Суррогатный (искусственный): создается искусственно, обычно целочисленный с автоинкрементом (ID_Клиента, Код_Заказа). Самый надежный вариант.

Альтернативный ключ: атрибут, который мог бы быть первичным ключом (уникален), но не выбран в этом качестве (например, Номер_паспорта при PK ID_Клиента).

Внешний ключ (Foreign Key, FK) – атрибут (или набор атрибутов) в одной сущности, который является первичным ключом в другой сущности. Он устанавливает и поддерживает связь между сущностями.

Пример: в сущности Заказ атрибут ID_Клиента является FK, ссылающимся на PK ID_Клиента в сущности Клиент.

Результатом атрибутивного анализа является детальная спецификация сущностей, которая представляет собой таблицу со следующими колонками для каждого атрибута:

1. Наименование сущности.
2. Наименование атрибута.
3. Описание/Назначение.
4. Тип данных и размер.
5. Обязательность (Да/Нет).
6. Уникальность (Да/Нет).
7. Примечание (PK, FK, альтернативный ключ и т.д.).

Пример выполнения лабораторной работы

Тема: «Библиотека колледжа».

Исходные данные (сущности из предыдущей лабораторной работы):

1. Книга (издание).
2. Экземпляр_книги (физическая копия)..
3. Читатель.
4. Выдача.

Шаг 1. Детализация атрибутов для каждой сущности.

Таблица 3 – Сущность «Книга»

Наименование атрибута	Описание	Тип данных и размер	Обя зат.	Уникальн.	Примечание
ISBN	Международный стандартный номер	Строка (17 символов)	Да	Да	PK
Название	Полное название книги	Строка (200 символов)	Да	Нет	
Автор	ФИО автора(ов)	Строка (150 символов)	Да	Нет	
Издательство	Название издательства	Строка (100 символов)	Нет	Нет	
Год_издания	Год публикации	Целое число (4 цифры)	Нет	Нет	
Аннотация	Краткое описание содержания	Текст (неогр. длина)	Нет	Нет	

Таблица 4 – Сущность «Экземпляр_книги»

Наименование атрибута	Описание	Тип данных и размер	Обя зат.	Уникальн.	Примечание
Инвентарный_номер	Уникальный номер экземпляра	Целое число	Да	Да	PK
ISBN	Ссылка на издание	Строка (17 символов)	Да	Нет	FK (Книга.ISBN)
Дата_поступления	Когда экземпляр поступил в фонд	Дата (ДД.ММ.ГГГГ)	Да	Нет	
Состояние	Физическое состояние	Строка (20 символов)	Да	Нет	('новое', 'хор', 'ветхое')
Стоимость	Балансовая стоимость	Число (10,2)	Нет	Нет	
Место_хранения	Стеллаж и полка	Строка (50 символов)	Да	Нет	

Таблица 5 – Сущность «Читатель»

Наименование атрибута	**Описание**	**Тип данных и размер**	**Обязат.**	**Уникальн.**	**Примечание**
Номер_билета	Номер читательского билета	Целое число	Да	Да	PK
Фамилия		Строка (50 символов)	Да	Нет	
Имя		Строка (50 символов)	Да	Нет	
Отчество		Строка (50 символов)	Нет	Нет	
Группа	Учебная группа (для студентов)	Строка (10 символов)	Нет	Нет	
Должность	Должность (для преподавателей)	Строка (50 символов)	Нет	Нет	
Телефон	Контактный телефон	Строка (15 символов)	Нет	Нет	
Адрес_электрон_почты		Строка (100 символов)	Нет	Нет	
Дата_регистрации	Дата выдачи билета	Дата (ДД.ММ.ГГГГ)	Да	Нет	

Таблица 6 – Сущность «Выдача»

Наименование атрибута	**Описание**	**Тип данных и размер**	**Обязат.**	**Уникальн.**	**Примечание**
ID_Выдачи	Уникальный номер операции	Целое число	Да	Да	PK

Наименование атрибута	Описание	Тип данных и размер	Обязат.	Уникальн.	Примечание
Инвентарный_номер	Какой экземпляр выдан	Целое число	Да	Нет	FK (Экземпляр_книги.Инв_номер)
Номер_билета	Кому выдан	Целое число	Да	Нет	FK (Читатель.Номер_билета)
Дата_выдачи	Дата выдачи книги читателю	Дата и Время	Да	Нет	
Срок_возврата	Планируемая дата возврата	Дата (ДД.ММ.ГГГГ)	Да	Нет	
Дата_возврата_факт	Фактическая дата возврата	Дата (ДД.ММ.ГГГГ)	Нет	Нет	
Отметка_о_штрафе	Наличие штрафа за просрочку	Логический (Да/Нет)	Да	Нет	По умолчанию 'Нет'

Вывод по примеру: В ходе атрибутивного анализа были детально описаны все характеристики сущностей предметной области «Библиотека». Для каждой сущности определен первичный ключ, типы данных и ограничения для атрибутов. Установлены связи между сущностями через внешние ключи. Полученная спецификация готова для создания физической модели базы данных.

Последовательность выполнения лабораторной работы

1. Возьмите за основу результат своей лабораторной работы по функциональному анализу (список сущностей и связей).
2. Для каждой сущности из вашего списка выполните следующее:

2.1 Выпишите все атрибуты, которые вы ранее определили.

2.2 Детализируйте каждый атрибут:

- Дайте краткое текстовое описание его назначения.
- Определите тип данных и его размер/формат.
- Укажите, является ли атрибут обязательным для заполнения (Да/Нет).
- Укажите, должны ли значения атрибута быть уникальными (Да/Нет).

2.3 Определите ключи:

- Выделите первичный ключ (PK). Обоснуйте выбор.
- Если есть другие уникальные атрибуты, отметьте их как альтернативные ключи.
- Найдите атрибуты, которые ссылаются на первичные ключи других сущностей. Помечайте их как внешние ключи (FK) с указанием, на какую сущность и какой атрибут они ссылаются.

3. Оформите отчет в виде таблицы (см. пример выше) для всех сущностей вашего варианта.

4. Сделайте вывод о проделанной работе.

Лабораторная работа №7 «Моделирование ER-диаграмм»

Теоретические сведения

ER-диаграмма (Entity-Relationship Diagram) – это графическая модель, которая наглядно отображает сущности предметной области, их атрибуты и связи между ними. Это основной инструмент визуализации концептуальной модели данных.

Типы ER‑ моделей.

1. Концептуальный уровень.

Первая верхнеуровневая модель для представления новой предметной области будущего проекта: что в ней есть и с чем нужно работать (табл. 7). Например, в ПО для транспортной компании будут сущности «Транспорт», «Груз», «Маршрут», «Накладная».

Таблица 7 – Концептуальная модель

Характеристика	Описание
Аудитория	Заказчики, бизнес-аналитики, предметные эксперты
Уровень абстракции	Высокий, близкий к естественному языку
Что отображается	• Ключевые сущности (объекты) • Связи между ними с типами (1:1, 1:M, M:N) • Основные атрибуты (без деталей)
Что НЕ отображается	• Первичные/внешние ключи • Типы данных • Технические ограничения
Нотация	Простая нотация «Воронья лапка», часто без атрибутов внутри сущностей
Результат	Общее понимание структуры данных всеми участниками проекта

ER-модель концептуального уровня нужна системному аналитику и заказчику, чтобы проверить, все ли термины учтены. Поэтому системный

аналитик, как правило, создаёт её самостоятельно и не привлекает технических специалистов из команды разработки.

2. Логический уровень

На этом уровне детализируют данные из концептуальной модели: к сущностям добавляют характеристики – атрибуты (табл. 8). Например, на логическом уровне описывают характеристики сущности «Транспорт»: марка и модель автомобиля, количество лошадиных сил, пробег, грузоподъёмность.

Таблица 8 – Логическая модель

Характеристика	Описание
Аудитория	Аналитики, проектировщики БД, разработчики
Уровень абстракции	Средний, переход от бизнес-понятий к технической реализации
Что отображается	• Все сущности • Все атрибуты с указанием PK/FK • Связи (уже без M:N - они устранены) • Типы данных (общие: число, строка, дата)
Что НЕ отображается	• Конкретные типы данных СУБД (varchar(50), int) • Индексы, партиции, физическое размещение
Нотация	Полноценная нотация «Воронья лапка» со всеми атрибутами
Результат	Техническая спецификация для создания БД в любой реляционной СУБД

Модель логического уровня тоже составляет системный аналитик, но уже не в одиночку. К работе подключают технических специалистов — разработчика или архитектора баз данных. Готовую логическую ER-модель нужно презентовать команде разработки. Разработчики проверяют, чтобы аналитик ничего не упустил, и согласовывают модель.

3. Физический уровень

На этом уровне описывают, как будет организована работа с данными: выбирают тип базы, её содержание и где данные будут хранить (табл. 9).

Например, выбирают реляционный тип базы данных и СУБД для работы с ней, перечисляют таблицы в базе и определяют, что она будет храниться на внутреннем сервере компании.

Таблица 9 – Физическая модель

Характеристика	Описание
Аудитория	Разработчики БД, администраторы БД
Уровень абстракции	Низкий, техническая реализация
Что отображается	• Таблицы (вместо сущностей) • Столбцы с конкретными типами данных СУБД • Ограничения (NOT NULL, UNIQUE, CHECK) • Индексы (первичные, внешние, дополнительные) • Триггеры, хранимые процедуры (опционально) • Оценка объема данных
Что НЕ отображается	Бизнес-смысл связей (остаются только FK)
Нотация	Часто используется диаграмма таблиц с указанием индексов, либо прямое описание в SQL
Результат	Готовые SQL-скрипты для создания БД

Над ER-моделью физического уровня в большей степени работают архитектор баз данных и разработчики, а системный аналитик только помогает в процессе.

Символы и нотации ER- диаграмм.

ER-модель – это общее представление данных, ER-диаграмма – представление модели, а нотация – графический язык для представления модели. Символы, с помощью которых описывают модель, – это нотации.

Для того чтобы построить ER-диаграмму, можно использовать разные нотации. Три самые известные из них:

1. Нотация IDEF1X. Её относят к фундаментальным, но на практике давно не используют, потому что есть более удобные варианты.

2. Нотация Чена. Классическая нотация, которая состоит из простых символов – прямоугольников, овалов и линий. Из-за этого нотацию часто используют для концептуальных моделей, которые презентуют заказчику. Человеку, который далёк от аналитики данных, проще разобраться в понятных диаграммах со знакомыми символами.

3. Нотация Мартина. Её ещё называют «воронья лапка» (от англ. Crow's Foot). Она компактнее нотации Чена, поэтому её используют для построения ER-моделей логического уровня, когда нужно описать в модели все атрибуты сущностей.

В нотациях Чена и Мартина есть одинаковые элементы: сущности, атрибуты и связи. Но эти элементы диаграмм обозначают разными символами (рис. 26).

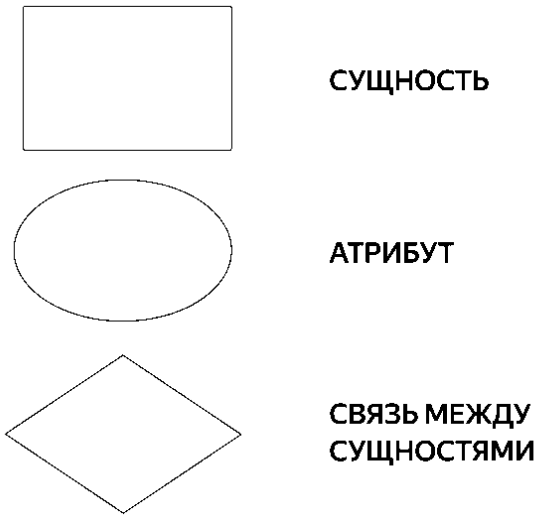


Рис. 26 Графические обозначения элементов нотации Чена

Элементы ER-диаграммы в нотации Чена соединяют линиями. Если линия соединяет две сущности, сверху обозначают тип связи:

– 1:1 – «один-к-одному»;
– 1:M – «один-ко-многим»;
– M:M – «многие-ко-многим».

В нотации Мартина сущность также вписывают в прямоугольник, а атрибуты и связи обозначают по-другому (рис. 27):

– атрибуты перечисляют прямо под сущностью,

– связи рисуют разными соединительными линиями.

«Воронья лапка» (Crow's Foot Notation) – это одна из самых популярных и наглядных нотаций (системы символов и правил) для построения ER-диаграмм (Entity-Relationship, «Сущность-Связь»). Её основная цель – наглядно показать, сколько экземпляров одной сущности может быть связано с экземплярами другой сущности.

Название произошло от специального символа, который напоминает след вороны (>|<) и обозначает сторону «многих» в связи.

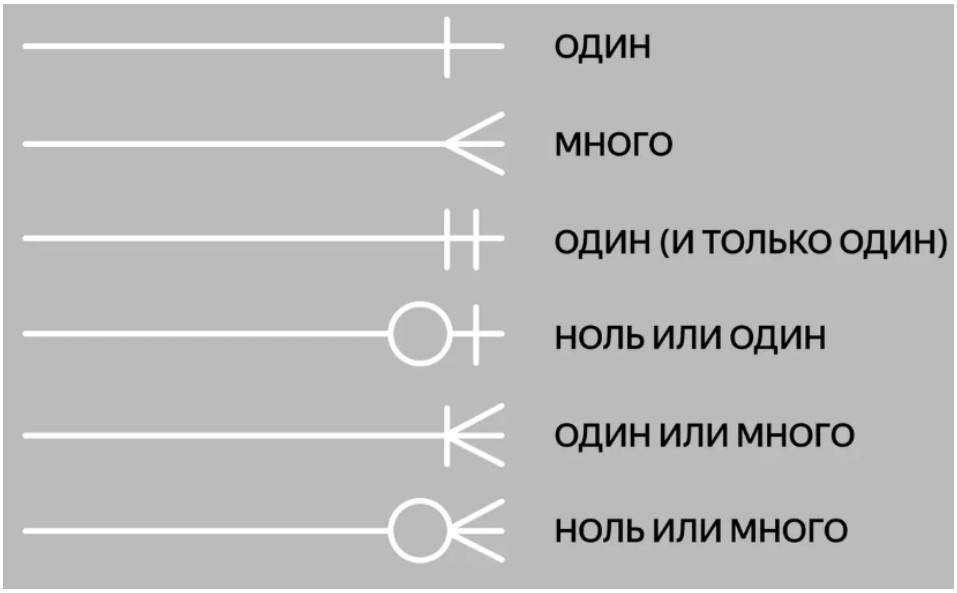


Рис. 27 Виды соединительных линий для иллюстрации типа связи между сущностями в нотации Мартина

Для того чтобы изобразить три типа связи в нотации Мартина, можно использовать разные комбинации. Например, связь «многие-ко-многим» можно изобразить так, как представлено на рис. 28.

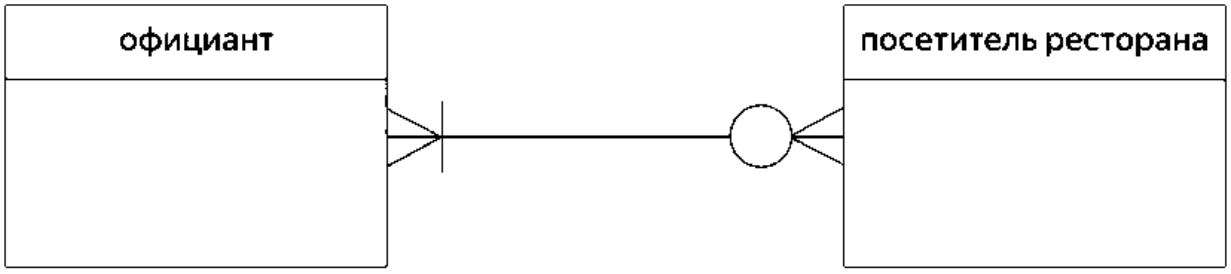


Рис. 28 Пример обозначения связи «многие-ко-многим»

Официант может обслуживать от нуля до множества посетителей ресторана. При этом одного посетителя ресторана должен обслуживать хотя бы один официант, а могут и несколько

А связь «один-ко-многим» может выглядеть так:

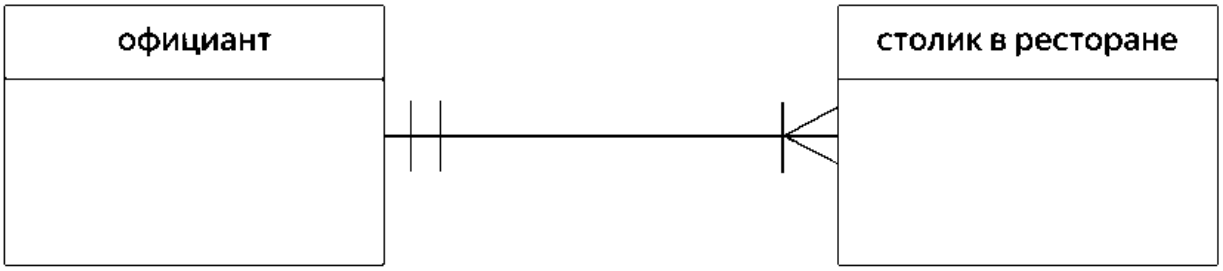


Рис. 29 Пример обозначения связи «один-ко-многим»

За каждым столиком в ресторане закреплён только один официант. При этом за каждым официантом может быть закреплено несколько столиков.

Примеры ER- диаграмм.

На примере сервиса по бронированию номеров в сети гостиниц рассмотрим, как выглядит одна и та же ER-модель в разных нотациях.

Сначала нужно выделить сущности ER-модели:

- гость;
- гостиница;
- номер.

У каждой сущности есть основные атрибуты, например, у сущности «гость» – это ФИО и номер паспорта, у «гостиницы» – её номер в сети и адрес, у «номера» – его порядковый номер в гостинице и категория.

Затем нужно установить связи между сущностями (рис. 30).

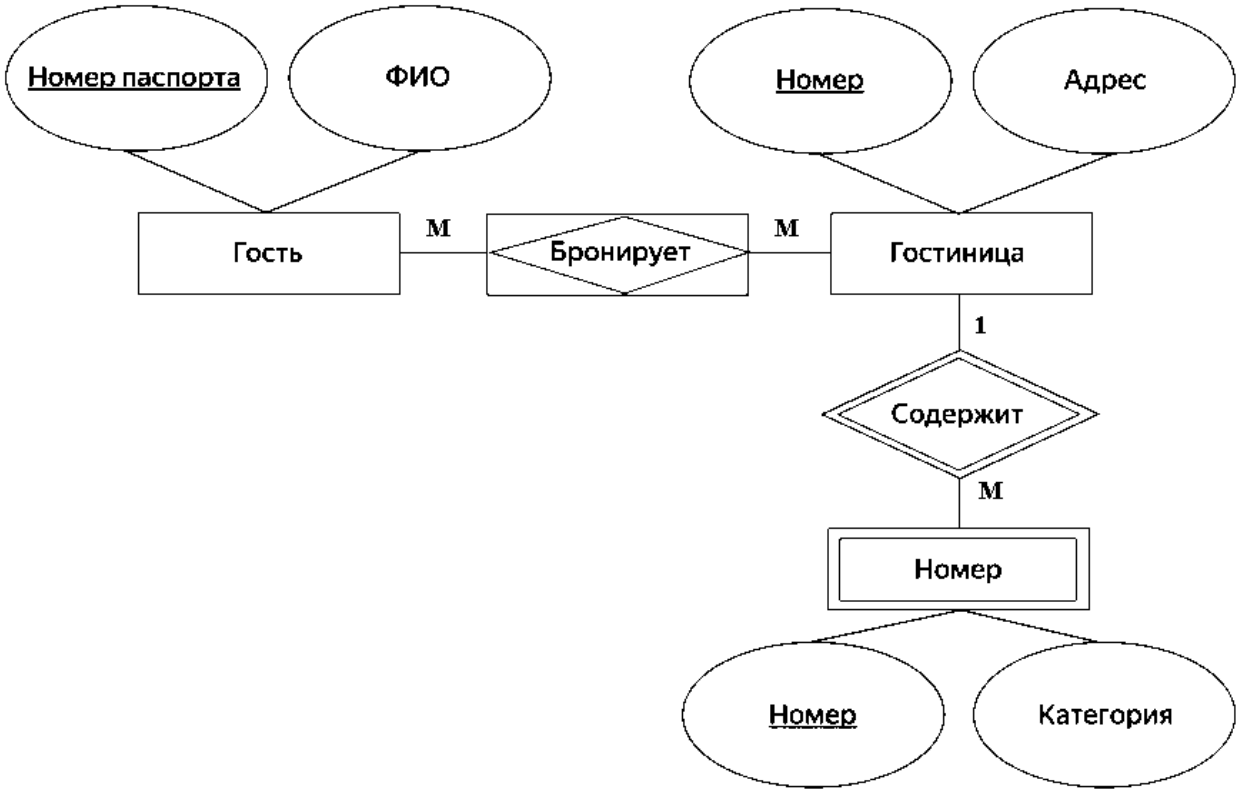


Рис. 30 ER-модель концептуального уровня в нотации Чена

ER-модель концептуального уровня в нотации Чена содержит прямоугольники с сущностями, овалы с атрибутами, ромбы со связями. Сущность в подчинении у другой сущности называют дочерней и помещают в прямоугольник с двойной рамкой. Ромб со связью между ними тоже обводят двойной рамкой

Между сущностями «Гость» и «Гостиница» установлена связь «многие-ко-многим» – много гостей могут бронировать много гостиниц. В нотации Чена такая связь становится самостоятельной сущностью, которую называют ассоциативной и обозначают ромбом внутри прямоугольника. Ассоциативная сущность между «Гостем» и «Гостиницей» – «Бронирование». На следующих уровнях ER-модели у неё появятся атрибуты, например, дата и номер бронирования.

Если строить ER-модель логического уровня в нотации Чена, она может сильно разрастись из-за большого количества атрибутов. Поэтому на следующем уровне можно построить модель в нотации Мартина (рис. 31).

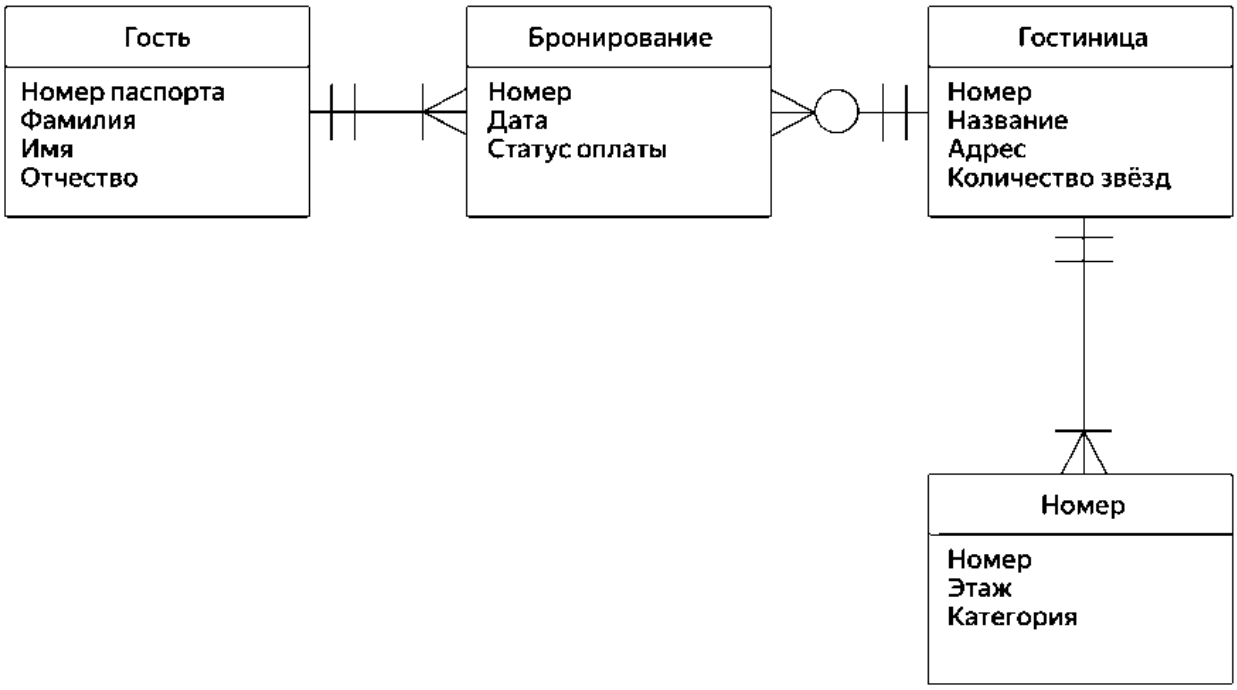


Рис. 31 ER-модель в нотации Мартина

В ER-модели в нотации Мартина атрибуты сущностей перечисляют в полях под ними. За счёт этого модель занимает меньше места и её структура менее запутана.

Пример выполнения лабораторной работы

1. Исходное описание предметной области.

Библиотека колледжа содержит книги. Каждая книга:

– Имеет название, ISBN, год издания.

– Может принадлежать одному или нескольким жанрам.

– Может иметь одного или нескольких авторов.

– Может существовать в нескольких физических экземплярах.

Читатели (студенты и преподаватели):

– Регистрируются в библиотеке, получая читательский билет.

– Могут брать несколько книг одновременно.

– Для каждой выдачи фиксируется дата выдачи и срок возврата.

Дополнительно:

– Книги распределены по отделам (абонемент, читальный зал).

– У каждого экземпляра есть инвентарный номер и состояние.

2. Концептуальная модель.

Шаг 1. Выделяем сущности (отвечаем на вопрос: «О ком/чем мы храним информацию?»):

– Книга (издание).

– Автор.

– Жанр.

– Экземпляр (физическая копия книги).

– Читатель.

– Отдел.

– Выдача.

Шаг 2. Определяем связи (отвечаем: «Как эти сущности связаны?») (рис. 32):

– M:M: Одна книга может иметь несколько авторов, один автор может написать несколько книг.

– M:M: Одна книга может принадлежать к нескольким жанрам, один жанр может включать много книг.

– 1:M: Одна книга может существовать в нескольких экземплярах.

– 1:M: Один отдел содержит много экземпляров.

– 1:M: Один читатель может иметь много выдач.

– 1:M: Один экземпляр может быть выдан много раз (в разное время).

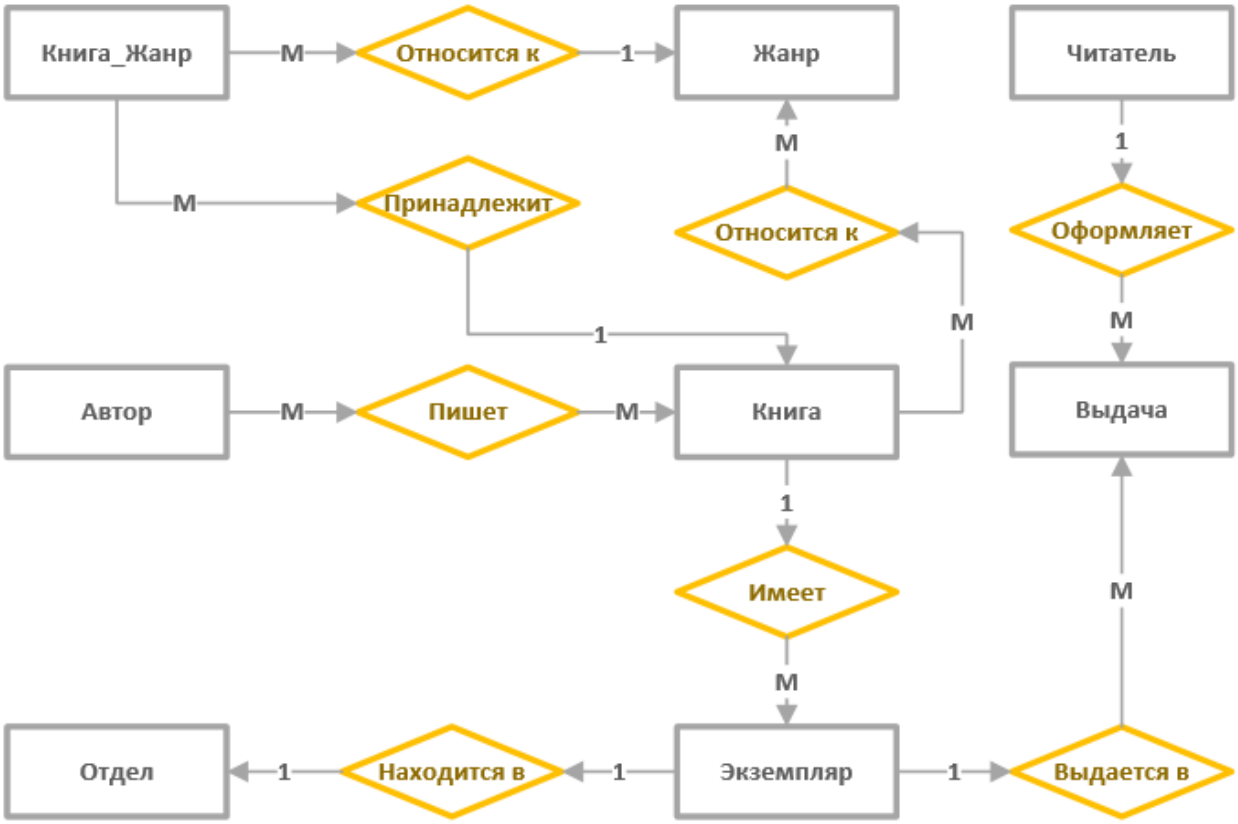


Рис. 32 Концептуальная модель

Описание концептуальной модели:

1. Книга связана с Автор связью M:M (многие-ко-многим).
2. Книга связана с Жанр связью M:M (показана через ассоциацию Книга_Жанр).
3. Книга связана с Экземпляр связью 1:M (одна книга - много экземпляров).
4. Отдел связан с Экземпляр связью 1:M (один отдел - много экземпляров).
5. Читатель связан с Выдача связью 1:M (один читатель - много выдач).
6. Экземпляр связан с Выдача связью 1:M (один экземпляр - много выдач во времени).

Вывод по концептуальной модели: модель отражает ключевые бизнес-сущности библиотеки и их взаимосвязи. Все участники проекта имеют общее понимание структуры данных.

3. Логическая модель.

Шаг 1. Устраняем связи M:M путем создания ассоциативных сущностей:

– Книга_Автор (для связи Книга-Автор).

– Книга_Жанр (для связи Книга-Жанр).

Шаг 2. Добавляем все атрибуты для каждой сущности.

Шаг 3. Определяем первичные (PK) и внешние ключи (FK).

Логическая модель в виде таблиц (табл. 10 – 18):

Таблица 10 – Таблица «Книга» (Book)

Название столбца	**Тип данных**	**Обязательный**	**Уникальный**	**Примечания**
ISBN	VARCHAR(17)	Да	Да	ПК
Название	VARCHAR(200)	Да	Нет	
Год_издания	INTEGER	Нет	Нет	
Издательство	VARCHAR(100)	Нет	Нет	
Аннотация	TEXT	Нет	Нет	
Количество_страниц	INTEGER	Нет	Нет	
Дата_добавления	TIMESTAMP	Нет	Нет	По умолчанию текущая дата

Таблица 11 – Таблица «Автор» (Author)

Название столбца	Тип данных	Обязательный	Уникальный	Примечания
ID_Автора	INTEGER	Да	Да	ПК, автоинкремент
Фамилия	VARCHAR(50)	Да	Нет	
Имя	VARCHAR(50)	Да	Нет	
Отчество	VARCHAR(50)	Нет	Нет	
Страна	VARCHAR(50)	Нет	Нет	

Таблица 12 – Таблица «Книга_Автор» (Book_Author)

Название столбца	Тип данных	Обязательный	Уникальный	Примечания
ID_Связи	INTEGER	Да	Да	ПК, автоинкремент
ISBN	VARCHAR(17)	Да	Нет	ВК → КНИГА.ISBN
ID_Автора	INTEGER	Да	Нет	ВК → АВТОР.ID_Автора

Таблица 13 – Таблица «Жанр» (Genre)

Название столбца	Тип данных	Обязательный	Уникальный	Примечания
Код_жанра	INTEGER	Да	Да	ПК, автоинкремент
Название_жанра	VARCHAR(50)	Да	Да	

Таблица 14 – Таблица «Книга_Жанр» (Book_Genre)

Название столбца	Тип данных	Обязатель-ный	Уникаль-ный	Примечания
ID_Связи	INTEGER	Да	Да	ПК, автоинкре-мент
ISBN	VARCHAR(17)	Да	Нет	ВК → КНИ-ГА.ISBN
Код_жанра	INTEGER	Да	Нет	ВК → ЖАНР.Код_жанра

Таблица 15 – Таблица «Отдел» (Department)

Название столбца	Тип данных	Обязатель-ный	Уникаль-ный	Примеча-ния
Код_отдела	INTEGER	Да	Да	ПК, автоин-кремент
Назва-ние_отдела	VARCHAR(50)	Да	Да	
Местоположе-ние	VARCHAR(100)	Нет	Нет	

Таблица 16 – Таблица «Экземпляр» (Copy)

Название столбца	Тип данных	Обязатель-ный	Уникаль-ный	Примечания
Инвентар-ный_номер	INTEGER	Да	Да	ПК
ISBN	VARCHAR(17)	Да	Нет	ВК → КНИ-ГА.ISBN
Код_отдела	INTEGER	Да	Нет	ВК → ОТ-ДЕЛ.Код_отдела
Да-та_поступления	DATE	Да	Нет	

Название столбца	Тип данных	Обязатель-ный	Уникаль-ный	Примечания
Состояние	VARCHAR(20)	Да	Нет	'новый', 'хороший', 'ветхий'
Стоимость	DECIMAL(10,2)	Нет	Нет	
Примечания	TEXT	Нет	Нет	

Таблица 17 – Таблица «Читатель» (Reader)

Название столб-ца	Тип данных	Обязатель-ный	Уникаль-ный	Примеча-ния
Номер_билета	INTEGER	Да	Да	ПК
Фамилия	VARCHAR(50)	Да	Нет	
Имя	VARCHAR(50)	Да	Нет	
Отчество	VARCHAR(50)	Нет	Нет	
Тип_читателя	VARCHAR(20)	Да	Нет	'студент', 'преподава-тель', 'со-трудник'
Груп-па_или_кафедра	VARCHAR(50)	Нет	Нет	
Телефон	VARCHAR(15)	Нет	Нет	
Email	VARCHAR(100)	Нет	Нет	
Дата_регистрации	DATE	Да	Нет	

Таблица 18– Таблица «Выдача» (Issue)

Название столбца	Тип данных	Обязательный	Уникальный	Примечания
ID_Выдачи	INTEGER	Да	Да	ПК, автоинкремент
Инвентарный_номер	INTEGER	Да	Нет	ВК → ЭКЗЕМПЛЯР.Инвентарный_номер
Номер_билета	INTEGER	Да	Нет	ВК → ЧИТАТЕЛЬ.Номер_билета
Дата_выдачи	TIMESTAMP	Да	Нет	
Плановый_срок_возврата	DATE	Да	Нет	
Фактическая_дата_возврата	DATE	Нет	Нет	
Статус	VARCHAR(20)	Да	Нет	'выдана', 'возвращена', 'просрочена'
Штраф	DECIMAL(10,2)	Нет	Нет	DEFAULT 0

Пояснения сокращений представлены в табл. 19.

Таблица 19 – Пояснение сокращений

Сокращение	Расшифровка	Описание
ПК	Первичный ключ	Уникальный идентификатор записи в таблице
ВК	Внешний ключ	Ссылка на первичный ключ другой таблицы
Да	Требуется	Поле обязательно для заполнения (NOT NULL)
Нет	Не требуется	Поле может быть пустым
VARCHAR(N)	Строка	Текст длиной до N символов

Сокращение	Расшифровка	Описание
INTEGER	Целое число	Целые числа без дробной части
DECIMAL(10,2)	Десятичное число	Число с 10 цифрами, 2 после запятой
DATE	Дата	Дата в формате ГГГГ-ММ-ДД
TIMESTAMP	Дата и время	Дата и время с точностью до секунды
TEXT	Текст	Длинный текст без ограничения длины

На основании определенных выше таблиц строится логическая модель данных в виде диаграммы с указанием связей между сущностями (рис. 33).

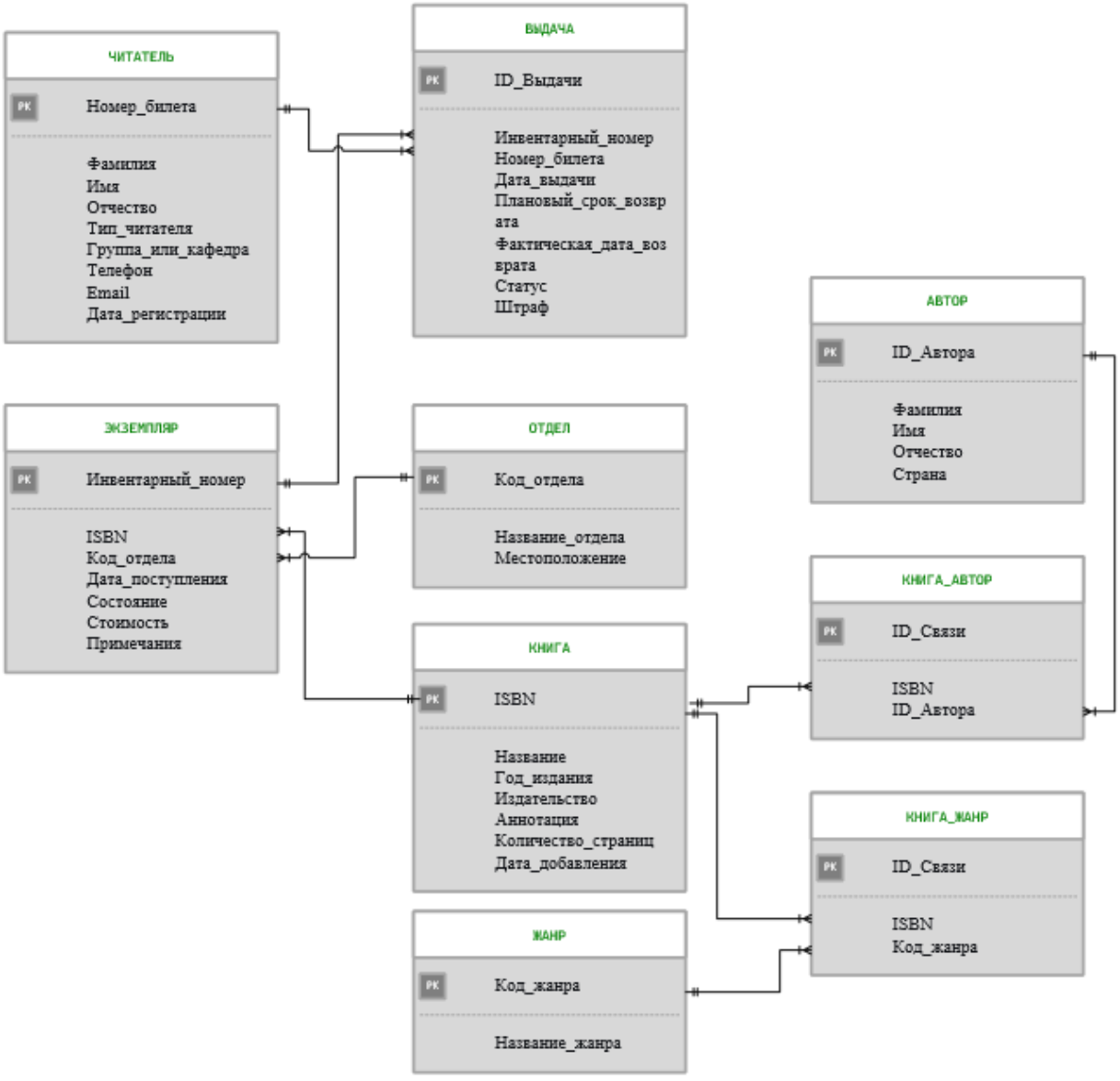


Рис. 33 Логическая модель данных

Связи между таблицами:

1. КНИГА (1) ←→ (M) КНИГА_АВТОР (M) ←→ (1) АВТОР
   - Одна книга может иметь несколько авторов.
   - Один автор может написать несколько книг.
2. КНИГА (1) ←→ (M) КНИГА_ЖАНР (M) ←→ (1) ЖАНР
   - Одна книга может относиться к нескольким жанрам.
   - Один жанр может включать множество книг.
3. КНИГА (1) ←→ (M) ЭКЗЕМПЛЯР
   - Одна книга может существовать в нескольких физических экземплярах.
   - Каждый экземпляр относится к одной конкретной книге.
4. ОТДЕЛ (1) ←→ (M) ЭКЗЕМПЛЯР
   - Один отдел может содержать множество экземпляров.
   - Каждый экземпляр хранится в одном отделе.
5. ЭКЗЕМПЛЯР (1) ←→ (M) ВЫДАЧА
   - Один экземпляр может быть выдан много раз (в разное время).
   - Каждая выдача относится к одному конкретному экземпляру.
6. ЧИТАТЕЛЬ (1) ←→ (M) ВЫДАЧА
   - Один читатель может оформить множество выдач.
   - Каждая выдача оформляется одним читателем.

Особенности логической модели:

1. Устранение связей M:M:
   - Связь «Книга-Автор» (M:M) заменена на две связи 1:M через ассоциативную таблицу КНИГА_АВТОР.
   - Связь «Книга-Жанр» (M:M) заменена на две связи 1:M через ассоциативную таблицу КНИГА_ЖАНР.

2. Суррогатные ключи:

- Для большинства таблиц введены искусственные числовые ключи (ID_Автора, Код_жанра и т.д.)
- Исключение: ISBN, Номер_билета, Инвентарный_номер – естественные ключи

3. Нормализация:
   - Каждая таблица содержит данные об одной сущности.
   - Отсутствуют повторяющиеся группы данных.
   - Все неключевые атрибуты зависят от первичного ключа.
4. Ограничения целостности:
   - Все внешние ключи имеют ссылки на существующие записи в родительских таблицах.
   - Указаны обязательные для заполнения поля.

Вывод по логической модели: создана детализированная модель со всеми атрибутами. Устранены связи M:M через ассоциативные сущности. Определены первичные и внешние ключи. Модель независима от конкретной СУБД и готова для преобразования в физическую модель.

Последовательность выполнения лабораторной работы

1. Выберите предметную область согласно своему индивидуальному варианту (лаб. раб. № 5)
2. Создайте две модели последовательно:
   - Концептуальная: все сущности, связи с типами.
   - Логическая: Все сущности, все атрибуты, устранение M:M.

Лабораторная работа №8 «Особенности реляционной модели и проектирование баз данных»

Теоретические сведения

Реляционная модель данных – это логическая модель данных, предложенная Эдгаром Коддом в 1970 году. В основе модели лежит математическое понятие отношения (relation).

В табл. 20 представлены основные элементы реляционной модели.

Таблица 20 – Основные элементы реляционной модели

Элемент	Определение	Пример
Отношение (Relation)	Таблица с данными	Таблица «Студенты»
Кортеж (Tuple)	Строка в таблице	Запись о конкретном студенте
Атрибут (Attribute)	Столбец в таблице	«Фамилия», «Имя», «Группа»
Домен (Domain)	Множество допустимых значений атрибута	Для «Оценка»: целые числа 2-5
Первичный ключ (Primary Key)	Атрибут(ы), уникально идентифицирующий кортеж	«Номер зачетки»
Внешний ключ (Foreign Key)	Атрибут, который ссылается на первичный ключ другой таблицы	«ID_Группы» в таблице «Студенты»

Свойства отношений:

1. Отсутствие дубликатов кортежей – нет одинаковых строк.
2. Неупорядоченность кортежей – порядок строк не имеет значения.
3. Неупорядоченность атрибутов – порядок столбцов не имеет значения.
4. Атомарность значений – каждое значение в ячейке неделимо.

Аномалии в ненормализованных таблицах.

Аномалии – проблемы, возникающие при работе с таблицами, содержащими избыточные данные.

Аномалии вставки (Insert Anomaly) – невозможность добавить данные без наличия других данных.

Пример: в таблице «Студенты_с_оценками» нельзя добавить нового студента, если он еще не получил ни одной оценки.

Аномалии удаления (Delete Anomaly) – потеря информации при удалении данных.

Пример: при удалении последней оценки по предмету теряется информация о самом предмете.

Аномалии обновления (Update Anomaly) – необходимость изменения одних и тех же данных в нескольких местах.

Пример: если студент перевелся в другую группу, нужно изменить группу во всех его записях с оценками.

Нормализация отношений.

Нормализация – процесс преобразования отношений к виду, исключающему аномалии.

1. Первая нормальная форма (1НФ).

Условия:

- Все значения атрибутов атомарны (неделимы).
- В таблице нет повторяющихся групп.

Пример нарушения 1НФ представлен в табл. 21.

Таблица 21 – Сущность «Заказ»

ID_Заказа	**Товары**
1	«Ноутбук, Мышь, Наушники»

Проблема: атрибут Товары содержит неатомарное значение (список).

Исправление (приведение к 1НФ) отображено в табл. 22.

Таблица 22 – Приведенная к 1НФ сущность «Заказ»

ID_Заказа	**Товар**
1	Ноутбук
1	Мышь
1	Наушники

2. Вторая нормальная форма (2НФ).

Условия:

- Таблица находится в 1НФ.
- Все неключевые атрибуты полностью зависят от всего составного первичного ключа.

Пример нарушения 2НФ (табл. 23):

Первичный ключ (ПК): {ID_Заказа, ID_Товара}

Таблица 23 – Сущность «Заказ_Подробно»

ID_Заказа	**ID_Товара**	**Название_Товара**	**Цена**	**Количество**
1	101	Ноутбук	50000	1

Проблема: Название_Товара и Цена зависят только от части ключа (ID_Товара), а не от всего ключа {ID_Заказа, ID_Товара}.

Исправление (приведение к 2НФ) отображено в табл. 24-25. При этом мы создаем две таблицы: Заказ_Позиция (содержит информацию о количестве товаров в заказе) и Товар (содержит информацию о товаре).

Таблица 24 – Сущность «Заказ_Позиция»

ID_Заказа	**ID_Товара**	**Количество**
1	101	1
1	102	2
2	101	1

Таблица 25 – Сущность «Товар»

ID_Товара	**Название_Товара**	**Цена**
101	Ноутбук	50000
102	Мышь	1500

Почему это лучше:

- Наглядно показывает проблему: видно, что ID_Товара=101 повторяется в разных заказах.
- Показывает необходимость нормализации: видна избыточность данных в исходной таблице.
- Демонстрирует правильную структуру: в Заказ_Позиция может быть несколько записей с одним ID_Товара в разных заказах.
- Соответствует реальной ситуации: в реальных БД один товар действительно может быть во многих заказах.

Как они связаны:

Связь один-ко-многим:

- Один товар → Много позиций в заказах.
- Товар.ID_Товара (первичный ключ) → Заказ_Позиция.ID_Товара (внешний ключ).

3. Третья нормальная форма (3НФ).

Условия:

- Таблица находится в 2НФ.
- Отсутствуют транзитивные зависимости (неключевые атрибуты не зависят от других неключевых атрибутов).

Пример нарушения 3НФ представлен в табл. 26.

Таблица 26 – Сущность «Студент»

ID_Студента	**ФИО**	**ID_Группы**	**Название_Группы**	**Куратор**
1	Иванов И.И.	ГРП-21	Программисты	Петрова П.П.

Проблема: неключевые атрибуты Название_Группы и Куратор зависят не от первичного ключа (ID_Студента), а от другого неключевого атрибута (ID_Группы). Это транзитивная зависимость.

Исправление (приведение к 3НФ) отображено в табл. 27-28. При этом мы создаем две таблицы: Студент и Группа.

Таблица 27 – Сущность «Студент»

ID_Студента	**ФИО**	**ID_Группы**
1	Иванов И.И.	ГРП-21

Таблица 28 – Сущность «Группа»

ID_Группы	**Название_Группы**	**Куратор**
ГРП-21	Программисты	Петрова П.П.

4. Нормальная форма Бойса-Кодда (НФБК/BCNF).

Усиленная 3НФ, где каждый детерминант (атрибут, от которого функционально зависит другой атрибут) является потенциальным ключом.

Функциональные зависимости.

Функциональная зависимость X → Y означает, что значения атрибута Y однозначно определяются значениями атрибута X.

Виды функциональных зависимостей:

- Полная: {A, B} → C (C зависит от A и B вместе).
- Частичная: A → C при {A, B} → C (C зависит только от A).
- Транзитивная: A → B и B → C, следовательно, A → C.

Правила вывода (аксиомы Армстронга):

- Рефлексивность: Если Y ⊆ X, то X → Y
- Пополнение: Если X → Y, то XZ → YZ
- Транзитивность: Если X → Y и Y → Z, то X → Z

В табл. 29 отражены способы обеспечения целостности данных в реляционной модели.

Таблица 29 – Целостность данных в реляционной модели

Тип целостности	Обеспечивается	Пример
Объектная (сущностная)	Первичными ключами	UNIQUE, PRIMARY KEY
Ссылочная	Внешними ключами	FOREIGN KEY
Пользовательская	Ограничениями CHECK	CHECK (Оценка BETWEEN 2 AND 5)
Каскадные операции	Правилами обновления/удаления	ON DELETE CASCADE

Пример выполнения лабораторной работы

Шаг 1. Анализ исходной ненормализованной таблицы (табл. 30).

Таблица 30 – Исходная таблица «Учебный_процесс»

Номер_зачетки	ФИО	Группа	Предмет	Преподаватель	Дата_сдачи	Оценка
2023001	Иванов И.И.	ИСП-21	БД	Петрова П.П.	2024-01-15	5
2023001	Иванов И.И.	ИСП-21	ООП	Сидоров С.С.	2024-01-20	4
2023002	Петров П.П.	ИСП-21	БД	Петрова П.П.	2024-01-15	4
2023002	Петров П.П.	ИСП-21	ООП	Сидоров С.С.	2024-01-20	3
2023001	Иванов И.И.	ИСП-21	Физ-ра	Кузнецов К.К.	2024-01-10	зачет

Шаг 2. Выявление аномалий (табл. 31).

Таблица 31 – Выявление аномалий

Тип аномалии	Пример из таблицы
Аномалии вставки	Нельзя добавить нового преподавателя, если он еще не ведет занятия
Аномалии удаления	При удалении последней оценки по предмету «Физ-ра» теряется информация о предмете и преподавателе
Аномалии обновления	Если студент переводится в другую группу, нужно изменить группу во всех его записях

Шаг 3. Выявление функциональных зависимостей (табл. 32).

Таблица 32 – Выявление функциональных зависимостей

Функциональная зависимость	Объяснение
1. Номер_зачетки → ФИО, Группа	Информация о студенте зависит от номера зачетки
2. Предмет → Преподаватель	Преподаватель зависит от предмета
3. {Номер_зачетки, Предмет} → Дата_сдачи, Оценка	Дата сдачи и оценка зависят и от студента, и от предмета
4. Группа → ?	Нет зависимости, так как в группе может быть много студентов

Шаг 4. Нормализация до 1НФ.

Проверка: таблица уже в 1НФ (все значения атомарны, нет повторяющихся групп).

Шаг 5. Нормализация до 2НФ.

Определяем потенциальный ключ: {Номер_зачетки, Предмет}

Выявляем частичные зависимости:

– ФИО зависит только от Номер_зачетки.

– Группа зависит только от Номер_зачетки.

– Преподаватель зависит только от Предмет.

Разделяем таблицы (табл. 33).

Таблица 33 – Разделение таблиц

Таблица: СТУДЕНТ	Таблица: ПРЕДМЕТ	Таблица: ОЦЕНКА
Номер_зачетки	**Предмет**	**Номер_зачетки**
ФИО	Преподаватель	**Предмет**
Группа		Дата_сдачи
		Оценка

Шаг 6. Нормализация до 3НФ (табл. 34).

Таблица 34 – Проверка транзитивных зависимостей

Таблица	**Проверка**	**Результат**
СТУДЕНТ	Группа не зависит от Номер_зачетки через другие атрибуты	✓ (нет транзитивных зависимостей)
ПРЕДМЕТ	Нет транзитивных зависимостей	✓
ОЦЕНКА	Нет транзитивных зависимостей	✓

Шаг 7. Определение ключей и связей (табл. 35).

Таблица 35 – Определение ключей и связей

Таблица	**Первичный ключ (ПК)**	**Внешние ключи (ВК)**
СТУДЕНТ	Номер_зачетки	нет
ПРЕДМЕТ	Предмет	нет
ОЦЕНКА	{Номер_зачетки, Предмет}	1. Номер_зачетки → СТУДЕНТ(Номер_зачетки) 2. Предмет → ПРЕДМЕТ(Предмет)

Шаг 8. Создание реляционных таблиц в СУБД через интерфейс.

1. Создание базы данных.

В обозревателе объектов (Object Explorer) справа:

– Раскройте узел сервера.

– Правой кнопкой на «Базы данных» → «New Database...» (рис. 34).

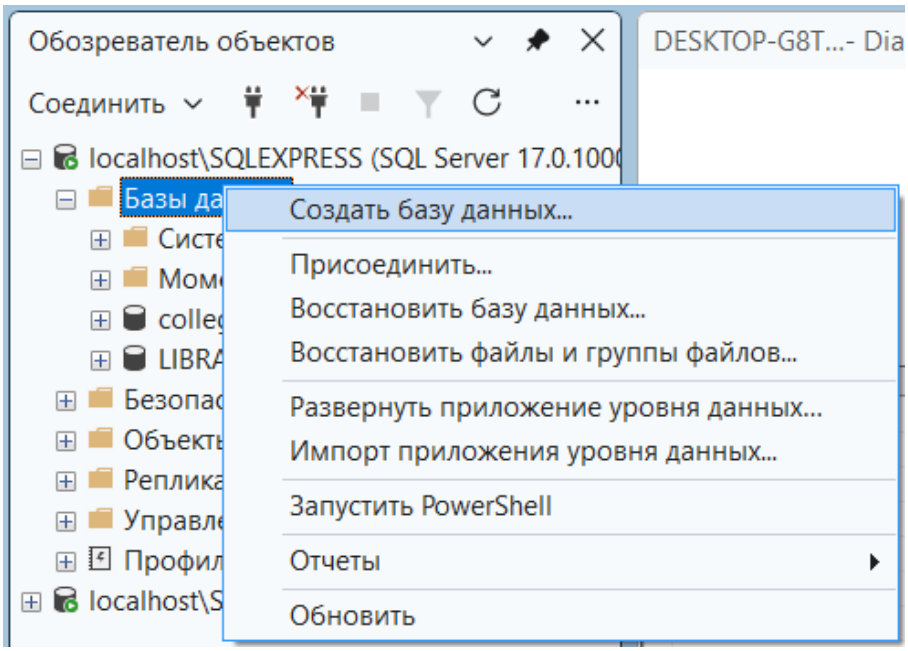


Рис. 34 Создание новой БД

В окне «Создание базы данных»:

– Имя базы данных: введите имя (например, УчебныйПроцесс) (рис. 35).

– Нажмите «OK».

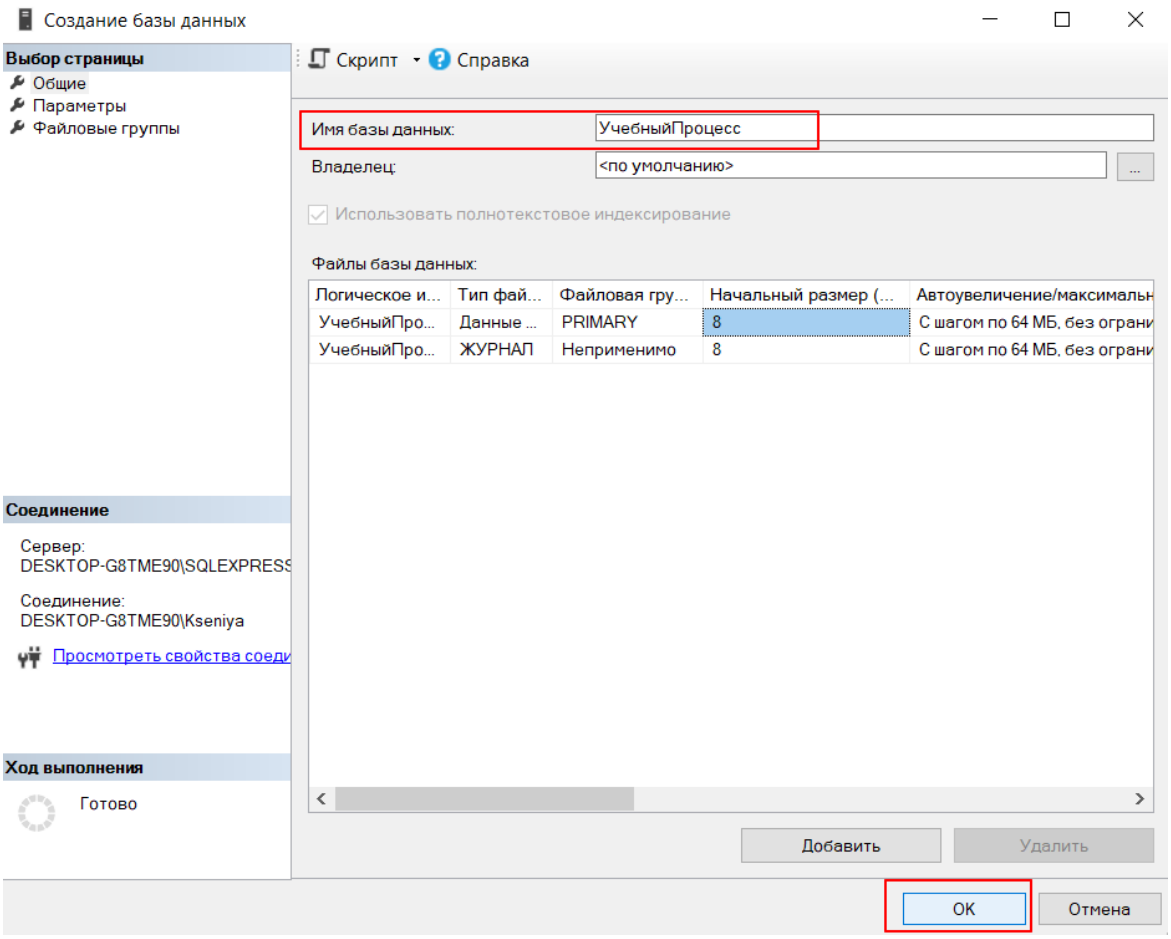


Рис. 35 Задание имени БД

2. Создание таблицы.

1. Открыть конструктор таблиц.

– В обозревателе объектов раскройте созданную базу данных.

– Правой кнопкой на папке «Таблицы» → «Создать» → «Таблица...» (рис. 36).

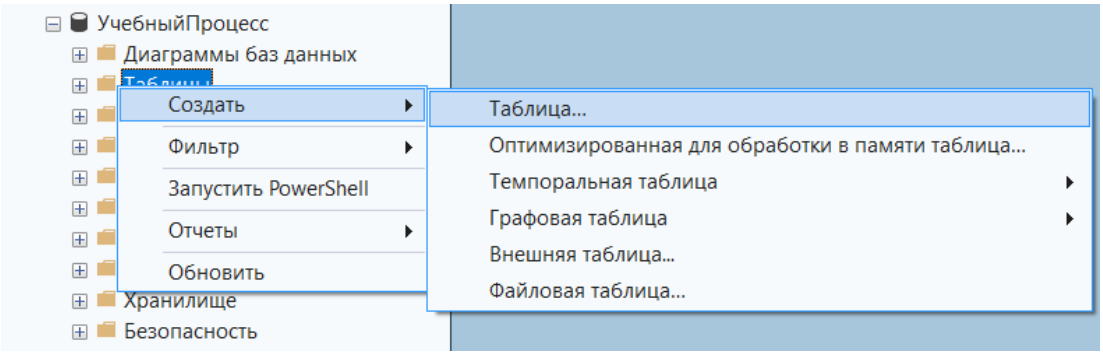


Рис. 36 Создание новой таблицы

2. Задать столбцы таблицы.

Например, создадим таблицу «Студент» (рис. 37):

– В первой строке в «Имя столбца» введите «Номер_зачетки».
– В «Тип данных» выберите int.
– В «Разрешить значения NULL» снимите галочку (это поле не может быть пустым).
– Повторите для остальных столбцов

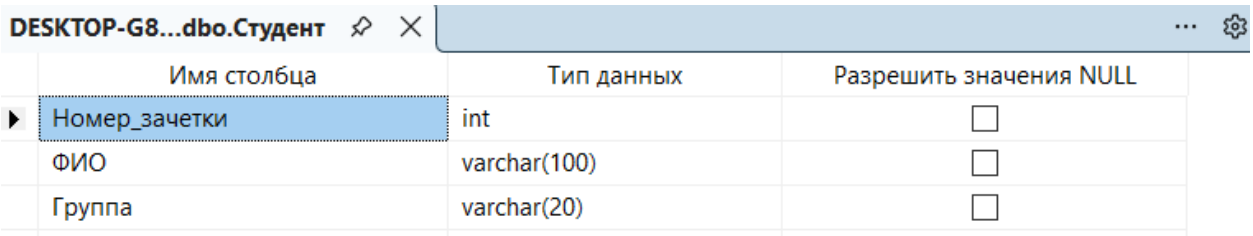


Рис. 37 Таблица «Студент»

3. Установить первичный ключ.

– Правой кнопкой на столбце Номер_зачетки
– Выберите «Задать первичный ключ» (рис. 38).

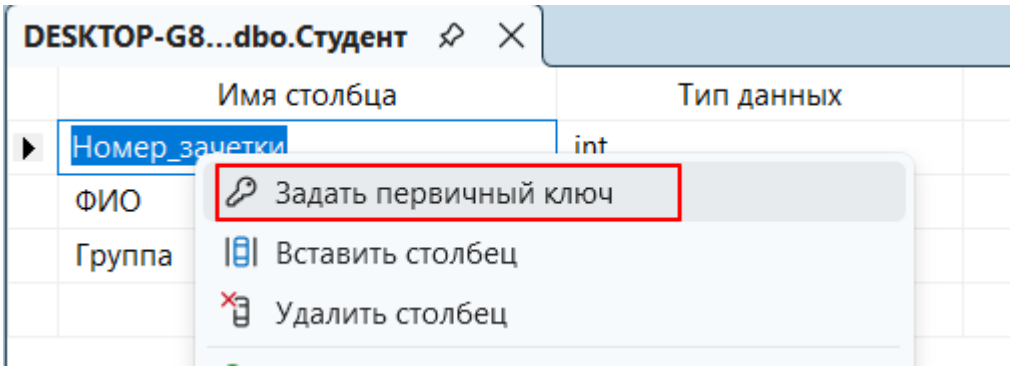


Рис. 38 Задание первичного ключа

– Появится значок ключа слева от столбца (рис. 39).

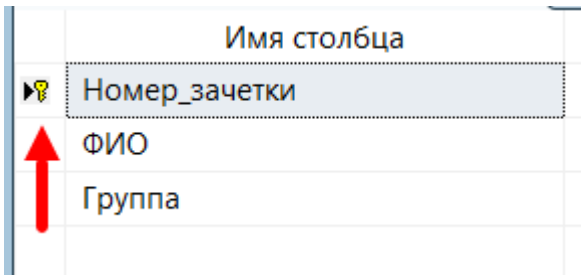


Рис. 39 Отображение первичного ключа

Примечание: если необходимо создать составной первичный ключ, то выделите нужные столбцы (Ctrl+клик на второй) и затем через правую кнопку мыши на выделенном выберите «Задать первичный ключ».

4. Сохранить таблицу.
   - Нажмите Ctrl+S или значок сохранения.
   - В диалоговом окне введите имя таблицы: СТУДЕНТ.
   - Нажмите «ОК».

5. По аналогии создаем таблицы «Предмет» и «Оценка».
6. Добавить внешние ключи.

Внешний ключ на СТУДЕНТ:

- В верхней панели конструктора нажмите кнопку «Отношения» (или Alt+F2) (рис. 40).

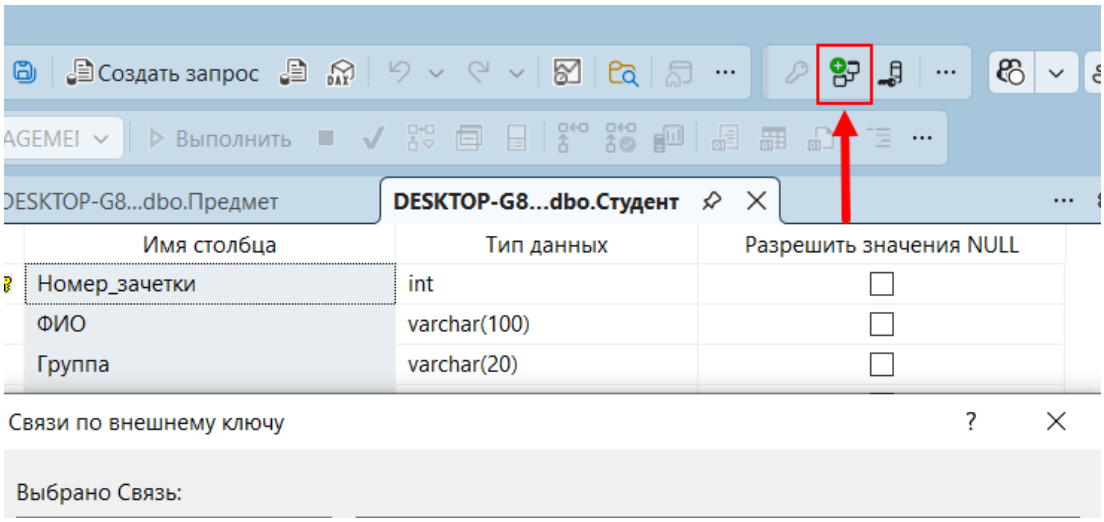


Рис. 40 Раздел «Отношения»

– В открывшемся окне «Связи по внешнему ключу» нажмите «Добавить» (рис. 41).

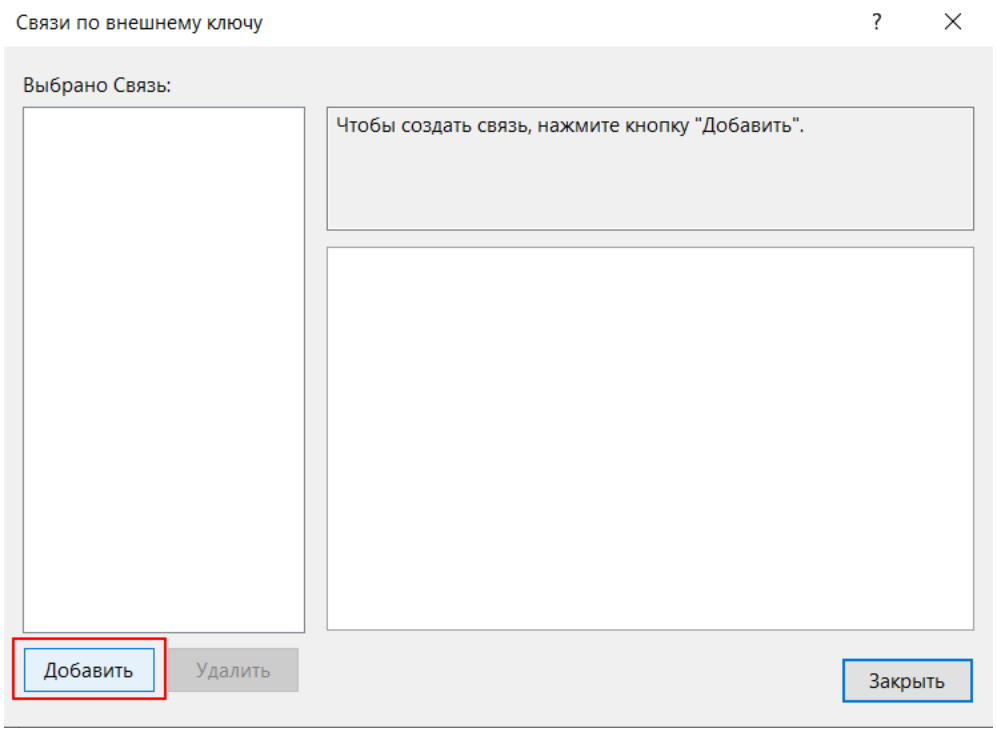


Рис. 41 Окно «Связи по внешнему ключу»

– В списке слева выберите новое отношение (например, FK_ОЦЕНКА_СТУДЕНТ) (рис. 42).

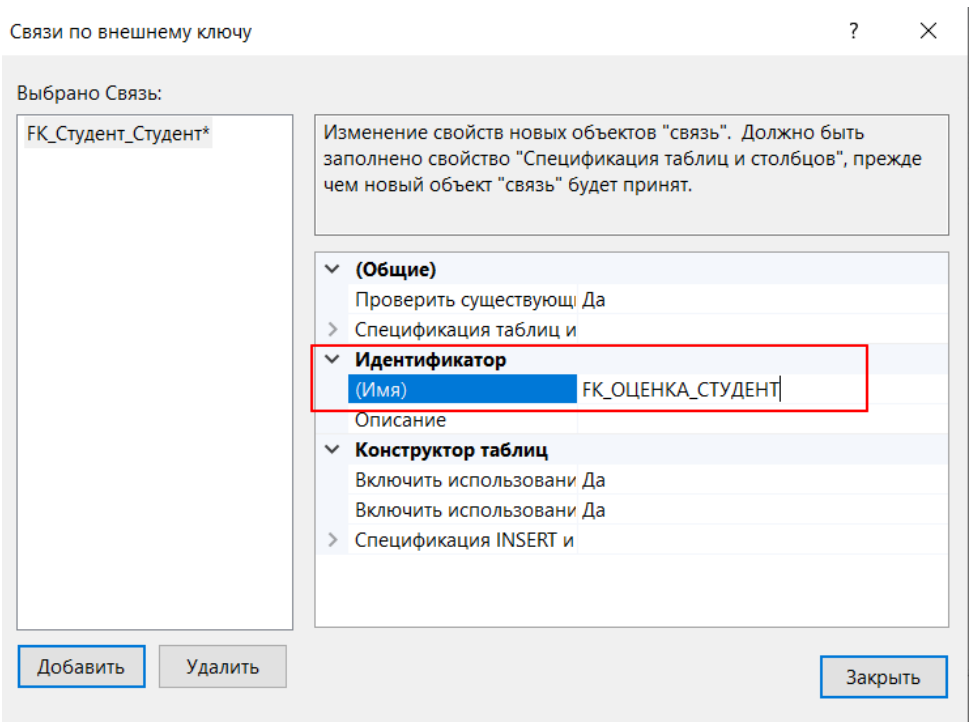


Рис. 42 Создание нового отношения

– В правой части окна нажмите «...» у свойства «Спецификация таблиц и столбцов» (рис. 43).

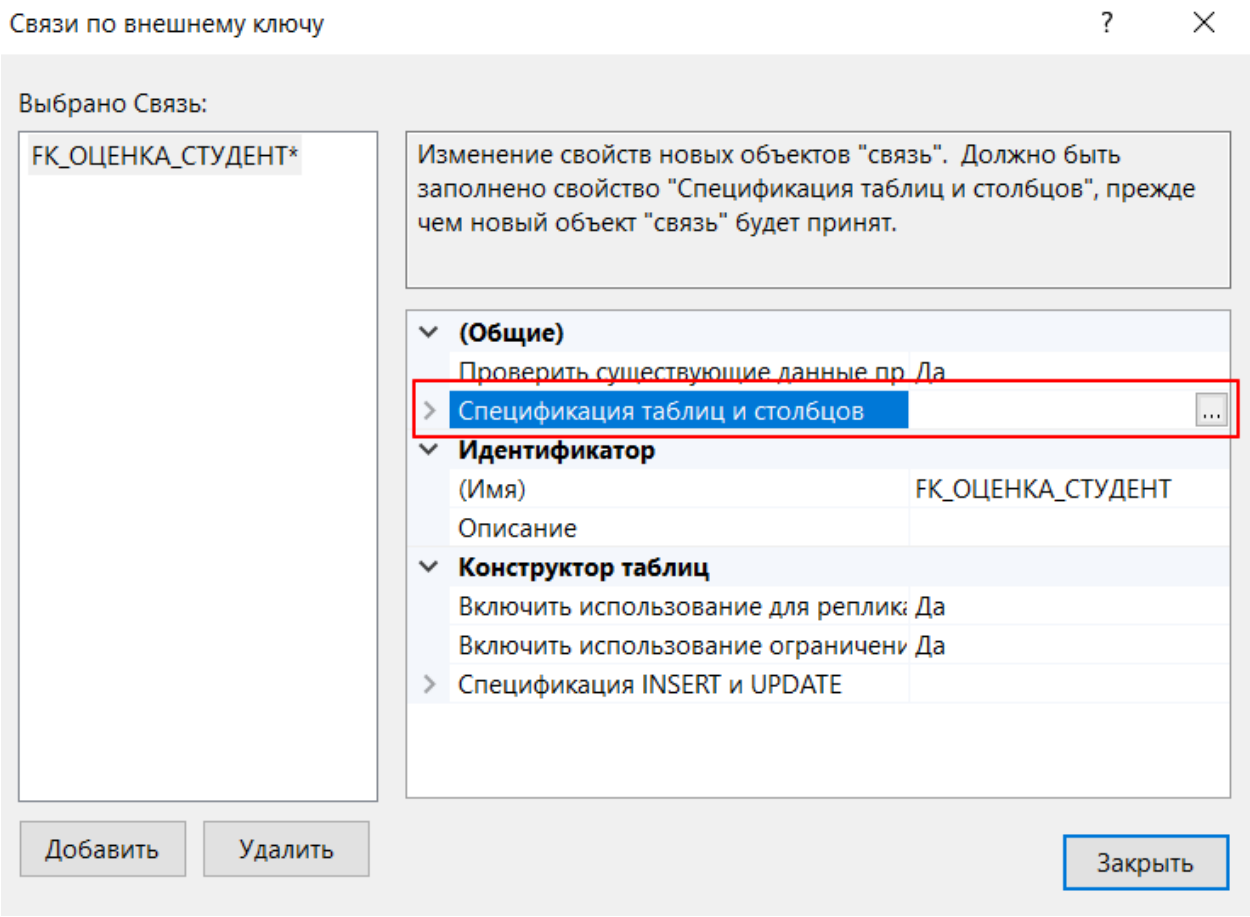


Рис. 43 Свойство «Спецификация таблиц и столбцов»

В окне «Таблицы и столбцы» (рис. 44):

– Таблица первичного ключа: выберите СТУДЕНТ.
– Столбец первичного ключа: выберите Номер_зачетки.
– Таблица внешнего ключа: автоматически ОЦЕНКА.
– Столбец внешнего ключа: выберите Номер_зачетки.
– Нажмите «ОК».

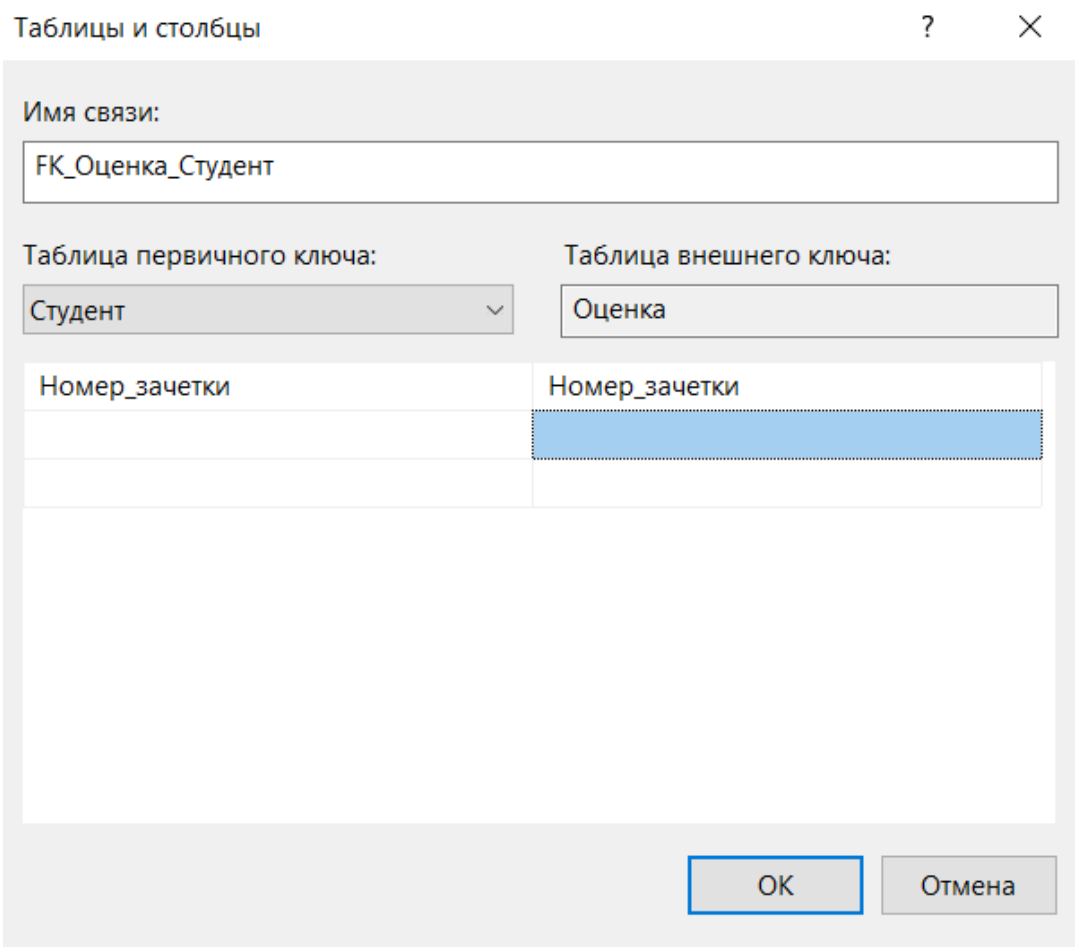


Рис. 44 Создание внешнего ключа

Внешний ключ на ПРЕДМЕТ:

– Снова нажмите «Добавить» в окне «Связи по внешнему ключу».
– Выберите новое отношение (например, FK_ОЦЕНКА_ПРЕДМЕТ).
– Нажмите «...» у «Спецификация таблиц и столбцов».

В окне «Таблицы и столбцы»:

– Таблица первичного ключа: выберите ПРЕДМЕТ.
– Столбец первичного ключа: выберите Предмет.
– Таблица внешнего ключа: автоматически ОЦЕНКА.
– Столбец внешнего ключа: выберите Предмет.
– Нажмите «OK».
– Закройте окно «Связи по внешнему ключу».

7. Сохранить таблицу.

Ctrl+S → имя таблицы: ОЦЕНКА → «OK».

3. Проверка созданных таблиц.

Способ 1: Через обозреватель объектов (рис. 45).

1. Раскройте базу данных → «Таблицы».
2. Увидите три созданные таблицы:
   - dbo.СТУДЕНТ
   - dbo.ПРЕДМЕТ
   - dbo.ОЦЕНКА

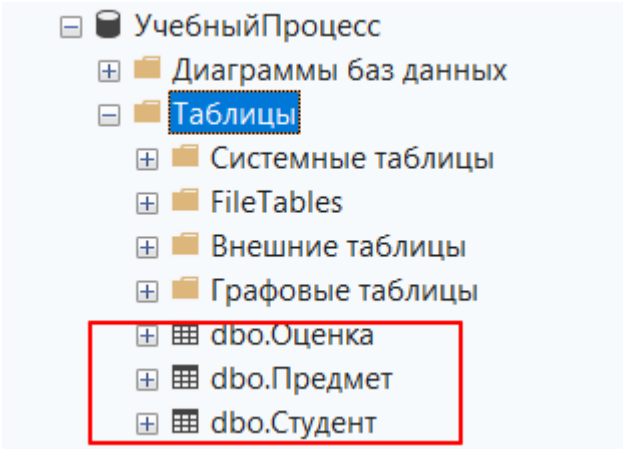


Рис. 45 Просмотр созданных таблиц через обозреватель объектов

Способ 2: Просмотр связей (диаграмма).

- Правой кнопкой на папке «Диаграммы баз данных» (в вашей БД)
- «Создать диаграмму баз данных» (рис. 46).

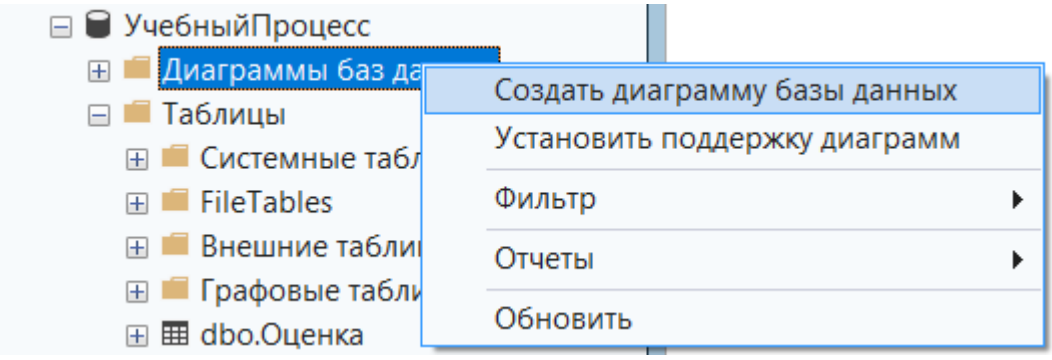


Рис. 46 Создание диаграммы базы данных

- Добавьте все три таблицы (рис. 47).

Примечание: при создании диаграммы может появиться ошибка «индекс находился вне границ массива». Для ее исправления необходимо заменить все русскоязычные названия на английские.

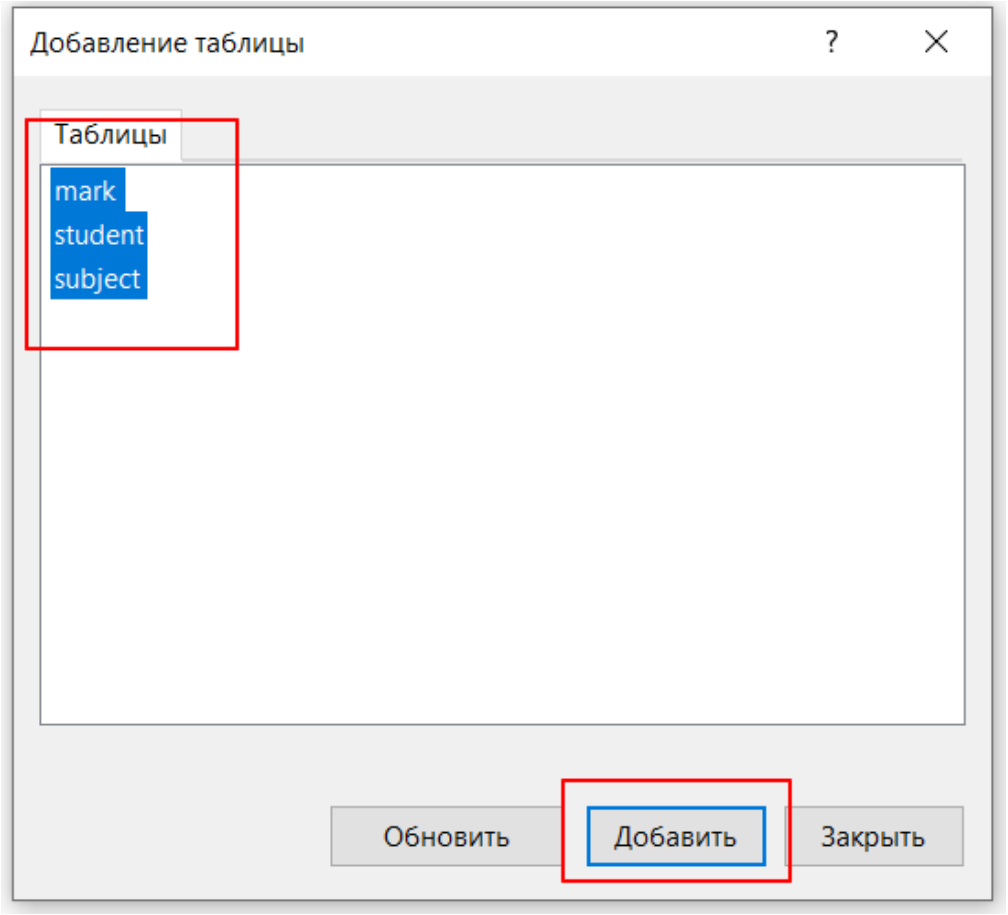


Рис. 47 Добавление таблиц на диаграмму

– Увидите связи между таблицами со значками (рис. 48).

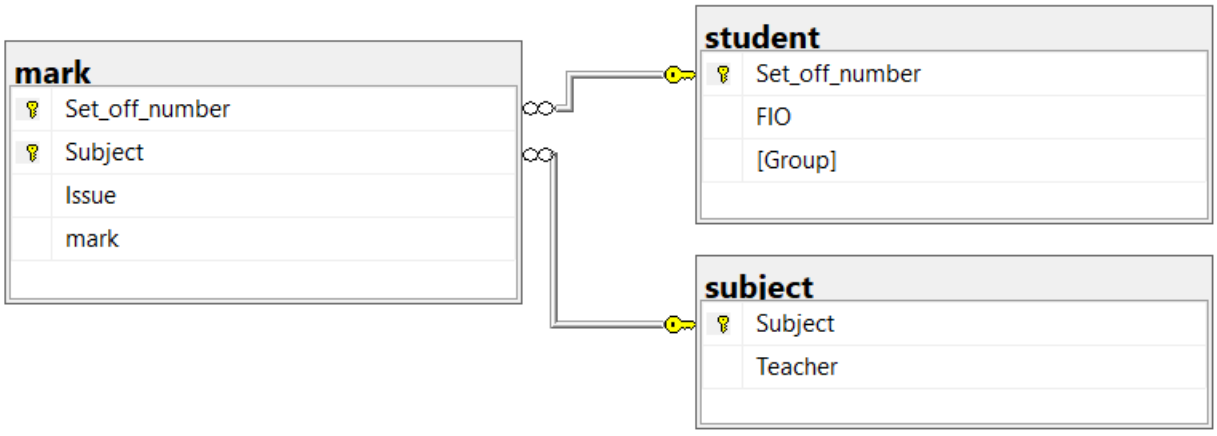


Рис. 48 Диаграмма созданной БД

4. Добавление тестовых данных через интерфейс.

- Правой кнопкой на таблице.
- «Изменить первые 200 строк».
- Введите данные.
- Данные сохраняются автоматически при переходе на другую строку.

Вывод: в результате нормализации устранены все аномалии, данные организованы в три таблицы, соответствующие 3НФ. Каждая таблица хранит информацию об одной сущности, исключена избыточность данных.

Последовательность выполнения лабораторной работы

1. Изучите теоретический материал о реляционной модели.
2. Согласно индивидуальному варианту задания выполните анализ исходной таблицы:
   - Заполните таблицу тестовыми данными (5-7 записей).
   - Определите потенциальный первичный ключ.
3. Опишите примеры возможных аномалий вставки, удаления и обновления.
4. Запишите все функциональные зависимости в формате $X \rightarrow Y$.
5. Приведите таблицу к 1НФ (если необходимо).
6. Приведите к 2НФ (разделите таблицы, устранив частичные зависимости).
7. Приведите к 3НФ (устраните транзитивные зависимости).
8. Определите первичные и внешние ключи для всех таблиц.
9. Нарисуйте схему связей между таблицами.

Варианты индивидуальных заданий

Вариант 1: Учет продаж в магазине.

Таблица 36 – ПРОДАЖИ

ID_чека	Дата	Продавец	Товар	Количество	Цена
1001	2024-01-15	Иванов	Хлеб	2	50
1001	2024-01-15	Иванов	Молоко	1	80
1002	2024-01-15	Петрова	Хлеб	1	50
1003	2024-01-16	Иванов	Яйца (10шт)	2	120

Вариант 2: Запись на прием к врачу

Таблица 37 – ПРИЕМЫ

№	Пациент	Телефон	Врач	Специальность	Дата
1	Иванов И.И.	89161234567	Петрова	Терапевт	2024-01-20
2	Сидоров С.С	89098765432	Петрова	Терапевт	2024-01-20
3	Иванов И.И.	89161234567	Кузнецов	Хирург	2024-01-22
4	Петров П.П.	89163456789	Петрова	Терапевт	2024-01-21

Вариант 3: Библиотечный учет

Таблица 38 – БИБЛИОТЕКА

Читат_билет	ФИО	Группа	Книга	Автор	Дата_выдачи
Б001	Иванов И.И.	ИСП-21	SQL за 21д	Бен Форта	2024-01-10
Б001	Иванов И.И.	ИСП-21	Python	Лутц	2024-01-12
Б002	Петров П.П.	ПР-22	SQL за 21д	Бен Форта	2024-01-11
Б003	Сидорова С.	ИСП-21	Python	Лутц	2024-01-09

Вариант 4: Учет оборудования

Таблица 39 – ОБОРУДОВАНИЕ

Инв_номер	Название	Тип	Ответственный	Кабинет	Телефон
КОМП001	Ноутбук	Компьютер	Иванов И.И.	101	1234
КОМП002	Ноутбук	Компьютер	Иванов И.И.	101	1234
ПРОЕК001	Проектор	Проектор	Петров П.П.	201	5678
КОМП003	ПК	Компьютер	Сидоров С.С.	102	9012

Вариант 5: Расписание занятий

Таблица 40 – РАСПИСАНИЕ

Группа	День	Пара	Предмет	Преподаватель	Аудитория
ИСП-21	Понедельник	1	БД	Петрова П.П.	101
ИСП-21	Понедельник	2	БД	Петрова П.П.	101
ИСП-21	Вторник	1	ООП	Сидоров С.С.	201
ПР-22	Понедельник	3	БД	Петрова П.П.	102

Вариант 6: Учет заказов в кафе

Таблица 41 – ЗАКАЗЫ_КАФЕ

№_заказа	Дата_время	Столик	Официант	Блюдо	Количество
100	2024-01-15	1	Иванова	Суп	2
100	2024-01-15	1	Иванова	Гарнир	2
101	2024-01-15	2	Петров	Суп	1
102	2024-01-15	3	Иванова	Десерт	3

Вариант 7: Учет сотрудников и отделов

Таблица 42 – СОТРУДНИКИ

Таб_номер	ФИО	Должность	Отдел	Телефон	Нач_отдела
001	Иванов И.И.	Программист	IT	123	Сидоров

Таб_номер	ФИО	Должность	Отдел	Телефон	Нач_отдела
002	Петров П.П.	Тестировщик	IT	124	Сидоров
003	Сидоров С.С	Нач_отдела	IT	125	NULL
004	Кузнецова	Бухгалтер	Бухгалтерия	126	Смирнова

Вариант 8: Учет проектов и задач

Таблица 43 – ПРОЕКТЫ

Проект	Руководитель	Исполнитель	Должность	Задача	Срок
Сайт	Иванов	Петров	Программист	Дизайн	2024-02-01
Сайт	Иванов	Сидоров	Дизайнер	Дизайн	2024-01-25
Сайт	Иванов	Петров	Программист	Верстка	2024-02-15
Мобильное	Сидоров	Кузнецов	Программист	Разработка	2024-03-01

Вариант 9: Учет аренды автомобилей

Таблица 44 – АРЕНДА

Договор	Клиент	Телефон	Автомобиль	Марка	Срок_аренды
A001	Иванов И.И.	89161234567	A123BC77	Toyota	7 дней
A002	Петров	89098765432	B456OP78	Honda	3 дня

Договор	Клиент	Телефон	Автомобиль	Марка	Срок_аренды
	П.П.				
А003	Иванов И.И.	89161234567	С789ТУ79	BMW	5 дней
А004	Сидоров С.С	89163456789	А123ВС77	Toyota	2 дня

Вариант 10: Учет спортивных секций

Таблица 45 – СПОРТ_СЕКЦИИ

Секция	Тренер	Телефон	Ученик	Возраст	Расписание
Футбол	Петров П.П.	89161234567	Иванов	15	Пн,Ср 18:00
Футбол	Петров П.П.	89161234567	Сидоров	16	Пн,Ср 18:00
Баскетбол	Кузнецов	89098765432	Петров	17	Вт,Чт 19:00
Футбол	Петров П.П.	89161234567	Кузнецов	15	Пн,Ср 18:00

Лабораторная работа №9 «Преобразование реляционной БД в сущности и связи»

Теоретические сведения

В предыдущих лабораторных работах мы прошли путь от анализа предметной области к созданию физической реляционной базы данных.

На практике часто возникает обратная задача: необходимо понять и документально описать структуру уже существующей базы данных. Этот процесс, иногда называемый обратным проектированием (reverse engineering), включает анализ схемы БД (таблиц, ограничений, индексов) и восстановление на её основе концептуальной модели.

Ключевые этапы преобразования.

1. Идентификация сущностей: каждая таблица в БД, как правило, соответствует одной сущности предметной области. Исключения могут составлять служебные или технические таблицы (например, для аудита).
2. Определение атрибутов: столбцы таблицы становятся атрибутами соответствующей сущности. Необходимо проанализировать их имена, типы данных и ограничения (NOT NULL, UNIQUE).
3. Выделение ключей.
4. Восстановление связей: внешние ключи прямо указывают на наличие связи между сущностями. Тип связи (1:1, 1:M, M:M) определяется по уникальности значений в связанных столбцах и структуре таблиц.

Отличие от прямого проектирования: при прямом проектировании мы идём от бизнес-логики к таблицам. При обратном преобразовании мы исходим из технической реализации (таблицы, ключи) и восстанавливаем бизнес-логику, что требует анализа имен и семантики данных.

Пример выполнения лабораторной работы

Исходная схема реляционной БД «Библиотека колледжа» (фрагмент):

В БД созданы следующие таблицы (табл. 46-49):

Таблица 46 – books (Книги)

Имя столбца	Тип данных	Ограничения	Комментарий
book_id	INT	PRIMARY KEY, NOT NULL	Уникальный ID книги
isbn	VARCHAR(17)	UNIQUE, NOT NULL	Международный номер
title	VARCHAR(200)	NOT NULL	Название книги
publication_year	INT		Год издания

Таблица 47 – authors (Авторы)

Имя столбца	Тип данных	Ограничения	Комментарий
author_id	INT	PRIMARY KEY, NOT NULL	Уникальный ID автора
first_name	VARCHAR(50)	NOT NULL	Имя
last_name	VARCHAR(50)	NOT NULL	Фамилия

Таблица 48 – book_authors (Связь Книги-Авторы)

Имя столбца	Тип данных	Ограничения	Комментарий
book_id	INT	NOT NULL, FOREIGN KEY REFERENCES books(book_id)	Ссылка на книгу
author_id	INT	NOT NULL, FOREIGN KEY REFERENCES authors(author_id)	Ссылка на автора
		PRIMARY KEY (book_id, author_id)	Составной первичный ключ

Таблица 49 – book_copies (Экземпляры книг)

Имя столбца	**Тип данных**	**Ограничения**	**Комментарий**
copy_id	INT	PRIMARY KEY, NOT NULL	Инвентарный номер
book_id	INT	NOT NULL, FOREIGN KEY REFERENCES books(book_id)	Ссылка на издание (книгу)
location	VARCHAR(50)	NOT NULL	Место хранения (стеллаж)
status	VARCHAR(20)	NOT NULL, CHECK (status IN ('available', 'issued'))	Текущий статус

Шаги преобразования:

1. Идентификация сущностей.
   - Таблица books → Сущность Книга (издание).
   - Таблица authors → Сущность Автор.
   - Таблица book_copies → Сущность ЭкземплярКниги (физическая копия).
   - Таблица book_authors → это ассоциативная таблица для реализации связи M:M. Она не становится самостоятельной сущностью на концептуальной диаграмме.
2. Определение атрибутов и ключей.

Для каждой сущности перечисляем атрибуты на основе столбцов соответствующих таблиц (табл. 50). Первичный ключ таблицы становится PK сущности.

Таблица 50 – Сущности и атрибуты, восстановленные из схемы БД

Сущность	Атрибуты	Примечание
Книга	**book_id**, isbn, title, publi-cation_year	PK book_id – суррогатный ключ. isbn – альтернативный уникальный ключ.
Автор	**author_id**, first_name, last_name	PK author_id – суррогатный ключ.
ЭкземплярКниги	**copy_id**, book_id (FK), location, status	PK copy_id. Атрибут book_id является внешним ключом, указывающим на связь с сущностью Книга.

3. Восстановление связей.

Связь между Книга и Автор: наличие ассоциативной таблицы book_authors с двумя внешними ключами однозначно указывает на связь «Многие ко многим» (M:M). Одна книга может быть написана несколькими авторами, один автор может написать несколько книг.

Связь между Книга и ЭкземплярКниги: в таблице book_copies есть внешний ключ book_id, ссылающийся на books(book_id). Один экземпляр связан только с одной книгой, но одной книге (изданию) может соответствовать много экземпляров. Это связь «Один ко многим» (1:M).

4. Построение концептуальной ER-диаграммы.

На основе проведенного анализа строится диаграмма. Связь M:M может быть изображена как есть или с явным указанием ассоциативной сущности (на логическом уровне). Для концептуального уровня изображаем непосредственно M:M (рис. 49).

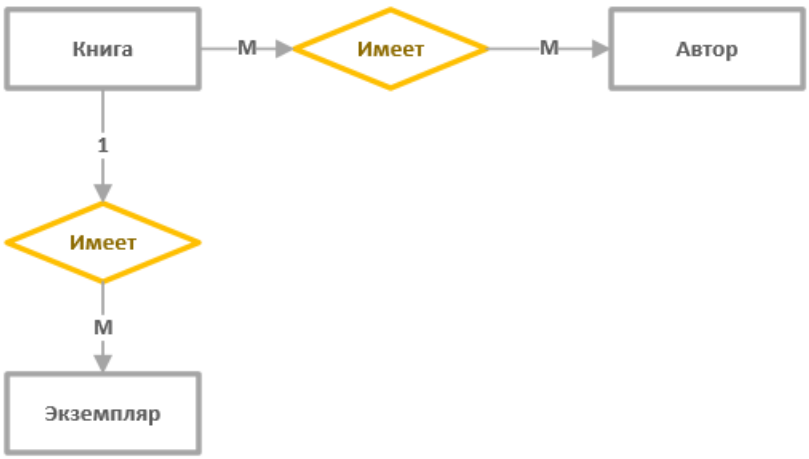


Рис. 49 Концептуальная модель

Пояснение:

Книга – M:M – Автор: Связь «многие-ко-многим».

Книга – 1:M – ЭкземплярКниги: Одна книга может существовать в виде многих физических экземпляров. Каждый экземпляр относится к одной конкретной книге.

Последовательность выполнения лабораторной работы

1. Выберите фрагмент схемы реляционной БД (описание таблиц, столбцов и ключей) согласно своему индивидуальному варианту.
2. Проанализируйте схему БД. Для каждой таблицы определите:
   - Её предполагаемое имя сущности.
   - Атрибуты (название, возможный тип и смысл).
   - Первичный ключ.
   - Внешние ключи (укажите, на какую таблицу/сущность они ссылаются).
3. Выявите и классифицируйте связи. Для каждой пары связанных сущностей определите тип связи (1:1, 1:M, M:M). Обоснуйте вывод, ссылаясь на структуру ключей.

4. Составьте отчет. Представьте результаты в виде двух таблиц:
   - Таблица «Восстановленные сущности и атрибуты».
   - Таблица «Восстановленные связи» со столбцами: Сущность А, Сущность Б, Тип связи (с обозначением 1:1, 1:M, M:M), Обоснование (на основании каких ключей сделан вывод).

5. Постройте концептуальную ER-диаграмму для своей схемы БД, используя полученные сущности и связи.

6. Сделайте вывод. Опишите, какие сложности возникли при интерпретации технических имен столбцов в бизнес-понятия. Все ли связи были очевидны? Какая дополнительная информация о предметной области могла бы помочь в анализе?

Варианты индивидуальных заданий

Вариант 1. Схема БД «Фитнес-центр»

Таблица 51 – Исходные таблицы БД «Фитнес-центр»

Сущность	**Атрибуты**
clients	client_id, passport_number, full_name, phone, birth_date
memberships	membership_id, client_id, type, start_date, end_date, status
group_classes	class_id, name, trainer_id, schedule_time, max_participants
class_registrations	registration_id, client_id, class_id, date

Вариант 2. Схема БД «Служба доставки»

Таблица 52 – Исходные таблицы БД «Служба доставки»

Сущность	**Атрибуты**
couriers	courier_id, full_name, phone, vehicle_type, status
orders	order_id, customer_address, recipient_address, weight, created_at
order_status_log	log_id, order_id, status, timestamp, location
order_assignments	assignment_id, order_id, courier_id, assigned_time

Вариант 3. Схема БД «Библиотека цифрового контента»

Таблица 53 – Исходные таблицы БД «Библиотека цифрового контента»

Сущность	**Атрибуты**
users	user_id, email, username, subscription_tier
media_items	item_id, title, author, media_type, genre, file_size
downloads	download_id, user_id, item_id, download_date, license_key
reviews	review_id, user_id, item_id, rating, comment_text

Вариант 4. Схема БД «Ремонтная мастерская»

Таблица 54 – Исходные таблицы БД «Ремонтная мастерская»

Сущность	**Атрибуты**
devices	device_id, customer_id, type, brand, serial_number, fault_description
customers	customer_id, full_name, phone, email
repair_orders	order_id, device_id, master_id, acceptance_date, estimated_cost, status
masters	master_id, full_name, specialization, qualification_level

Вариант 5. Схема БД «Коворкинг-центр»

Таблица 55 – Исходные таблицы БД «Коворкинг-центр»

Сущность	**Атрибуты**
workspaces	workspace_id, zone, seats, has_monitor, daily_rate
companies	company_id, name, contact_person, phone
bookings	booking_id, workspace_id, company_id, date, time_slot
invoices	invoice_id, booking_id, total_amount, issued_date, payment_status

Вариант 6. Схема БД «Агентство мероприятий»

Таблица 56 – Исходные таблицы БД «Агентство мероприятий»

Сущность	**Атрибуты**
events	event_id, client_id, name, event_date, venue, budget
clients	client_id, company_name, contact_name, phone
vendors	vendor_id, service_type, company_name, rating
event_vendors	event_vendor_id, event_id, vendor_id, service_details, agreed_price

Вариант 7. Схема БД «Лаборатория исследований»

Таблица 57 – Исходные таблицы БД «Лаборатория исследований»

Сущность	**Атрибуты**
samples	sample_id, research_project_id, collection_date, sample_type, storage_location
projects	project_id, title, lead_scientist_id, start_date, funding_source
scientists	scientist_id, full_name, department, academic_degree
experiments	experiment_id, sample_id, scientist_id, procedure, result, conclusion

Вариант 8. Схема БД «Студия звукозаписи»

Таблица 58 – Исходные таблицы БД «Студия звукозаписи»

Сущность	**Атрибуты**
artists	artist_id, stage_name, real_name, genre
albums	album_id, title, artist_id, release_date, label
tracks	track_id, album_id, title, duration, bpm
studio_sessions	session_id, track_id, engineer_id, booking_date, studio_room, duration_hours
engineers	engineer_id, full_name, specialization

Вариант 9. Схема БД «Прокат оборудования»

Таблица 59 – Исходные таблицы БД «Прокат оборудования»

Сущность	**Атрибуты**
equipment	item_id, name, category, daily_rental_price, condition_status
customers	customer_id, full_name, passport_data, deposit_amount
rental_contracts	contract_id, customer_id, start_date, planned_end_date
contract_items	contract_item_id, contract_id, item_id, actual_return_date, damage_notes

Вариант 10. Схема БД «Онлайн-голосование/опросы»

Таблица 60 – Исходные таблицы БД «Онлайн-голосование/опросы»

Сущность	**Атрибуты**
polls	poll_id, title, creator_id, start_date, end_date, is_anonymous
users	user_id, username, registration_date
questions	question_id, poll_id, question_text, question_type
answers	answer_id, question_id, user_id, answer_text, submission_time

Вариант 11. Схема БД «Учет посещаемости мероприятий»

Таблица 61 – Исходные таблицы БД «Учет посещаемости мероприятий»

Сущность	**Атрибуты**
events	event_id, name, organizer, event_date_time, venue, capacity
participants	participant_id, ticket_code, full_name, email, registration_type
check_ins	check_in_id, participant_id, event_id, check_in_time, channel

Вариант 12. Схема БД «Система бронирования переговорных»

Таблица 62 – Исходные таблицы БД «Система бронирования переговорных»

Сущность	**Атрибуты**
meeting_rooms	room_id, floor, seats, has_projector, has_board
employees	employee_id, full_name, department, phone_extension
bookings	booking_id, room_id, booker_id, start_time, end_time, meeting_topic
booking_participants	booking_participant_id, booking_id, employee_id

Вариант 13. Схема БД «Фермерское хозяйство (учет урожая)»

Таблица 63 – Исходные таблицы БД «Фермерское хозяйство (учет урожая)»

Сущность	**Атрибуты**
fields	field_id, area_hectares, crop_type, soil_type
harvests	harvest_id, field_id, harvest_date, yield_kg
storage	storage_id, harvest_id, warehouse_section, initial_weight, current_weight
sales	sale_id, harvest_id, buyer, sale_date, sold_weight_kg, price_per_kg

Вариант 14. Схема БД «Платформа онлайн-курсов»

Таблица 64 – Исходные таблицы БД «Платформа онлайн-курсов»

Сущность	**Атрибуты**
courses	course_id, title, instructor_id, category, price, is_published
instructors	instructor_id, full_name, bio, specialization
students	student_id, email, full_name, registration_date
enrollments	enrollment_id, student_id, course_id, enrollment_date, completion_percentage
lessons	lesson_id, course_id, title, video_url, duration_minutes, order

Вариант 15. Схема БД «Учет инвентаря в отеле»

Таблица 65 – Исходные таблицы БД «Учет инвентаря в отеле»

Сущность	**Атрибуты**
inventory_items	item_id, name, category, model, serial_number, purchase_date
hotel_rooms	room_id, room_number, floor, room_type
item_placements	placement_id, item_id, room_id, placement_date
maintenance_log	log_id, item_id, issue_description, report_date, fix_date, technician

Лабораторная работа №10 «Проектирование реляционной БД»

Теоретические сведения

В предыдущих лабораторных работах был пройден полный цикл проектирования базы данных.

Физическое проектирование – это финальный этап, на котором логическая модель данных (списки сущностей, атрибутов и связей) реализуется в конкретной системе управления базами данных (СУБД). Результатом является работающая база данных, готовая к использованию приложениями.

Ключевые шаги физического проектирования в SSMS:

1. Создание базы данных: определение имени, файлов данных (*.mdf) и журналов (*.ldf), их начального размера и местоположения.
2. Создание таблиц: для каждой сущности логической модели создается таблица. Каждый атрибут становится столбцом с определенным типом данных SQL Server (например, INT, VARCHAR(n), DATE, DECIMAL).
3. Определение ключей.
4. Добавление ограничений (CONSTRAINTS):
   - NOT NULL: Запрещает пустые значения в столбце.
   - UNIQUE: Гарантирует уникальность значений в столбце (кроме NULL).
   - CHECK: Задает условие для значений столбца (например, rating >= 1 AND rating <= 5).
   - DEFAULT: Задает значение по умолчанию для столбца при вставке новой строки.
5. Заполнение тестовыми данными: вставка в таблицы реалистичных данных для проверки структуры и целостности связей с помощью инструкции INSERT.
6. Проверка связей и целостности.

Последовательность выполнения лабораторной работы

На основе описания сущностей и атрибутов из вашего варианта лабораторной работы № 9 создать в SQL Server Management Studio (SSMS) полноценную базу данных.

1. Подключение к серверу и создание базы данных.

1.1 Запустите SQL Server Management Studio 22.

1.2 Подключитесь к вашему экземпляру SQL Server (например, localhost\SQLEXPRESS) с использованием аутентификации Windows или SQL Server.

1.3 В обозревателе объектов щелкните правой кнопкой мыши на папке «Базы данных».

1.4 Выберите «Создать базу данных...».

1.5 В открывшемся окне:

- В поле «Имя базы данных» введите название на английском языке, соответствующее вашему варианту.
- Оставьте остальные настройки по умолчанию.
- Нажмите «ОК».

2. Создание таблиц.

2.1 В обозревателе объектов разверните только что созданную базу данных.

2.2 Щелкните правой кнопкой мыши на папке «Таблицы» и выберите «Создать» -> «Таблица...».

2.3 Откроется конструктор таблиц. Для каждой сущности из вашего задания создайте отдельную таблицу:

- В столбце «Имя столбца» введите название атрибута.
- В столбце «Тип данных» выберите соответствующий тип из выпадающего списка.

– Снимите галочку «Разрешить значения NULL» для обязательных к заполнению полей.

2.4 Определение первичного ключа (PK):

– Щелкните правой кнопкой мыши на строке столбца, который должен быть первичным ключом (обычно это *_id).
– Выберите «Задать первичный ключ». Слева от столбца появится значок ключа.

2.5 Сохраните таблицу, нажав Ctrl+S или на значок «Сохранить» на панели инструментов.

2.6 В диалоговом окне введите имя таблицы (например, books, authors). Нажмите «ОК».

2.7 Повторите шаги 2.2-2.6 для всех сущностей из вашего варианта.

3. Создание связей.

3.1 Откройте конструктор таблицы, которая должна содержать внешний ключ.

3.2 На панели инструментов конструктора таблиц нажмите кнопку «Связи по внешнему ключу».

3.3 В открывшемся диалоговом окне нажмите кнопку «Добавить».

3.4 В правой части окна, в свойствах новой связи, нажмите на кнопку с тремя точками (…) в строке «Спецификация таблиц и столбцов».

3.5 В новом окне:

– «Таблица первичного ключа»: выберите родительскую таблицу.
– «Столбец первичного ключа»: выберите соответствующий первичный ключ.
– «Таблица внешнего ключа»: здесь будет указана текущая таблица.

– «Столбец внешнего ключа»: выберите столбец, который ссылается на PK.

– Нажмите «ОК».

3.6 Нажмите «Закрыть» в окне «Связи по внешнему ключу».

3.7 Сохраните изменения в таблице (Ctrl+S). SSMS может запросить подтверждение на пересоздание таблицы – согласитесь.

3.8 Повторите для всех необходимых связей в вашей схеме.

4. Заполнение таблиц тестовыми данными.

4.1 В обозревателе объектов найдите созданную таблицу.

4.2 Щелкните на ней правой кнопкой мыши и выберите «Изменить первые 200 строк».

4.3 В открывшейся сетке введите минимум по 10 осмысленных записей в каждую таблицу.

5. Проверка целостности и выполнение запросов.

Создайте новый запрос к вашей базе данных (кнопка «Создать запрос» на панели инструментов) и выполните следующие команды:

5.1 Просмотр данных:

```
-- Просмотр содержимого всех таблиц

SELECT * FROM books;

SELECT * FROM authors;

SELECT * FROM book_authors;
```

Примечание: названия таблиц необходимо заменить в соответствии со своим вариантом.

5.2 Проверка связи «один-ко-многим»:

```
-- Пример: показать все книги с их авторами (через промежуточную таблицу)
```

```
SELECT b.title, a.first_name, a.last_name
FROM books b
INNER JOIN book_authors ba ON b.book_id = ba.book_id
INNER JOIN authors a ON ba.author_id = a.author_id;
```

Примечание: названия таблиц и атрибутов необходимо заменить в соответствии со своим вариантом.

5.3 Проверка ограничений:

- Попробуйте вставить запись с дублирующимся значением первичного ключа. Должна возникнуть ошибка.
- Попробуйте вставить запись во внешний ключ со значением, которого нет в родительской таблице. Должна возникнуть ошибка ссылочной целостности.

Лабораторная работа №11 «Проектирование реляционной БД. Нормализация таблиц»

Теоретические сведения

Нормализация – процесс организации данных в базе данных для устранения избыточности и аномалий (вставки, обновления, удаления). В предыдущих работах мы уже сталкивались с основами нормализации (Л.р. №8), а также создавали таблицы без избыточности (Л.р. №10). В этой работе мы сосредоточимся на практическом применении нормализации для исправления плохой структуры таблиц.

Ключевые аномалии:

- Аномалия вставки: невозможность добавить данные, пока не существуют другие данные.
- Аномалия обновления: необходимость изменения одних и тех же данных в нескольких местах.
- Аномалия удаления: потеря важной информации при удалении записей.

Нормальные формы на практике в SQL Server:

1. Первая нормальная форма (1НФ):
   - Все значения в столбцах атомарны (неделимы).
   - В таблице нет повторяющихся групп.
   - В SSMS это означает, что каждый столбец имеет простой тип данных, а не массивы или списки.
2. Вторая нормальная форма (2НФ):
   - Таблица находится в 1НФ.
   - Все неключевые атрибуты полностью зависят от всего составного первичного ключа.
   - В SSMS это проверяется анализом функциональных зависимостей.

3. Третья нормальная форма (3НФ):
   - Таблица находится в 2НФ.
   - Отсутствуют транзитивные зависимости (неключевые атрибуты не зависят от других неключевых атрибутов).

Процесс нормализации в SSMS включает:

1. Анализ существующей ненормализованной таблицы.
2. Выявление функциональных зависимостей.
3. Разделение таблицы на несколько связанных таблиц.
4. Создание новых таблиц с соответствующими ключами.
5. Перенос данных из исходной таблицы в нормализованные.
6. Удаление исходной ненормализованной таблицы (опционально).

Основные правила SQL:

- Регистр: SQL не чувствителен к регистру, но принято писать ключевые слова (CREATE, TABLE) заглавными буквами
- Точка с запятой (;): завершает команду (в SQL Server GO также разделяет команды)
- Комментарии: начинаются с -- для одной строки или /* */ для нескольких строк

Рассмотрим запрос создания базы данных:

```
CREATE DATABASE OnlineShop_Normalization;
```

CREATE DATABASE – команда «создать базу данных».

OnlineShop_Normalization – ИМЯ базы данных.

; – конец команды.

Правила именования:

- Без пробелов.
- Без русских букв.
- Лучше использовать английские слова.

– Можно использовать знак подчеркивания _.

Рассмотрим еще один запрос:

```
USE OnlineShop_Normalization;
```

USE – команда «использовать».

OnlineShop_Normalization – имя базы, которую мы создали.

Теперь все следующие команды будут выполняться в этой базе.

Чтобы создать таблицу необходимо воспользоваться командой CREATE TABLE:

```
CREATE TABLE OrderData (
```

CREATE TABLE – команда «создать таблицу».

OrderData – ИМЯ таблицы.

(– начало перечисления столбцов.

Структура описания столбцов: OrderID INT,

Формат: Имя_столбца Тип_данных,

OrderID – имя столбца.

INT – тип данных «целое число».

, – разделитель столбцов.

Шаблоны для работы с БД:

```
-- СОЗДАТЬ БАЗУ ДАННЫХ
CREATE DATABASE ИмяБазы;

-- ВЫБРАТЬ БАЗУ ДАННЫХ
USE ИмяБазы;

-- СОЗДАТЬ ТАБЛИЦУ
CREATE TABLE ИмяТаблицы (
  Столбец1 ТипДанных1,
  Столбец2 ТипДанных2,
  Столбец3 ТипДанных3
);
```

```
-- УДАЛИТЬ ТАБЛИЦУ
DROP TABLE ИмяТаблицы;

-- УДАЛИТЬ БАЗУ ДАННЫХ
        DROP DATABASE ИмяБазы;
```

Пример выполнения лабораторной работы

Исходная таблица «Заказы в интернет-магазине».

Шаг 1. Создание базы данных и ненормализованной таблицы.

```
-- Создание базы данных
CREATE DATABASE OnlineShop_Normalization;
GO

USE OnlineShop_Normalization;
GO

-- Создание ненормализованной таблицы
CREATE TABLE OrderData (
  OrderID INT,
  OrderDate DATE,
  CustomerID INT,
  CustomerName VARCHAR(100),
  CustomerEmail VARCHAR(100),
  CustomerCity VARCHAR(50),
  ProductID INT,
  ProductName VARCHAR(100),
  Category VARCHAR(50),
  UnitPrice DECIMAL(10,2),
  Quantity INT,
  TotalPrice DECIMAL(10,2),
  ShippingAddress VARCHAR(200)
);
        GO
```

На рис. 50 представлен результат создания ненормализованной таблицы.

DESKTOP-G8...o.OrderData

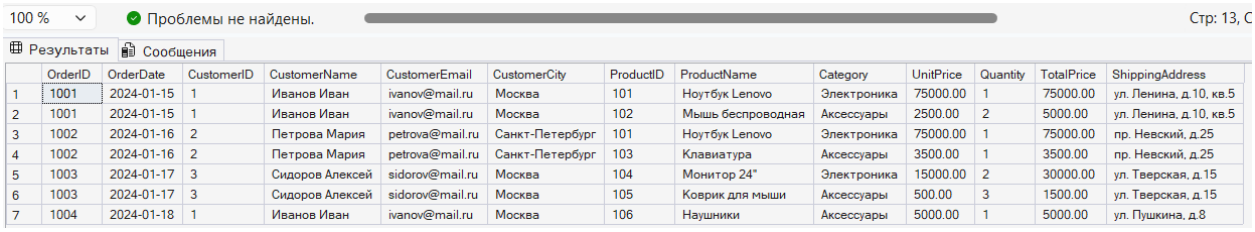
Имя столбца	Тип данных	Разрешить знач...
OrderID	int	☑
OrderDate	date	☑
CustomerID	int	☑
CustomerName	varchar(100)	☑
CustomerEmail	varchar(100)	☑
CustomerCity	varchar(50)	☑
ProductID	int	☑
ProductName	varchar(100)	☑
Category	varchar(50)	☑
UnitPrice	decimal(10, 2)	☑
Quantity	int	☑
TotalPrice	decimal(10, 2)	☑
ShippingAddress	varchar(200)	☑
		☐

Рис. Созданная таблица в SSMS

Шаг 2. Заполнение тестовыми данными.

```
-- Вставка тестовых данных
INSERT INTO OrderData VALUES
(1001, '2024-01-15', 1, 'Иванов Иван', 'ivanov@mail.ru', 'Москва', 101, 'Ноутбук Lenovo',
'Электроника', 75000.00, 1, 75000.00, 'ул. Ленина, д.10, кв.5'),
(1001, '2024-01-15', 1, 'Иванов Иван', 'ivanov@mail.ru', 'Москва', 102, 'Мышь беспроводная',
'Аксессуары', 2500.00, 2, 5000.00, 'ул. Ленина, д.10, кв.5'),
(1002, '2024-01-16', 2, 'Петрова Мария', 'petrova@mail.ru', 'Санкт-Петербург', 101, 'Ноутбук
Lenovo', 'Электроника', 75000.00, 1, 75000.00, 'пр. Невский, д.25'),
(1002, '2024-01-16', 2, 'Петрова Мария', 'petrova@mail.ru', 'Санкт-Петербург', 103, 'Клавиа-
тура', 'Аксессуары', 3500.00, 1, 3500.00, 'пр. Невский, д.25'),
(1003, '2024-01-17', 3, 'Сидоров Алексей', 'sidorov@mail.ru', 'Москва', 104, 'Монитор 24"',
'Электроника', 15000.00, 2, 30000.00, 'ул. Тверская, д.15'),
(1003, '2024-01-17', 3, 'Сидоров Алексей', 'sidorov@mail.ru', 'Москва', 105, 'Коврик для мы-
ши', 'Аксессуары', 500.00, 3, 1500.00, 'ул. Тверская, д.15'),
(1004, '2024-01-18', 1, 'Иванов Иван', 'ivanov@mail.ru', 'Москва', 106, 'Наушники', 'Аксессу-
ары', 5000.00, 1, 5000.00, 'ул. Пушкина, д.8');
GO
-- Проверка вставленных данных
        SELECT * FROM OrderData;
```

На рис. 51 продемонстрирован результат проверки вставленных данных.



100 % Проблемы не найдены. Стр: 13, С

Результаты Сообщения

	OrderID	OrderDate	CustomerID	CustomerName	CustomerEmail	CustomerCity	ProductID	ProductName	Category	UnitPrice	Quantity	TotalPrice	ShippingAddress
1	1001	2024-01-15	1	Иванов Иван	ivanov@mail.ru	Москва	101	Ноутбук Lenovo	Электроника	75000.00	1	75000.00	ул. Ленина, д.10, кв.5
2	1001	2024-01-15	1	Иванов Иван	ivanov@mail.ru	Москва	102	Мышь беспроводная	Аксессуары	2500.00	2	5000.00	ул. Ленина, д.10, кв.5
3	1002	2024-01-16	2	Петрова Мария	petrova@mail.ru	Санкт-Петербург	101	Ноутбук Lenovo	Электроника	75000.00	1	75000.00	пр. Невский, д.25
4	1002	2024-01-16	2	Петрова Мария	petrova@mail.ru	Санкт-Петербург	103	Клавиатура	Аксессуары	3500.00	1	3500.00	пр. Невский, д.25
5	1003	2024-01-17	3	Сидоров Алексей	sidorov@mail.ru	Москва	104	Монитор 24"	Электроника	15000.00	2	30000.00	ул. Тверская, д.15
6	1003	2024-01-17	3	Сидоров Алексей	sidorov@mail.ru	Москва	105	Коврик для мыши	Аксессуары	500.00	3	1500.00	ул. Тверская, д.15
7	1004	2024-01-18	1	Иванов Иван	ivanov@mail.ru	Москва	106	Наушники	Аксессуары	5000.00	1	5000.00	ул. Пушкина, д.8

Рис. 51 Отображение тестовых данных

Шаг 3. Анализ проблем и функциональных зависимостей.

Выявленные проблемы:

1. Избыточность данных: информация о клиенте (имя, email, город) повторяется для каждого заказа.
2. Аномалия обновления: при смене email клиента нужно обновить все связанные записи.
3. Аномалия вставки: нельзя добавить нового клиента без заказа.
4. Аномалия удаления: при удалении последнего заказа клиента теряется информация о клиенте.

Функциональные зависимости:

1. OrderID → OrderDate, CustomerID, CustomerName, CustomerEmail, CustomerCity, ShippingAddress
2. CustomerID → CustomerName, CustomerEmail, CustomerCity
3. ProductID → ProductName, Category, UnitPrice
4. {OrderID, ProductID} → Quantity, TotalPrice
5. TotalPrice = UnitPrice * Quantity (вычисляемая зависимость)

Первичный ключ: {OrderID, ProductID} (составной)

Шаг 4. Нормализация до 1НФ.

Проверка 1НФ:

✓ Все значения атомарны

✓ Нет повторяющихся групп в столбцах

✗ Но есть смешение разных сущностей (клиент, продукт, заказ)

Таблица уже находится в 1НФ формально, но содержит данные о разных сущностях.

Шаг 5. Нормализация до 2НФ.

Выявление частичных зависимостей:

- CustomerName, CustomerEmail, CustomerCity зависят только от CustomerID (часть ключа через транзитивность)
- ProductName, Category, UnitPrice зависят только от ProductID (часть ключа)
- OrderDate, ShippingAddress зависят только от OrderID (часть ключа)

Создание нормализованных таблиц (рис. 52):

```
-- Таблица Клиенты
CREATE TABLE Customers (
  CustomerID INT PRIMARY KEY,
  CustomerName VARCHAR(100) NOT NULL,
  CustomerEmail VARCHAR(100),
  CustomerCity VARCHAR(50)
);
GO

-- Таблица Продукты
CREATE TABLE Products (
  ProductID INT PRIMARY KEY,
  ProductName VARCHAR(100) NOT NULL,
  Category VARCHAR(50),
  UnitPrice DECIMAL(10,2) NOT NULL
);
GO

-- Таблица Заказы (заголовки)
CREATE TABLE Orders (
  OrderID INT PRIMARY KEY,
  OrderDate DATE NOT NULL,
  CustomerID INT NOT NULL,
  ShippingAddress VARCHAR(200)
);
GO
```

```
-- Таблица Позиции заказов (детали)
CREATE TABLE OrderItems (
  OrderItemID INT IDENTITY(1,1) PRIMARY KEY,
  OrderID INT NOT NULL,
  ProductID INT NOT NULL,
  Quantity INT NOT NULL,
  UnitPrice DECIMAL(10,2) NOT NULL
);
        GO
```

	Имя столбца	Тип данных	Разрешить знач...
🔑	CustomerID	int	☐
	CustomerName	varchar(100)	☐
	CustomerEmail	varchar(100)	☑
	CustomerCity	varchar(50)	☑

Рис. 52 Нормализованная таблица «Клиенты»

Шаг 6. Нормализация до 3НФ.

Проверка транзитивных зависимостей:

1. В таблице Customers: нет транзитивных зависимостей (все атрибуты зависят от PK).
2. В таблице Products: нет транзитивных зависимостей.
3. В таблице Orders: нет транзитивных зависимостей.
4. В таблице OrderItems: нет транзитивных зависимостей.

Все таблицы находятся в 3НФ.

Шаг 7. Создание внешних ключей.

```
-- Связь Заказы -> Клиенты
ALTER TABLE Orders
ADD CONSTRAINT FK_Orders_Customers
FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID);
GO
```

```
-- Связь Позиции заказов -> Заказы
ALTER TABLE OrderItems
ADD CONSTRAINT FK_OrderItems_Orders
FOREIGN KEY (OrderID) REFERENCES Orders(OrderID);
GO

-- Связь Позиции заказов -> Продукты
ALTER TABLE OrderItems
ADD CONSTRAINT FK_OrderItems_Products
FOREIGN KEY (ProductID) REFERENCES Products(ProductID);
GO
```

Разберем по частям:

1. ALTER TABLE Orders
   - ALTER TABLE – это команда для изменения существующей таблицы.
   - Orders – имя таблицы, которую мы изменяем.
2. ADD CONSTRAINT FK_Orders_Customers
   - ADD CONSTRAINT – добавить ограничение (constraint). Ограничение – это правило, которое применяется к данным в таблице.
   - FK_Orders_Customers – имя ограничения. Обычно для внешних ключей имя начинается с FK, затем указываются имена таблиц, которые он связывает.

В данном случае: FK (внешний ключ) для связи таблицы Orders с таблицей Customers.

3. FOREIGN KEY (CustomerID)
   - FOREIGN KEY – указывает, что мы создаем внешний ключ.

- (CustomerID) – это столбец в текущей таблице (Orders), который будет внешним ключом.

4. REFERENCES Customers(CustomerID)
   - REFERENCES – указывает на таблицу и столбец, на которые ссылается внешний ключ.
   - Customers – имя таблицы, на которую ссылаемся.
   - (CustomerID) – столбец в таблице Customers, на который ссылается наш внешний ключ.

Таким образом, мы говорим:

«В таблице Orders есть столбец CustomerID, который ссылается на столбец CustomerID в таблице Customers».

Это создает связь между двумя таблицами. Теперь в таблице Orders в столбце CustomerID можно помещать только такие значения, которые существуют в столбце CustomerID таблицы Customers. Это обеспечивает целостность данных.

Кроме того, обычно внешний ключ препятствует удалению записей из таблицы Customers, если на них есть ссылки в таблице Orders.

Вторая и третья команды аналогично создают внешние ключи в таблице OrderItems, ссылающиеся на таблицы Orders и Products.

Важно: Для успешного создания внешнего ключа необходимо, чтобы:

- Таблица, на которую ссылаемся (например, Customers), существовала.
- Столбец, на который ссылаемся (например, CustomerID в таблице Customers), был первичным ключом или имел ограничение UNIQUE.
- Типы данных столбцов, которые связываются, были одинаковыми.

Шаг 8. Перенос данных.

```
-- Вставка уникальных клиентов
INSERT INTO Customers (CustomerID, CustomerName, CustomerEmail, CustomerCity)
SELECT DISTINCT CustomerID, CustomerName, CustomerEmail, CustomerCity
FROM OrderData;
GO

-- Вставка уникальных продуктов
INSERT INTO Products (ProductID, ProductName, Category, UnitPrice)
SELECT DISTINCT ProductID, ProductName, Category, UnitPrice
FROM OrderData;
GO

-- Вставка заголовков заказов
INSERT INTO Orders (OrderID, OrderDate, CustomerID, ShippingAddress)
SELECT DISTINCT OrderID, OrderDate, CustomerID, ShippingAddress
FROM OrderData;
GO

-- Вставка позиций заказов
INSERT INTO OrderItems (OrderID, ProductID, Quantity, UnitPrice)
SELECT OrderID, ProductID, Quantity, UnitPrice
FROM OrderData;
    GO
```

Разберем по частям:

Что такое INSERT ... SELECT?

Это команда для копирования данных из одной таблицы в другую. Вместо ручного ввода данных мы берем их из существующей таблицы.

Простая аналогия:

1. У вас есть список покупок на листочке (таблица OrderData)
2. Вы переписываете этот список в разные блокноты:
   - В блокнот «Клиенты» (таблица Customers).
   - В блокнот «Продукты» (таблица Products).
   - И т.д.

INSERT INTO – команда «вставить в».

Customers – имя таблицы, куда вставляем данные.

(CustomerID, CustomerName, CustomerEmail, CustomerCity) – список столбцов, в которые будем вставлять.

Важно: порядок столбцов должен совпадать с порядком в SELECT.

SELECT – команда «выбрать».

DISTINCT – «уникальные» (без повторов).

CustomerID, CustomerName, CustomerEmail, CustomerCity – какие столбцы выбираем

FROM – «из».

OrderData – имя таблицы, откуда берем данные.

Шаг 9. Проверка перенесенных данных (рис. 53).

```
-- Проверка количества записей
SELECT
  'Customers' AS TableName, COUNT(*) AS RecordCount FROM Customers
UNION ALL
SELECT 'Products', COUNT(*) FROM Products
UNION ALL
SELECT 'Orders', COUNT(*) FROM Orders
UNION ALL
        SELECT 'OrderItems', COUNT(*) FROM OrderItems;
```

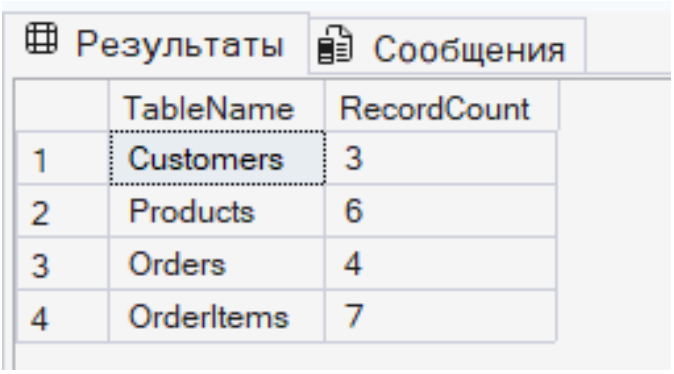


Рис. 53 Результат проверки перенесенных данных

Шаг 10. Проверка целостности данных (рис. 54).

```
-- Сводный запрос для проверки целостности
SELECT
  o.OrderID,
  o.OrderDate,
  c.CustomerName,
  c.CustomerEmail,
  p.ProductName,
  p.Category,
  oi.Quantity,
  oi.UnitPrice,
  (oi.Quantity * oi.UnitPrice) AS TotalPrice,
  o.ShippingAddress
FROM Orders o
JOIN Customers c ON o.CustomerID = c.CustomerID
JOIN OrderItems oi ON o.OrderID = oi.OrderID
JOIN Products p ON oi.ProductID = p.ProductID
        ORDER BY o.OrderID, p.ProductName;
```

Результаты | Сообщения

	OrderID	OrderDate	CustomerName	CustomerEmail	ProductName	Category	Quantity	UnitPrice	TotalPrice	ShippingAddress
1	1001	2024-01-15	Иванов Иван	ivanov@mail.ru	Мышь беспроводная	Аксессуары	2	2500.00	5000.00	ул. Ленина, д.10, кв.5
2	1001	2024-01-15	Иванов Иван	ivanov@mail.ru	Ноутбук Lenovo	Электроника	1	75000.00	75000.00	ул. Ленина, д.10, кв.5
3	1002	2024-01-16	Петрова Мария	petrova@mail.ru	Клавиатура	Аксессуары	1	3500.00	3500.00	пр. Невский, д.25
4	1002	2024-01-16	Петрова Мария	petrova@mail.ru	Ноутбук Lenovo	Электроника	1	75000.00	75000.00	пр. Невский, д.25
5	1003	2024-01-17	Сидоров Алексей	sidorov@mail.ru	Коврик для мыши	Аксессуары	3	500.00	1500.00	ул. Тверская, д.15
6	1003	2024-01-17	Сидоров Алексей	sidorov@mail.ru	Монитор 24"	Электроника	2	15000.00	30000.00	ул. Тверская, д.15
7	1004	2024-01-18	Иванов Иван	ivanov@mail.ru	Наушники	Аксессуары	1	5000.00	5000.00	ул. Пушкина, д.8

Рис. 54 Результат проверки целостности данных

1. Первый JOIN – присоединяем таблицу Customers:
   - JOIN – «присоединить» (объединить с другой таблицей)
   - Customers – имя таблицы для объединения
   - c – псевдоним для таблицы Customers
   - ON – «при условии» (указываем условие соединения)
   - o.CustomerID = c.CustomerID – условие: CustomerID в Orders равен CustomerID в Customers
2. Второй JOIN – присоединяем таблицу OrderItems:
   - OrderItems – таблица позиций заказов
   - oi – псевдоним для OrderItems

– o.OrderID = oi.OrderID – условие: OrderID в Orders равен OrderID в OrderItems

3. Третий JOIN – присоединяем таблицу Products:
   – Products – таблица товаров
   – p – псевдоним для Products
   – oi.ProductID = p.ProductID – условие: ProductID в OrderItems равен ProductID в Products
4. Сортировка результатов:
   – ORDER BY – «упорядочить по»
   – o.OrderID – сначала сортируем по номеру заказа
   – p.ProductName – внутри одного заказа сортируем по названию товара

Шаг 11. Проверка устранения аномалий (рис. 55).

```
-- Тест 1: Добавление нового клиента без заказа (раньше было невозможно)
INSERT INTO Customers (CustomerID, CustomerName, CustomerEmail, CustomerCity)
VALUES (4, 'Новиков Дмитрий', 'novikov@mail.ru', 'Казань');

-- Тест 2: Обновление email клиента в одном месте
UPDATE Customers
SET CustomerEmail = 'ivanov.new@mail.ru'
WHERE CustomerID = 1;

-- Проверка, что обновилось во всех заказах
SELECT DISTINCT c.CustomerEmail
FROM Orders o
JOIN Customers c ON o.CustomerID = c.CustomerID
WHERE o.CustomerID = 1;

-- Тест 3: Удаление заказа (должны удалиться только позиции этого заказа)
DELETE FROM OrderItems WHERE OrderID = 1004;
DELETE FROM Orders WHERE OrderID = 1004;

-- Проверка, что клиент остался в базе
        SELECT * FROM Customers WHERE CustomerID = 1;
```

Результаты Сообщения

	CustomerEmail
1	ivanov.new@mail.ru

	CustomerID	CustomerName	CustomerEmail	CustomerCity
1	1	Иванов Иван	ivanov.new@mail.ru	Москва

Рис. 55 Результат проверки устранения аномалий

Шаг 12. Дополнительное улучшение – вычисляемое поле (рис. 56).

```
-- Добавление вычисляемого поля TotalPrice в OrderItems
ALTER TABLE OrderItems
ADD TotalPrice AS (Quantity * UnitPrice) PERSISTED;
GO

-- Проверка вычисляемого поля
SELECT OrderID, ProductID, Quantity, UnitPrice, TotalPrice
	FROM OrderItems;
```

Результаты Сообщения

	OrderID	ProductID	Quantity	UnitPrice	TotalPrice
1	1001	101	1	75000.00	75000.00
2	1001	102	2	2500.00	5000.00
3	1002	101	1	75000.00	75000.00
4	1002	103	1	3500.00	3500.00
5	1003	104	2	15000.00	30000.00
6	1003	105	3	500.00	1500.00

Рис. 56 Результат реализации вычисляемого поля

Анализ и выводы:

Преимущества нормализованной структуры:

1. Устранена избыточность данных:

- Данные о клиентах хранятся один раз (было 7 повторений, стало 3 записи).
- Данные о продуктах хранятся один раз (было 7 повторений, стало 6 записей).

2. Устранены аномалии:
   - Вставки: можно добавлять клиентов без заказов.
   - Обновления: изменение email клиента происходит в одном месте.
   - Удаления: удаление заказа не приводит к потере информации о клиенте.
3. Улучшена целостность данных:
   - Внешние ключи гарантируют, что в заказах есть только существующие клиенты и продукты.
   - Типы данных и ограничения NOT NULL обеспечивают корректность данных.
4. Гибкость структуры:
   - Легко добавлять новые атрибуты к клиентам или продуктам.
   - Возможность хранить историю изменений цен продуктов.

Сравнение объемов данных:

До нормализации: 7 записей × 13 столбцов = 91 значение

После нормализации:

- Customers: 3 записи × 4 столбца = 12 значений
- Products: 6 записей × 4 столбца = 24 значения
- Orders: 4 записи × 4 столбца = 16 значений
- OrderItems: 7 записей × 5 столбцов = 35 значений

Итого: 87 значений (экономия за счет устранения дублирования).

Трудности и решения:

1. *Проблема:* нужно было правильно определить составной первичный ключ в исходной таблице.

Решение: проанализировали функциональные зависимости и выбрали {OrderID, ProductID}

2. *Проблема:* при переносе данных могли возникнуть дубликаты.

Решение: использовали SELECT DISTINCT при вставке в Customers и Products.

3. *Проблема*: нужно было сохранить вычисляемое поле TotalPrice.

Решение: добавили вычисляемое поле в OrderItems с ключевым словом PERSISTED.

Вывод:

Нормализация до 3НФ позволила создать оптимальную структуру базы данных для интернет-магазина. Новая структура устраняет избыточность данных, предотвращает аномалии и обеспечивает целостность данных через систему внешних ключей. Хотя нормализация немного усложняет запросы (требуются JOIN), она значительно улучшает поддерживаемость и надежность системы. Для дальнейшего улучшения можно рассмотреть добавление индексов на часто используемые поля для оптимизации производительности.

Последовательность выполнения лабораторной работы

1. Создайте новую базу данных.
2. Создайте ненормализованную таблицу по индивидуальному варианту:
   - Выполните SQL-код для создания таблицы из вашего задания.
   - Заполните таблицу тестовыми данными (минимум 5-7 записей).
3. Проанализируйте структуру и выявите проблемы:
   - Составьте список функциональных зависимостей.

– Определите потенциальный первичный ключ.

– Выявите повторяющиеся данные и транзитивные зависимости.

4. Выполните приведение к 1НФ.
5. Выполните приведение к 2НФ.
6. Выполните приведение к 3НФ.
7. Выполните создание связей и перенос данных.
8. Выполните запросы для проверки целостности.
9. Проверьте устранение аномалий.

Варианты индивидуальных заданий

Вариант 1. Таблица «Заказы в интернет-магазине»

```
CREATE TABLE OrderData (
  OrderID INT,
  OrderDate DATE,
  CustomerID INT,
  CustomerName VARCHAR(100),
  CustomerEmail VARCHAR(100),
  CustomerCity VARCHAR(50),
  ProductID INT,
  ProductName VARCHAR(100),
  Category VARCHAR(50),
  UnitPrice DECIMAL(10,2),
  Quantity INT,
  TotalPrice DECIMAL(10,2),
  ShippingAddress VARCHAR(200)
        );
```

Вариант 2. Таблица «Больничные записи»

```
CREATE TABLE HospitalRecords (
  PatientID INT,
  PatientName VARCHAR(100),
  PatientBirthDate DATE,
  DoctorID INT,
  DoctorName VARCHAR(100),
  Specialization VARCHAR(50),
  VisitDate DATE,
  DiagnosisCode VARCHAR(20),
  DiagnosisName VARCHAR(200),
  MedicationID INT,
  MedicationName VARCHAR(100),
  Dosage VARCHAR(50),
  PrescriptionDate DATE
        );
```

Вариант 3. Таблица «Библиотечные записи»

```
CREATE TABLE LibraryRecords (
  BookID INT,
  BookTitle VARCHAR(200),
  AuthorID INT,
  AuthorName VARCHAR(100),
  AuthorNationality VARCHAR(50),
  Genre VARCHAR(50),
  ReaderID INT,
  ReaderName VARCHAR(100),
  ReaderType VARCHAR(20),
  BorrowDate DATE,
  ReturnDate DATE,
  FineAmount DECIMAL(8,2)
);
```

Вариант 4. Таблица «Проекты и задачи»

```
CREATE TABLE ProjectData (
  ProjectID INT,
  ProjectName VARCHAR(100),
  ManagerID INT,
  ManagerName VARCHAR(100),
  ManagerDepartment VARCHAR(50),
  TaskID INT,
  TaskDescription VARCHAR(500),
  EmployeeID INT,
  EmployeeName VARCHAR(100),
  EmployeePosition VARCHAR(50),
  StartDate DATE,
  EndDate DATE,
  Status VARCHAR(20)
);
```

Вариант 5. Таблица «Спортивные мероприятия»

```
CREATE TABLE SportsEvents (
  EventID INT,
  EventName VARCHAR(100),
  EventDate DATE,
  SportType VARCHAR(50),
  TeamID INT,
  TeamName VARCHAR(100),
  CoachID INT,
  CoachName VARCHAR(100),
  PlayerID INT,
  PlayerName VARCHAR(100),
  PlayerPosition VARCHAR(30),
  Score INT,
  Venue VARCHAR(100)
);
```

Вариант 6. Таблица «Учет оборудования»

```
CREATE TABLE EquipmentLog (
```

```
  EquipmentID INT,
  EquipmentName VARCHAR(100),
  EquipmentType VARCHAR(50),
  PurchaseDate DATE,
  PurchasePrice DECIMAL(10,2),
  DepartmentID INT,
  DepartmentName VARCHAR(100),
  EmployeeID INT,
  EmployeeName VARCHAR(100),
  AssignmentDate DATE,
  MaintenanceDate DATE,
  MaintenanceType VARCHAR(50),
  Cost DECIMAL(8,2)
);
```

Вариант 7. Таблица «Гостиничные бронирования»

```
CREATE TABLE HotelBookings (
  BookingID INT,
  GuestID INT,
  GuestName VARCHAR(100),
  GuestPhone VARCHAR(20),
  RoomID INT,
  RoomType VARCHAR(30),
  RoomPrice DECIMAL(8,2),
  CheckInDate DATE,
  CheckOutDate DATE,
  ServiceID INT,
  ServiceName VARCHAR(50),
  ServicePrice DECIMAL(6,2),
  TotalAmount DECIMAL(10,2));
```

Вариант 8. Таблица «Учебные группы»

```
CREATE TABLE GroupData (
  GroupID INT,
  GroupName VARCHAR(50),
  Specialization VARCHAR(100),
  Year INT,
  StudentID INT,
  StudentName VARCHAR(100),
  StudentBirthDate DATE,
  CourseID INT,
  CourseName VARCHAR(100),
  TeacherID INT,
  TeacherName VARCHAR(100),
  Grade DECIMAL(3,1)
);
```

Вариант 9. Таблица «Конференции»

```
CREATE TABLE ConferenceData (
  ConferenceID INT,
  ConferenceName VARCHAR(200),
  ConferenceDate DATE,
  Location VARCHAR(100),
```

```
  SpeakerID INT,
  SpeakerName VARCHAR(100),
  SpeakerAffiliation VARCHAR(150),
  TopicID INT,
  TopicTitle VARCHAR(200),
  ParticipantID INT,
  ParticipantName VARCHAR(100),
  RegistrationType VARCHAR(30)
);
```

Вариант 10. Таблица «Складские записи»

```
CREATE TABLE WarehouseData (
  ProductID INT,
  ProductName VARCHAR(100),
  SupplierID INT,
  SupplierName VARCHAR(100),
  SupplierContact VARCHAR(50),
  Category VARCHAR(50),
  UnitPrice DECIMAL(8,2),
  WarehouseID INT,
  WarehouseLocation VARCHAR(100),
  Quantity INT,
  LastDelivery DATE,
  NextDelivery DATE
);
```

Вариант 11. Таблица «Аренда автомобилей»

```
CREATE TABLE CarRentalData (
  RentalID INT,
  CustomerID INT,
  CustomerName VARCHAR(100),
  CustomerLicense VARCHAR(20),
  CarID INT,
  CarModel VARCHAR(50),
  CarType VARCHAR(30),
  DailyRate DECIMAL(6,2),
  StartDate DATE,
  EndDate DATE,
  InsuranceType VARCHAR(30),
  InsuranceCost DECIMAL(6,2),
  TotalCost DECIMAL(8,2)
);
```

Вариант 12. Таблица «Ресторанные заказы»

```
CREATE TABLE RestaurantOrders (
  OrderID INT,
  TableNumber INT,
  WaiterID INT,
  WaiterName VARCHAR(100),
  CustomerID INT,
  CustomerName VARCHAR(100),
  DishID INT,
  DishName VARCHAR(100),
  DishCategory VARCHAR(50),
  DishPrice DECIMAL(6,2),
  Quantity INT,
  OrderTime DATETIME,
  TotalAmount DECIMAL(8,2)
);
```

Вариант 13. Таблица «Фитнес-центр»

```
CREATE TABLE FitnessCenterData (
  MemberID INT,
  MemberName VARCHAR(100),
  MembershipType VARCHAR(30),
  TrainerID INT,
  TrainerName VARCHAR(100),
  TrainerSpecialization VARCHAR(50),
  ClassID INT,
  ClassName VARCHAR(50),
  ClassSchedule DATETIME,
  AttendanceDate DATE,
  PaymentID INT,
  PaymentAmount DECIMAL(6,2),
  PaymentDate DATE
);
```

Вариант 14. Таблица «IT-поддержка»

```
CREATE TABLE ITSupportData (
  TicketID INT,
  EmployeeID INT,
  EmployeeName VARCHAR(100),
  EmployeeDepartment VARCHAR(50),
  IssueType VARCHAR(50),
  IssueDescription VARCHAR(500),
  TechnicianID INT,
  TechnicianName VARCHAR(100),
  TechnicianSpecialization VARCHAR(50),
  ResolutionDate DATE,
  ResolutionStatus VARCHAR(30),
  ResolutionNotes VARCHAR(1000)
        );
```

Лабораторная работа №12 «Создание проекта БД. Создание БД. Редактирование и модификация таблиц»

Теоретические сведения

На предыдущих этапах проектирования базы данных были выполнены:

- анализ предметной области;
- построение концептуальной и логической моделей;
- нормализация отношений.

Данная лабораторная работа является завершающим этапом – физической реализацией проекта базы данных в конкретной СУБД. Основное внимание уделяется практическому созданию объектов БД, а также операциям по изменению их структуры.

Основные SQL-операторы для работы с объектами БД

1. Операторы определения данных (DDL – Data Definition Language)
   - CREATE DATABASE – создание базы данных.
   - CREATE TABLE – создание таблицы.
   - ALTER TABLE – изменение структуры существующей таблицы.
   - DROP TABLE – удаление таблицы.
   - ADD CONSTRAINT – добавление ограничений целостности (первичные, внешние ключи, проверки).
2. Операторы манипуляции данными (DML – Data Manipulation Language)
   - INSERT – вставка новых записей.
   - UPDATE – обновление существующих записей.
   - DELETE – удаление записей.
3. Важные аспекты модификации таблиц

При изменении структуры таблиц необходимо учитывать:

– Целостность данных: при добавлении/удалении столбцов, изменении типов данных или ограничений могут возникнуть конфликты с существующими данными.
– Зависимости объектов: таблицы могут быть связаны внешними ключами, иметь индексы, триггеры, что может осложнить модификацию.
– Согласованность: изменения должны соответствовать логической модели данных.

В данной работе используется графический интерфейс SQL Server Management Studio (SSMS) и прямое выполнение SQL-скриптов.

Описание предметной области для выполнения лабораторной работы

В качестве предметной области используется ваш индивидуальный вариант из лабораторной работы №5.

Вы будете реализовывать в СУБД тот проект базы данных, который был спроектирован ранее.

Пример выполнения лабораторной работы

Шаг 1. Создание базы данных и таблиц.

1. Создание новой базы данных.
   – Откройте SSMS и подключитесь к вашему экземпляру SQL Server.
   – Создайте новую базу данных. Используйте графический интерфейс или выполните SQL-скрипт (рис. 57).

```
CREATE DATABASE CollegeLibrary;
GO

USE CollegeLibrary;
GO
```

Рис. 57 Пример SQL-скрипта для создания новой БД

2. Создание таблиц согласно логической модели.
   - Для каждой сущности создайте таблицу, указав все атрибуты с соответствующими типами данных, ограничениями NOT NULL и UNIQUE, где необходимо.
   - Определите первичные ключи.
   - Создайте все таблицы вашего проекта (рис. 58).

```
CREATE TABLE Book (
    BookID INT PRIMARY KEY IDENTITY(1,1),
    ISBN VARCHAR(17) UNIQUE NOT NULL,
    Title NVARCHAR(200) NOT NULL,
    PublicationYear INT,
    Publisher NVARCHAR(100)
);
GO
```

Рис. 58 Пример для сущности «Книга»

3. Установка связей между таблицами (внешние ключи).
   - Для каждой связи «один-ко-многим» добавьте внешний ключ в дочерней таблице (рис. 59).
   - Для связей «многие-ко-многим» убедитесь, что создана ассоциативная таблица и на её поля установлены внешние ключи.

```
ALTER TABLE BookCopy
ADD CONSTRAINT FK_BookCopy_Book
FOREIGN KEY (BookID) REFERENCES Book(BookID);
GO
```

Рис. 59 Пример для связи «Книга – Экземпляр»

4. Заполнение таблиц тестовыми данными.
   - Вставьте в каждую таблицу не менее 10 осмысленных записей, используя оператор INSERT (рис. 60).

– Убедитесь, что данные согласованы (например, внешние ключи ссылаются на существующие записи).

```
INSERT INTO Book (ISBN, Title, PublicationYear, Publisher)
VALUES
    ('978-5-4461-1234-5', N'Программирование на Python', 2023, N'Питер'),
    ('978-5-94836-567-8', N'Базы данных для начинающих', 2022, N'Эксмо');
GO
```

Рис. 60 Пример заполнения таблицы «Книга»

5. Проверка создания объектов и целостности данных.
   – В обозревателе объектов SSMS убедитесь, что все таблицы созданы.
   – Выполните несколько SELECT-запросов с соединениями, чтобы проверить связи.
   – Попробуйте вставить некорректные данные (например, дубликат первичного ключа, несуществующий внешний ключ) и убедитесь, что ограничения работают.

Шаг 2. Редактирование и модификация таблиц.

Предположим, что в ходе эксплуатации базы данных возникла необходимость внести следующие изменения в структуру:

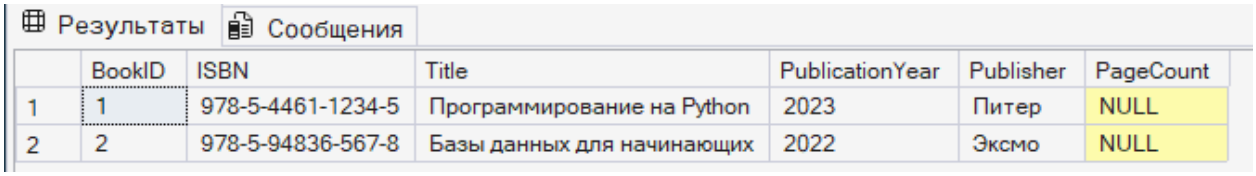
1. Добавление нового столбца в существующую таблицу.

Задание: в таблицу Book добавьте столбец PageCount (количество страниц) типа INT, разрешающий NULL.

SQL-команда:

```
ALTER TABLE Book
ADD PageCount INT NULL;
```

Проверка: выполните запрос SELECT * FROM Book; и убедитесь, что столбец появился (рис. 61).



Результаты | Сообщения

	BookID	ISBN	Title	PublicationYear	Publisher	PageCount
1	1	978-5-4461-1234-5	Программирование на Python	2023	Питер	NULL
2	2	978-5-94836-567-8	Базы данных для начинающих	2022	Эксмо	NULL

Рис. 61 Проверка добавления нового столбца

2. Изменение типа данных или размера столбца.

Задание: увеличьте максимальную длину столбца Publisher в таблице Book с NVARCHAR(100) до NVARCHAR(150).

SQL-команда:

```
ALTER TABLE Book
ALTER COLUMN Publisher NVARCHAR(150);
```

Примечание: если в столбце уже есть данные, убедитесь, что новое ограничение длины их не нарушает.

3. Добавление ограничения CHECK на столбец.

Задание: добавьте ограничение, что PublicationYear в таблице Book не может быть раньше 1900.

SQL-команда:

```
ALTER TABLE Book
ADD CONSTRAINT CHK_Book_PublicationYear
CHECK (PublicationYear >= 1900);
```

Проверка: попробуйте вставить или обновить запись с годом менее 1900. Должна возникнуть ошибка.

4. Добавление столбца с вычисляемым значением.

Задание: в таблицу BookCopy добавьте вычисляемый столбец Age, который показывает, сколько лет экземпляру (разница между текущим годом

и годом издания книги). Предполагается, что в BookCopy есть внешний ключ BookID, а год издания хранится в таблице Book.

SQL-команда:

```
ALTER TABLE BookCopy
ADD Age AS (YEAR(GETDATE()) -
    (SELECT PublicationYear FROM Book
     WHERE Book.BookID = BookCopy.BookID));
```

5. Удаление столбца из таблицы.

Задание: удалите ранее добавленный столбец PageCount из таблицы Book.

SQL-команда:

```
ALTER TABLE Book
DROP COLUMN PageCount;
```

Внимание: убедитесь, что на столбец не ссылаются другие объекты (индексы, ограничения, вычисляемые столбцы).

6. Переименование таблицы или столбца.

Задание: переименуйте таблицу Book в Publication.

SQL-команда:

```
EXEC sp_rename 'Book', 'Publication';
```

Примечание: старайтесь избегать переименований в рабочей БД, так как это может сломать существующие запросы, хранимые процедуры и приложения.

7. Создание индекса для повышения производительности.

Задание: создайте некластеризованный индекс по столбцу Title таблицы Book (или Publication).

SQL-команда:

```
CREATE INDEX IX_Book_Title ON Book(Title);
```

Последовательность выполнения лабораторной работы

По результатам работы необходимо предоставить:

1. Скрипт создания БД, содержащий:
   - Создание базы данных.
   - Создание всех таблиц с ключами.
   - Вставку тестовых данных.
2. Скрипт модификации БД, содержащий все операции из шага 2 представленного выше примера выполнения лабораторной работы.
3. Снимки экрана:
   - Обозреватель объектов SSMS с созданными таблицами.
   - Результаты нескольких проверочных запросов SELECT.
   - Окно выполнения скрипта с сообщениями об успешном выполнении команд.
4. Выводы:
   - Опишите, какие трудности возникли при создании и изменении таблиц.
   - Как обеспечивается целостность данных при выполнении операций ALTER TABLE?
   - В каких случаях изменение структуры существующей таблицы может быть опасным?

Лабораторная работа №13 «Задание ключей. Создание основных объектов БД»

Теоретические сведения

В предыдущей лабораторной работе было выполнено создание и модификация структуры таблиц. Данная лабораторная работа углубляет понимание работы с ключами и основными объектами базы данных, которые обеспечивают целостность, производительность и безопасность данных.

Ключи и их типы в реляционных базах данных.

1. Первичный ключ (Primary Key, PK).

Назначение: однозначно идентифицирует каждую запись в таблице.

Требования: уникальность, неизменность, ненулевое значение (NOT NULL).

Типы:

- Естественный ключ: использует существующие бизнес-данные (паспорт, ISBN).
- Суррогатный ключ: искусственно созданный идентификатор (обычно автоинкрементное число).
- Составной ключ: состоит из нескольких столбцов.

2. Внешний ключ (Foreign Key, FK)

Назначение: обеспечивает ссылочную целостность между таблицами.

Типы связей:

- Один-ко-многим (1:M): самый распространенный тип.
- Один-к-одному (1:1): редко используется, обычно для разделения таблиц.
- Многие-ко-многим (M:M): реализуется через промежуточную таблицу.

3. Альтернативные ключи (Unique Constraints)

Назначение: гарантирует уникальность значений в столбце, но не является первичным ключом.

Отличие от PK: допускает одно значение NULL (в зависимости от СУБД).

Дополнительные объекты базы данных.

1. Индексы (Indexes)

Назначение: ускоряют поиск и сортировку данных.

Типы:

- Кластеризованный: определяет физический порядок данных (только один на таблицу).
- Некластеризованный: создает отдельную структуру для быстрого поиска.
- Составной: по нескольким столбцам.
- Уникальный: гарантирует уникальность индексируемых значений.
- Покрывающий (Covering): содержит все столбцы, необходимые для запроса.

2. Представления (Views)

Назначение: виртуальные таблицы, представляющие данные из одной или нескольких таблиц.

Преимущества:

- Безопасность (скрытие структуры данных).
- Упрощение сложных запросов.
- Абстракция данных.

Типы:

- Простое представление: на основе одного запроса.
- Индексированное представление: материализованное, с физическим хранением данных.

3. Ограничения (Constraints)
   - CHECK: Проверка значений по условию.
   - DEFAULT: Значение по умолчанию.
   - NOT NULL: Запрет пустых значений.

4. Последовательности (Sequences) и Identity

Назначение: автоматическая генерация уникальных числовых значений для суррогатных ключей.

Практические аспекты работы с ключами в SQL Server.

Создание первичного ключа:

```
-- Способ 1: При создании таблицы
CREATE TABLE TableName (
  ID INT PRIMARY KEY,
  ...
);

-- Способ 2: С отдельным ограничением
CREATE TABLE TableName (
  ID INT NOT NULL,
  ...
  CONSTRAINT PK_TableName PRIMARY KEY (ID)
);

-- Способ 3: Добавление к существующей таблице
ALTER TABLE TableName
ADD CONSTRAINT PK_TableName PRIMARY KEY (ID);
```

Создание внешнего ключа с правилами каскадирования:

```
ALTER TABLE ChildTable
ADD CONSTRAINT FK_Child_Parent
FOREIGN KEY (ParentID) REFERENCES ParentTable(ID)
ON DELETE CASCADE  -- Удаление дочерних записей при удалении родительской
ON UPDATE NO ACTION; -- Запрет обновления родительского ключа
```

Варианты каскадных операций:

– NO ACTION (по умолчанию) – запрет операции, если есть зависимые записи.

– CASCADE – каскадное выполнение той же операции.

– SET NULL – установка NULL в зависимых записях.

– SET DEFAULT – установка значения по умолчанию.

Пример выполнения лабораторной работы

В качестве примера будем использовать предметную область «Библиотека колледжа».

<u>База данных должна содержать следующие таблицы:</u>

1. Книга (Book) - информация о книжных изданиях.
2. Автор (Author) - информация об авторах.
3. Книга_Автор (BookAuthor) - ассоциативная таблица для связи M:M.
4. Экземпляр (BookCopy) - информация о физических экземплярах книг.
5. Читатель (Reader) - информация о читателях.
6. Выдача (Loan) - информация о выдаче книг читателям.

<u>Требования к ключам:</u>

1. Все таблицы должны иметь первичный ключ.
2. Для большинства таблиц использовать суррогатные ключи (IDENTITY).
3. Установить все необходимые внешние ключи.
4. Для поля Email в таблице Читатель создать уникальный ключ.
5. Создать составной уникальный ключ для ассоциативной таблицы.

Шаг 1. Создание базы данных.

```
-- Удаление БД, если существует (для учебных целей)
USE master;
GO

IF EXISTS (SELECT name FROM sys.databases WHERE name = 'CollegeLibrary')
  DROP DATABASE CollegeLibrary;
GO

-- Создание новой БД
CREATE DATABASE CollegeLibrary;
GO

USE CollegeLibrary;
```

Шаг 2. Создание таблиц с различными типами ключей.

1. Таблица Author (Автор) с суррогатным первичным ключом:

```
CREATE TABLE Author (
  AuthorID INT IDENTITY(1,1) PRIMARY KEY,
  FirstName NVARCHAR(50) NOT NULL,
  LastName NVARCHAR(50) NOT NULL,
  BirthDate DATE NULL,
  Country NVARCHAR(50) NULL
);
```

2. Таблица Book (Книга) с естественным ключом (ISBN) и суррогатным PK:

```
CREATE TABLE Book (
  BookID INT IDENTITY(1,1) PRIMARY KEY,
  ISBN VARCHAR(17) UNIQUE NOT NULL, -- Альтернативный уникальный ключ
  Title NVARCHAR(200) NOT NULL,
  PublicationYear INT NOT NULL CHECK (PublicationYear >= 1800 AND Publica-
tionYear <= YEAR(GETDATE())),
  Publisher NVARCHAR(100) NULL,
  PageCount INT NULL CHECK (PageCount > 0),
```

```
  Price DECIMAL(10,2) NULL DEFAULT 0.00
);
```

3. Таблица BookAuthor (ассоциативная) с составным первичным ключом:

```
CREATE TABLE BookAuthor (
  BookID INT NOT NULL,
  AuthorID INT NOT NULL,
  Role NVARCHAR(50) NULL, -- Например: "автор", "соавтор", "редактор"

  -- Составной первичный ключ
  CONSTRAINT PK_BookAuthor PRIMARY KEY (BookID, AuthorID),

  -- Внешние ключи
  CONSTRAINT FK_BookAuthor_Book FOREIGN KEY (BookID)
    REFERENCES Book(BookID) ON DELETE CASCADE,

  CONSTRAINT FK_BookAuthor_Author FOREIGN KEY (AuthorID)
    REFERENCES Author(AuthorID) ON DELETE CASCADE
);
```

4. Таблица Reader (Читатель) с уникальным ограничением на Email:

```
CREATE TABLE Reader (
  ReaderID INT IDENTITY(1,1) PRIMARY KEY,
  FirstName NVARCHAR(50) NOT NULL,
  LastName NVARCHAR(50) NOT NULL,
  Email NVARCHAR(100) UNIQUE NOT NULL, -- Уникальный ключ
  Phone VARCHAR(20) NULL,
  RegistrationDate DATE NOT NULL DEFAULT GETDATE(),
  ReaderType NVARCHAR(20) NOT NULL CHECK (ReaderType IN ('студент', 'пре-
подаватель', 'сотрудник')),
```

```
  GroupName NVARCHAR(50) NULL,

  -- Составной уникальный ключ (для студентов одной группы не может быть полных тезок)
  CONSTRAINT UQ_Reader_FullName_Group UNIQUE (FirstName, LastName, GroupName)
);
```

5. Таблица BookCopy (Экземпляр) с каскадными внешними ключами:

```
CREATE TABLE BookCopy (
  CopyID INT IDENTITY(1,1) PRIMARY KEY,
  BookID INT NOT NULL,
  InventoryNumber VARCHAR(50) UNIQUE NOT NULL,
  AcquisitionDate DATE NOT NULL DEFAULT GETDATE(),
  ShelfLocation NVARCHAR(50) NOT NULL,
  Status NVARCHAR(20) NOT NULL DEFAULT 'доступна'
    CHECK (Status IN ('доступна', 'выдана', 'ремонт', 'списана')),

  CONSTRAINT FK_BookCopy_Book FOREIGN KEY (BookID)
    REFERENCES Book(BookID) ON DELETE CASCADE
);
```

6. Таблица Loan (Выдача) с проверками дат:

```
CREATE TABLE Loan (
  LoanID INT IDENTITY(1,1) PRIMARY KEY,
  CopyID INT NOT NULL,
  ReaderID INT NOT NULL,
  LoanDate DATE NOT NULL DEFAULT GETDATE(),
  DueDate DATE NOT NULL,
  ReturnDate DATE NULL,
  FineAmount DECIMAL(10,2) NULL DEFAULT 0.00,
```

```
    -- Проверка, что дата возврата не раньше даты выдачи
    CONSTRAINT CHK_Loan_Dates CHECK (
      (ReturnDate IS NULL) OR
      (ReturnDate >= LoanDate)
    ),

    -- Проверка, что срок возврата не раньше даты выдачи
    CONSTRAINT CHK_Loan_DueDate CHECK (DueDate >= LoanDate),

    -- Внешние ключи
    CONSTRAINT FK_Loan_BookCopy FOREIGN KEY (CopyID)
      REFERENCES BookCopy(CopyID),

    CONSTRAINT FK_Loan_Reader FOREIGN KEY (ReaderID)
      REFERENCES Reader(ReaderID)
  );
```

Шаг 3. Создание различных типов индексов.

1. Некластеризованный индекс для часто используемых полей поиска:

```
-- Индекс для поиска книг по названию
CREATE NONCLUSTERED INDEX IX_Book_Title
ON Book(Title);
GO


-- Индекс для поиска читателей по фамилии и имени
CREATE NONCLUSTERED INDEX IX_Reader_Name
ON Reader(LastName, FirstName);
GO
```

```
-- Индекс для поиска авторов по фамилии
CREATE NONCLUSTERED INDEX IX_Author_LastName
ON Author(LastName);
GO
```

2. Составной индекс для частых запросов с несколькими условиями:

```
-- Индекс для поиска выдач по читателю и дате
CREATE NONCLUSTERED INDEX IX_Loan_Reader_Date
ON Loan(ReaderID, LoanDate DESC);
GO


-- Индекс для поиска экземпляров по книге и статусу
CREATE NONCLUSTERED INDEX IX_BookCopy_Book_Status
ON BookCopy(BookID, Status);
GO
```

3. Уникальный индекс (кроме PK):

```
-- Уже создан через UNIQUE CONSTRAINT, но можно и явно:
CREATE UNIQUE INDEX UIX_Reader_Email
ON Reader(Email);
```

4. Индекс с включенными столбцами (Covering Index):

```
-- Индекс для часто запрашиваемых данных о выдаче
CREATE NONCLUSTERED INDEX IX_Loan_Covering
ON Loan(CopyID, ReaderID)
INCLUDE (LoanDate, DueDate, ReturnDate);
```

Шаг 4. Создание различных типов представлений.

1. Простое представление для часто используемого запроса:

```
-- Представление доступных книг
```

```
CREATE VIEW v_AvailableBooks AS
SELECT
  b.BookID,
  b.ISBN,
  b.Title,
  b.PublicationYear,
  b.Publisher,
  COUNT(bc.CopyID) AS AvailableCopies
FROM Book b
LEFT JOIN BookCopy bc ON b.BookID = bc.BookID
  AND bc.Status = 'доступна'
GROUP BY b.BookID, b.ISBN, b.Title, b.PublicationYear, b.Publisher;
```

2. Представление для сложного объединения данных:

```
-- Подробная информация о выданных книгах
CREATE VIEW v_LoanDetails AS
SELECT
  l.LoanID,
  l.LoanDate,
  l.DueDate,
  l.ReturnDate,
  l.FineAmount,

  r.ReaderID,
  r.FirstName + ' ' + r.LastName AS ReaderName,
  r.Email,
  r.ReaderType,

  bc.CopyID,
  bc.InventoryNumber,
  bc.Status AS CopyStatus,
```

```
    b.BookID,
    b.Title AS BookTitle,
    b.ISBN,

    -- Подзапрос для получения списка авторов
    (SELECT STRING_AGG(a.FirstName + ' ' + a.LastName, ', ')
     FROM Author a
     INNER JOIN BookAuthor ba ON a.AuthorID = ba.AuthorID
     WHERE ba.BookID = b.BookID) AS Authors

FROM Loan l
INNER JOIN Reader r ON l.ReaderID = r.ReaderID
INNER JOIN BookCopy bc ON l.CopyID = bc.CopyID
INNER JOIN Book b ON bc.BookID = b.BookID;
```

3. Представление с вычисляемыми полями:

```
-- Статистика по читателям
CREATE VIEW v_ReaderStatistics AS
SELECT
    r.ReaderID,
    r.FirstName + ' ' + r.LastName AS ReaderName,
    r.ReaderType,
    r.RegistrationDate,

    -- Количество текущих выдач
    (SELECT COUNT(*)
     FROM Loan l
     WHERE l.ReaderID = r.ReaderID
       AND l.ReturnDate IS NULL) AS ActiveLoans,
```

```
 -- Общее количество выдач
 (SELECT COUNT(*)
  FROM Loan l
  WHERE l.ReaderID = r.ReaderID) AS TotalLoans,

 -- Сумма штрафов
 (SELECT ISNULL(SUM(l.FineAmount), 0)
  FROM Loan l
  WHERE l.ReaderID = r.ReaderID) AS TotalFines,

 -- Последняя дата выдачи
 (SELECT MAX(l.LoanDate)
  FROM Loan l
  WHERE l.ReaderID = r.ReaderID) AS LastLoanDate

FROM Reader r;
```

Шаг 5. Создание и использование Sequence.

1. Создание последовательности для специальных номеров:

```
CREATE SEQUENCE seq_InventoryNumber
 AS INT
 START WITH 1000
 INCREMENT BY 1
 MINVALUE 1000
 MAXVALUE 9999
 CYCLE; -- Начинать сначала после достижения максимума
```

2. Использование последовательности при вставке данных:

```
-- Таблица с использованием Sequence
```

```
CREATE TABLE SpecialCollection (
  ItemID INT PRIMARY KEY,
  InventoryNumber VARCHAR(10) NOT NULL DEFAULT
    'SC-' + FORMAT(NEXT VALUE FOR seq_InventoryNumber, '0000'),
  Title NVARCHAR(200) NOT NULL,
  Description NVARCHAR(MAX) NULL
);
```

Шаг 6. Вставка данных с соблюдением всех ограничений.

```
-- Вставка авторов
INSERT INTO Author (FirstName, LastName, BirthDate, Country)
VALUES
  (N'Лев', N'Толстой', '1828-09-09', N'Россия'),
  (N'Фёдор', N'Достоевский', '1821-11-11', N'Россия'),
  (N'Антон', N'Чехов', '1860-01-29', N'Россия'),
  (N'Джордж', N'Оруэлл', '1903-06-25', N'Великобритания'),
  (N'Эрих', N'Ремарк', '1898-06-22', N'Германия');
GO

-- Вставка книг
INSERT INTO Book (ISBN, Title, PublicationYear, Publisher, PageCount, Price)
VALUES
  ('978-5-389-12345-6', N'Война и мир', 1869, N'Азбука', 1225, 850.00),
  ('978-5-17-98765-4', N'Преступление и наказание', 1866, N'АСТ', 672, 450.00),
  ('978-5-04-112233-4', N'1984', 1949, N'Эксмо', 320, 350.00),
  ('978-5-699-55667-8', N'Три товарища', 1936, N'Эксмо', 480, 420.00),
  ('978-5-367-09876-5', N'Вишневый сад', 1904, N'Просвещение', 128, 280.00);
GO

-- Вставка связей книга-автор
INSERT INTO BookAuthor (BookID, AuthorID, Role)
VALUES
  (1, 1, 'автор'),
  (2, 2, 'автор'),
  (3, 4, 'автор'),
  (4, 5, 'автор'),
  (5, 3, 'автор');
GO

-- Вставка читателей
INSERT INTO Reader (FirstName, LastName, Email, Phone, ReaderType, GroupName)
VALUES
  (N'Иван', N'Иванов', 'ivanov@college.ru', '+7(912)345-67-89', 'студент', 'ИСП-21'),
  (N'Мария', N'Петрова', 'petrova@college.ru', '+7(923)456-78-90', 'студент', 'ИСП-21'),
  (N'Алексей', N'Сидоров', 'sidorov@college.ru', NULL, 'преподаватель', NULL),
  (N'Елена', N'Кузнецова', 'kuznetsova@college.ru', '+7(934)567-89-01', 'сотрудник', NULL),
```

```
  (N'Дмитрий', N'Новиков', 'novikov@college.ru', '+7(945)678-90-12', 'студент', 'ПР-22');
GO

-- Вставка экземпляров книг
INSERT INTO BookCopy (BookID, InventoryNumber, AcquisitionDate, ShelfLocation, Status)
VALUES
  (1, 'INV-001', '2023-01-15', 'Стеллаж А, полка 1', 'доступна'),
  (1, 'INV-002', '2023-01-15', 'Стеллаж А, полка 1', 'выдана'),
  (2, 'INV-003', '2023-02-20', 'Стеллаж Б, полка 2', 'доступна'),
  (3, 'INV-004', '2023-03-10', 'Стеллаж В, полка 3', 'ремонт'),
  (4, 'INV-005', '2023-03-15', 'Стеллаж Г, полка 4', 'доступна'),
  (5, 'INV-006', '2023-04-01', 'Стеллаж Д, полка 5', 'доступна');
GO

-- Вставка выдач
INSERT INTO Loan (CopyID, ReaderID, LoanDate, DueDate, ReturnDate, FineAmount)
VALUES
  (2, 1, '2024-01-10', '2024-01-31', NULL, 0.00),
  (3, 2, '2024-01-12', '2024-02-02', '2024-01-30', 0.00),
  (5, 3, '2024-01-15', '2024-02-15', NULL, 0.00),
  (1, 4, '2023-12-01', '2023-12-30', '2024-01-05', 50.00), -- Просрочка
  (6, 1, '2024-01-18', '2024-02-08', NULL, 0.00);
        GO
```

Шаг 7. Тестирование ограничений целостности.

1. Проверка уникальности первичного ключа:

```
-- Попытка вставить дубликат PK (должна вызвать ошибку)
BEGIN TRY
  INSERT INTO Reader (ReaderID, FirstName, LastName, Email, ReaderType)
  VALUES (1, N'Тест', N'Тестов', 'test@test.ru', 'студент');
  PRINT 'ОШИБКА: Дубликат PK был вставлен!';
END TRY
BEGIN CATCH
  PRINT 'УСПЕХ: Ошибка при вставке дубликата PK: ' + ERROR_MESSAGE();
END CATCH
```

2. Проверка внешнего ключа:

```
-- Попытка вставить несуществующий внешний ключ (должна вызвать ошибку)
BEGIN TRY
  INSERT INTO Loan (CopyID, ReaderID, LoanDate, DueDate)
  VALUES (999, 1, GETDATE(), DATEADD(day, 30, GETDATE()));
```

```
  PRINT 'ОШИБКА: Вставлен несуществующий FK!';
 END TRY
 BEGIN CATCH
  PRINT 'УСПЕХ: Ошибка при вставке несуществующего FK: ' + ER-
ROR_MESSAGE();
 END CATCH
```

3. Проверка ограничения CHECK:

```
-- Попытка вставить отрицательное количество страниц (должна вызвать ошибку)
BEGIN TRY
  INSERT INTO Book (ISBN, Title, PublicationYear, PageCount)
  VALUES ('978-5-11111-111-1', N'Тестовая книга', 2024, -100);
  PRINT 'ОШИБКА: Вставлено отрицательное значение PageCount!';
END TRY
BEGIN CATCH
  PRINT 'УСПЕХ: Ошибка при вставке отрицательного PageCount: ' + ER-
ROR_MESSAGE();
END CATCH
```

4. Проверка уникального ограничения (UNIQUE):

```
-- Попытка вставить дубликат email (должна вызвать ошибку)
BEGIN TRY
  INSERT INTO Reader (FirstName, LastName, Email, ReaderType)
  VALUES (N'Пётр', N'Иванов', 'ivanov@college.ru', 'студент');
  PRINT 'ОШИБКА: Дубликат email был вставлен!';
END TRY
BEGIN CATCH
  PRINT 'УСПЕХ: Ошибка при вставке дубликата email: ' + ERROR_MESSAGE();
END CATCH
```

5. Проверка каскадного удаления:

```
-- Проверка количества записей до удаления
SELECT 'До удаления книги:', COUNT(*) AS BookCount FROM Book;
SELECT 'До удаления экземпляров:', COUNT(*) AS CopyCount FROM BookCopy;


-- Удаление книги (должно каскадно удалить экземпляры)
DELETE FROM Book WHERE BookID = 3;
```

```
-- Проверка количества записей после удаления
SELECT 'После удаления книги:', COUNT(*) AS BookCount FROM Book;
SELECT 'После удаления экземпляров:', COUNT(*) AS CopyCount FROM BookCopy;
GO
```

Шаг 8. Просмотр метаданных базы данных.

1. Просмотр всех таблиц и их столбцов:

```
SELECT
  t.name AS TableName,
  c.name AS ColumnName,
  ty.name AS DataType,
  c.max_length AS MaxLength,
  c.is_nullable AS IsNullable,
  CASE WHEN pk.index_id IS NOT NULL THEN 'PK' ELSE '' END AS PrimaryKey
FROM sys.tables t
INNER JOIN sys.columns c ON t.object_id = c.object_id
INNER JOIN sys.types ty ON c.user_type_id = ty.user_type_id
LEFT JOIN (
  SELECT ic.object_id, ic.column_id, i.index_id
  FROM sys.indexes i
  INNER JOIN sys.index_columns ic ON i.object_id = ic.object_id AND i.index_id = ic.index_id
  WHERE i.is_primary_key = 1
) pk ON c.object_id = pk.object_id AND c.column_id = pk.column_id
WHERE t.name NOT LIKE 'sys%'
ORDER BY t.name, c.column_id;
```

2. Просмотр всех ограничений:

```
SELECT
  t.name AS TableName,
```

```
  c.name AS ConstraintName,
  c.type_desc AS ConstraintType,
  col.name AS ColumnName
FROM sys.tables t
INNER JOIN sys.objects c ON t.object_id = c.parent_object_id
LEFT JOIN sys.index_columns ic ON c.parent_object_id = ic.object_id
  AND ic.index_id = (SELECT index_id FROM sys.indexes WHERE object_id =
c.parent_object_id AND is_primary_key = 1)
LEFT JOIN sys.columns col ON ic.object_id = col.object_id AND ic.column_id =
col.column_id
WHERE c.type IN ('PK', 'FK', 'UQ', 'CK', 'D')
ORDER BY t.name, c.type_desc, c.name;
```

3. Просмотр всех индексов:

```
SELECT
  t.name AS TableName,
  i.name AS IndexName,
  i.type_desc AS IndexType,
  i.is_unique AS IsUnique,
  STRING_AGG(col.name, ', ') WITHIN GROUP (ORDER BY ic.key_ordinal) AS In-
dexedColumns
FROM sys.tables t
INNER JOIN sys.indexes i ON t.object_id = i.object_id
INNER JOIN sys.index_columns ic ON i.object_id = ic.object_id AND i.index_id =
ic.index_id
INNER JOIN sys.columns col ON ic.object_id = col.object_id AND ic.column_id =
col.column_id
WHERE i.type > 0 -- Исключить heap таблицы
  AND i.is_primary_key = 0 -- Исключить PK (они тоже индексы)
  AND i.name IS NOT NULL
GROUP BY t.name, i.name, i.type_desc, i.is_unique
ORDER BY t.name, i.name;
```

Последовательность выполнения лабораторной работы

Для выполнения лабораторной работы используйте предметную область согласно вашему индивидуальному варианту из предыдущей работы.

Задание 1: Создание дополнительных объектов.

1. Создайте составной индекс для таблиц.
2. Создайте представление.
3. Создайте уникальный индекс для таблиц.
4. Добавьте ограничение DEFAULT для полей в таблице.

Задание 2: Модификация существующих объектов.

1. Измените внешний ключ в таблице для поля, чтобы при удалении из таблицы записи устанавливали значение= NULL.
2. Добавьте новое ограничение CHECK в таблицу.
3. Создайте индекс для полнотекстового поиска по полю в таблице.

Задание 3: Тестирование производительности.

1. Сравните производительность запросов с использованием индексов и без них.
2. Проанализируйте план выполнения для сложного запроса с JOIN 3-х таблиц.
3. Создайте покрывающий индекс для часто выполняемого запроса.

Лабораторная работа №14 «Задание значений и ограничений поля. Проверка введенного в поле значения. Отображение данных числового типа и типа дата»

Теоретические сведения

На предыдущих этапах были изучены основы нормализации, создания таблиц, первичных и внешних ключей. Данная работа углубляет понимание доменной целостности (Domain Integrity), которая обеспечивает корректность данных в каждом отдельном столбце согласно его типу и бизнес-правилам.

Доменная целостность – это совокупность правил, определяющих допустимые значения для каждого отдельного столбца таблицы. Её обеспечение является фундаментом для качества данных и выполняется на трёх уровнях:

1. Декларативный уровень: через встроенные механизмы СУБД (типы данных, ограничения).
2. Процедурный уровень: через триггеры и хранимые процедуры (более сложная логика).
3. Уровень приложения: через проверки в клиентском коде (дополнительная защита).

Данная лабораторная работа фокусируется на декларативном уровне как наиболее эффективном и производительном.

Система типов данных SQL Server как первичное ограничение.

Выбор типа данных – это первое и главное ограничение, накладываемое на данные. SQL Server предоставляет богатую систему типов, которые можно классифицировать для целей проектирования (табл. 66).

Таблица 66 – Основные типы данных для обеспечения целостности

Категория	**Тип данных**	**Описание / Назначение**	**Пример значения**	**Размер / Особенности**
Точные числовые	INT	Целые числа. Оптимальный выбор для суррогатных ключей и счетчиков.	-2147483648 до 2147483647	4 байта
	DECIMAL(p, s) / NUMERIC(p,s)	Числа с фиксированной точностью и масштабом. Для финансовых операций, оценок.	12345.67	p (точность) до 38, s (масштаб) ≤ p
Приблизительные числовые	FLOAT(n)	Числа с плавающей точкой. Для научных расчетов, где важна широта диапазона, а не абсолютная точность.	1.79E+308	Зависит от n
Символьные	VARCHAR(n) / NVARCHAR(n)	Строки переменной длины. NVARCHAR хранит Unicode. Для имен, описаний, текстов.	'Иванов'	n – макс. число символов (до 8000/4000). Экономит место на диске.
	CHAR(n) / NCHAR(n)	Строки фиксированной длины. Для кодов с постоянной длиной (ISBN, серия паспорта).	'AB-1234'	Занимает полный размер n, даже если строка короче. Быстрее для поиска.
Дата и время	DATE	Только дата (без времени). Для даты рождения, события.	'2024-01-19'	3 байта
	DATETIME2	Дата и время с высокой точностью. Предпо-	'2024-01-19 14:30:25.1234567'	Точность до 100 наносекунд

Категория	Тип данных	Описание / Назначение	Пример значения	Размер / Особенности
		чтительнее устаревшего DATETIME.		
Логические	BIT	Логическое значение. Для флагов (Да/Нет, Включен/Выключен).	0 (FALSE), 1 (TRUE)	Может занимать 1 бит в составе группы
Двоичные	BINARY(n) / VARBINARY(n)	Последовательности байтов. Для хранения хэшей, небольших файлов, шифрованных данных.	0x1A2B3C	Аналогично CHAR/VARCHAR

Практическое правило выбора типа:

Для ключей: INT IDENTITY (суррогатный) или VARCHAR/CHAR для естественных кодов.

Для денежных сумм: DECIMAL(19,4) (стандарт для финансовых систем) или MONEY.

Для дат: всегда используйте DATE для чистых дат и DATETIME2 вместо DATETIME.

Для текста: NVARCHAR для поддержки любого языка, VARCHAR – только для данных в одной кодировке.

Ограничения (CONSTRAINTS) для расширенной проверки данных.

Тип данных задаёт общие рамки, но бизнес-правила требуют более точных ограничений. Они создаются командой CONSTRAINT (табл. 67).

Таблица 67 – Виды ограничений и их применение

Ограничение	**Синтаксис (пример)**	**Цель**	**Практический пример из предметной области "Колледж"**
NOT NULL	LastName NVARCHAR(50) NOT NULL	Запрещает наличие NULL в столбце. Фактически преобразует опциональный атрибут в обязательный.	ФИО студента, Название дисциплины.
UNIQUE	Email NVARCHAR(100) UNIQUE	Гарантирует уникальность всех значений в столбце (кроме NULL, если он разрешён). Может быть несколько на таблицу.	Email преподавателя, Номер зачетной книжки.
CHECK	CONSTRAINT CHK_Grade CHECK (Grade BETWEEN 2 AND 5)	Проверяет значение по логическому выражению. Основной инструмент для реализации сложных бизнес-правил на уровне БД.	Оценка (2-5), Дата сдачи >= Дата начала семестра, Количество часов > 0.
DEFAULT	Status NVARCHAR(20) DEFAULT 'активен'	Задаёт значение, которое подставляется при вставке, если для столбца значение не указано. Ускоряет ввод и обеспечивает согласованность.	Статус заявки (по умолчанию "новая"), Дата создания записи (по умолчанию GETDATE()).
PRIMARY KEY	StudentID INT PRIMARY KEY / CONSTRAINT PK_Order PRIMARY KEY (OrderID, ProductID)	Комбинация UNIQUE + NOT NULL. Обеспечивает уникальную идентификацию строки. Может быть составным.	ID студента, составной ключ (ID студента, ID дисциплины, Дата) для таблицы посещаемости.
FOREIGN KEY	FOREIGN KEY (GroupID) REFERENCES Groups(ID) ON DELETE CASCADE	Обеспечивает ссылочную целостность. ON DELETE/UPDATE правила критичны для поддержания целостности базы.	ГруппаID в таблице Студенты. ON DELETE CASCADE – удаление группы удаляет всех её студентов. ON DELETE SET NULL – студенты остаются, но поле GroupID обнуляется.

Приоритет и каскадные операции для FOREIGN KEY:

Правила ON DELETE и ON UPDATE определяют реакцию на изменение в родительской таблице. Их выбор – стратегическое решение.

- NO ACTION (по умолчанию): запрещает операцию, если существуют зависимые записи. Самый безопасный, требует явного управления.
- CASCADE: повторяет операцию для всех зависимых записей. Опасен, но может быть полезен для жестко связанных данных (например, удаление заказа → удаление всех его позиций).
- SET NULL / SET DEFAULT: устанавливает в зависимом столбце NULL или значение по умолчанию. Требует, чтобы столбец допускал NULL или имел DEFAULT. Полезен для «мягких» связей.

Проверка введенных значений и обработка ошибок.

Когда ограничение нарушено, SQL Server генерирует ошибку и откатывает всю транзакцию (если не используется обработка исключений). Понимание структуры ошибки важно для отладки.

Коды распространенных ошибок целостности:

- 547 – Конфликт ограничения FOREIGN KEY или CHECK.
- 2627 – Нарушение уникальности (PRIMARY KEY, UNIQUE).
- 515 – Попытка вставить NULL в поле с NOT NULL.

Конструкция TRY...CATCH: позволяет перехватить ошибку на стороне сервера и выполнить альтернативные действия (логирование, повторная попытка, возврат пользователю понятного сообщения), вместо полного прекращения работы скрипта.

Отображение данных: форматирование чисел и дат.

База данных хранит данные во внутреннем формате. Для презентационного слоя (отчеты, интерфейсы) требуется преобразование. SQL

Server предлагает два основных подхода: функция CONVERT() и FORMAT() (табл. 68).

Таблица 68 – Сравнение функций для форматирования

Задача	Функция CONVERT()	Функция FORMAT()	Рекомендация
Суть	Преобразование типа данных с указанием стиля.	Форматирование значения с использованием строка формата .NET.	
Пример даты	CONVERT(VARCHAR, GETDATE(), 104) -> '19.01.2024'	FORMAT(GETDATE(), 'dd.MM.yyyy', 'ru-RU') -> '19.01.2024'	Используйте CONVERT для производительности. FORMAT в 30-50 раз медленнее, но гораздо гибче.
Пример числа	CONVERT(VARCHAR, 1234567.89, 1) -> '1,234,567.89'	FORMAT(1234567.89, 'N2', 'ru-RU') -> '1 234 567,89'	Для простых стилей (даты) – CONVERT. Для сложного форматирования (пробелы разрядов, валюта) – FORMAT.
Производительность	Высокая. Встроенная, низкоуровневая функция.	Низкая. Выполняется в среде CLR (.NET).	Не используйте FORMAT в запросах к большим наборам данных или в часто вызываемых процедурах.
Гибкость	Ограничена предопределёнными стилями (около 30 для дат).	Очень высокая. Любой формат .NET, поддержка локалей.	Идеально для итоговых отчетов, где важен внешний вид.
Локализация	Ограниченная (стили для разных регионов).	Полная. Второй параметр – код культуры ('ru-RU', 'en-US').	Для мультиязычных приложений FORMAT незаменим.

Дополнительная функция CAST(): более стандартизирована (входит в ANSI SQL), но не поддерживает стили форматирования. Используется в основном для преобразования типов (CAST(Price AS INT)).

Практические рекомендации и выводы по проектированию.

1. Принцип «от общего к частному»: сначала назначайте максимально общий, но подходящий тип данных (например, NVARCHAR(255)), затем уточняйте ограничения (CHECK, UNIQUE).
2. Ограничения vs. Триггеры: всегда отдавайте предпочтение ограничениям (CHECK, FOREIGN KEY) там, где это возможно. Они декларативны, надёжнее и эффективнее триггеров.
3. Место проверки: основная проверка – в БД (доменная и ссылочная целостность). Проверка в приложении – для удобства пользователя и разгрузки сервера, но не заменяет проверки в БД.
4. Форматирование – задача презентации: Стремитесь хранить данные в «чистом» виде (DATE, DECIMAL), а форматировать на уровне выборки или, ещё лучше, в клиентском приложении. Это сохраняет гибкость.

Примеры заданий для лабораторной работы

Задание 1. Создание таблицы с комплексными ограничениями.

На основе схемы БД «Библиотека» создадим таблицу с использованием различных ограничений CHECK и DEFAULT:

```
-- Пример для предметной области "Библиотека"
CREATE TABLE Reader (
  Reader_ID INT IDENTITY(1,1) PRIMARY KEY,
  LastName NVARCHAR(50) NOT NULL,
  FirstName NVARCHAR(50) NOT NULL,
  Email NVARCHAR(100) UNIQUE,
  RegistrationDate DATE NOT NULL DEFAULT GETDATE(), -- Дата по умолчанию - те-
кущая
  MembershipType NVARCHAR(20) NOT NULL DEFAULT 'Standard' CHECK (Member-
shipType IN ('Standard', 'Premium', 'Student')),
  BirthDate DATE,
  -- Проверка на разумную дату рождения (человеку не меньше 12 и не больше 120 лет)
```

```
  CONSTRAINT CHK_Reader_BirthDate CHECK (BirthDate <= DATEADD(YEAR, -12,
GETDATE()) AND BirthDate >= DATEADD(YEAR, -120, GETDATE()))
);

CREATE TABLE Book_Loan (
  Loan_ID INT IDENTITY(1,1) PRIMARY KEY,
  Book_ID INT NOT NULL,
  Reader_ID INT NOT NULL,
  Loan_Date DATE NOT NULL DEFAULT GETDATE(),
  Planned_Return_Date DATE NOT NULL,
  Actual_Return_Date DATE NULL,
  -- Проверка 1: Планируемая дата возврата не может быть раньше даты выдачи
  -- Проверка 2: Фактическая дата возврата, если указана, не может быть раньше даты вы-
дачи
  CONSTRAINT CHK_Loan_Dates CHECK (
    Planned_Return_Date >= Loan_Date AND
    (Actual_Return_Date IS NULL OR Actual_Return_Date >= Loan_Date)
  ),
  FOREIGN KEY (Book_ID) REFERENCES Book(Book_ID),
  FOREIGN KEY (Reader_ID) REFERENCES Reader(Reader_ID)
        );
```

Что проверить: попробуйте вставить данные, нарушающие условия CHECK (например, MembershipType 'Invalid' или Planned_Return_Date в прошлом). Проанализируйте сообщения об ошибках от SQL Server.

Задание 2. Проверка вводимых данных и обработка ошибок.

Напишем скрипт, который пытается вставить или обновить данные в созданных таблицах, и обрабатывает возможные ошибки целостности с помощью конструкции TRY...CATCH:

```
BEGIN TRY
  -- Попытка вставить запись с нарушением CHECK ограничения
  INSERT INTO Reader (LastName, FirstName, MembershipType, BirthDate)
  VALUES ('Иванов', 'Иван', 'InvalidType', '2010-01-01');
  PRINT 'Данные успешно добавлены.';
END TRY
BEGIN CATCH
  PRINT 'Ошибка при вставке данных:';
  PRINT 'Сообщение: ' + ERROR_MESSAGE();
  PRINT 'Код ошибки: ' + CAST(ERROR_NUMBER() AS VARCHAR(10));
END CATCH;
```

Задание 3. Отображение данных числового типа и типа дата.

Напишем запросы, которые форматируют вывод числовых и датовых полей из наших таблиц.

1. Форматирование даты:

```
-- Различные форматы отображения даты выдачи книги
SELECT
  Loan_ID,
  Loan_Date AS [Дата выдачи (стандартно)],
  CONVERT(VARCHAR, Loan_Date, 104) AS [Дата выдачи (DD.MM.YYYY)], -- Немецкий формат
  FORMAT(Loan_Date, 'dd.MM.yyyy', 'ru-RU') AS [Дата выдачи (FORMAT)],
  FORMAT(Loan_Date, 'yyyy-MM-dd') AS [Дата выдачи (ISO)]
FROM Book_Loan;
```

2. Форматирование чисел (например, для отображения штрафа):

```
-- Предположим, в таблице Book_Loan есть столбец FineAmount DECIMAL(10,2)
SELECT
  Loan_ID,
  FineAmount AS [Штраф (число)],
  CAST(FineAmount AS DECIMAL(10,2)) AS [Штраф (2 знака)],
  FORMAT(FineAmount, 'N2', 'ru-RU') AS [Штраф (формат с пробелом)],
  '₽' + FORMAT(FineAmount, 'N2', 'ru-RU') AS [Штраф с валютой]
FROM Book_Loan
WHERE FineAmount > 0;
```

Последовательность выполнения лабораторной работы

1. Анализ предметной области: определите, какие ограничения (CHECK, DEFAULT) логично добавить к вашим сущностям.

2. Модификация схемы БД: используя знания из лабораторных работ №12 и №13, создайте или измените 2-3 таблицы, добавив в них не менее 3-х различных ограничений CHECK и DEFAULT.

3. Написание и выполнение скриптов: создайте скрипты из заданий 2 и 3. Вставьте как корректные, так и некорректные данные, чтобы проверить работу ограничений.

4. Форматирование вывода: напишите не менее 3-х запросов, которые по-разному форматируют даты и числа из ваших таблиц.

5. Вывод по работе должен содержать ответы на вопросы: Какие типы ограничений наиболее эффективны для предотвращения некорректного ввода? В чем практическая польза от использования CHECK и DEFAULT? Когда удобнее использовать CONVERT, а когда FORMAT для вывода дат?

Лабораторная работа №15 «Редактирование, добавление и удаление записей в таблице. Применение логических условий к записям. Открытие, редактирование и пополнение табличного файла»

Теоретические сведения

В SQL Server данные представляются как множества, а не как последовательности записей. Это определяет декларативный характер операций DML – мы описываем что нужно изменить, а не как это сделать.

Три уровня обработки DML-операций:

1. Логический уровень – оптимизатор определяет оптимальный план выполнения
2. Физический уровень – движок хранилища выполняет физические изменения
3. Уровень журналирования – все изменения фиксируются в журнале транзакций

Детальная анатомия операции INSERT.

1. Механизм выполнения INSERT:
   - Проверка ограничений (CHECK, FOREIGN KEY, UNIQUE).
   - Резервирование места в страницах данных.
   - Запись в журнал транзакций.
   - Фактическая вставка данных.
   - Обновление индексов.

Таблица 69 – Сравнение методов INSERT

Метод	Синтаксис	Преимущества	Недостатки	Производительность	Использование памяти
Простой INSERT	INSERT ... VALUES (...)	Простота, читаемость	Одна строка за раз	Низкая	Минимальная
Множественный INSERT	INSERT ... VALUES (...), (...)	Эффективность	Ограничение 1000 строк	Высокая	Умеренная
INSERT-SELECT	INSERT ... SELECT ...	Гибкость, преобразование	Сложность запроса	Зависит от SELECT	Зависит от объема
BULK INSERT	BULK INSERT ...	Максимальная скорость	Требует файл	Очень высокая	Большая
SELECT INTO	SELECT ... INTO ...	Создание+вставка	Только новая таблица	Высокая	Большая

2. Внутренние структуры данных при INSERT (рис. 62):

```
-- Демонстрация использования скрытых столбцов
SELECT
  %%lockres%% AS [Блокировка ресурса],
  %%physloc%% AS [Физическое расположение],
  sys.fn_PhysLocFormatter(%%physloc%%) AS [Формат расположения]
FROM Book
WHERE BookID = 1;
```

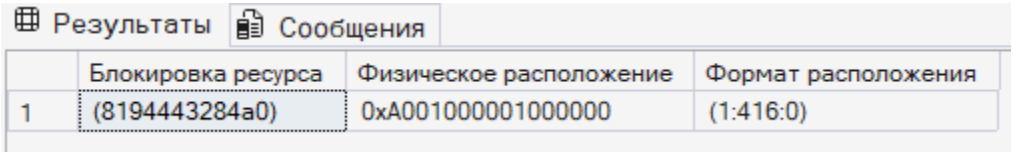


Рис. 63 Отображение результата

Критические аспекты:

- HEAP vs. Clustered Table: вставка в кучу (HEAP) быстрее, но поиск медленнее.
- FILLFACTOR: определяет заполненность страниц (по умолчанию 100%).
- PAGE SPLIT: при вставке в середину кластеризованного индекса происходит разделение страницы.

Расширенный анализ операции UPDATE.

Два режима выполнения UPDATE:

1. In-place update (обновление на месте):
   - Используется, когда не меняется размер строки.
   - Не меняется физическое расположение.
   - Минимальные блокировки.
2. Not-in-place update (перемещающее обновление):
   - Требуется при изменении размера данных.
   - Старая версия помечается как «призрак» (ghost).
   - Новая версия создается в другом месте.

```
-- запрос для мониторинга UPDATE операций
SELECT
  r.session_id,
  r.command,
  t.text AS [SQL Query],
```

```
    r.reads,
    r.writes,
    r.cpu_time,
    r.status,
    r.start_time,
    r.total_elapsed_time AS [Elapsed MS],
    r.blocking_session_id AS [Blocked By],
    r.wait_type,
    r.wait_time AS [Wait Time MS]
FROM sys.dm_exec_requests r
CROSS APPLY sys.dm_exec_sql_text(r.sql_handle) t
WHERE r.command LIKE '%UPDATE%'
  OR r.command LIKE '%INSERT%'
  OR r.command LIKE '%DELETE%'
  OR t.text LIKE '%UPDATE%'
ORDER BY r.cpu_time DESC;
```

Таблица 70 – Таблица оптимизаций UPDATE

Оптимизация	**Принцип работы**	**Когда применяется**	**Эффект**
Halloween Protection	Защита от повторного чтения обновленных строк	При обновлении с ORDER BY	Предотвращает бесконечные циклы
Split/Sort/Collapse	Разделение сложного UPDATE	При обновлении через представления	Упрощение плана выполнения
Predicate Pushdown	Раннее применение условий	В запросах с JOIN	Уменьшение обрабатываемых строк
Index Maintenance	Отложенное обновление индексов	При массовых обновлениях	Улучшение производительности

Операция DELETE.

Поэтапный процесс DELETE:

- Поиск строк для удаления (использует индексы).
- Проверка внешних ключей (если есть зависимые таблицы).
- Запись в журнал транзакций.
- Пометка строк как «призраков» (ghost records).
- Очистка «призраков» фоновым процессом ghost cleanup.
- Освобождение страниц (если они полностью пусты).

Таблица 71 – Детальное сравнение DELETE и TRUNCATE

Критерий	**DELETE**	**TRUNCATE**	**Примечания**
Синтаксис	DELETE FROM table WHERE condition	TRUNCATE TABLE table	TRUNCATE не поддерживает WHERE
Журналирование	Полное журналирование каждой строки	Минимальное (только освобождение страниц)	TRUNCATE значительно быстрее
Триггеры	Вызываются для каждой строки	Не вызываются	Важно для аудита
Идентификаторы	Не сбрасывает IDENTITY	Сбрасывает счетчик IDENTITY	TRUNCATE эквивалентен DROP+CREATE
Ограничения	Проверяет FOREIGN KEY	Требует отключения FOREIGN KEY или CASCADE	TRUNCATE требует больших привилегий
Блокировки	Строковые/страничные блокировки	Блокировка таблицы на уровне схема	TRUNCATE блокирует всю таблицу
Производительность	Медленно для больших объемов	Очень быстро	Разница в 1000+ раз
Откат	Полностью откаты-	Можно откатить	TRUNCATE в тран-

Критерий	DELETE	TRUNCATE	Примечания
	вается	только до начала операции	закции откатывается

Оптимизация массовых удалений:

```
-- Стратегия чанкирования для удаления старых возвращенных книг
DECLARE @BatchSize INT = 1000; -- Размер пакета (можно уменьшить для тестирования)
DECLARE @RowsAffected INT = 1;
DECLARE @TotalDeleted INT = 0;
DECLARE @StartTime DATETIME = GETDATE();

PRINT 'Начало чанкированного удаления...';
PRINT 'Размер пакета: ' + CAST(@BatchSize AS VARCHAR) + ' строк';

WHILE @RowsAffected > 0
BEGIN
  BEGIN TRANSACTION;

  -- Удаляем старые возвращенные книги (старше 5 лет)
  DELETE TOP (@BatchSize) FROM Loans
  WHERE Status = 'Возвращена'
   AND ReturnDate < DATEADD(YEAR, -5, GETDATE());

  SET @RowsAffected = @@ROWCOUNT;
  SET @TotalDeleted = @TotalDeleted + @RowsAffected;

  COMMIT TRANSACTION;

  -- Выводим прогресс
  IF @RowsAffected > 0
  BEGIN
    PRINT 'Удалено пакетов: ' + CAST(@RowsAffected AS VARCHAR)
      + ', всего: ' + CAST(@TotalDeleted AS VARCHAR);

    -- Пауза для уменьшения нагрузки (можно настроить или убрать)
    -- WAITFOR DELAY '00:00:00.100'; -- 100 миллисекунд
  END
END

PRINT 'Чанкированное удаление завершено.';
PRINT 'Всего удалено строк: ' + CAST(@TotalDeleted AS VARCHAR);
        PRINT  'Время  выполнения:  '  +  CAST(DATEDIFF(SECOND,  @StartTime,
GETDATE()) AS VARCHAR) + ' сек.';
```

Уровни изоляции транзакций.

В табл. 72 представлены проблемы параллельного доступа.

Таблица 71 – Проблемы параллельного доступа

Проблема	**Описание**	**Пример**	**Решение в SQL Server**
Lost Update (Потерянное обновление)	Два обновления, второе перезаписывает первое	Два кассира обновляют баланс	Блокировки, уровни изоляции
Dirty Read (Грязное чтение)	Чтение незафиксированных данных	Чтение данных из откаченной транзакции	READ COMMITTED или выше
Non-repeatable Read (Неповторяемое чтение)	Разные значения при повторном чтении	Баланс изменился между SELECT	REPEATABLE READ или выше
Phantom Read (Фантомное чтение)	Появление новых строк между чтениями	Новые заказы появились между считываниями	SERIALIZABLE
Snapshot Isolation Violations (Нарушения изоляции снапшотов)	Конфликты обновления в снапшотах	Два обновления одной строки	Механизм управления версиями

Уровни изоляции в SQL Server (табл. 72):

```
-- Каждая команда устанавливает свой уровень изоляции отдельно

-- Уровень 1: READ UNCOMMITTED (разрешает грязные чтения)
SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;
PRINT 'Установлен уровень изоляции: READ UNCOMMITTED';

-- Уровень 2: READ COMMITTED (по умолчанию, предотвращает грязные чтения)
SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
PRINT 'Установлен уровень изоляции: READ COMMITTED';

-- Уровень 3: REPEATABLE READ (предотвращает неповторяемые чтения)
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
PRINT 'Установлен уровень изоляции: REPEATABLE READ';

-- Уровень 4: SERIALIZABLE (полная изоляция, предотвращает фантомные чтения)
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
PRINT 'Установлен уровень изоляции: SERIALIZABLE';
```

```
-- Уровень 5: SNAPSHOT (управление версиями строк)
-- Требует предварительной настройки базы данных
-- ALTER DATABASE LibraryLab15 SET ALLOW_SNAPSHOT_ISOLATION ON;
SET TRANSACTION ISOLATION LEVEL SNAPSHOT;
PRINT 'Установлен уровень изоляции: SNAPSHOT';
```

Таблица 72 – Сравнение уровней изоляции

Уровень	Dirty Reads	Non-repeatable Reads	Phantoms	Блокировки	Управление версиями	Производительность
READ UNCOMMITTED	да	да	да	Минимальные	Нет	Очень высокая
READ COMMITTED	нет	да	да	Умеренные	Нет (или да, для READ_COMMITTED_SNAPSHOT)	Высокая
REPEATABLE READ	да	нет	да	Высокие	Нет	Средняя
SERIALIZABLE	да	нет	нет	Максимальные	Нет	Низкая
SNAPSHOT	да	нет	нет	Оптимистичные	Да, в tempdb	Зависит от нагрузки

Механизм блокировок (Locking).

Типы блокировок:

- Shared (S) – для чтения.
- Exclusive (X) – для изменения.
- Update (U) – для потенциального обновления.

– Intent (IS, IX, SIX) – намерение заблокировать на нижнем уровне.

– Schema (Sch-M, Sch-S) – для изменения схемы.

Мониторинг блокировок:

```
-- Анализ текущих блокировок
SELECT
  t.request_session_id AS [Session ID],
  db_name(resource_database_id) AS [Database],
  CASE resource_type
    WHEN 'OBJECT' THEN object_name(resource_associated_entity_id)
    WHEN 'DATABASE' THEN 'DATABASE'
    WHEN 'PAGE' THEN 'Page ' + cast(resource_description as varchar)
    ELSE resource_type
  END AS [Resource],
  request_mode AS [Lock Type],
  request_status AS [Status],
  request_owner_type AS [Owner Type]
FROM sys.dm_tran_locks t
WHERE resource_database_id = db_id('LibraryLab15')
ORDER BY request_session_id, resource_type;
```

Управление версиями строк (Row Versioning).

Архитектура RCSI и Snapshot Isolation:

```
-- КОМАНДЫ ВЫПОЛНЯЮТСЯ ОТДЕЛЬНО, КАЖДАЯ В СВОЕМ ОКНЕ ЗАПРОСА

-- Шаг 1: Убедитесь, что вы подключены к базе данных master
USE master;
GO

-- Шаг 2: Завершите все активные подключения к целевой базе данных
-- (Если вы единственный пользователь, этот шаг можно пропустить)
ALTER DATABASE LibraryLab15
SET SINGLE_USER WITH ROLLBACK IMMEDIATE;
GO

-- Шаг 3: Включите READ_COMMITTED_SNAPSHOT (отдельная команда)
ALTER DATABASE LibraryLab15
SET READ_COMMITTED_SNAPSHOT ON;
GO

-- Шаг 4: Включите ALLOW_SNAPSHOT_ISOLATION (отдельная команда)
ALTER DATABASE LibraryLab15
SET ALLOW_SNAPSHOT_ISOLATION ON;
GO

-- Шаг 5: Верните базу в многопользовательский режим
ALTER DATABASE LibraryLab15
SET MULTI_USER;
```

```
GO
-- Шаг 6: Вернитесь к вашей базе данных
USE LibraryLab15;
GO
```

Временное хранилище (tempdb):

- Старые версии строк копируются в tempdb.
- Каждая версия имеет метку времени транзакции.
- Очистка старых версий фоновым процессом.

Структура версионированной строки:

[Заголовок строки] → [Данные] → [14 байт сведений о версиях] → [Указатель на старую версию в tempdb]

Расширенные методы импорта и экспорта.

1. Архитектура массовых операций

Bulk Insert механизм:

Три режима работы BULK INSERT:

1. Non-logged (минимальное журналирование):
   - Требует TABLOCK.
   - Требует пустую таблицу или использование bcp.
   - Максимальная производительность.
2. Fully-logged (полное журналирование):
   - Используется по умолчанию.
   - Все изменения журналируются.
   - Медленнее, но безопаснее.
3. Partially-logged (частичное журналирование):
   - Журналируются только выделения страниц.
   - Требует простую модель восстановления.

```
-- Оптимизированный BULK INSERT
BULK INSERT Books
FROM 'C:\Data\books_100k.csv'
WITH (
  DATAFILETYPE = 'char',
  FIELDTERMINATOR = ',',
  ROWTERMINATOR = '\n',
  TABLOCK,          -- Минимальное журналирование
  ORDER (ISBN ASC),      -- Если есть кластеризованный индекс
  BATCHSIZE = 10000,     -- Размер пакета
  MAXERRORS = 100,       -- Максимальное количество ошибок
  CHECK_CONSTRAINTS,     -- Проверка ограничений
  FIRE_TRIGGERS       -- Вызов триггеров (медленнее)
          );
```

Использование OPENROWSET:

```
-- Чтение из Excel
SELECT *
FROM OPENROWSET(
  'Microsoft.ACE.OLEDB.12.0',
  'Excel 12.0;Database=C:\Data\books.xlsx;HDR=YES',
  'SELECT * FROM [Sheet1$]'
);

-- Чтение из CSV с помощью драйвера ODBC
SELECT *
FROM OPENROWSET(
  'MSDASQL',
  'Driver={Microsoft Text Driver (*.txt; *.csv)};DefaultDir=C:\Data;',
  'SELECT * FROM books.csv'
);

-- Чтение JSON
SELECT *
FROM OPENROWSET(
  BULK 'C:\Data\books.json',
  SINGLE_CLOB
) AS j
CROSS APPLY OPENJSON(j.BulkColumn)
WITH (
  ISBN VARCHAR(17),
  Title NVARCHAR(200),
  Author NVARCHAR(150)
          );
```

Последовательность выполнения лабораторной работы

1. Выполните добавление записей (INSERT) в рамках своей предметной области согласно предыдущей работы:

   – Простой INSERT с указанием значений.

   – INSERT с множественными значениями.

   – INSERT с выборкой данных (INSERT-SELECT).

2. Выполните редактирование записей (UPDATE):

   – Обновление одного поля для одной записи.

   – Обновление нескольких полей.

   – Обновление с использованием вычисляемых значений.

   – Условное обновление с CASE.

   – Проверка обновлений.

3. Выполните удаление записей (DELETE):

   – Удаление одной записи.

   – Каскадное удаление.

   – Удаление дубликатов.

4. Выполните применение логических условий.

Лабораторная работа №16 «Создание ключевых полей. Задание индексов. Установление и удаление связей между таблицами»

Теоретические сведения

В предыдущих работах были рассмотрены основы реляционной модели, нормализации, а также создание ключей и ограничений. Данная работа фокусируется на практических аспектах управления этими объектами в работающей базе данных.

Индексы: назначение и типы.

Индекс – это объект базы данных, ускоряющий поиск, сортировку и соединение данных за счет создания дополнительной структуры (своего рода «оглавления»).

Кластеризованный индекс (Clustered): определяет физический порядок хранения данных в таблице. Таблица может иметь только один кластеризованный индекс. Часто совпадает с PK.

Некластеризованный индекс (Non-Clustered): создает отдельную структуру, указывающую на данные. Таблица может иметь множество некластеризованных индексов. Используется для ускорения поиска по часто используемым полям, не входящим в PK.

Составной индекс: создается по нескольким столбцам. Порядок столбцов в индексе критически важен для его эффективности.

Покрывающий индекс (Covering): индекс, который содержит все столбцы, необходимые для выполнения конкретного запроса, что позволяет избежать обращений к самой таблице (к ее данным).

Связи между таблицами и их поддержка.

Связи реализуются через механизм внешних ключей. При определении связи можно задать правила каскадных операций для поддержания целостности при изменениях:

ON DELETE: что делать с дочерними записями при удалении родительской?

NO ACTION (по умолчанию): запретить удаление, если существуют зависимые записи.

CASCADE: удалить все зависимые записи.

SET NULL: установить NULL в столбце внешнего ключа у зависимых записей (столбец должен допускать NULL).

SET DEFAULT: установить значение по умолчанию в столбце внешнего ключа.

ON UPDATE: аналогичные правила для случая обновления значения первичного ключа в родительской таблице.

Последовательность выполнения лабораторной работы

Используйте свою предметную область из лабораторной работы №5 и созданную в предыдущих работах базу данных.

1. Для одной из своих основных таблиц:
   - Создайте составной первичный ключ из двух полей, если это логически оправдано. Объясните в отчете свой выбор.
   - Создайте уникальное ограничение на поле, которое не должно повторяться (например, email, VIN-код, серийный номер).
2. Проанализируйте потенциальные запросы в вашей БД. Создайте не менее двух индексов для их оптимизации:
   - Один простейший некластеризованный индекс для поиска по одному часто используемому полю.

- Один составной или покрывающий индекс для запроса, который включает условия (WHERE), сортировку (ORDER BY) или соединения (JOIN).
- Выполните целевой запрос с включенным планом выполнения (Ctrl+M). Сделайте скриншот плана и объясните, как используется созданный индекс.

3. Модифицируйте связи в вашей БД:

- Для одной из связей «один-ко-многим» измените правило каскадного удаления с NO ACTION на CASCADE. Напишите и выполните скрипт.
- Проверьте работу правила: создайте тестовые данные, удалите запись в родительской таблице и убедитесь, что зависимые записи удалились.
- Восстановите исходное состояние, вернув правило NO ACTION.

2 ОБЩАЯ ХАРАКТЕРИСТИКА САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ

Самостоятельная работа - целенаправленная, планируемая в рамках учебного плана деятельность студентов, которая осуществляется по заданию, при методическом руководстве и контроле преподавателя, но без его непосредственного участия. Самостоятельная работа студентов является одной из важнейших составляющих образовательного процесса.

В учебном процессе учебного заведения выделяют два вида самостоятельной работы: аудиторная и внеаудиторная.

Аудиторная самостоятельная работа выполняется на учебных занятиях под непосредственным руководством преподавателя и по его заданию.

Внеаудиторная – планируемая учебная, учебно-исследовательская, научно-исследовательская работа студентов, выполняемая во внеаудиторное время по заданию и при методическом руководстве преподавателя, но без его непосредственного участия.

Целью самостоятельной работы студентов является:

- систематизация и закрепление полученных теоретических знаний и практических умений;
- углубление и расширение теоретических знаний;
- формирование умений использовать нормативную, правовую, справочную документацию и специальную литературу;
- развитие познавательных способностей и активности студентов, творческой инициативы, самостоятельности, ответственности, организованности;
- формирование самостоятельности мышления, способностей к саморазвитию, совершенствованию и самоорганизации;
- формирование общих и профессиональных компетенций.

Самостоятельная работа студентов должна быть хорошо спланирована и организована. При планировании такой работы необходимо учитывать условия, обеспечивающие её успешное выполнение:

- чёткое определение преподавателем объёма и содержания самостоятельной работы;
- определение видов консультативной помощи;
- постановка цели самостоятельной работы и критерии её оценки;
- виды и формы контроля её выполнения.

Выполняя самостоятельную работу под контролем преподавателя, студент должен:

- освоить минимум знаний;
- планировать свою самостоятельную работу в соответствии разработанным графиком;
- выполнять самостоятельную работу и отчитываться по ее результатам в соответствии с графиком представления результатов, видами и сроками отчетности по самостоятельной работе студентов.

В процессе самостоятельной работы студент приобретает навыки самоорганизации, самоконтроля, самоуправления, саморефлексии и становится активным самостоятельным субъектом учебной деятельности.

Таким образом, самостоятельная работа студентов оказывает важное влияние на формирование личности будущего специалиста.

Самостоятельная работа студентов является обязательной для каждого студента, объем ее определяется учебным планом в соответствии с требованиями Государственных образовательных стандартов.

При изучении тем дисциплины студенты выполняют следующие виды самостоятельной работы:

- проработка конспектов занятий, учебных изданий и специальной технической литературы;
- составление конспекта, тематических схем, таблиц;
- подготовка к лабораторным работам и практическим занятиям с использованием методических рекомендаций преподавателя;
- оформление отчетов по лабораторным работам и практическим занятиям, подготовка к их защите;
- моделирование и решение производственных процессов и ситуационных задач;
- подготовка презентаций;
- работа с электронными ресурсами в сети Интернет;
- подготовка к семинару;
- подготовка к зачетам, экзаменам.

Технология организации самостоятельной работы студентов включает использование информационных и материально-технических ресурсов образовательного учреждения. Материально-техническое и информационно - техническое обеспечение самостоятельной работы студентов включает в себя:

- библиотеку с читальным залом, укомплектованную в соответствии с существующими нормами;
- учебно-методическую базу учебных кабинетов, лабораторий и методического центра;
- компьютерные классы с возможностью работы в Интернет;
- базы практики в соответствии с заключенными договорами;
- аудитории для консультационной деятельности;
- учебную и учебно-методическую литературу, разработанную с учетом увеличения доли самостоятельной работы студентов, и иные методические материалы.

Перед выполнением внеаудиторной самостоятельной работы преподаватель проводит инструктаж по выполнению задания, в котором указывает цель задания, его содержание, сроки выполнения, ориентировочный объем работы, основные требования к результатам работы, критерии оценки. Во время выполнения студентами внеаудиторной самостоятельной работы и при необходимости преподаватель может проводить консультации. Самостоятельная работа может осуществляться

индивидуально или группами студентов в зависимости от цели, объема, конкретной тематики самостоятельной работы, уровня сложности, уровня умений обучающихся.

3 МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ВЫПОЛНЕНИЮ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

Общие методические рекомендации студенту при изучении тем дисциплины.

Большая часть самостоятельной работы выполняется студентом вне учебных занятий при подготовке домашних заданий. Общие требования к выполнению этого вида самостоятельной работы заключаются в следующем:

- активно работать на уроке, усваиваю основную часть нового материала;
- если что-то непонятно, не стесняться задавать вопросы преподавателю;
- большое задание необходимо разбивать на части и работать над каждой из них в отдельности;
- выполняя домашнее задание, надо не просто думать, что надо сделать, а еще и решать, с помощью каких средств и приемов этого можно добиться;
- в процессе приготовления домашнего задания необходимо делать перерывы;
- готовиться к докладам, рефератам, защите курсовых работ и проектов, практических и лабораторных занятий надо заранее, равномерно распределяя нагрузку, а не оставлять такую ответственную работу на последний день;
- изучая заданный материал, сначала надо его понять, а уже потом запомнить;
- научиться находить интересующую нужную информацию с помощью компьютера;
- не стесняться обращаться за помощью к взрослым и однокурсникам;
- надо составлять план устного ответа и проверять себя;
- на письменном столе должно лежать только то, что необходимо для выполнения одного задания. После его завершения со стола убираются уже использованные материалы, и кладутся те учебные принадлежности, которые необходимы для выполнения следующего задания;
- нужно решить, в какой последовательности лучше выполнять задания и сколько времени понадобится на каждое из них;
- трудный материал урока лучше повторить в тот же день, чтобы сразу закрепить его и запомнить;
- читая учебник, надо задавать самому себе вопросы по тексту.

Подготовка тематических сообщений, докладов, рефератов

Реферат доклад, сообщение (от латинского refero - передаю, сообщаю) - краткое письменное изложение материала по определенной теме с целью привития студентам навыков самостоятельного поиска и анализа информации, формирования умения подбора и изучения литературных

источников, используя при этом дополнительную научную, методическую и периодическую литературу.

Тема реферата выбирается по желанию студента из списка, предлагаемого преподавателем. Тема может быть сформулирована студентом самостоятельно.

Выбранная тема согласовывается с преподавателем.

После выбора темы требуется:

- составить план реферата;
- подобрать необходимую информацию;
- изучить подобранную информацию;
- составить текст реферата.

План реферата должен включать в себя введение, основной текст и заключение. Во введении аргументируется актуальность выбранной темы, указываются цели и задачи исследования. В нем также отражается методика исследования и структура работы. Основная часть работы предполагает освещение материала в соответствии с планом. В заключении излагаются основные выводы и рекомендации по теме исследования.

Реферат оформляется согласно требованиям, установленным в учебном заведении. Он должен содержать: титульный лист, оглавление и список использованной литературы. На титульном листе указываются: название учебного заведения, название профессионального модуля, междисциплинарного курса, тема работы, курс, группа, фамилии, имена, отчества студента и руководителя работы, название города, в котором находится учебное заведение, год написания данной работы. Реферат может содержать приложения в форме схем, образцов документов и другие изображения в соответствии с темой исследования. Все страницы работы, включая оглавление и список литературы, нумеруются по порядку с титульного листа (на нем цифра не ставится) до последней страницы без пропусков и повторений. Введение, заключение, новые главы, список использованных источников и литературы должны начинаться с нового листа. Подбор литературы производится студентом из предложенного преподавателем списка литературы. Текст реферата необходимо набирать на компьютере на одной стороне листа. Размер левого поля 30 мм, правого - 15 мм, верхнего - 20 мм, нижнего - 20 мм. Шрифт - Times New Roman, размер - 14, межстрочный интервал - 1,5. Фразы, начинающиеся с новой строки, печатаются с абзацным отступом от начала строки (1,25 см). Реферат, выполненный небрежно, неразборчиво, без соблюдения требований по оформлению, возвращается студенту без проверки с указанием причин возврата на титульном листе.

Критерии оценки:

- знание и понимание проблемы;
- умение систематизировать и анализировать материал, четко и обоснованно формулировать выводы;
- «трудозатратность» (объем изученной литературы, добросовестное отношение к анализу проблемы);

- самостоятельность, способность к определению собственной позиции по проблеме и к практической адаптации материала, недопустимость плагиата;
- выполнение необходимых формальностей (точность в цитировании и указании источника текстового фрагмента, аккуратность оформления).

Проработка занятый, учебных изданий и специальной технической литературы

Работа с конспектом лекций по темам междисциплинарных курсов заключается в том, что студент после рассмотрения темы на учебных занятиях в период между очередными лекциями изучает материал конспекта. При этом непонятные положения конспекта необходимо выяснять у преподавателя на консультациях или при чтении основной и дополнительной литературы.

При работе с книгой необходимо научиться правильно ее читать, вести записи. Для подбора литературы в библиотеке используются алфавитный и систематический каталоги. Правильный подбор учебников рекомендуется преподавателем. Необходимая литература может быть также указана в методических разработках. Изучая материал по учебнику, следует переходить к следующему вопросу только после правильного уяснения предыдущего, описывая на бумаге все выкладки и определения (в том числе те, которые в учебнике опущены или на лекции даны для самостоятельного вывода). Полезно составлять опорные конспекты. При изучении материала по учебнику, полезно в тетради (на специально отведенных полях) дополнять конспект лекций, написанный на учебных занятиях. Там же следует отмечать вопросы, выделенные студентом для консультации с преподавателем. Выводы, полученные в результате изучения, рекомендуется в конспекте выделять, чтобы они при пропитывании записей лучше запоминались. Различают два вида чтения; первичное и вторичное. Первичное - эти внимательное, неторопливое чтение, при котором можно остановиться на трудных местах. После него не должно остаться ни одного непонятного слова. Содержание не всегда может быть понятно после первичного чтения. Задача вторичного чтения - полное усвоение смысла целого (по счету это чтение может быть и не вторым, а третьим или четвертым).

Чтение научного текста является частью познавательной деятельности. Ее цель - извлечение из текста необходимой информации. От того на сколько осознанна читающим собственная внутренняя установка при обращении к печатному слову (найти нужные сведения, усвоить информацию полностью или частично, критически проанализировать материал и т.п.) во многом зависит эффективность осуществляемого действия. Выделяют четыре основные установки в чтении научного текста:

- информационно-поисковая, задача которой - найти, выделить искомую информацию;

- усваивающая, при которой усилия читателя направлены на то, чтобы как можно полнее осознать и запомнить как сами сведения, излагаемые автором, так и всю логику его рассуждений;
- аналитико-критическая - читатель стремится критически осмыслить материал, проанализировав его, определив свое отношение к нему;
- творческая, создающая у читателя готовность в том или ином виде использовать суждения автора, ход его мыслей, результат наблюдения, разработанную методику, дополнить их, подвергнуть новой проверке.

Самостоятельная работа при чтении учебной литературы начинается с изучения конспекта материала, полученного при слушании лекций преподавателя. Полученную информацию необходимо осмыслить. При необходимости, в конспект лекций могут быть внесены схемы, эскизы рисунков, другая дополнительная информация.

Составление конспекта, тематических схем, таблиц

При изучении нового материала, как правило, составляется конспект. Конспект - изложение текста, которому присущи краткость, связность и последовательность. При этом максимально точно записываются формулы, определения, схемы, трудные для запоминания места.

При оформлении конспекта необходимо стремиться к емкости каждого предложения. Мысли автора книги следует излагать кратко, заботясь о стиле и выразительности написанного. Число дополнительных элементов конспекта должно быть логически обоснованным, записи должны распределяться в определенной последовательности, отвечающей логической структуре текста. Для уточнения и дополнения необходимо оставлять поля. Овладение навыками конспектирования требует от студента целеустремленности, повседневной самостоятельной работы.

Классификация конспектов:

- плановый конспект, для чего сначала нужно написать план текста, а затем на пункты плана делаются комментарии: свободно изложенный текст либо цитаты;
- обзорный конспект - краткое изложение данной темы с использованием нескольких источников;
- текстуальный конспект состоит из цитат одного текста;
- свободный конспект предполагает цитаты текста и собственные формулировки прочитанного текста;
- сложный - конспект, в котором отражается определенная тема или вопрос;
- хронологический конспект отражает последовательность событий;
- опорный конспект, в котором излагается информация в виде опорных знаков, слов, сигналов.

Методические рекомендации по составлению конспекта:

- определить цель написания конспекта;

- внимательно прочитать текст, уточнить в справочной литературе непонятные слова;
- выделить основные смысловые части текста;
- определить главное, составить план;
- кратко сформулировать основные положения текста, отметить аргументацию автора;
- составить текст конспекта, изложив информацию кратко и своими словами, четко следуя пунктам плана, записи следует вести четко, ясно;
- грамотно записывать цитаты, учитывая лаконичность, значимость мысли;
- в тексте конспекта желательно приводить не только тезисные положения, но и их доказательства.

При составлении тематических схем, таблиц необходимо внимательно прочитать текст соответствующий параграф учебника. Продумать «конструкцию» таблицы или схемы, расположение порядковых номеров, терминов, примеров и пояснений (и прочего). Начертить схему или таблицу и заполнить ее графы необходимым содержимым.

Подготовка к лабораторным работам и практическим занятиям, оформление отчетов по лабораторным работам и практическим занятиям, подготовка к их защите

Программы профессиональных модулей предусматривают выполнение практических и лабораторных занятий.

Лабораторное занятие - форма учебного занятия, ведущей дидактической целью которого является экспериментальное подтверждение и проверка существующих теоретических положений (законов, зависимостей), формирование учебных и профессиональных практических умений и навыков.

Практическое занятие - это одна из форм учебной работы, которая ориентирована на закрепление изученного теоретического материала, его более глубокое усвоение и формирование умения применять теоретические знания в практических целях. Особое внимание на практических занятиях уделяется выработке учебных или профессиональных навыков. Такие навыки формируются в процессе выполнения конкретных заданий - упражнений, задач - под руководством и контролем преподавателя.

Подготовка к практическим и лабораторным занятиям заключается в работе с конспектом лекций по данной теме, в изучении соответствующего раздела учебника или учебного пособия, в просмотре дополнительной литературы. Этапы подготовки к практическому или лабораторному занятию заключаются в следующем: освежить в памяти теоретические сведения, полученные на лекциях и в процессе самостоятельной работы, подобрать необходимую учебную и справочную литературу. Отобрать те материалы, которые позволят в полной мере реализовать цели и задачи предстоящей работы. Еще раз проверить соответствие отобранного материала. Студент

должен прийти на лабораторное или практическое занятие подготовленным по данной теме.

При выполнении заданий практического или лабораторного занятия студент должен быть ознакомлен преподавателем с целью и ходом выполнения задания и, по необходимости, с правилами техники безопасности. Если у студентов во время выполнения заданий возникают вопросы, то преподаватель консультирует студентов. Порядок выполнения того или иного задания излагается в инструкционных картах или рабочих тетрадях.

После проведения занятия студент представляет письменный отчет, который оформляется в соответствие с принятыми в образовательном учреждении правилами. Отчеты оформляются на листах писчей бумаги формата А4 или в специальных рабочих тетрадях, разработанных преподавателем. Содержание отчета указано в инструкционных картах или рабочих тетрадях.

При подготовке к защите практических и лабораторных занятий студент должен ответить на контрольные вопросы, указанные также в инструкционных картах или рабочих тетрадях, проштудировав при этом конспект лекций, учебную литературу.

Моделирование и решение производственных процессов и ситуационных задач

При изучении дисциплины очень часто студенту приходится сталкиваться с профессиональными задачами и ситуациями, которые необходимо решить самостоятельно, как во время аудиторной работы, так и во время внеаудиторной. При решении таких задач необходимо:

- провести анализ ситуации для определения проблемы в целом; представить ситуацию и себя в качестве действующего в ней лица; проанализировать ошибочные или правильные действия всех участников ситуации;
- определить проблемные узлы - возможные причины и прогнозируемые последствия развития данной ситуации;
- рассмотреть условное прогнозирование развития ситуации: определить окончательную гипотезу, представить обоснованный и доказательный прогноз вероятностного развития ситуации; предложить варианты действий, обоснованные теоретически и, по возможности, подкрепленные практическим личным опытом, опираясь на принципы профессиональной этики; определить способы и методы воздействия на предлагаемую ситуацию;
- сформулировать итоговые выводы, используя профессиональные термины, доказательства правильности своего решения.

Подготовка презентаций

Подготовка презентации позволит студенту логически выстроить изучаемый материал, систематизировать его, сформировать

коммуникативные компетенции. Материал презентации представляется в виде текста, схем, диаграмм, таблиц, которые призваны дополнить текстовую информацию или передать ее в более наглядном виде. Желательно избегать в презентации изображений, не несущих смысловой нагрузки, если они не являются частью стилевого оформления. Цвет графических изображений не должен резко контрастировать с общим стилевым оформлением слайдов, иллюстрации рекомендуется сопровождать пояснительным текстом.

Анимационные эффекты используются для привлечения внимания слушателей или для демонстрации динамики развития какого - либо процесса. В этих случаях использование анимации оправдано, но не стоит чрезмерно насыщать презентацию такими эффектами, иначе это вызовет негативную реакцию аудитории.

Звуковое сопровождение должно отражать суть или подчеркивать особенность темы слайда, презентации. Фоновая музыка не должна отвлекать внимание слушателей и заглушать слова докладчика.

Оптимальное количество слайдов, как правило, десять - пятнадцать. Для оформления слайдов презентации рекомендуется использовать несложные шаблоны, соблюдать единый стиль. Не рекомендуется на одном слайде использовать более трех цветов. Смену слайдов для управления презентацией докладчиком желательно устанавливать по щелчку без времени. Шрифт, выбираемый для презентации, должен обеспечивать читаемость информации на экране и соответствовать выбранному шаблону оформления. Не желательно использовать разные шрифты в одной презентации.

Алгоритм выстраивания презентации должен соответствовать логической структуре работы и отражать последовательность ее этапов. Независимо от алгоритма выстраивания презентации на первом слайде рекомендуется выносить следующие данные: полное наименование образовательной организации; тема презентации; фамилия, имя, отчество студента; специальность обучения; фамилия, имя, отчество руководителя. Последний слайд должен содержать фразу «Спасибо за внимание».

Работа с электронными ресурсами в сети Интернет

Для повышения эффективности самостоятельной работы студент должен учиться работать в поисковой системе сети Интернет, в электронно-библиотечной системе и использовать найденную информацию при подготовке к занятиям.

Интернет сегодня - правомерный источник научных статей, статистической и аналитической информации, и использование его наряду с книгами давно уже стало нормой. Однако, несмотря на то, что ресурсы Интернета позволяют достаточно быстро и эффективно осуществлять поиск необходимой информации, следует помнить о том, что эта информация может быть неточной или вовсе не соответствовать действительности. В связи с этим при поиске материала по заданной тематике следует обращать

внимание на научные труды признанных авторов, которые посоветовали вам преподаватели.

Поиск информации можно вести по автору, заглавию, виду издания, году издания или издательству. Также в сети Интернет доступна услуга по скачиванию методических указаний и учебных пособий, подбору необходимой учебной и научно - технической литературы.

Подготовка к семинару

Семинар – это особая форма учебно-теоретических занятий, которая, как правило, служит дополнением к лекционному курсу. Семинар обычно посвящен детальному изучению отдельной темы.

Этапы подготовки к семинару:

- проанализировать тему семинара, подумать о цели и основных проблемах, вынесенных на обсуждение;
- внимательно прочитать материал, данный преподавателем по этой теме на лекции;
- изучить рекомендованную литературу, делая при этом конспекты прочитанного или выписки, которые понадобятся при обсуждении на семинаре;
- постараться сформулировать свое мнение по каждому вопросу и аргументированно его обосновать;
- записать возникшие во время самостоятельной работы с учебниками и научной литературой вопросы, чтобы затем на семинаре получить на них ответы.

При подготовке к семинарским занятиям следует руководствоваться указаниями и рекомендациями преподавателя, использовать основную и дополнительную литературу из представленного им списка.

При подготовке доклада на семинарское занятие желательно заранее обсудить с преподавателем перечень используемой литературы, за день до семинарского занятия предупредить его о необходимых для представления материала технических средствах. Напечатанный текст доклада представить преподавателю на рецензию.

Подготовка к зачетам, экзаменам

Изучение выше перечисленных тем дисциплины завершается зачетами или экзаменами.

Подготовка к зачету или экзамену способствует закреплению, углублению и обобщению знаний, получаемых в процессе обучения, а также применению их к решению практических задач. Готовясь к зачету или экзамену, студент ликвидирует имеющиеся пробелы в знаниях, углубляет, систематизирует и упорядочивает свои знания. На зачете или экзамене студент демонстрирует то, что он приобрел в процессе обучения конкретным темам междисциплинарных курсов или модулям в целом.

Экзаменационная сессия - это серия экзаменов, установленных учебным планом. Между экзаменами, согласно графику их проведения,

дается интервал времени в несколько дней. Не следует думать, что их достаточно для успешной подготовки к экзаменам. В эти дни нужно систематизировать уже имеющиеся знания. На консультации перед экзаменом студентов познакомят с основными требованиями, ответят на возникшие у них вопросы. Поэтому посещение консультаций обязательно.

Требования к организации подготовки студента к экзаменам те же, что и при занятиях в течение семестра, но соблюдаться они должны более строго. Во-первых, очень важно соблюдение режима дня: сон не менее 8 часов в сутки, занятия должны заканчиваться не позднее, чем за 2-3 часа до сна.

Оптимальное время занятий - утренние и дневные часы. В перерывах между занятиями рекомендуются прогулки на свежем воздухе, неутомительные занятия спортом. Во-вторых, наличие хороших собственных конспектов лекций. Даже в том случае, если была пропущена какая- либо лекция, необходимо во время ее восстановить, обдумать, снять возникшие вопросы для того, чтобы запоминание материала было осознанным. В-третьих, при подготовке к зачету или экзамену у студента должен быть хороший учебник или конспект литературы, прочитанной по указанию преподавателя в течение семестра. Здесь можно эффективно использовать листы опорных конспектов. Вначале следует просмотреть весь материал по сдаваемой теме, отметить для себя трудные вопросы, обязательно в них разобраться. В заключение еще раз целесообразно повторить основные положения. Систематическая подготовка к занятиям в течение семестра позволит использовать время экзаменационной сессии для систематизации знаний.

Правила подготовки к экзамену:

- сориентироваться во всем материале и обязательно расположить его согласно экзаменационным вопросам или вопросам, обсуждаемым на семинарах, учебных занятиях. Эта работа может занять много времени, но все остальное - уже технические детали, главное - это ориентировка в материале;
- постараться максимально запомнить материал, переосмыслить его, рассмотреть альтернативные идеи;
- подготовить «шпаргалки», главный смысл которых систематизация и оптимизация знаний, однако пользоваться таким подспорьем не рекомендуется. Это очень сложная и важная для студента работа, более сложная и важная, чем простое поглощение массы учебной информации. Если студент самостоятельно подготовил такие «шпаргалки», то, скорее всего, он и экзамены сдавать будет более уверенно, так как у него уже сформирована общая ориентировка в сложном материале. Как это ни парадоксально, но использование «шпаргалок» часто позволяет отвечающему студенту лучше демонстрировать свои познания, точнее - ориентировку в знаниях, что намного важнее знания «запомненного» и «тут же забытого» после сдачи экзамена.

При ответе на экзамене студент сначала должен продемонстрировать преподавателю усвоенный по программе обучения материал, и лишь после этого высказать иную, желательно аргументированную точку зрения.

4 МЕТОДИКА ВЫПОЛНЕНИЯ ВНЕАУДИТОРНОЙ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

1. Получить у преподавателя задание и необходимую литературу.
2. Найти предложенную литературу на образовательном портале или в библиотеке.
3. Изучить имеющуюся литературу в электронном или печатном виде, прочитать материалы лекций, практических и (или) семинарских занятий по теме.
4. Изучить методические рекомендации.
5. Оформить работу в тетради или на компьютере в соответствии с требованиями преподавателя.
6. Сдать самостоятельную работу преподавателю, предварительно ответив на вопросы для самоконтроля.

5 МЕТОДЫ КОНТРОЛЯ И ОЦЕНКА ВНЕАУДИТОРНОЙ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

Контроль результатов самостоятельной работы проводится преподавателем одновременно с текущим и промежуточным контролем знаний обучающихся. Для контроля самостоятельной работы обучающегося используются разнообразные формы и методы: фронтальный, индивидуальный, выборочный, самоконтроль, защита презентации, участие в семинарском занятии, ответы на контрольные вопросы и т. д. При контроле результатов самостоятельной работы используются следующие критерии:

- уровень освоения обучающимся учебного материала;
- умение обучающегося использовать теоретические знания при выполнении заданий;
- обоснованность и чёткость изложения ответа;
- оформления материала в соответствии с требованиями.

Критерии оценки выполненной обучающимися работы:

оценка «5» - работа выполнена без ошибок; чисто, без исправлений; тема раскрыта полностью;

оценка «4» - работа выполнена с незначительными ошибками; тема раскрыта не полностью;

оценка «3» - работа выполнена со значительными ошибками; тема практически не раскрыта;

оценка «2» - работа не выполнена.

6 ТЕМЫ ИНДИВИДУАЛЬНЫХ ЗАДАНИЙ ДЛЯ ВЫПОЛНЕНИЯ КУРСОВОЙ РАБОТЫ

Тема 1. Платформа для организации гибридных мероприятий (конференций, митапов)

Описание: система для планирования мероприятий с возможностью онлайн-участия, оффлайн-посещения, управления сессиями, спикерами и интерактивными активностями.

Таблица 6.1 – Описание предметной области варианта №1

Сущность	Атрибуты
Событие	Код события, Название, Описание, Дата и время начала, Дата и время окончания, Тип (онлайн/оффлайн/гибрид), Максимум участников, Срок регистрации, Статус (черновик/опубликовано/завершено)
Сессия	Код сессии, Код события, Название сессии, Код спикера, Время начала, Время окончания, Комната или ссылка, Тип сессии (лекция/воркшоп/панель)
Участник	Код участника, ФИО, Электронная почта, Телефон, Тип участия (онлайн/оффлайн), Дата регистрации
Регистрация	Код регистрации, Код участника, Код события, Дата регистрации, Тип билета (стандартный/VIP), Статус посещения (зарегистрирован/присутствовал/отменено)
Онлайн-взаимодействие	Код взаимодействия, Код сессии, Код участника, Текст вопроса, Время вопроса, Количество голосов

Тема 2. Система мониторинга умного города.

Описание: база данных для сбора и анализа данных с датчиков городской инфраструктуры (освещение, мусор, парковка, качество воздуха).

Таблица 6.2 – Описание предметной области варианта №2

Сущность	Атрибуты
Тип датчика	Код типа, Название, Описание, Единица измерения, Тип данных (целое/дробное/логическое)
Датчик	Код датчика, Код типа, Код локации, Дата установки, Дата последнего обслуживания, Статус (активен/неактивен/ошибка)
Локация	Код локации, Район, Улица, Координаты, Тип локации (улица/парк/площадь)
Показание	Код показания, Код датчика, Метка времени, Значение, Признак аномалии
Аномалия	Код аномалии, Код показания, Время обнаружения, Код назначенного сотрудника, Статус устранения (зафиксировано/в работе/решено), Примечания по решению
Сотрудник службы	Код сотрудника, ФИО, Отдел, Телефон, График смен

Тема 3. Платформа для коллективного инвестирования в стартапы.

Описание: система, где инвесторы вкладывают небольшие суммы в стартапы, отслеживают их развитие и получают отчетность.

Таблица 6.3 – Описание предметной области варианта №3

Сущность	Атрибуты
Стартап-проект	Код стартапа, Название, Код основателя, Описание, Отрасль, Целевая сумма, Текущие инвестиции, Доля предложенная (%), Статус (сбор средств/в работе/выход/закрыт)
Основатель	Код основателя, ФИО, Электронная почта, Телефон, Биография, Ссылка на профиль
Инвестор	Код инвестора, ФИО, Электронная почта, ИНН, Статус аккредитации (аккредитованный/розничный), Общая сумма инвестиций
Инвестиция	Код инвестиции, Код инвестора, Код стартапа, Сумма, Дата инвестиции, Доля приобретена
Отчет о прогрессе	Код отчета, Код стартапа, Квартал, Год, Достигнутые вехи, Финансовый итог, Проблемы, Следующие цели

Сущность	Атрибуты
Транзакция выплаты	Код транзакции, Код инвестиции, Сумма выплаты, Дата выплаты, Тип транзакции (дивиденды/доход от продажи)

Тема 4. Система управления цепочками поставок с блокчейн-трекингом.

Описание: отслеживание товара от производителя до потребителя с фиксацией каждого этапа в неизменяемом журнале.

Таблица 6.4 – Описание предметной области варианта №4

Сущность	Атрибуты
Продукт	Код продукта, Название, Номер партии, Код производителя, Дата производства, Срок годности, Спецификации
Участник цепи	Код участника, Название, Роль (производитель/дистрибьютор/склад/ритейлер), Адрес, Публичный ключ
Этап поставки	Код этапа, Код продукта, Код отправителя, Код получателя, Метка времени начала, Метка времени окончания, Лог температуры, Хэш этапа
Сертификат	Код сертификата, Код продукта, Выдающий орган, Тип сертификата (органик/честная торговля/безопасность), Дата выдачи, Ссылка на проверку

Тема 5. База данных для киберспортивной команды.

Описание: учет игроков, тренировок, тактических разработок и детальной статистики по матчам.

Таблица 6.5 – Описание предметной области варианта №5

Сущность	Атрибуты
Игрок	Код игрока, Игровой псевдоним, Настоящее имя, Роль, Дата вступления, Дата окончания контракта, Зарплата
Матч	Код матча, Название турнира, Дата матча, Название команды со-

Сущность	Атрибуты
	перника, Результат (победа/поражение), Длительность игры, Ссылка на запись
Статистика игрока	Код статистики, Код матча, Код игрока, Игровой персонаж, Убийства, Смерти, Помощи, Золото в минуту, Нанесенный урон, Купленные предметы
Тренировочный матч	Код тренировки, Запланированные дата и время, Имя соперника, Фокус области (ранняя игра/командные бои), Заметки тренера

Тема 6. Платформа для аренды профессионального оборудования.

Описание: система бронирования дорогостоящего профессионального оборудования (камеры, дроны, инструменты) с системой страховки и проверки.

Таблица 6.6 – Описание предметной области варианта №6

Сущность	Атрибуты
Единица оборудования	Код единицы, Код владельца, Код категории, Модель, Серийный номер, Стоимость покупки, Цена аренды в день, Сумма залога, Местонахождение, Статус (доступно/арендовано/на обслуживании)
Владелец	Код владельца, Название компании, Контактное лицо, Налоговый номер, Банковские реквизиты, Рейтинг
Арендатор	Код арендатора, ФИО, Паспортные данные, Номер водительского удостоверения, Статус верификации (на проверке/подтвержден/отклонен), Рейтинг арендатора
Бронирование	Код бронирования, Код единицы, Код арендатора, Дата начала, Дата окончания, Общая стоимость, Опция страховки, Статус брони (подтверждено/активно/завершено/отменено)
Акт приема-передачи	Код акта, Код бронирования, Фото при передаче, Фото при возврате, Отмеченные повреждения, Возвращенный депозит

Тема 7. Система управления экологическими проектами и углеродными кредитами.

Описание: учет проектов по восстановлению лесов, ВИЭ, торговля верифицированными углеродными единицами.

Таблица 6.7 – Описание предметной области варианта №7

Сущность	Атрибуты
Экопроект	Код проекта, Название, Тип (лесовосстановление/солнечная электростанция/улавливание метана), Местоположение, Дата начала, Расчетный объем CO2 в год, Орган верификации
Верификация	Код верификации, Код проекта, Аудиторская компания, Дата аудита, Выпущенный объем единиц, Номер сертификата, Срок действия
Покупатель кредитов	Код покупателя, Название компании, Отрасль, Цель по устойчивому развитию, Всего куплено единиц
Сделка с единицами	Код сделки, Код проекта, Код покупателя, Количество единиц, Цена за единицу, Дата сделки, Статус расчетов (в ожидании/завершено)

Тема 8. База данных для диджитал-арт галереи и торговли NFT.

Описание: каталогизация цифровых произведений искусства, управление авторами, коллекциями и историей владения.

Таблица 6.8 – Описание предметной области варианта №8

Сущность	Атрибуты
Цифровое искусство	Код произведения, Код художника, Название, Описание, Дата создания, Хэш файла, Ссылка на превью, Категория искусства, Блокчейн выпуска
Художник	Код художника, Псевдоним, Адрес кошелька, Биография, Ссылки на соцсети
Коллекция	Код коллекции, Название, Код куратора, Тема, Дата запуска

Сущность	Атрибуты
Владение NFT	Код владения, Код произведения, Адрес кошелька владельца, Дата приобретения, Цена приобретения, Адрес предыдущего владельца

Тема 9. Система планирования для кейтерингового бизнеса.

Описание: управление заказами на выездное обслуживание, складом продуктов, бригадами и маршрутами.

Таблица 6.9 – Описание предметной области варианта №9

Сущность	Атрибуты
Мероприятие заказчика	Код мероприятия, Код клиента, Тип мероприятия, Дата, Количество гостей, Адрес места, Особые пожелания
Заказ на кейтеринг	Код заказа, Код мероприятия, Код пакета меню, Общая стоимость, Внесенный депозит, Статус оплаты, Количество обслуживающего персонала, Присутствие шеф-повара
Пакет меню	Код пакета, Название, Описание, Цена с человека, Включенные блюда
Ингредиент	Код ингредиента, Название, Единица измерения, Текущий запас, Минимальный запас, Код поставщика

Тема 10. Платформа для микро-обучения сотрудников.

Описание: корпоративная система для создания коротких учебных модулей, их назначения и отслеживания прогресса.

Таблица 6.10 – Описание предметной области варианта №10

Сущность	Атрибуты
Учебный модуль	Код модуля, Название, Тема, Тип контента (видео/тест/статья), Длительность в минутах, Код автора, Дата создания
Сотрудник	Код сотрудника, ФИО, Отдел, Должность, Дата приема
Назначение	Код назначения, Код модуля, Код сотрудника, Код назначившего, Срок выполнения, Статус выполнения (назначено/в процессе/завершено/просрочено)
Попытка прохождения	Код попытки, Код назначения, Время начала, Время окончания, Балл, Журнал ответов

Тема 11. Система управления парком дронов для агромониторинга.

Описание: учет БПЛА, полетных заданий, сельхозучастков и собранных данных (мультиспектральные снимки).

Таблица 6.11 – Описание предметной области варианта №11

Сущность	Атрибуты
Модель БПЛА	Код модели, Название производителя, Максимальное время полета, Тип датчиков
Конкретный дрон	Серийный номер дрона, Код модели, Дата ввода в эксплуатацию, Статус (исправен/на обслуживании/списан)
Полетное задание	Код задания, Серийный номер дрона, Код участка, Запланированная дата, Цель облета, Выполнено (да/нет)
Поле/Участок	Код участка, Номер поля, Площадь, Культура, Координаты границ
Собранные данные	Код данных, Код задания, Тип снимка, Дата съемки, Ссылка на файл с данными

Тема 12. База данных для театральной площадки.

Описание: учет постановок, артистов, ролей, репетиций, костюмов и реквизита.

Таблица 6.12 – Описание предметной области варианта №12

Сущность	Атрибуты
Постановка	Код постановки, Название, Режиссер, Дата премьеры, Статус (подготовка/в репертуаре/архив)
Артист	Код артиста, ФИО, Амплуа, Дата рождения, Трудовой договор до
Роль	Код роли, Код постановки, Название роли, Код артиста, Тип (главная/второстепенная/ввод)
Расписание репетиций	Код репетиции, Код постановки, Дата и время, Зал, Цель репетиции
Костюм	Инвентарный номер костюма, Название, Эпоха, Материал, Код постановки, Состояние

Тема 13. Платформа для совместного создания научных статей.

Описание: управление статьями, соавторами, источниками, рецензиями и версиями.

Таблица 6.13 – Описание предметной области варианта №13

Сущность	Атрибуты
Научная статья	Код статьи, Название, Аннотация, Предметная область, Текущая версия, Статус (черновик/на рецензии/опубликована)
Соавтор	Код соавтора, ФИО, Ученая степень, Доля авторского вклада, Порядок в списке авторов
Исследовательский ин-	Код института, Полное название, Страна, Город

Сущность	Атрибуты
ститут	
Источник	Код источника, Библиографическое описание, Тип (книга/статья/сайт), Год издания
Рецензия	Код рецензии, Код статьи, Рецензент, Дата, Текст рецензии, Рекомендация (принять/доработать/отклонить)

Тема 14. Система бронирования коворкингов и переговорных.

Описание: управление рабочими местами, тарифами, клиентами и бронированиями.

Таблица 6.14 – Описание предметной области варианта №14

Сущность	Атрибуты
Рабочее место/Комната	Код места, Тип (открытое место/кабина/переговорная), Номер/название, Вместимость, Оснащение (проектор/доска)
Тариф	Код тарифа, Название, Тип (почасовой/дневной/месячный), Цена, Описание
Компания-клиент	Код компании, Название, ИНН, Контактное лицо, Номер договора
Бронирование	Код бронирования, Код места, Код компании, Дата и время начала, Дата и время окончания, Тема встречи, Статус (подтверждено/завершено/отменено)

Тема 15. База данных для благотворительного фонда.

Описание: учет благополучателей, доноров, программ помощи, пожертвований и отчетов.

Таблица 6.15 – Описание предметной области варианта №15

Сущность	Атрибуты
Благополучатель	Код получателя, ФИО или название семьи, Категория (малоимущая/многодетная/инвалид), Адрес, Описание потребности
Донор	Код донора, ФИО или название организации, Контактные данные, Предпочтительный тип помощи
Программа помощи	Код программы, Название, Цель, Бюджет программы, Дата начала, Дата окончания
Пожертвование	Код пожертвования, Код донора, Тип (денежное/натуральное), Сумма или описание товара, Дата, Код программы
Отчет об использовании	Код отчета, Код программы, Период, Израсходованная сумма, Оказанная помощь, Фотоотчет

Тема 16. Система контроля доступа на основе биометрии.

Описание: учет сотрудников, устройств считывания, событий доступа, графиков работы и отклонений.

Таблица 6.16 – Описание предметной области варианта №16

Сущность	Атрибуты
Сотрудник	Табельный номер, ФИО, Отдел, Должность, Биометрический шаблон (ссылка), График работы
Устройство считывания	Код устройства, Тип (турникет/дверь), Место установки, Режим работы
Событие доступа	Код события, Табельный номер, Код устройства, Тип события (вход/выход), Метка времени, Результат (успешно/отказано)
Отклонение	Код отклонения, Табельный номер, Дата, Тип (опоздание/ранний уход), Продолжительность, Причина

Тема 17. Платформа для каршеринга электромобилей.

Описание: учет электромобилей, зарядных станций, пользователей, поездок и платежей.

Таблица 6.17 – Описание предметной области варианта №17

Сущность	Атрибуты
Электромобиль	Государственный номер, Марка и модель, Год выпуска, Пробег, Текущий заряд батареи (%), Статус (свободен/арендован/на зарядке)
Зарядная станция	Код станции, Адрес, Количество разъемов, Типы разъемов, Статус (работает/неисправна)
Пользователь	Код пользователя, ФИО, Номер водительского удостоверения, Рейтинг, Баланс
Поездка	Код поездки, Государственный номер, Код пользователя, Время начала, Время окончания, Пробег поездки, Стоимость

Тема 18. База данных для производства подкастов.

Описание: учет подкаст-шоу, выпусков, участников, рекламных интеграций и задач для монтажа.

Таблица 6.18 – Описание предметной области варианта №18

Сущность	Атрибуты
Подкаст-шоу	Код шоу, Название, Ведущий, Тематика, Периодичность выхода
Выпуск (Эпизод)	Код выпуска, Код шоу, Номер выпуска, Название, Дата записи, Дата публикации, Длительность, Ссылка на аудиофайл
Участник	Код участника, ФИО, Роль (ведущий/гость), Контакт для связи
Рекламная интеграция	Код интеграции, Код выпуска, Рекламодатель, Условия интеграции, Стоимость

Тема 19. Система управления инцидентами и ИТ-активами.

Описание: учет ИТ-активов, сотрудников, заявок в службу поддержки и их решений.

Таблица 6.19 – Описание предметной области варианта №19

Сущность	Атрибуты
ИТ-актив	Инвентарный номер, Тип (ноутбук/монитор/лицензия ПО), Модель, Серийный номер, Код ответственного сотрудника, Дата ввода
Сотрудник-пользователь	Табельный номер, ФИО, Отдел, Должность, Контактный телефон
Заявка в поддержку	Номер заявки, Табельный номер заявителя, Дата и время создания, Категория проблемы (железо/ПО/сеть), Описание, Срочность, Статус (новая/в работе/решено)
Решение	Код решения, Номер заявки, Специалист, Дата решения, Описание решения, Потраченное время

Тема 20. Платформа для подписки на боксы с товарами.

Описание: учет подписчиков, вариантов боксов, товаров, ежемесячных поставок и отзывов.

Таблица 6.20 – Описание предметной области варианта №20

Сущность	Атрибуты
Подписчик	Код подписчика, ФИО, Адрес доставки, Электронная почта, Дата начала подписки
Вариант бокса	Код варианта, Название (например, "Косметический", "Книжный"), Цена в месяц, Описание
Товар в ассортименте	Артикул товара, Название, Категория, Производитель, Стоимость закупки

Сущность	Атрибуты
Ежемесячная поставка	Код поставки, Код подписчика, Код варианта бокса, Месяц поставки, Дата отправки, Трек-номер
Отзыв подписчика	Код отзыва, Код поставки, Текст отзыва, Оценка (1-5), Дата

Тема 21. База данных для агентства приключенческого туризма.

Описание: учет сложных туров (альпинизм, дайвинг, сафари), включая гидов, снаряжение, разрешения и медицинские справки клиентов.

Таблица 6.21 – Описание предметной области варианта №21

Сущность	Атрибуты
Тур	Код тура, Название, Уровень сложности, Локация, Дата начала, Дата окончания, Максимальный размер группы, Базовая цена, Необходимые разрешения
Гид	Код гида, ФИО, Номер сертификации, Специализация, Контакт для экстренной связи, Владение языками
Клиент	Код клиента, ФИО, Паспортные данные, Медицинские заметки, Контакт для экстренной связи, Уровень опыта
Бронирование тура	Код бронирования, Код тура, Код клиента, Дата бронирования, Оплаченная сумма, Опция страховки, Подписана расписка

Тема 22. Система мониторинга здоровья сельскохозяйственных животных.

Описание: отслеживание индивидуальных показателей здоровья, продуктивности, родословной и ветеринарных мероприятий.

Таблица 6.22 – Описание предметной области варианта №22

Сущность	Атрибуты
Животное	Идентификационный номер (чип), Номер бирки, Дата рождения, Пол, Порода, Номер матери, Номер отца, Статус (теленок/продуктивное/продано/пало)
Показатель здоровья	Код показателя, Номер животного, Дата измерения, Вес, Температура, Надой молока (для дойных), Примечания
Ветеринарное событие	Код события, Номер животного, Тип события (вакцинация/болезнь/отел/обрезка копыт), Дата события, ФИО ветеринара, Введенный препарат, Дозировка, Следующий срок
Рацион	Код рациона, Группа животных, Состав корма, Суточная норма (кг), Дата ввода

Тема 23. Платформа для краудсорсинга идей внутри компании.

Описание: сбор идей от сотрудников, их оценка экспертами, формирование проектных групп и отслеживание реализации.

Таблица 6.23 – Описание предметной области варианта №23

Сущность	Атрибуты
Идея	Код идеи, Код автора, Заголовок, Описание, Дата подачи, Категория, Текущая фаза (подана/на рассмотрении/одобрена/отклонена/в реализации)
Экспертная оценка	Код оценки, Код идеи, Код эксперта, Балл реализуемости, Балл влияния, Балл новизны, Комментарии, Дата оценки
Проект реализации	Код проекта, Код идеи, Код руководителя проекта, Выделенный бюджет, Дата начала, Дедлайн, Статус (планирование/исполнение/завершено)
Участник проекта	Код участия, Код проекта, Код сотрудника, Роль в проекте, Выделенное время (%)

Тема 24. База данных для студии разработки мобильных игр.

Описание: учет задач, ресурсов (арт, звук, код), версий сборок, багрепортов от тестеров.

Таблица 6.24 – Описание предметной области варианта №24

Сущность	Атрибуты
Игровой проект	Код проекта, Внутреннее кодовое название, Публичное название, Жанр, Целевые платформы, Код ведущего разработчика, Статус (препродакшн/альфа/бета/поддержка)
Задача	Код задачи, Код проекта, Код исполнителя, Заголовок, Описание, Тип задачи (арт/программирование/дизайн/тестирование), Приоритет, Плановые часы, Фактические часы, Статус (к выполнению/в работе/готово)
Арт-актив	Код актива, Код проекта, Имя файла, Тип актива (персонаж/интерфейс/фон), Версия, Код художника, Ссылка на исходник
Сборка	Код сборки, Код проекта, Номер версии, Дата сборки, Ссылка на скачивание, Примечания к выпуску, Стабильная сборка

Тема 25. Система управления качеством на производстве.

Описание: фиксация дефектов, контрольные точки, аудиты, корректирующие действия и связь с партиями продукции.

Таблица 6.25 – Описание предметной области варианта №25

Сущность	Атрибуты
Контрольная точка	Код точки, Идентификатор производственной линии, Название, Контролируемый параметр, Минимальное допустимое значение, Максимальное допустимое значение
Запись контроля	Код записи, Код точки, Код инспектора, Номер производственной партии, Измеренное значение, Время контроля, Результат (годен/брак)

Сущность	Атрибуты
Дефект	Код дефекта, Код записи, Код типа дефекта, Описание, Количество пораженных единиц
Корректирующее действие	Код действия, Код дефекта, Назначено отделу, Описание, Срок выполнения, Дата проверки эффективности

Тема 26. Платформа для найма и управления фрилансерами.

Описание: биржа для заказчиков и исполнителей с системой портфолио, отзывов и безопасных сделок.

Таблица 6.26 – Описание предметной области варианта №26

Сущность	Атрибуты
Фрилансер	Код фрилансера, ФИО, Специализация, Ссылка на портфолио, Почасовая ставка, Доступен с, Доступен до, Рейтинг
Заказ	Код заказа, Код клиента, Заголовок, Описание, Бюджет, Дедлайн, Требуемые навыки, Статус заказа (опубликован/в работе/на проверке/завершен/спор)
Отклик	Код отклика, Код заказа, Код фрилансера, Сопроводительное письмо, Предложенная сумма, Предложенные сроки, Дата подачи, Статус (рассматривается/принят/отклонен)
Этап работы	Код этапа, Код заказа, Название этапа, Описание, Срок выполнения, Ссылка на результат, Подтверждение клиента, Оплата по этапу

Тема 27. Система учета и обслуживания лифтового оборудования.

Описание: учет лифтов, графики ТО, история поломок, заявки от жильцов, работа аварийной службы.

Таблица 6.27 – Описание предметной области варианта №27

Сущность	Атрибуты
Лифт	Код лифта, Адрес здания, Номер подъезда, Производитель, Модель, Год установки, Дата последнего капремонта, Грузоподъемность, Статус (рабочий/на обслуживании/неисправен)
Плановое ТО	Код графика, Код лифта, Вид обслуживания, Плановая дата, Фактическая дата, Код техника, Затраченные часы, Замененные запчасти
Заявка на ремонт	Номер заявки, Код лифта, Кто сообщил, Дата и время сообщения, Описание проблемы, Срочность (низкая/средняя/высокая/аварийная), Статус заявки (принята/назначена/исправлена)
Аварийный вызов	Код вызова, Номер заявки, Код диспетчера, Код выездной бригады, Время выезда, Время прибытия, Количество эвакуированных

Тема 28. База данных для организации фестиваля уличной еды.

Описание: управление участниками (фудтраками), локациями на фестивале, расписанием, жюри, голосованием посетителей.

Таблица 6.28 – Описание предметной области варианта №28

Сущность	Атрибуты
Участник-фудтрак	Код фудтрака, Название бизнеса, Тип кухни, Владелец, Номер лицензии, Фото фудтрака, Соцсети
Локация на фестивале	Код локации, Зона (главная сцена/тихая семейная), Номер места, Наличие электричества, Наличие воды
Бронирование места	Код брони, Код фудтрака, Код локации, День фестиваля, Оплаченный взнос
Оценка жюри	Код оценки, Код фудтрака, Код члена жюри, Критерий (вкус/подача/оригинальность), Балл (1-10), Комментарии

Тема 29. Платформа для управления личными финансами.

Описание: учет доходов/расходов, категоризация, планирование бюджета, отслеживание финансовых целей, аналитика.

Таблица 6.29 – Описание предметной области варианта №29

Сущность	Атрибуты
Счет	Код счета, Код пользователя, Название счета, Тип счета (наличные/дебетовая карта/кредитная карта/вклад), Валюта, Начальный баланс, Текущий баланс
Транзакция	Код транзакции, Код счета, Сумма, Дата, Контрагент, Код категории, Описание, Тип (доход/расход/перевод)
Категория	Код категории, Код пользователя, Название, Код родительской категории, Тип (доход/расход)
Цель накопления	Код цели, Код пользователя, Название цели, Целевая сумма, Целевая дата, Текущая сумма, Код счета для автоперевода

Тема 30. Система сопровождения беженцев и мигрантов.

Описание: учет подопечных, их потребностей (жилье, документы, язык), волонтеров, мероприятий и оказанной помощи.

Таблица 6.30 – Описание предметной области варианта №30

Сущность	Атрибуты
Подопечный	Код подопечного, Регистрационный номер дела, ФИО, Дата рождения, Страна происхождения, Дата прибытия, Количество членов семьи, Текущий статус (нужно жилье/оформление документов/языковые курсы/трудоустроен)
Потребность	Код потребности, Код подопечного, Тип потребности (жилье/юридическая помощь/медицина/еда), Описание, Приоритет, Статус (открыта/в работе/решена)

Сущность	Атрибуты
Волонтер	Код волонтера, ФИО, Навыки, Владение языками, Доступные часы в неделю, Код назначенной потребности
Предоставленная услуга	Код услуги, Код потребности, Код волонтера, Дата оказания, Описание, Результат
Партнерская организация	Код организации, Название, Контактное лицо, Предоставляемые услуги, Действие соглашения до

Тема 31. Система управления арендой яхт и катеров.

Описание: учет плавсредств, владельцев, клиентов, бронирования, экипажа и дополнительных услуг.

Таблица 6.31 – Описание предметной области варианта №31

Сущность	Атрибуты
Плавсредство	Код плавсредства, Название, Тип (яхта/катер/парусник), Вместимость (чел.), Длина, Год постройки, Порт приписки, Стоимость аренды в сутки, Статус (доступно/арендовано/на обслуживании)
Владелец	Код владельца, ФИО или название компании, Контактный телефон, Реквизиты для выплат
Клиент	Код клиента, ФИО, Паспортные данные, Опыт управления (да/нет), Номер сертификата (если есть)
Бронирование	Код бронирования, Код плавсредства, Код клиента, Дата начала, Дата окончания, Общая стоимость, Статус (подтверждено/завершено/отменено)
Член экипажа	Код сотрудника, ФИО, Должность (капитан/матрос/повар), Контактные данные, Опыт работы (лет)

Тема 32. Платформа для обмена жильем (хоумсвоппинг).

Описание: система для временного обмена жилыми помещениями между пользователями, включая проверку репутации и договоренностей.

Таблица 6.32 – Описание предметной области варианта №32

Сущность	Атрибуты
Пользователь	Код пользователя, ФИО, Город проживания, Репутация (баллы), Дата регистрации, Верифицирован (да/нет)
Жилье	Код жилья, Код владельца, Тип (квартира/дом/вилла), Адрес, Описание, Вместимость, Удобства, Доступные даты
Предложение обмена	Код предложения, Код жилья (предлагаемого), Код пользователя (ищущего), Желаемые даты, Статус (активно/забронировано/отменено)
Бронирование обмена	Код бронирования, Код предложения, Код жилья (искомого), Дата подтверждения, Условия обмена, Статус (подтверждено/завершено/отменено)
Отзыв об обмене	Код отзыва, Код бронирования, Автор отзыва, Текст отзыва, Оценка (1-5), Дата

Тема 33. База данных для клининговой компании.

Описание: учет клиентов, объектов уборки, сотрудников, графика работ и используемых средств.

Таблица 6.33 – Описание предметной области варианта №33

Сущность	Атрибуты
Клиент	Код клиента, Название компании или ФИО, Контактное лицо, Телефон, Адрес, Тип объекта (офис/квартира/торговый центр)
Объект уборки	Код объекта, Код клиента, Адрес объекта, Площадь (кв. м.), Тип уборки (ежедневная/генеральная/после ремонта), Частота уборки

Сущность	Атрибуты
Сотрудник клининга	Код сотрудника, ФИО, Должность (уборщик/менеджер), График работы, Контактный телефон
График работ	Код графика, Код объекта, Код сотрудника, Дата и время уборки, Фактическое время выполнения, Статус (запланировано/выполнено/отменено)
Средство для уборки	Код средства, Название, Производитель, Расход на кв. м., Остаток на складе

Тема 34. Система управления лояльностью в розничной сети.

Описание: учет покупателей, карт лояльности, бонусных баллов, транзакций и акций.

Таблица 6.34 – Описание предметной области варианта №34

Сущность	Атрибуты
Покупатель	Код покупателя, ФИО, Номер телефона, Электронная почта, Дата регистрации
Карта лояльности	Номер карты, Код покупателя, Дата выдачи, Текущий баланс баллов, Статус (активна/заблокирована)
Транзакция	Код транзакции, Номер карты, Дата и время, Сумма покупки, Начислено баллов, Списано баллов
Акция	Код акции, Название, Описание, Условия, Дата начала, Дата окончания
Начисление баллов	Код начисления, Код транзакции, Количество баллов, Тип начисления (покупка/акция), Дата

Тема 35. Платформа для планирования и ухода за садом/огородом.

Описание: учет растений, участков, графика работ (полив, удобрение), погодных условий и урожая.

Таблица 6.35 – Описание предметной области варианта №35

Сущность	Атрибуты
Растение	Код растения, Название, Тип (овощ/фрукт/цветок), Сорт, Дата посадки, Период созревания (дней)
Участок	Код участка, Название грядки, Площадь, Тип почвы, Освещенность
График ухода	Код графика, Код растения, Тип работы (полив/прополка/удобрение), Периодичность (дни), Последнее выполнение
Погодные условия	Код записи, Дата, Температура, Осадки (мм), Влажность воздуха (%), Влияние на график ухода
Урожай	Код урожая, Код растения, Дата сбора, Количество, Качество (отличное/хорошее/удовлетворительное)

Тема 36. База данных для службы доставки еды (рестораны-партнеры).

Описание: учет ресторанов, курьеров, заказов, клиентов и отзывов.

Таблица 6.36 – Описание предметной области варианта №36

Сущность	Атрибуты
Ресторан-партнер	Код ресторана, Название, Кухня, Адрес, Контактный телефон, Время работы
Курьер	Код курьера, ФИО, Телефон, Транспорт (пеший/велосипед/авто), Статус (свободен/занят)
Заказ	Код заказа, Код клиента, Код ресторана, Состав заказа, Сумма, Адрес доставки, Время заказа, Статус (принят/готовится/у курьера/доставлен)
Клиент	Код клиента, ФИО, Телефон, Адрес, Предпочтения (аллергии и т.п.)

Сущность	Атрибуты
Отзыв	Код отзыва, Код заказа, Оценка еды (1-5), Оценка доставки (1-5), Комментарий

Тема 37. Система управления арендой строительной техники.

Описание: учет единиц техники, владельцев, клиентов, бронирования и технического обслуживания.

Таблица 6.37 – Описание предметной области варианта №37

Сущность	Атрибуты
Строительная техника	Код техники, Тип (экскаватор/бетономешалка/подъемник), Модель, Год выпуска, Грузоподъемность, Стоимость аренды в час, Статус (доступна/арендована/на ремонте)
Владелец	Код владельца, Название компании, Контактное лицо, Телефон
Клиент	Код клиента, Название компании, ИНН, Контактное лицо, Телефон
Бронирование	Код бронирования, Код техники, Код клиента, Дата начала, Дата окончания, Место использования, Общая стоимость, Статус (подтверждено/завершено/отменено)
Техническое обслуживание	Код обслуживания, Код техники, Дата, Вид работ, Затраты, Исполнитель

Тема 38. Платформа для организации волонтерских акций.

Описание: учет акций, организаторов, волонтеров, необходимых ресурсов и отчетов.

Таблица 6.38 – Описание предметной области варианта №38

Сущность	Атрибуты
Акция	Код акции, Название, Описание, Дата и время, Место, Необходимое количество волонтеров, Статус (планируется/идет/завершена)
Организатор	Код организатора, Название организации, Контактное лицо, Телефон
Волонтер	Код волонтера, ФИО, Телефон, Навыки, Предпочтительные типы акций
Участие	Код участия, Код акции, Код волонтера, Статус участия (заявлен/подтвержден/принял участие), Часы участия
Необходимый ресурс	Код ресурса, Код акции, Название ресурса, Требуемое количество, Единица измерения

Тема 39. База данных для сервиса по ремонту бытовой техники.

Описание: учет клиентов, видов техники, мастеров, заявок и запчастей.

Таблица 6.39 – Описание предметной области варианта №39

Сущность	Атрибуты
Клиент	Код клиента, ФИО, Адрес, Телефон
Вид техники	Код вида, Название (холодильник/стиральная машина и т.д.), Производитель, Типовые неисправности
Мастер	Код мастера, ФИО, Специализация, Контактный телефон, График работы
Заявка на ремонт	Код заявки, Код клиента, Код вида техники, Описание проблемы, Дата и время вызова, Код мастера, Статус (принята/в работе/завершена)
Использованная запчасть	Код использования, Код заявки, Название запчасти, Количество, Стоимость

Тема 40. Система управления библиотекой электронных книг.

Описание: учет электронных книг, авторов, читателей, подписок и истории чтения.

Таблица 6.40 – Описание предметной области варианта №40

Сущность	Атрибуты
Электронная книга	Код книги, Название, Автор, Жанр, Год издания, Формат файла, Размер файла, Ссылка на файл
Автор	Код автора, ФИО, Биография, Страна
Читатель	Код читателя, ФИО, Электронная почта, Дата регистрации, Тип подписки (бесплатный/премиум)
Подписка	Код подписки, Код читателя, Тип подписки, Дата начала, Дата окончания, Стоимость
История чтения	Код записи, Код читателя, Код книги, Дата начала чтения, Дата окончания чтения, Прогресс (%)

Тема 41. Платформа для организации карпулинга (совместных поездок).

Описание: учет водителей, пассажиров, маршрутов, поездок и отзывов.

Таблица 6.41 – Описание предметной области варианта №41

Сущность	Атрибуты
Водитель	Код водителя, ФИО, Марка и модель автомобиля, Госномер, Вместимость, Контактный телефон
Пассажир	Код пассажира, ФИО, Контактный телефон, Рейтинг
Маршрут	Код маршрута, Пункт отправления, Пункт назначения, Примерное время в пути, Расстояние

Сущность	Атрибуты
Поездка	Код поездки, Код водителя, Код маршрута, Дата и время отправления, Стоимость с человека, Свободные места, Статус (запланирована/завершена/отменена)
Бронирование места	Код бронирования, Код поездки, Код пассажира, Количество мест, Статус (подтверждено/отменено)

Тема 42. База данных для студии танцев и фитнеса.

Описание: учет тренеров, клиентов, абонементов, расписания занятий и посещений.

Таблица 6.42 – Описание предметной области варианта №42

Сущность	Атрибуты
Тренер	Код тренера, ФИО, Специализация (танцы/йога/фитнес), Контактный телефон, График работы
Клиент	Код клиента, ФИО, Дата рождения, Контактный телефон, Уровень подготовки
Абонемент	Код абонемента, Код клиента, Тип (разовый/месячный/годовой), Дата начала, Дата окончания, Количество занятий
Занятие	Код занятия, Код тренера, Тип занятия, Время начала, Продолжительность, Зал, Максимальное количество участников
Посещение	Код посещения, Код клиента, Код занятия, Дата, Отметка о присутствии

Тема 43. Система управления складом товаров для интернет-магазина.

Описание: учет товаров, поставщиков, приходных/расходных ордеров, инвентаризаций.

Таблица 6.43 – Описание предметной области варианта №43

Сущность	Атрибуты
Товар	Код товара, Название, Категория, Производитель, Единица измерения, Минимальный остаток
Поставщик	Код поставщика, Название компании, Контактное лицо, Телефон, Адрес
Приходный ордер	Номер ордера, Код поставщика, Дата, Общая сумма, Статус (ожидается/получено)
Расходный ордер	Номер ордера, Код клиента (из заказа), Дата, Общая сумма, Статус (собран/отправлен)
Инвентаризация	Код инвентаризации, Дата проведения, Ответственный, Результат (излишек/недостача)

Тема 44. Платформа для организации фотосессийю

Описание: учет фотографов, клиентов, локаций, заказов и готовых работ.

Таблица 6.44– Описание предметной области варианта №44

Сущность	Атрибуты
Фотограф	Код фотографа, ФИО, Специализация (свадьба/портрет/пейзаж), Контактные данные, Портфолио (ссылка)
Клиент	Код клиента, ФИО, Контактный телефон, Электронная почта
Локация	Код локации, Название, Адрес, Тип (студия/природа/интерьер), Стоимость аренды в час
Заказ	Код заказа, Код клиента, Код фотографа, Код локации, Дата и время, Продолжительность, Общая стоимость, Статус (запланировано/выполнено/отменено)

Сущность	Атрибуты
Готовая работа	Код работы, Код заказа, Ссылка на альбом, Дата сдачи, Количество фотографий

Тема 45. База данных для службы психологической поддержки.

Описание: учет психологов, клиентов, сеансов, методик и прогресса.

Таблица 6.45 – Описание предметной области варианта №45

Сущность	Атрибуты
Психолог	Код психолога, ФИО, Специализация, Образование, Контактные данные, График приема
Клиент	Код клиента, ФИО, Контактный телефон, Возраст, Причина обращения
Сеанс	Код сеанса, Код психолога, Код клиента, Дата и время, Продолжительность, Темы обсуждения, Рекомендации
Методика	Код методики, Название, Описание, Для каких проблем применяется
Прогресс клиента	Код записи, Код клиента, Дата оценки, Самочувствие (баллы), Заметки психолога

Тема 46. Система управления арендой музыкальных инструментов.

Описание: учет инструментов, владельцев, клиентов, бронирования и состояния инструментов.

Таблица 6.46 – Описание предметной области варианта №46

Сущность	Атрибуты
Музыкальный инструмент	Код инструмента, Тип (гитара/фортепиано/скрипка), Модель, Год изготовления, Состояние, Стоимость аренды в день, Статус (доступен/арендован/на ремонте)
Владелец	Код владельца, ФИО или название магазина, Контактный телефон,

Сущность	Атрибуты
	Адрес
Клиент	Код клиента, ФИО, Паспортные данные, Контактный телефон, Опыт игры
Бронирование	Код бронирования, Код инструмента, Код клиента, Дата начала, Дата окончания, Залог, Общая стоимость, Статус (активно/завершено/отменено)
Проверка состояния	Код проверки, Код инструмента, Дата, Обнаруженные дефекты, Кто проверил

Тема 47. Платформа для изучения иностранных языков с репетиторами.

Описание: учет репетиторов, учеников, расписания занятий, материалов и прогресса.

Таблица 6.47 – Описание предметной области варианта №47

Сущность	Атрибуты
Репетитор	Код репетитора, ФИО, Преподаваемый язык, Уровень, Контактные данные, Стоимость часа
Ученик	Код ученика, ФИО, Изучаемый язык, Текущий уровень, Цель обучения
Занятие	Код занятия, Код репетитора, Код ученика, Дата и время, Продолжительность, Тема, Ссылка на запись (если онлайн)
Учебный материал	Код материала, Название, Тип (текст/видео/упражнение), Язык, Уровень сложности, Ссылка на ресурс
Прогресс ученика	Код записи, Код ученика, Дата тестирования, Уровень, Навыки (аудирование/говорение/чтение/письмо)

Тема 48. База данных для автосервиса.

Описание: учет клиентов, автомобилей, заказов-нарядов, выполненных работ и запчастей.

Таблица 6.48 – Описание предметной области варианта №48

Сущность	Атрибуты
Клиент	Код клиента, ФИО, Контактный телефон, Адрес
Автомобиль	Код автомобиля, Код клиента, Марка, Модель, Год выпуска, Гос-номер, VIN
Заказ-наряд	Номер заказа, Код автомобиля, Дата приема, Описание проблемы, Статус (принят/в работе/готов)
Выполненная работа	Код работы, Номер заказа, Название работы, Стоимость нормочаса, Затраченное время, Общая стоимость
Запчасть	Код запчасти, Название, Производитель, Цена, Количество на складе

Тема 49. Система управления проектами в рекламном агентстве.

Описание: учет клиентов, проектов, задач, сотрудников и бюджетов.

Таблица 6.49 – Описание предметной области варианта №49

Сущность	Атрибуты
Клиент	Код клиента, Название компании, Контактное лицо, Телефон, Электронная почта
Проект	Код проекта, Название, Код клиента, Менеджер, Дата начала, Дедлайн, Бюджет, Статус (планирование/в работе/сдан)
Задача	Код задачи, Код проекта, Исполнитель, Описание, Приоритет, Дата начала, Дата окончания, Статус (к выполнению/в работе/готово)

Сущность	Атрибуты
Сотрудник	Код сотрудника, ФИО, Должность, Отдел, Контактные данные
Бюджет проекта	Код записи бюджета, Код проекта, Статья расходов, Плановая сумма, Фактическая сумма

Тема 50. Платформа для организации настольных игротек.

Описание: учет игр, игротек, мероприятий, участников и отзывов.

Таблица 6.50 – Описание предметной области варианта №50

Сущность	Атрибуты
Настольная игра	Код игры, Название, Издатель, Количество игроков, Время партии, Возрастное ограничение, Сложность
Игротека	Код игротеки, Название, Адрес, Контактный телефон, Время работы
Мероприятие	Код мероприятия, Код игротеки, Дата и время, Тематика, Код ведущего, Максимальное количество участников
Участник мероприятия	Код участия, Код мероприятия, Код посетителя, Дата регистрации, Статус (зарегистрирован/принял участие)
Посетитель	Код посетителя, ФИО, Контактный телефон, Предпочтения по играм

СПИСОК РЕКОМЕНДУЕМЫХ ИСТОЧНИКОВ

1. Нестеров, С. А. Базы данных : учебник и практикум для среднего профессионального образования / С. А. Нестеров. — 2-е изд. — Москва : Издательство Юрайт, 2025. — 258 с. — (Профессиональное образование). — ISBN 978-5-534-18087-9. — Текст : электронный // Образовательная платформа Юрайт [сайт].
2. Стружкин, Н. П. Базы данных: проектирование : учебник для среднего профессионального образования / Н. П. Стружкин, В. В. Годин. — Москва : Издательство Юрайт, 2026. — 477 с. — (Профессиональное образование). — ISBN 978-5-534-11635-9. — Текст : электронный // Образовательная платформа Юрайт [сайт]. ный
3. Советов, Б. Я. Базы данных : учебник для среднего профессионального образования / Б. Я. Советов, В. В. Цехановский, В. Д. Чертовской. — 4-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2026. — 403 с. — (Профессиональное образование). — ISBN 978-5-534-18784-7. — Текст : электронный // Образовательная платформа Юрайт [сайт].
4. Стружкин, Н. П. Базы данных: проектирование. Практикум : учебник для среднего профессионального образования / Н. П. Стружкин, В. В. Годин. — Москва : Издательство Юрайт, 2025. — 291 с. — (Профессиональное образование). — ISBN 978-5-534-08140-4. — Текст : электронный // Образовательная платформа Юрайт [сайт]. —
5. Гордеев, С. И. Организация баз данных в 2 ч. Часть 1 : учебник для среднего профессионального образования / С. И. Гордеев, В. Н. Волошина. — 2-е изд., испр. и доп. — Москва : Издательство Юрайт, 2025. — 310 с. — (Профессиональное образование). — ISBN 978-5-534-11626-7. — Текст : электронный // Образовательная платформа Юрайт [сайт].
6. Гордеев, С. И. Организация баз данных в 2 ч. Часть 2 : учебник для среднего профессионального образования / С. И. Гордеев, В. Н. Волошина. — 2-е изд., испр. и доп. — Москва : Издательство Юрайт, 2025. — 513 с. — (Профессиональное образование). — ISBN 978-5-534-11625-0. — Текст : электронный // Образовательная платформа Юрайт [сайт].
7. Илюшечкин, В. М. Основы использования и проектирования баз данных : учебник для среднего профессионального образования / В. М. Илюшечкин. — Москва : Издательство Юрайт, 2025. — 213 с. — (Профессиональное образование). — ISBN 978-5-534-01283-5. — Текст : электронный // Образовательная платформа Юрайт [сайт].

Учебное издание

ОСНОВЫ ПРОЕКТИРОВАНИЯ БАЗ ДАННЫХ. ЧАСТЬ 1

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Ростовский государственный университет путей сообщения» (ФГБОУ ВО РГУПС)

Адрес университета:

344038, г. Ростов н/Д, пл. Ростовского Стрелкового Полка Народного Ополчения, д. 2, www.rgups.ru