

Н. П. Сидорова

# БАЗЫ ДАННЫХ

## ПРАКТИКУМ ПО ПРОЕКТИРОВАНИЮ РЕЛЯЦИОННЫХ БАЗ ДАННЫХ

Учебное пособие





Государственное бюджетное образовательное учреждение высшего образования  
Московской области

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ**

**ИНСТИТУТ ТЕХНИКИ И ЦИФРОВЫХ ТЕХНОЛОГИЙ**

**ФАКУЛЬТЕТ**

**ИНФОКОММУНИКАЦИОННЫХ СИСТЕМ И ТЕХНОЛОГИЙ**

**КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И УПРАВЛЯЮЩИХ СИСТЕМ**

**Н. П. Сидорова**

# **БАЗЫ ДАННЫХ**

## **ПРАКТИКУМ ПО ПРОЕКТИРОВАНИЮ РЕЛЯЦИОННЫХ БАЗ ДАННЫХ**

*Учебное пособие*



**Москва  
Берлин  
2020**

УДК 004.65(076)  
ББК 32.972.34я7+16.3я7  
С 34

**Рецензенты:**

*Хорев П. Б.*, профессор кафедры Прикладной математики НИУ МЭИ,  
кандидат технических наук, доцент;

*Погодин А. В.*, кандидат технических наук, доцент

**Сидорова, Н. П.**

С 34      Базы данных: практикум по проектированию реляционных баз данных :  
учебное пособие / Н. П. Сидорова. — Москва ; Берлин : Директ-Медиа,  
2020. — 92 с.

ISBN 978-5-4499-0799-8

Учебное пособие включает восемь практических работ, отражающих основные этапы проектирования реляционных баз данных: инфологическое проектирование, логическое проектирование на основе реляционной модели данных, реализацию базы данных средствами СУБД. Каждая работа содержит необходимые теоретические сведения, используемые для выполнения работы. Построение моделей баз данных предполагает применение CASE-средств. Для реализации базы данных используется СУБД Access.

Практикум предназначен студентам, обучающимся по направлениям подготовки «Прикладная информатика», «Информационные системы и технологии» очной и заочной форм обучения. Практические задания, включенные в учебное пособие, будут полезны студентам и других направлений подготовки при изучении вопросов, связанных с проектированием и реализацией реляционных баз данных.

УДК 004.65(076)  
ББК 32.972.34я7+16.3я7

ISBN 978-5-4499-0799-8

© Сидорова Н. П., текст, 2020

© Издательство «Директ-Медиа», оформление, 2020

## ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ .....	5
Практическая работа 1. ПРОЕКТИРОВАНИЕ ER-МОДЕЛИ БД.....	8
Практическая работа 2. РАЗРАБОТКА РЕЛЯЦИОННОЙ МОДЕЛИ БД.....	26
Практическая работа 3. ПРОЕКТИРОВАНИЕ ПРАВИЛ ЦЕЛОСТНОСТИ БД И ФИЗИЧЕСКОЙ МОДЕЛИ БД.....	33
Практическая работа 4. СОЗДАНИЕ ТАБЛИЦ БД В СУБД ACCESS.....	42
Практическая работа 5. СОЗДАНИЕ ЗАПРОСОВ В СУБД ACCESS.....	51
Практическая работа 6. СОЗДАНИЕ ОТЧЕТОВ В СУБД ACCESS.....	63
Практическая работа 7. СОЗДАНИЕ ИНТЕРФЕЙСА К БД ACCESS .....	69
Практическая работа 8. ЦЕЛОСТНОСТЬ ДАННЫХ В БАЗЕ ДАННЫХ .....	79
ЗАКЛЮЧЕНИЕ .....	84
ЛИТЕРАТУРА .....	85
Приложение. ВАРИАНТЫ ПРЕДМЕТНОЙ ОБЛАСТИ ДЛЯ ПРАКТИЧЕСКИХ ЗАДАНИЙ .....	86

## ВВЕДЕНИЕ

Широкое применение информационных систем вызывает большой интерес к вопросам проектирования баз данных (БД), которые являются их основой [4, 5, 11]. Качество полученной БД во многом определяется результатами её проектирования. В настоящее время широко используются БД, основанные на реляционной модели данных – реляционные базы данных (РБД). Проектирование модели РБД предполагает определение набора таблиц БД и формирование структуры каждой таблицы. Модели БД должны удовлетворять следующим свойствам [5, 6, 10]:

- представление семантики предметной области, которое предполагает отражение в модели реально существующих в предметной области объектов;
- поддержка эффективного управления данными, связанного с их хранением во внешней памяти и обработкой;
- устранение избыточности данных, которая обусловлена дублированием данных в БД;
- поддержка целостности данных.

Модель БД играет особую роль в процессе проектирования, т.к. представляет информационную модель предметной области [4, 7]. Понятие предметной области, как и многие понятия, используемые в информатике, не имеет точного формального определения. Однако оно широко применяется для того, чтобы ограничить область использования данных из БД. Выделение предметной области БД позволяет сформировать перечень тех сведений об объектах реального мира (сущностей), данные о которых будет содержать БД. Анализ предметной области БД позволяет выделить в ней информационные объекты [7], сведения о которых будут храниться и обрабатываться в БД.

Построение модели БД является непростой задачей, поэтому её решение разбивается на несколько этапов. На каждом этапе разрабатываются модели различного уровня:

- 1) **семантическая модель** предметной области формируется на основе основных понятий, используемых в ней, определяет уровень инфологического моделирования;
- 2) **логическая модель** строится в терминах абстрактных моделей данных и формирует концептуальное представление БД;

3) **физическая модель** определяет, как будут храниться данные во внешней памяти, какие управляющие элементы необходимо предусмотреть для обеспечения эффективной работы с ней.

Такой подход определяет следующие этапы проектирования БД.

1. **Проектирование инфологической модели.** Оно выполняется на основе анализа предметной области. Анализ предметной области позволяет выделить основные информационные объекты, правила их взаимодействия, информационные процессы. Как правило, результатом анализа является формализованное описание процессов предметной области, которое служит основой для построения инфологической модели.

2. **Разработка логической модели** связана, в первую очередь, с выбором типа логической модели.

3. **Проектирование физической модели** позволяет определить способы размещения элементов логической модели во внешней памяти таким образом, чтобы их хранение и управление ими было оптимально.

Для проектирования БД в настоящее время широко применяются специальные программные средства, автоматизирующие разработку моделей различного уровня – CASE-средства (Computer Aided Software Engineering – компьютерные средства разработки программного обеспечения). CASE-средства позволяют применять современные подходы к процессу проектирования программного обеспечения, применять методологию нисходящего проектирования:

- широко использовать автоматизацию для разработки, визуализации и документирования моделей различного уровня;
- создавать библиотеку (репозиторий) моделей, в которой хранятся их описания, на основе репозитория можно создавать новые модели посредством изменения или доработки уже созданных;
- организовывать одновременную работу с моделью БД нескольких разработчиков;
- автоматизировать стандартные процедуры, связанные с документированием, проверкой, интеграцией модели БД.

CASE-средства позволяют решать следующие задачи в процессе проектирования БД.

1. **Разработка моделей процессов обработки информации** в предметной области. Такие модели, как правило, представляются в виде структурных моделей. Они позволяют представить процессы в

виде иерархических диаграмм, задающих функции процесса. Анализ полученных моделей позволяет сформировать состав информационных объектов и определить функции обработки данных. Эта информация является основой для разработки инфологической модели БД.

2. Разработка инфологической модели базируется на использовании модели «сущность-связь». Она позволяет конкретизировать набор характеристик каждого объекта и определить связи между ними.

3. Преобразование модели «сущность-связь» в реляционную модель БД.

4. Автоматическое создание описания БД на языке SQL выбранной СУБД.

5. Автоматическое создание описаний на языке программирования программных компонентов (описание структуры таблиц БД, создание индексов и триггеров БД, экранные формы).

Для анализа информационных процессов могут быть использованы такие программные продукты, как AllFusion Process Modeler, Microsoft Visio, Modelio Open Source.

Для проектирования моделей БД при выполнении практических работ используется CASE-средств AllFusion ERwin Data Modeler или Microsoft Visio. Физическая модель при выполнении практических заданий основывается на модели выбранной СУБД. Для создания БД на основе разработанных моделей при выполнении практических заданий используется СУБД Access.

Практические работы 1–3 позволяют приобрести навыки использования CASE-средств построения моделей БД. В практической работе 1 студенты получают навыки проектирования ER-модели заданной предметной области. В практической работе 2 проводится проектирование реляционной модели БД. В практической работе 3 приобретаются навыки проектирования статических и ссылочных правил целостности для реляционной модели.

Практические работы 4–8 позволяют приобрести навыки использования СУБД Access для реализации БД. В практической работе 4 изучается технология переноса разработанной модели БД в СУБД Access. Практическая работа 5 посвящена приобретению навыков разработки QBE-запросов в СУБД Access. Практическое занятие 6 позволяет приобрести навыки использования инструментальных средств СУБД Access для разработки интерфейса с объектами БД.

## **Практическая работа 1**

### **ПРОЕКТИРОВАНИЕ ER-МОДЕЛИ БД**

**Цель работы** – приобретение навыков разработки ER-модели заданной предметной области с помощью CASE-средств.

#### **1. Теоретические основы**

Для концептуального описания предметной области при проектировании БД используют семантические модели. Они позволяют, с одной стороны, сохранить смысловое описание предметной области, а с другой – представить предметную область в виде формализованной модели. Такие модели используются на этапе инфологического моделирования. В них представляются те смысловые понятия, которые существуют в предметной области и будут отражаться в БД.

Для решения проблемы представления семантики предметной области было создано несколько моделей. Наиболее популярной является модель, предложенная П. Ченом [2, 3], которая известна как модель «сущность-связь» (ER-модель). Её преимуществом является относительная простота, которая, тем не менее, позволяет разрабатывать большие и сложные семантические модели БД. Следует отметить, что существует большое количество CASE-средств, которые включают широкий набор инструментов для работы с такой моделью. При выполнении практической работы используется ER-модель.

В основе построения ER-модели лежит представление предметной области в виде набора взаимосвязанных сущностей. Поэтому основными элементами модели являются сущность (Entity) и связь (Relationship). Каждый реально существующий объект предметной области обладает набором характеристик (свойств), которые необходимо представить в информационной модели. Для представления таких объектов в ER-модели используется сущность. Свойства сущности описывают характеристики реально существующего объекта, а связи определяют смысловые ассоциации, которые можно выделить между сущностями в предметной области.

**Сущность** может определяться как для реально существующего объекта, так и для некоторого абстрактного понятия предметной области [5, 6, 7, 10, 11]. Конкретные значения характеристик каждого объекта будут храниться в БД. Реальный объект (например,



**Преподаватель**) задаёт физический объект, в то время как абстрактный объект (например **Лекция**) определяет некоторое абстрактное понятие, связанное с реальным объектом в предметной области. Все объекты предметной области, для которых можно определить общий набор характеристик, необходимо объединить в одну сущность-понятие (далее сущность). Характеристики сущности-понятия называются атрибутами. Они должны определять существенные для заданной предметной области свойства сущности и иметь уникальные (неповторяющиеся) имена. Задавая имена сущностей и атрибутов, следует придерживаться терминологии предметной области. При этом надо учитывать, что наименование характеристик разных сущностей в предметной области могут совпадать (например, свойство **Фамилия** можно определить для различных сущностей: преподаватель, студент, автор учебника, клиент и т.п.). Желательно, однако, определять имена атрибутов разных сущностей уникальными, это позволит легче понимать модель БД.

**Экземпляр сущности** определяет отдельный объект из этого класса и задаётся конкретными значениями его атрибутов.

При определении набора атрибутов сущности необходимо выделить идентифицирующий атрибут, значение которого позволяет определить конкретную сущность-экземпляр.

В ER-модели можно использовать сущности различных типов:

- сильные и слабые сущности,
- простые и сложные сущности.

Существование сильной сущности не зависит от наличия каких-либо других сущностей. Слабые сущности могут существовать только при наличии связи с некоторой сильной сущностью. Например, слабая сущность **Кредит** зависит от существования сильной сущности **Клиент**.

Простая сущность содержит все свои характеристики. Для сложного объекта предметной области такие характеристики могут быть определены в нескольких сущностях. В этом случае для представления сложного объекта в ER-модели используется сложная сущность.

Вторым элементом в ER-модели является связь. **Связь** в самом общем смысле определяет ассоциацию между сущностями. Связь задаёт смысловое взаимодействие сущностей. В ER-модели следует использовать только бинарные связи, т.е. связи только между двумя

сущностями, которые можно отнести к одному из допустимых типов связей [4, 5, 6].

Связь **ОДИН-К-ОДНОМУ** (1:1) можно задать между сущностями тогда, когда в каждый момент времени один экземпляр родительской сущности (например **Преподаватель**) ассоциируется в предметной области с 1 (или 0) экземпляром дочерней сущности (например **Кафедра**).

Связь **ОДИН-КО-МНОГИМ** (1:M) определяется между сущностями в том случае, когда один экземпляр родительской сущности связан с несколькими экземплярами дочерней сущности, например, между сущностями **Преподаватель** и **Дисциплина**.

Связь **МНОГИЕ-КО-МНОГИМ** (M:N) определяет такое взаимодействие сущностей, при котором некоторое множество экземпляров родительской сущности связано с несколькими экземплярами дочерней сущности. Например, такую связь можно определить между сущностями **Товар** и **Заказ** в том случае, если в один заказ можно включить несколько товаров.

Для однозначной интерпретации ER-модели все её элементы должны иметь описание:

- сущность описывается именем, желательно также определить смысловую интерпретацию сущности, которая определяет её назначение в модели;
- атрибут описывается своим именем, типом допустимых значений, описанием особенностей использования атрибута, также желательно привести краткую смысловую интерпретацию атрибута, которая позволит пояснить его появление в структуре сущности;
- связь описывается именем, которое определяет его смысловую нагрузку, и типом.

Одним из обязательных свойств БД является **целостность** данных. Целостность данных описывается набором правил, которое позволяет учесть специфику работы с данными в предметной области. Это позволяет защитить данные от потери или неправильных действий. В общем случае выделение смысловых правил целостности не является обязательным элементом проектирования ER-модели, но их выделение позволит в дальнейшем построить более качественную БД.

Рассмотрим процесс разработки ER-модели для конкретного примера предметной области: работу курсов дополнительного обучения, в которой организуется работа по проведению групповых

занятий по различным дисциплинам. В этой предметной области можно выделить следующие сущности: **Студент**, **Группа**, **Дисциплина**, **Преподаватель**.

Сущность **Студент** может быть задана следующим набором атрибутов: фамилия студента, номер зачётной книжки, дата рождения. Для конкретного студента эти атрибуты задаются конкретными значениями, например, Фамилия: Смирнов; номер зачётной книжки: 12345; дата рождения 12.10.2000.

Для сущности **Дисциплина** определим следующие атрибуты: название, уровень подготовки, вид итогового документа.

Для сущности **Группа** выделим следующие атрибуты: номер группы, дата формирования.

Задавая связи между сущностями, необходимо основываться на тех правилах, которые существуют в предметной области. Например, если один студент может учиться только в одной группе, то между сущностью **Студент** (родительская сущность) и сущностью **Группа** (дочерняя сущность) определяется связь типа 1:1 (рис. 1.1).

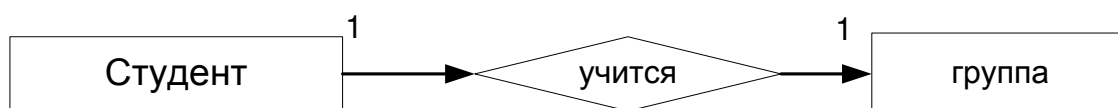


Рисунок 1.1 – Пример связи 1:1

С другой стороны, т.к. одна группа состоит из нескольких студентов, то между родительской сущностью **Группа** и дочерней сущностью **Студент** определена связь 1:M (рис. 1.2).

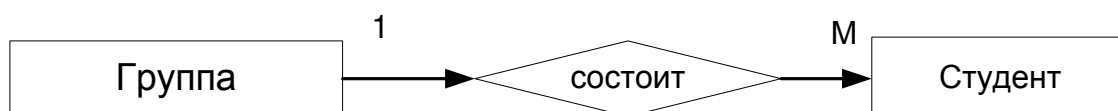


Рисунок 1.2 – Пример связи 1:M

Каждый студент может изучать произвольное количество дисциплин, поэтому между сущностью **Студент** и **Дисциплина** следует определить связь типа M:N (рис. 1.3).

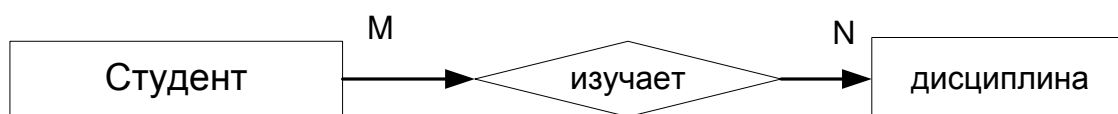


Рисунок 1.3 – Пример связи N:M

## 2. CASE-средства проектирования инфологической модели

### 2.1. Проектирование ER-модели в среде AllFusion ERwin Data Modeler

CASE-средство AllFusion ERwin Data Modeler (ERwin DM) широко применяется для проектирования БД [8, 9]. Оно позволяет создать модели различного типа (рис. 1.4): **Logical**, **Physical**, **Logical/Physical**, **Match template**.

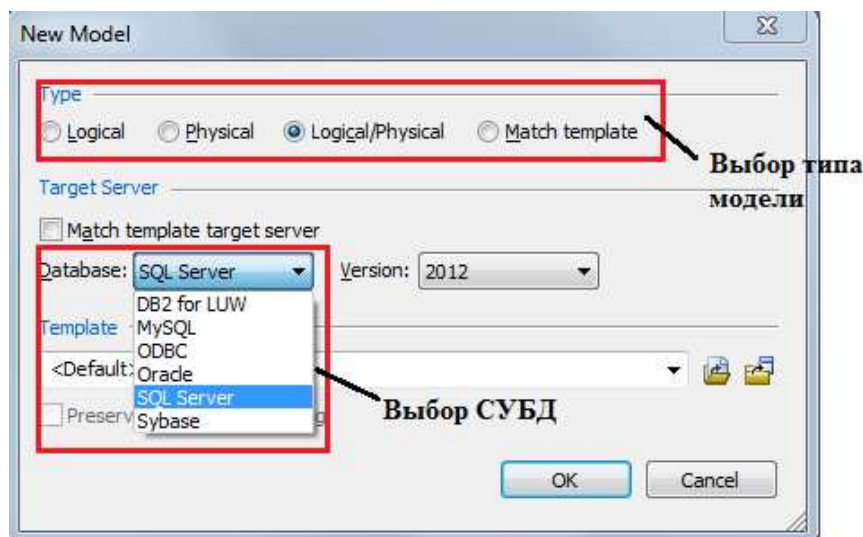


Рисунок 1.4 – Окно выбора типа модели и СУБД

Модель типа **Logical** используется для работы только с ER-моделью предметной области. Модель типа **Physical** используется для работы только с реляционной моделью БД. Модель типа **Logical/Physical** позволяет моделировать БД на инфологическом и логическом уровнях её проектирования и строить ER-модель и реляционную модель БД. Выбор типа модели осуществляется в начале создания модели. В рамках выполнения практических работ разрабатываются модели БД, имеющие тип **Logical/Physical**.

Если выбрана модель типа **Logical/Physical** или **Physical**, то можно выбрать тип целевой СУБД (рис.1.4).

Основное меню среды включает (рис. 1.5) возможности разработки логической и физической модели, документирование модели, выполнение процессов прямого и обратного проектирования. Это CASE-средство позволяет создавать графические модели БД и документировать их. **ERwin DM** использует удобный графический интерфейс, который позволяет разрабатывать ER-модели любой сложности. В среде **ERwin DM** ER-модель строится в терминах

логической модели, элементами которой являются сущности, атрибуты и связи.

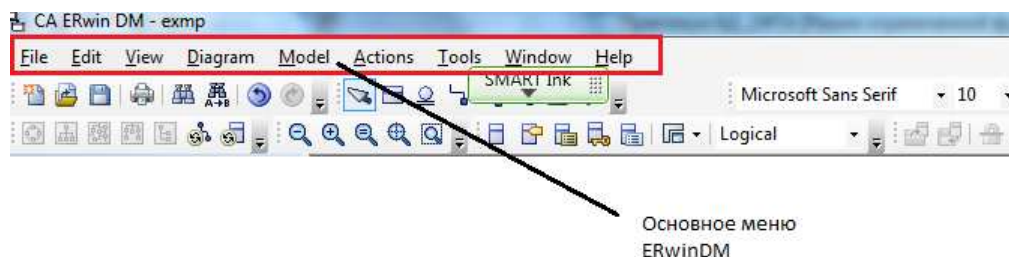
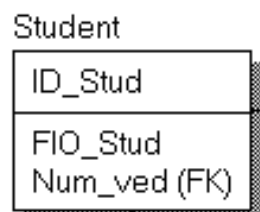


Рисунок 1.5 – Главное меню CASE-средства AllFusion ERwin Data Modeler

В **ERwin DM** прямоугольник (рис. 1.6.) обозначает сущность. Имя сущности располагается над прямоугольником. Для описания сущности используется контекстное меню, которое вызывается правой кнопкой мыши. Для того чтобы задать описание сущности, необходимо выбрать пункт **Properties** и задать уникальное имя, её смысловое описание и особенности взаимодействия с другими сущностями. От того, насколько полно будет представлена эта информация, в дальнейшем зависит качество модели БД.

Логическая модель в **Erwin** может быть представлена различными способами: на уровне сущностей, на уровне атрибутов, на уровне ключей. Наиболее полное представление модели задаётся уровнем атрибутов.

Рисунок 1.6 – Пример сущности в Erwin DM







Для задания описателей атрибутов сущности в контекстном меню выбирается пункт **Attributes**, меню которого позволяет вызвать **Редактор свойств**. В окне **Редактора свойства** задаются основные описатели атрибута: имя атрибута, тип принимаемого значения. Полезно также задать краткое описание атрибута, смысловые правила использования значений атрибута. Это позволит, во-первых, правильно интерпретировать назначение атрибута в модели, а во-вторых, позволит создать описание модели средствами **Erwin DM**. Имена атрибутов отображаются внутри прямоугольника. В верхней части прямоугольника, задающего сущность, отражаются идентифицирующие атрибуты, в нижней – остальные атрибуты.

Обязательным описателем каждого атрибута является тип значений, который должен быть выбран в среде **ERwin DM** из набора допустимых типов:

- данные целого типа – Integer;
- вещественные данные – Real;
- данные, задающие дату и время, – Date;
- символьные данные – Varchar;
- числовые данные в денежном формате – Money.

Между сущностями в **ERwin DM** допускаются следующие типы связей [8]:

-  – задаёт связь «многие-ко-многим»;
-  – задаёт идентифицирующую связь «один-ко-многим»;
-  – задает неидентифицирующую связь «один-ко-многим»;
-  – определяет категориальную связь, которая позволяет определять в модели сложные сущности, реализуя отношения типа тип-подтип.

Для выбора типа связи между сущностями необходимо выделить соответствующую связь (рис. 1.7) и задать её для двух выделенных сущностей.



Рисунок 1.7 – Типы связи между сущностями

Для определения связи 1:N в **ERwin DM** используется два типа связей: идентифицирующая связь «один-ко-многим» и неидентифицирующая связь «один-ко-многим». Идентифицирующая связь связывает сильную (родительскую) сущность и слабую сущность (дочернюю). При использовании такой связи дочерняя сущность автоматически преобразуется в зависимую сущность. На графической модели она представляется прямоугольником с закругленными углами. Поскольку экземпляр зависимой сущности не может существовать без существования связанного с ним экземпляра родительской сущности, то в этом случае в среде **ERwin DM** [8, 9] идентифицирующие атрибуты родительской сущности автоматически

копируются в идентифицирующие атрибуты дочерней сущности (миграция атрибутов). В дочерней сущности скопированные атрибуты помечаются символами FK (внешний ключ). На рисунке 1.8. приведён пример миграции ключей при определении неидентифицирующей связи между сущностями **Преподаватель** и **Курс**.

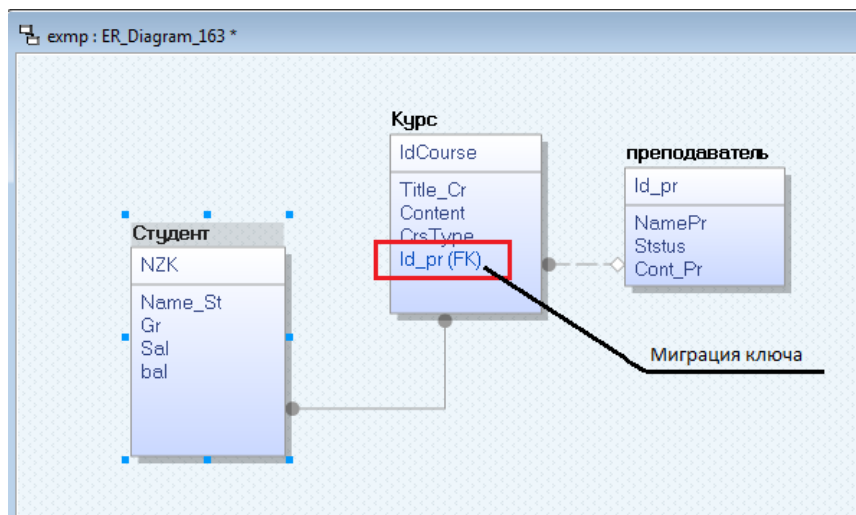


Рисунок 1.8– Задание связи между сущностями

Чтобы описать связь, необходимо в контекстном меню, вызываемом правой кнопкой мыши, выбрать пункт **Properties** (свойства), который вызывает редактор связей, и определить имя связи, её тип и мощность (рис. 1.9).

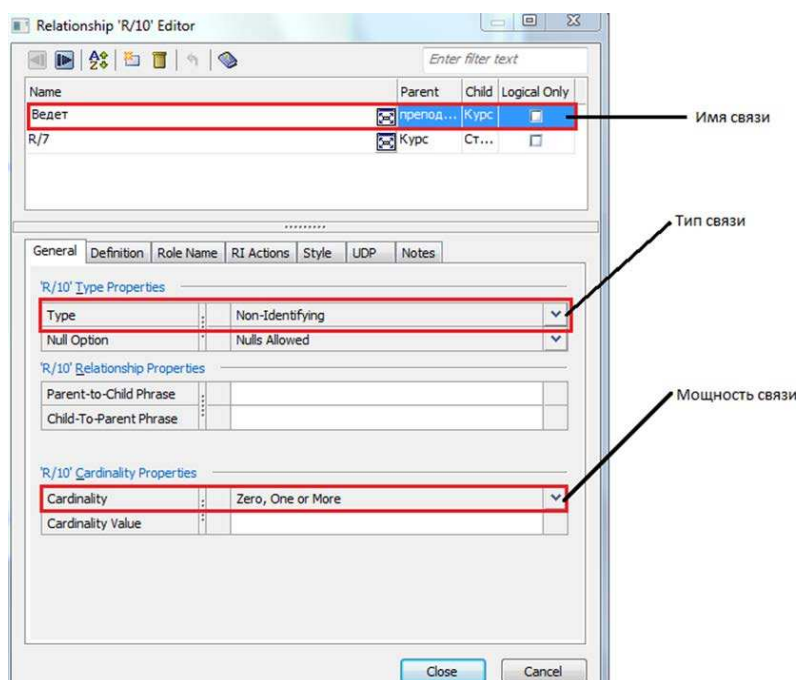


Рисунок 1.9 – Задание свойств связи



Мощность выбирается из определённых в среде **ERwin DM** типов (рис. 1.10):

- связи типа **Zero, One or More** и **One or More (P)** используются для обозначения связи типа 1:M (отличие между этими двумя типами в **ERwin DM** заключается в том, что в случае использования типа **Zero, One or More** в дочерней сущности могут отсутствовать связанные с родительской сущностью экземпляры, а при использовании связи типа **One or More (P)** в дочерней сущности обязательно должны быть экземпляры, связанные с родительской сущностью, такая связь на графической модели помечается символом **P**);

- **Zero or One (Z)** определяет связь 1:1;

- связь, помечаемая на модели цифрой, позволяет определить точное количество экземпляров дочерней сущности, которое связано с одним экземпляром родительской сущности.

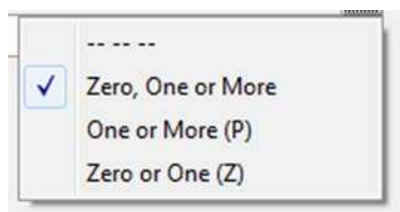


Рисунок 1.10 – Типы мощности связи

В общем случае выделяют следующие виды слабых сущностей:

- **характеристическая** – дополняет характеристики родительской сущности и связана только с одной сущностью;

- **ассоциативная** – используется для задания взаимодействия двух сущностей и связана с каждой из них;

- **категориальная** – используется для определения сложной сущности на основе нескольких слабых сущностей.

Пример слабой сущности **Контракт** представлен на рисунке 1.11.

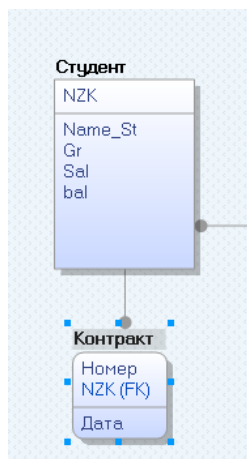


Рисунок 1.11 – Зависимая сущность



Задание идентифицирующей связи в дочерней сущности **Контракт** автоматически создаётся копия идентифицирующего атрибута родительской сущности **Студент – NZK (FK)**, которые помечаются символами (FK).

## 2.2. Проектирование ER-модели в среде Microsoft Visio

**Microsoft Visio** нельзя в полной мере отнести к CASE-средствам проектирования БД. Это среда разработки графических объектов. Однако благодаря большому количеству шаблонов, определённых в ней, её можно использовать для разработки модели БД.

Для разработки ER-модели при выполнении практического занятия рекомендуется использовать шаблон для создания БД в нотации **IDEF1X** (рис. 1.12), который находится в категории «Программное обеспечение».

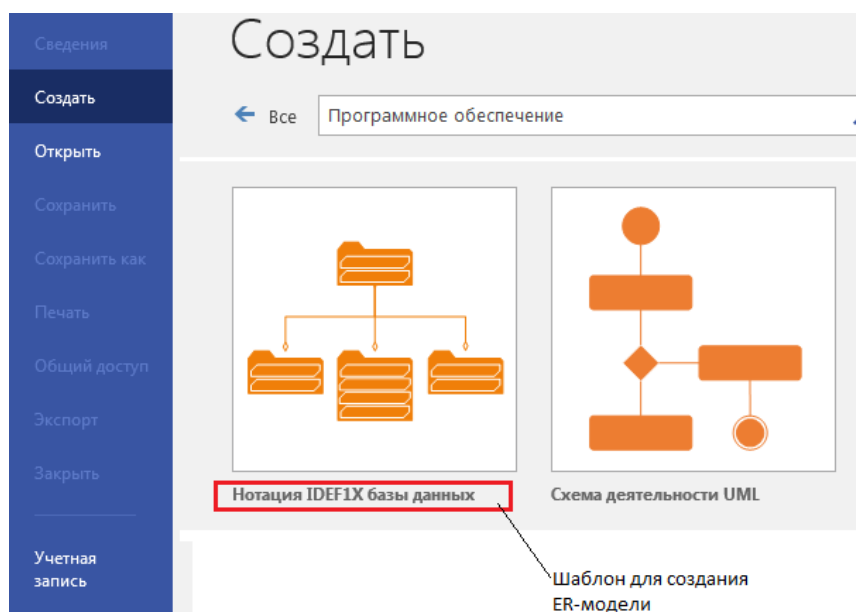


Рисунок 1.12 – Выбор шаблона для разработки ER-модели в среде Microsoft Visio

Основное меню **Microsoft Visio** (рис. 1.13) позволяет выполнять следующие действия по работе с графическим объектом:

- выполнять стандартные действия с файлом – открытие файла, создание нового файла, сохранение – вкладка **Файл**;
- создавать и редактировать графический объект – вкладка **Главная**;
- вставлять различные элементы графического объекта: иллюстрации, части схемы, ссылки, текст – вкладка **Вставка**;
- изменять параметры страницы – вкладка **Конструктор**;

- определять связь с внешними данными – вкладка **Данные**;
- создавать новый процесс – вкладка **Процесс**;
- проверять правописание, добавлять примечания, отмечать исправления и др. – вкладка **Рецензирование**;
- определять вид отображения графического объекта на экране дисплея – вкладка **Вид**.

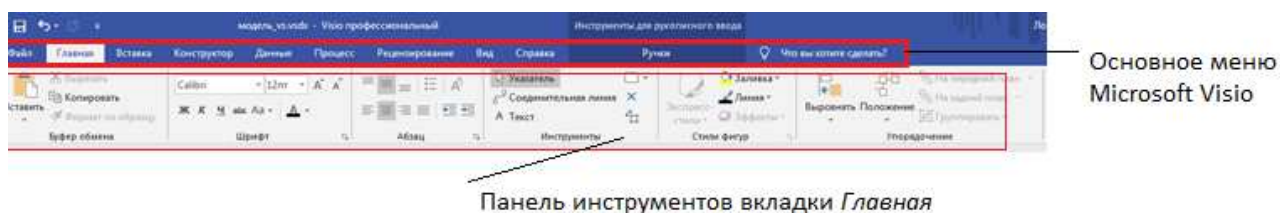

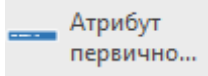
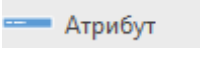


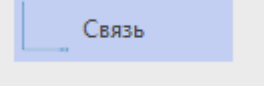

Рисунок 1.13 – Основное меню Microsoft Visio

При использовании шаблона **Нотация IDEF1X** для базы данных **Microsoft Visio** предоставляет следующий набор элементов (рис. 1.14):

- сущность  **Сущность** используется для определения имени сущности и включает описание атрибутов;

- атрибут первичного ключа  позволяет определить атрибут, идентифицирующий экземпляр сущности (атрибутов первичного ключа может быть несколько, в этом случае он является составным);

- атрибут  позволяет определить атрибут сущности, не относящейся к первичному ключу;

- связь  определяет все виды связей, предусмотренные в ER-модели (рекомендуется обозначать тип связи, используя инструмент добавления текста – инструмент  панели инструментов (рис. 1.13)).

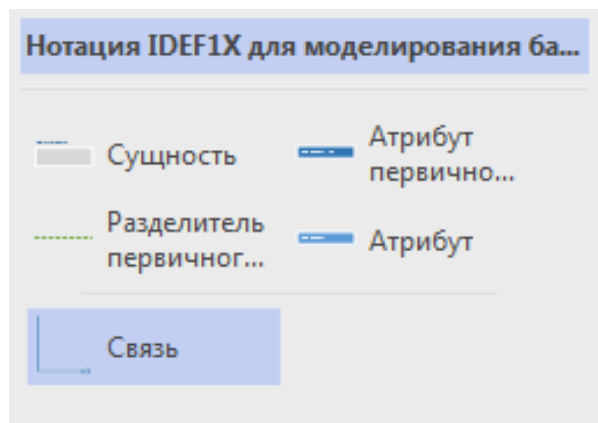


Рисунок 1.14 – Элементы для создания ER-модели в Microsoft Visio

Пример построения модели БД в среде Microsoft Visio представлен на рисунке 1.15.

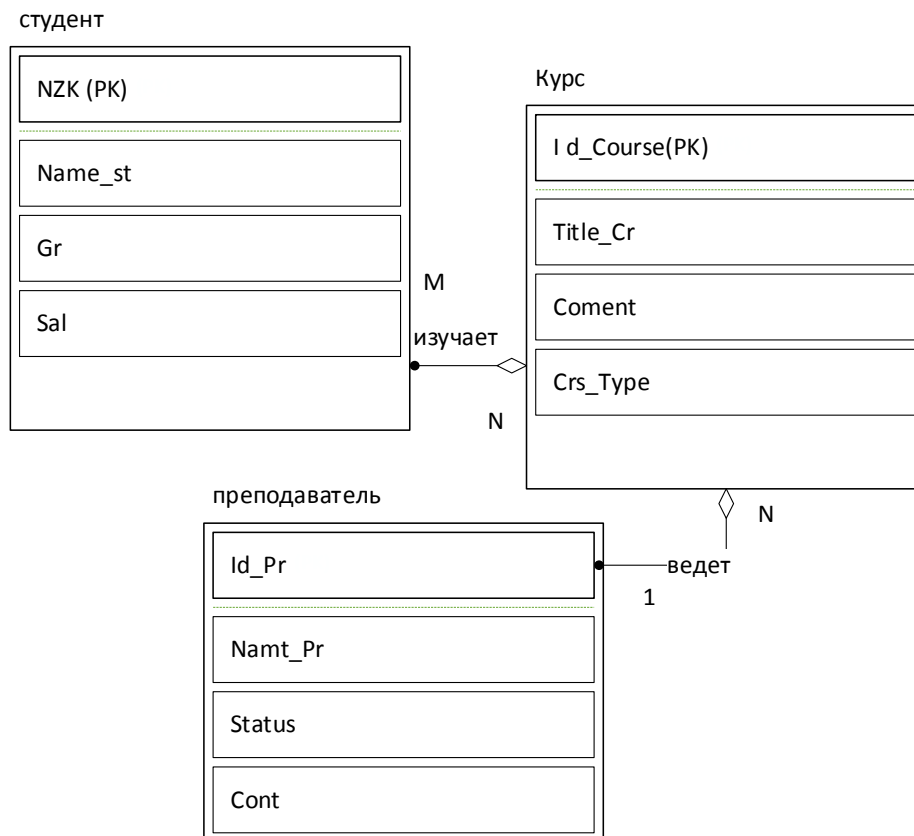


Рисунок 1.15 – ER-модель в среде Microsoft Visio

### 3. Средства документирования ER-модели

Документирование модели является важным и обязательным элементом разработки модели. Различные CASE-средства имеют разные инструменты для документирования.

### 3.1. Документирование модели в ERwin Data Modeler

CASE-средство **ERwin DM** обладает специальными возможностями для автоматического создания описания разработанной модели. В его состав включены возможности использования стандартных видов отчёта по модели и средства разработки отчёта на основе созданного шаблона. Эти возможности определены во вкладке **Tools** основного меню (рис. 1.5.). Набор стандартных отчётов определяется имеющимися шаблонами. Для создания собственного шаблона отчёта по модели используется редактор отчётов – **Report Designer** (рис. 1.16).

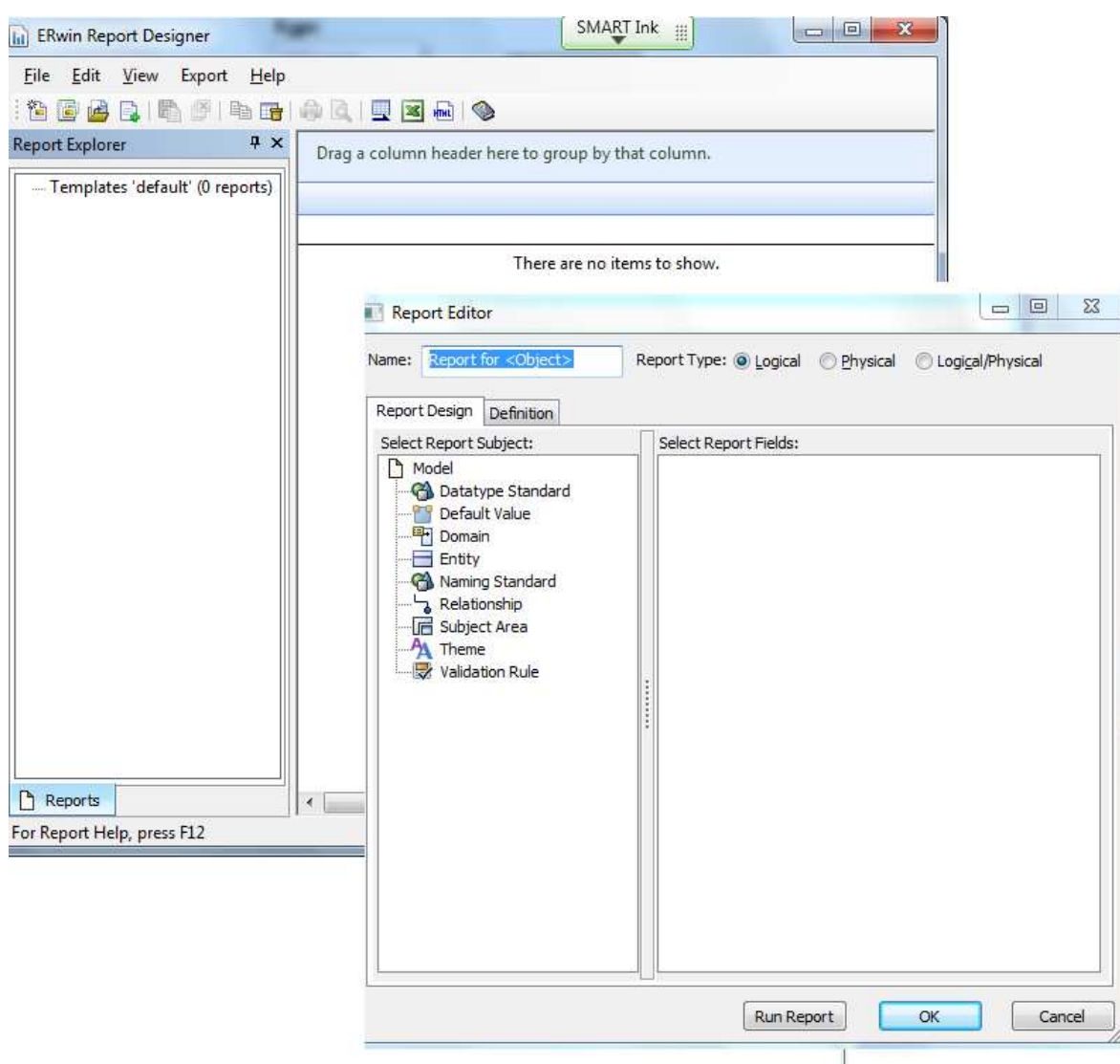


Рисунок 1.16 – Создание шаблона отчета в редакторе отчетов

Для формирования своего шаблона отчёта необходимо выделить в правой части **Редактора отчётов** элемент модели. В левой части **Редактора отчётов** будут отображаться свойства выбранного

элемента. Для включения выбранного свойства в шаблон отчёта достаточно поставить знак ☒ у этого элемента (рис. 1.17).

Для формирования отчёта достаточно нажать кнопку **Run Report**. Созданный отчёт (рис. 1.18) можно сохранить в различных форматах. Примеры сформированных отчётов по модели представлены на рисунках 1.18 и 1.19.

При использовании среды **Microsoft Visio** отчёт создаётся отдельно средствами **Microsoft Word**. Для этого можно использовать табличную форму описания модели, аналогичную представленной на рисунке 1.18.

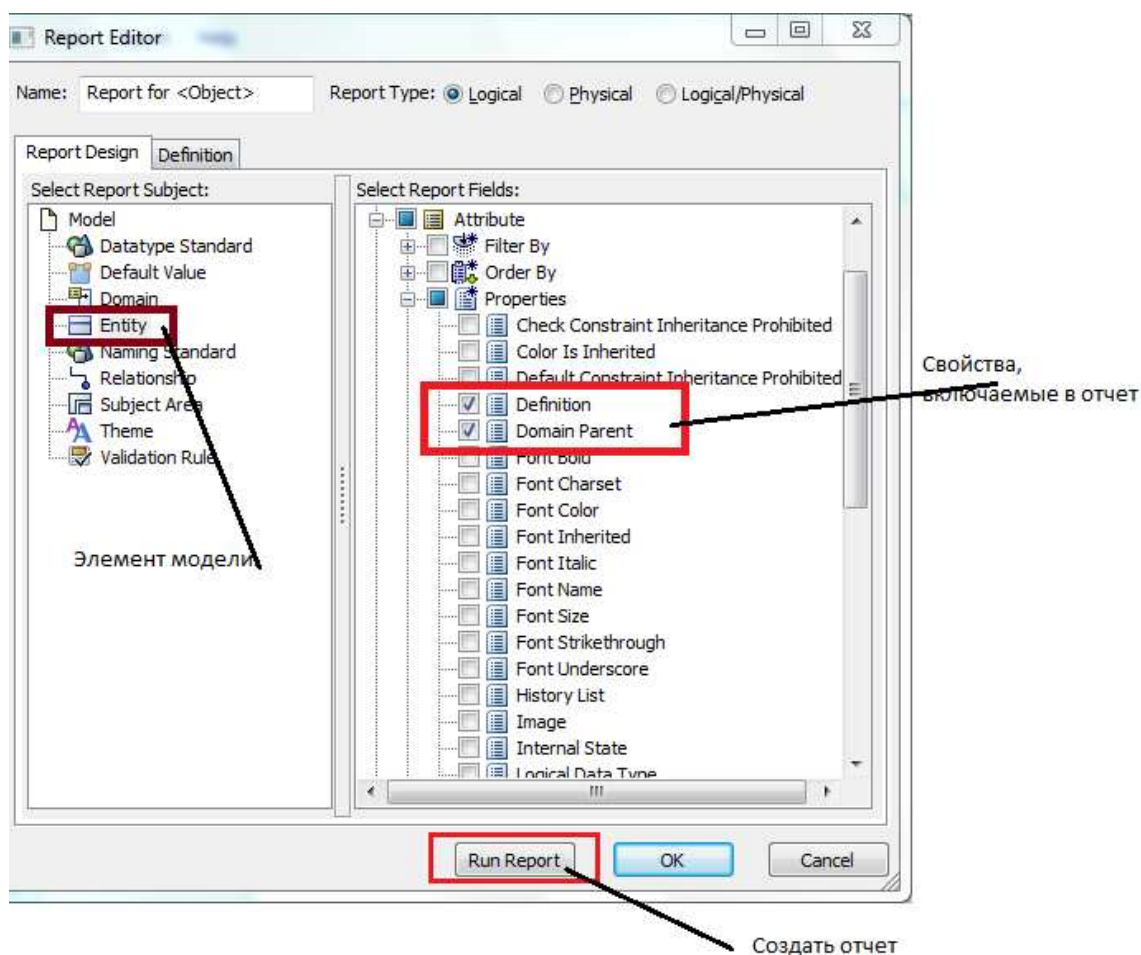


Рисунок 1.17 – Формирование шаблона отчета

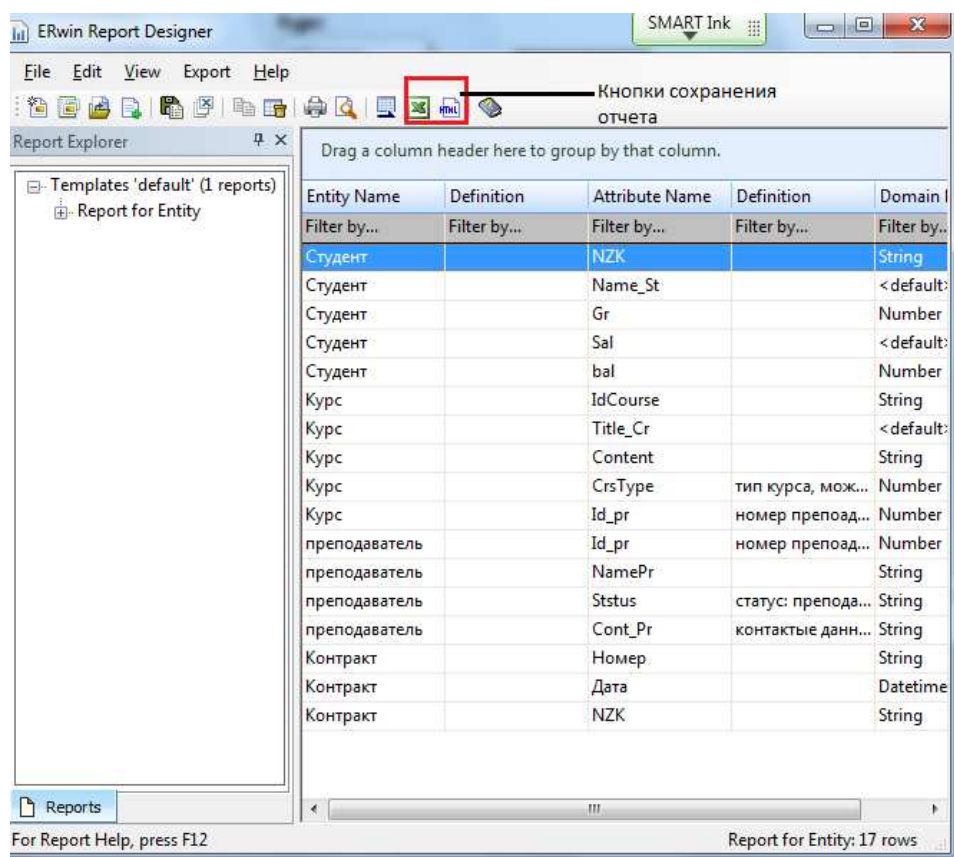


Рисунок 1.18 – Отчёт по модели, созданный в среде Erwin DM

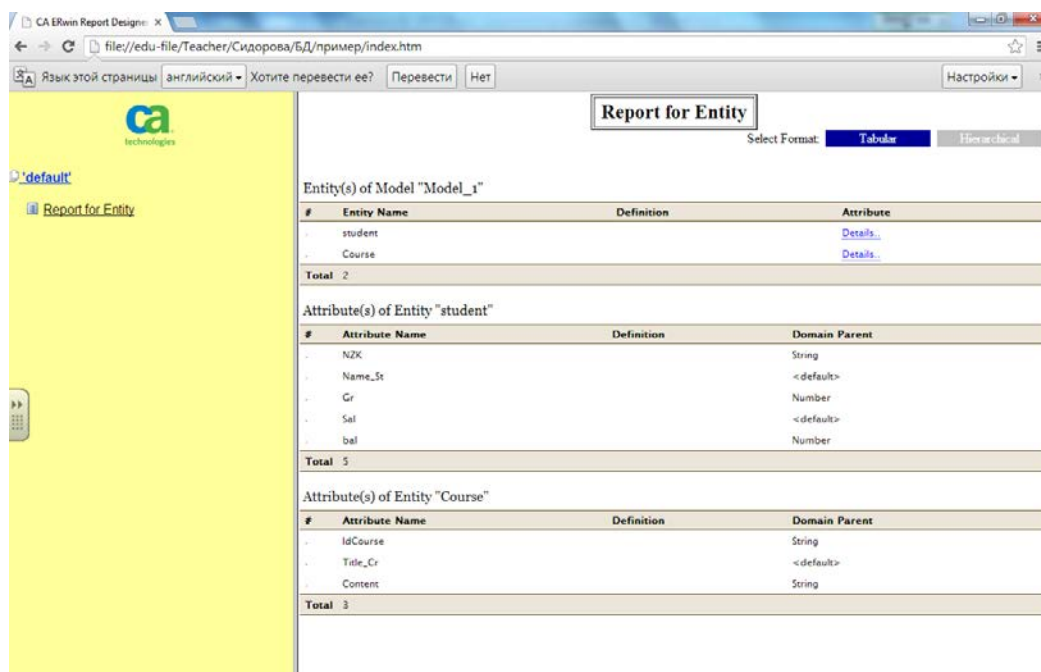


Рисунок 1.19 – Отчёт по модели в формате html

### 3.2. Документирование модели в среде Microsoft Visio

Среда Microsoft Visio не располагает средствами документирования модели. Поэтому для описания созданной модели

рекомендуется использовать средства разработки текстовых документов, например, **Word**. Описания ER-модели должно включать следующие разделы:

- состав сущностей и их краткая характеристика;
- состав атрибутов каждой сущности и их краткое описание;
- описание связей между сущностями.

Такое описание может быть выполнено в форме таблиц. Фрагмент описания ER-модели, представленной на рисунке 1.15, приведён в таблицах 1.1 – 1.3.

Таблица 1.1 – Описание сущностей ER-модели

№	Имя сущности	Описание
1.	студент	Учащийся вуза, который может обучаться по одному направлению подготовки
2.	дисциплина	Дисциплина учебного плана направления подготовки
3.	преподаватель	Сотрудник вуза, который ведёт занятия по дисциплинам учебного плана

Таблица 1.2 – Описание свойств сущностей

№	Имя свойства	Описание	Ограничения
Сущность студент			
1.	NZK	Номер зачётной книжки	Уникальное значение для каждого студента
2.	Name_st	Полное имя студента, содержит фамилию имя и отчество	
3.	Gr	Номер группы, в которой обучается студент	
4.	Sal	Стипендия	Размер стипендии $\geq 0$
Сущность дисциплина			
1.	Id_Course	Код дисциплины	Задаётся номером дисциплины по учебному плану, уникальное значение для каждой дисциплины
2.	Titile_Cr	Название дисциплины по учебному плану	
3.	Coment	Краткое описание дисциплины	
4.	Crs_Type	Тип дисциплины	Может принимать значения: 1 – 4



Таблица 1.3 – Описание связей ER-модели

№	Имя связи	Тип связи	Описание
1.	изучает	многие-ко-многим	Много студентов изучает несколько дисциплин
2.	ведёт	один-ко-многим	Один преподаватель ведёт несколько дисциплин

#### 4. Последовательность выполнения практической работы

1. Запустить CASE-средство разработки модели БД.

2. Ознакомиться с её интерфейсом среды моделирования, её элементами для построения ER-модели.

3. Провести анализ описания предметной области и выделить набор сущностей.

**Указание.** Варианты заданий для выбора предметной области приведены в Приложении I. Студент может конкретизировать описание предметной области, приведённое в варианте задания.

Набор сущностей должен обязательно включать указанные в варианте задания сущности, но также может включать дополнительные сущности на основе уточнённого описания предметной области.

При определении состава свойств (атрибутов) сущности следует учитывать указанный в задании минимальный набор атрибутов и запросы на обработку.

4. Используя CASE-средство, построить и описать ER-модель для заданной предметной области. Описание каждой сущности должно содержать её имя, назначение в модели БД.

Выделить атрибуты каждой сущности.

**Указание.** При определении состава свойств (атрибутов) сущности следует учитывать указанный в задании минимальный набор атрибутов и запросы на обработку.

5. Определить для каждой сущности набор атрибутов и задать их описание. Описание каждого атрибута должно содержать его имя, тип, назначение атрибута в модели, смысловые правила, определяющие специфику использования атрибута.

6. Задать идентифицирующий атрибут для каждой сущности.

7. При необходимости выделить сложные сущности.

8. Провести анализа бизнес-правил, приведённых в варианте задания, и определить связи между сущностями. Описать каждую



связь. Описание связи включает её смысловое имя, тип, краткое смысловое описание.

9. Построить отчёт по созданной модели, используя возможности среды моделирования.

10. Оформить отчёт по практической работе.

### **5. Требования к оформлению отчета**

Отчёт по практической работе включает:

- 1) титульный лист;
- 2) описание заданной предметной области;
- 3) состав выделенных сущностей и их краткое описание;
- 4) графическую ER-модель;
- 5) описание ER-модели, созданное с помощью CASE-средства;

**Указания:** Описание должно включать все элементы модели: сущности, атрибуты, связи.

- 6) краткие ответы на контрольные вопросы.

### **Контрольные вопросы**

1. Определите назначение инфологической модели БД.
2. Что такое сущность-понятие?
3. Что определяет сущность-экземпляр?
4. Какие ограничения существуют для слабой сущности?
5. Назовите обязательные описатели атрибута сущности.
6. Что определяет связь между сущностями?
7. Что такое идентифицирующий атрибут сущности?
8. Дайте интерпретацию связи 1:1 в ER-модели. Приведите пример использования такой связи.
9. Дайте интерпретацию связи 1:M в ER-модели. Приведите пример использования такой связи.
10. Поясните на примере интерпретацию связи M:N.

## Практическая работа 2

### РАЗРАБОТКА РЕЛЯЦИОННОЙ МОДЕЛИ БД

**Цель работы** – формирование навыков разработки нормализованной реляционной модели БД.

#### 1. Теоретические основы

Автором реляционной модели данных (РМД) Э. Коддом было доказано, что любое представление данных может быть сведено к совокупности двумерных таблиц особого вида – **relation (отношение)**. В качестве основы реляционной модели Э. Кодд использовал математический аппарат теории множеств. На её основе он разработал формализованное описание и сформулировал алгебру отношений (набор операций с ними). Формальное описание отношения [9] определяется как подмножество декартового произведения образующих его множеств (**доменов**):

$$R \subseteq D_1 \times D_2 \times \dots \times D_n, \text{ где } D_i - i\text{-ый домен отношения.}$$

Таким образом, отношение определяет множество, каждым элементом которого является **кортеж** вида  $(d_1, d_2, \dots, d_n)$ , где каждый  $d_i \in D_i$ . Каждый домен характеризует некоторое множество. Атрибут определяет использование домена в конкретном отношении. Например, есть домен, содержащий все целые числа. Тогда можно задать атрибут, содержащий только числа, задающие оценки за успеваемость. Каждый атрибут в отношении задаётся своим именем и определён на одном домене. Множество имен атрибутов отношения определяет **схему отношения**.

На практике чаще используют табличную интерпретацию понятия отношения, которая является более наглядной. В этом случае говорят о реляционных таблицах. При этом имена столбцов таблицы соответствуют именам доменов отношения. Само отношение представляется набором строк таблицы. Реляционные таблицы обладают следующими свойствами:

- 1) в отношении нет одинаковых строк;
- 2) строки таблицы не упорядочены;
- 3) все значения полей таблицы атомарны, т.е. неделимы в процессе их обработки.

Модель реляционной БД (РБД) определяется набором входящих в неё отношений (таблиц) [6, 10, 11]. Модель РБД должны

обеспечивать её корректное использование, простоту модификации, минимальную избыточность данных. Если модель БД не соответствует этим требованиям, то это приводит к увеличению времени обработки данных. В некоторых случаях использование некорректной модели БД может приводить к ошибкам обработки данных. Чаще всего такая ситуация связана с наличием избыточных данных, которые вызывают аномалии обновления данных. В модели РБД могут присутствовать следующие виды аномалий:

- аномалия, связанная с обновлением данных;
- аномалия, связанная с добавлением данных;
- аномалия, связанная с удалением данных.

Наличие аномалий в БД связано со схемой отношения, а именно, с набором её атрибутов. Оценка качества БД основывается на оценке каждого отдельного отношения. Для проверки качества отношения применяют **нормальные формы** (НФ). В теории РБД различают несколько нормальных форм, каждая из которых характеризуется своим набором ограничений. В основе этих ограничений лежат **функциональные зависимости** атрибутов отношения. Функциональную зависимость можно формализовать следующим образом.

Приведём несколько определений, используемых при нормализации модели РБД.

Пусть задано некоторое отношение  $R$ , в котором определены атрибуты  $X$  и  $Y$ . Говорят, что  $Y$  **функционально зависит** от  $X$  ( $X \rightarrow Y$ ), если в любой момент времени каждому значению  $X$  в отношении  $R$  соответствует единственное значение  $Y$ .

Введём понятие **полной функциональной зависимости**. Пусть задано некоторое отношение  $R$ , в котором определены атрибуты  $X$  и  $Y$ . Атрибут  $Y$  функционально полно зависит от составного атрибута  $X$ , если он функционально зависит от  $X$  и не зависит функционально от любого подмножества атрибута  $X$ .

Теперь введём определения нормальных форм, используемые при выполнении практического задания [9, с. 13].

«**Определение.** Отношение находится в первой нормальной форме (1НФ) тогда и только тогда, когда каждое поле отношения содержит атомарное значение.

**Определение.** Ключ отношения – это атрибут (простой или составной), который однозначно определяет кортеж отношения.

**Определение.** Отношение находится во второй нормальной форме, если оно находится в первой нормальной форме, и каждый неключевой атрибут отношения функционально полно зависит от ключа.

**Определение.** Отношение находится в третьей нормальной форме, если оно находится во второй нормальной форме, и каждый неключевой атрибут нетранзитивно зависит от первичного ключа.

**Определение.** Отношение находится в **нормальной форме Бойса-Кодда (НФБК)** тогда и только тогда, когда любая функциональная зависимость между его атрибутами сводится к полной функциональной зависимости от **вероятностного** ключа».

Таким образом, каждая нормальная форма определяет свой набор ограничений, которым должны удовлетворять атрибуты одной таблицы. Для обеспечения качественной работы РБД в большинстве случаев достаточно, чтобы она удовлетворяла ограничениям третьей нормальной формы или нормальной формы Бойса-Кодда. Для этого, необходимо выделить все функциональные зависимости между атрибутами каждой отдельно взятой таблицы, выделить неприводимое множество таких функциональных зависимостей и провести их анализ. В случае нарушения ограничений третьей нормальной формы или нормальной формы Бойса-Кодда необходимо выполнить нормализацию этой таблицы. Для этого выполняют **декомпозицию** (разбиение) таблицы. При этом в модели РБД увеличивается число таблиц, и структура таблиц становится проще. Правильная декомпозиция обладает следующими свойствами:

- соединение без потерь, которое позволяет получить исходное отношение  $R$  из  $C_i$ ;
- сохранение в наборе  $C_i$  функциональных зависимостей, определённых в  $R$ .

## **2. Логическое проектирование РБД**

На этапе разработки логической модели определяется [9, 10] такой набор отношений, который позволяет адекватно отображать семантику сущностей предметной области и является в некотором смысле оптимальным (эффективным, удобным и т.д.). Для этого необходимо решить следующие задачи:

- сформировать набор отношений БД;
- определить набор атрибутов для каждого отношения;

- обосновать выбор первичного ключа отношения (простого или составного) (при этом предпочтение отдаётся простым ключам);
- выделить ограничения целостности на атрибуты и отношения БД.

Набор отношений должен удовлетворять следующим требованиям:

- минимальная избыточность данных;
- обоснованный выбор первичных ключей, первичные ключи по возможности должны быть минимальными;
- каждое отношение БД должно находиться в третьей нормальной форме или нормальной форме Бойса-Кодда.

Процесс перехода от ER-модели к модели РБД описан в [9, с. 14] и включает следующие шаги.

1. «Каждая простая сущность представляется отношением.

2. Каждый атрибут сущности представляется возможным атрибутом отношения. При этом более точно определяется тип данных исходя из возможностей СУБД.

3. Атрибуты уникального идентификатора сущности определяют первичный ключ отношения.

4. Связи М:1 (и 1:1) реализуются с помощью атрибутов, определяемых как внешний ключ. В этом случае в дочернем отношении создаётся копия первичного ключа родительского отношения. При реализации связи между сильной и слабой сущностью копия первичного ключа родительского отношения становится частью составного первичного ключа дочернего отношения. При реализации связи между двумя сильными сущностями копия первичного ключа родительского отношения становится неключевым атрибутом дочернего отношения.

5. Связи типа М:N представляются отдельным отношением, первичный ключ которого формируется на основе первичных ключей двух связываемых отношений».

### **3. Разработки реляционной модели**

#### **3.1. Разработка реляционной модели в AllFusion ERwin Data Modeler**

Для разработки реляционной модели БД (РМД) в CASE-средстве **ERwin DM** используются инструменты работы с моделью типа **Physical** (рис. 2.1.).

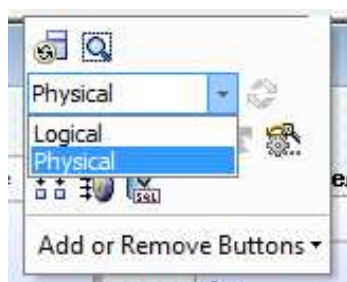


Рисунок 2.1 – Выбор типа модели

Набор инструментов модели этого типа отличается от инструментов, используемых при построении инфологической модели (рис. 2.2). Также отличаются и дополнительные средства, определённые во вкладках **Tools** и **Actions**. Они включают действия, выполняемые с моделью типа **Physical**.



Рисунок 2.2 – Панель инструментов для разработки РМД

При переходе от модели типа **Logical** к модели типа **Physical** CASE-средство **ERwin DM** автоматически выполняет преобразование элементов модели одного типа в элементы модели другого. Результатом их выполнения является реляционная модель, которая гарантированно находится в 1НФ (рис. 2.3).

Средства документирования реляционной модели аналогичны средствам документирования инфологической модели и описаны в первой практической работе.

### 3.2. Разработка реляционной модели в Microsoft Visio

В отличие от CASE-средства **ERwin DM** **Microsoft Visio** не выполняет автоматического преобразования моделей. Поэтому все шаги перехода от ER-модели к реляционной модели базы данных, определённые в разделе 2 этого практического задания, выполняются вручную. Таким образом, при разработке реляционной модели необходимо с помощью дополнительных атрибутов в таблице задать все связи между сущностями (рис. 2.4).

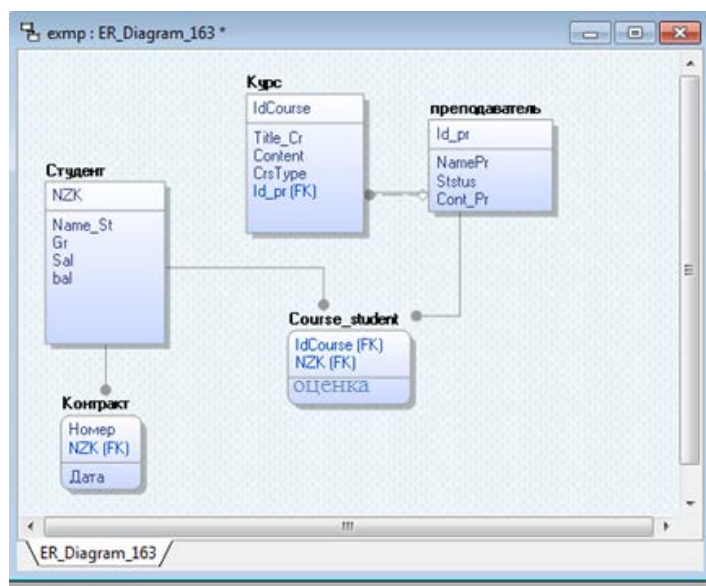


Рисунок 2.3 – Пример реляционной модели БД в среде Erwin DM

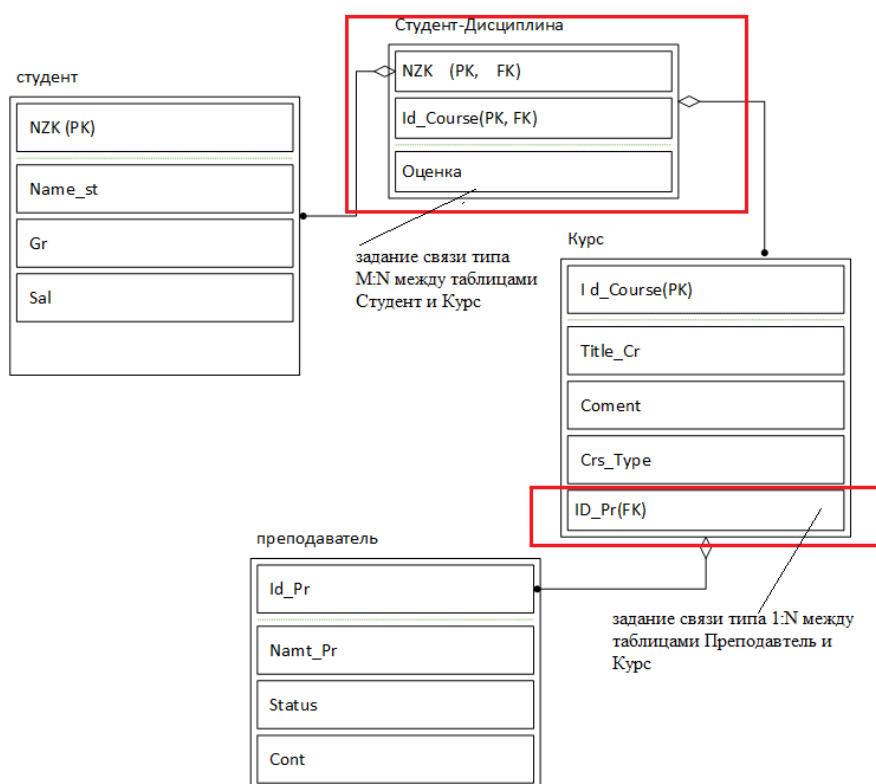


Рисунок 2.4 – Пример реляционной модели БД в среде Microsoft Visio

#### 4. Последовательность выполнения практической работы

1. Запустить среду моделирования и открыть созданную на практической работе 1 модель БД.

2. Используя алгоритм перехода от ER-модели к реляционной модели БД, разработать реляционную модель.

3. Для каждого отношения выделить все функциональные зависимости, определённые на его атрибутах и провести их анализ.

4. Проверить каждое отношение на ограничения третьей нормальной формы или нормальной формы Бойса-Кодда.

5. При необходимости выполнить нормализацию отношений, приведя модель РБД к требуемой нормальной форме.

6. Оформить отчёт по практической работе.

### **5. Требования к оформлению отчета**

Отчёт по практической работе должен включать следующие разделы:

- 1) титульный лист;
- 2) графическая модель РБД;
- 3) описание реляционной модели:
  - описание отношений: имя, краткое описание;
  - описание атрибутов каждого отношения: имя, тип, краткое описание;
  - первичные и внешние ключи для каждой таблицы;
  - функциональные зависимости для каждой таблицы;
  - обоснование нахождения модели РБД в 3НФ;
- 4) ответы на контрольные вопросы.

### **Контрольные вопросы**

1. Определите задачи этапа логического проектирования БД.
2. Что задаёт домен?
3. Назовите отличие атрибута отношения и домена отношения.
4. Что определяет схема отношения?
5. Объясните понятие функциональной зависимости, полной функциональной зависимости. Приведите примеры.
6. Какие ограничения заданы для 1НФ?
7. Назовите ограничения, определённые для отношения во 2НФ?
8. Что такое транзитивная зависимость? Приведите пример.
9. Каким свойствам должна удовлетворять декомпозиция отношения?
10. Определите описатели атрибутов отношения в среде ERwin DM.
11. Опишите алгоритм перехода от ER-модели к реляционной модели БД.



### Практическая работа 3

## ПРОЕКТИРОВАНИЕ ПРАВИЛ ЦЕЛОСТНОСТИ БД И ФИЗИЧЕСКОЙ МОДЕЛИ БД

**Цель работы** – разработать правила целостности БД и описать физическую модель БД на языке SQL выбранной СУБД.

### 1. Теоретические основы

Целостность – обязательное свойство РБД. Она гарантирует сохранность и корректность обработки данных в СУБД. Проектирование правил целостности является важным этапом, во многом определяющим эффективность её использования.

Правила целостности определяются на основе специфики предметной области и задают:

- сущностную целостность, т.е. гарантируют отсутствие повторяющихся строк в таблице;
- доменную целостность, т.е. определяют правила для значений в одном столбце таблицы;
- связная целостность.

Правила, определяющие доменную целостность, задаются логическим выражением. Его значение вычисляется на основе данных из БД и может принимать значение **Истина** или **Ложь**. С их помощью можно задать допустимые значения для атрибута.

СУБД автоматически проверяет выполнение правил целостности и в случае их нарушения не вносит изменение в состояние БД. В зависимости от способа их описания различают [9]:

- статические, которые задаются при описании объектов БД;
- динамические, которые реализуются специальными программными объектами БД.

В практической работе разрабатываются только статические правила, для которых можно использовать различные описатели атрибутов [9]:

- тип данных и описатель первичного ключа;
- логические выражения;
- значения по умолчанию;
- связную целостности и пр.

Например, для модели базы данных, приведённой в практической работе 2 (рис. 2.4), можно задать следующие правила

целостности: декларативное в таблице **Студент-Дисциплина** атрибут **Оценка** может принимать значения 2, 3, 4, 5.

При определении правил ссылочной целостности СУБД автоматически включает механизмы их проверки, которые обеспечивают согласованные изменения данных в связанных таблицах. Для правил ссылочной целостности определён стандартный набор стратегий, которые реализуются во всех реляционных СУБД [8, 9].

1. Ограничения на обновление (удаление или изменение), связанное с появлением «повисших ссылок» – **RESTRICT** (ограничить). В этом случае не будет выполняться обновление данных в таблице (дочерней или родительской), если имеется хотя бы один кортеж (родительской или дочерней) таблицы, который ссылается на обновляемый кортеж.

2. Каскадное обновление (изменение или удаление) – **CASCADE**. В этом случае разрешается обновление первичного ключа в родительской таблице. Однако будут обновлены (изменены или удалены) все значения внешних ключей во всех кортежах дочерних таблиц, связанных с обновляемым кортежем.

3. Задание неопределённого значения **NULL-SET NULL** (установить значение NULL) для всех внешних ключей дочерней таблицы, ссылающихся на обновленный (изменённый или удалённый) первичный ключ родительской таблицы.

4. Задание значения по умолчанию – **SET DEFAULT** в атрибутах внешнего ключа дочерней таблицы при изменении или удалении первичного ключа родительской таблицы.

5. Не учитывать целостность и выполнить обновление, не обращая внимания на нарушения ссылочной целостности – **IGNORE** (игнорировать).

Правила ссылочной целостности необходимо определить для каждой связи. Выбор стратегии поддержания целостности определяется правилами, которые существуют в предметной области.

Определим правила ссылочной целостности для модели РБД, приведённой в практической работе 2. Для поддержания ссылочной целостности между таблицами **Курс** и **Студент-Дисциплина** можно выбрать стратегию каскадного изменения, которое позволит изменить код курса во всех строках таблицы **Студент-Дисциплина** при изменении кода курса в таблице **Курс**.

## 2. Средства задания целостности в среде AllFusion ERwin Data Modeler

Инструменты для определения статических правил целостности в **Erwin DM** [8] позволяют определить статические правила целостности для атрибутов (доменная целостность) и связей (ссылочная целостность).

Для задания ограничений на значения атрибутов используются описатели типа данных, логические выражения, описатель первичного ключа. Логические выражения можно определить, используя вкладку **Constraint** (рис. 3.1) редактора свойств атрибута.

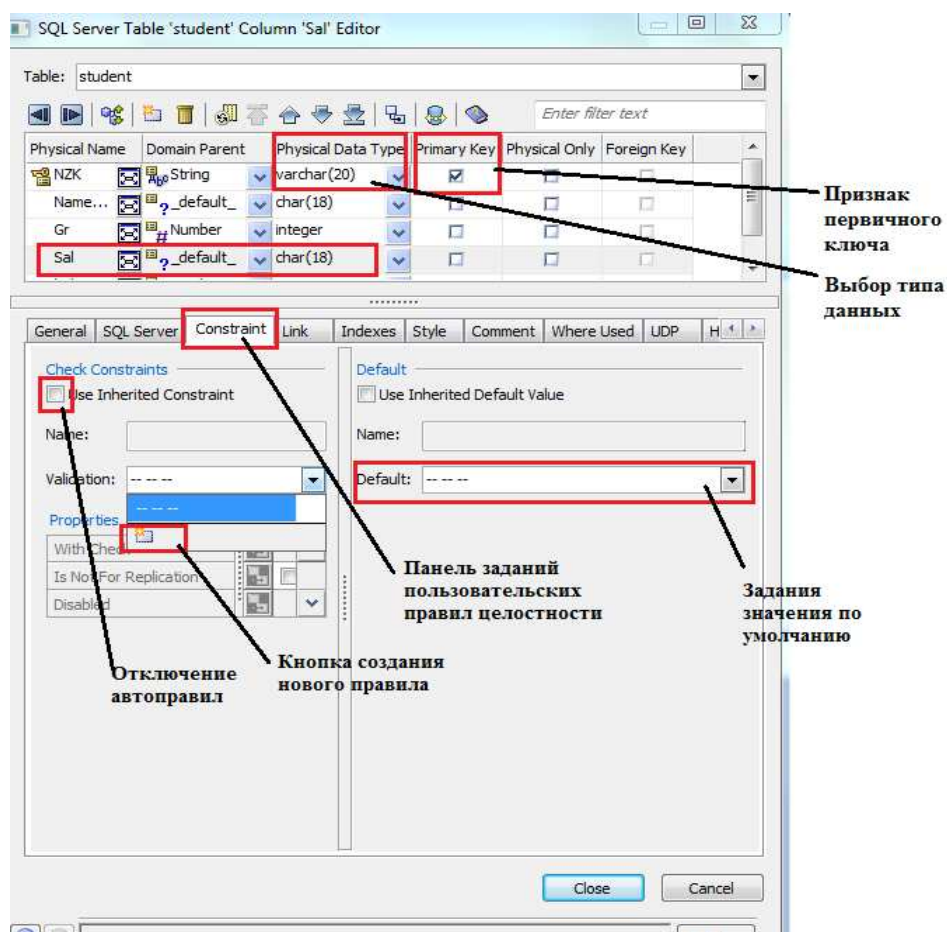



Рисунок 3.1 – Задание ограничений целостности для атрибута

Для задания дополнительных правил целостности необходимо открыть панель задания правил целостности для описываемого столбца. Нажать кнопку  для создания нового правила. В открывшемся окне редактора правил выбрать тип правила и задать значения для него (рис. 3.2).

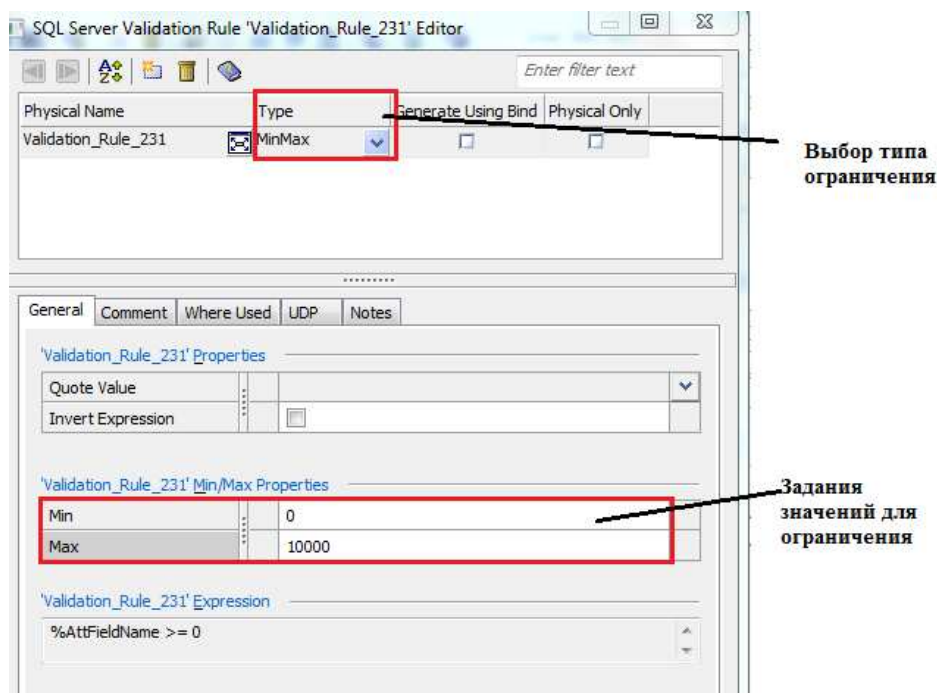


Рисунок 3.3 – Редактор правил целостности атрибута

CASE-средство **ERwin DM** поддерживает следующие виды правил целостности для столбца таблицы:

- задание диапазона значений;
- задания списка значений;
- задание правила в виде логического условия.

На рисунке 3.3 приведён пример задания ограничений в виде списка допустимых значений.

Правила ссылочной целостности используют стандартные стратегии управления целостностью, описанные ранее. При создании физической модели на основе описания РБД **ERwin DM** автоматически создаёт описание триггеров БД.

Правила ссылочной целостности определяются при задании свойств связи. Для каждой связи могут быть заданы требования по обработке операций **INSERT/UPDATE/DELETE** для родительской и дочерней таблиц. **ERwin DM** способствует возможности задания стандартных стратегий поддержания ссылочной целостности.

Для определения правил ссылочной целостности необходимо определить свойство **RI Action** в редакторе свойств связи (рис. 3.4). Для каждой связи можно установить свои правила ссылочной целостности при выполнении операций изменения данных. Для этого в строке описания целостности следует выбрать правило из выпадающего списка.

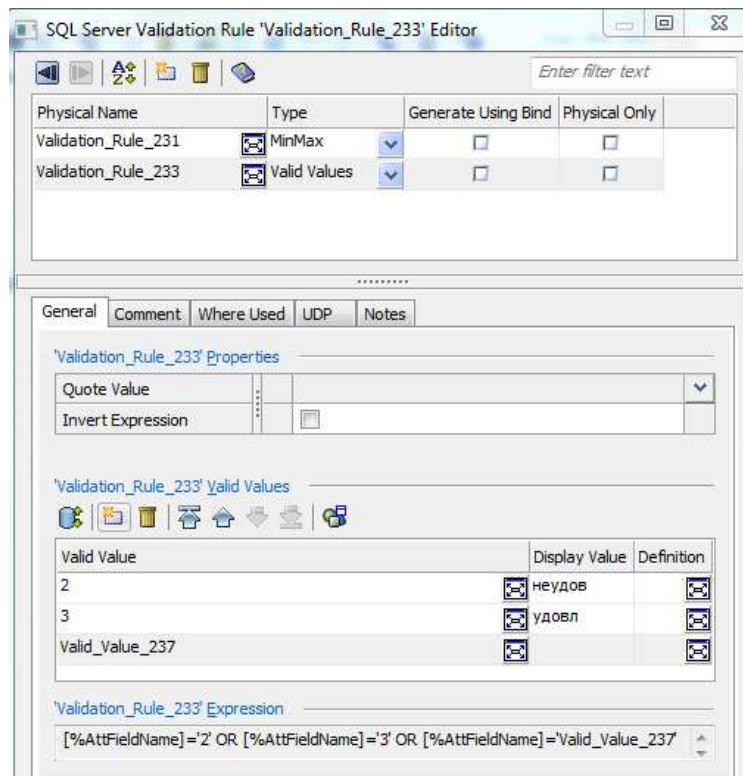


Рисунок 3.3 – Задание списка допустимых значений

На основе выбранных стратегий среда **ERwin DM** автоматически создаст на основе шаблонов триггеры на диалекте SQL целевой СУБД.

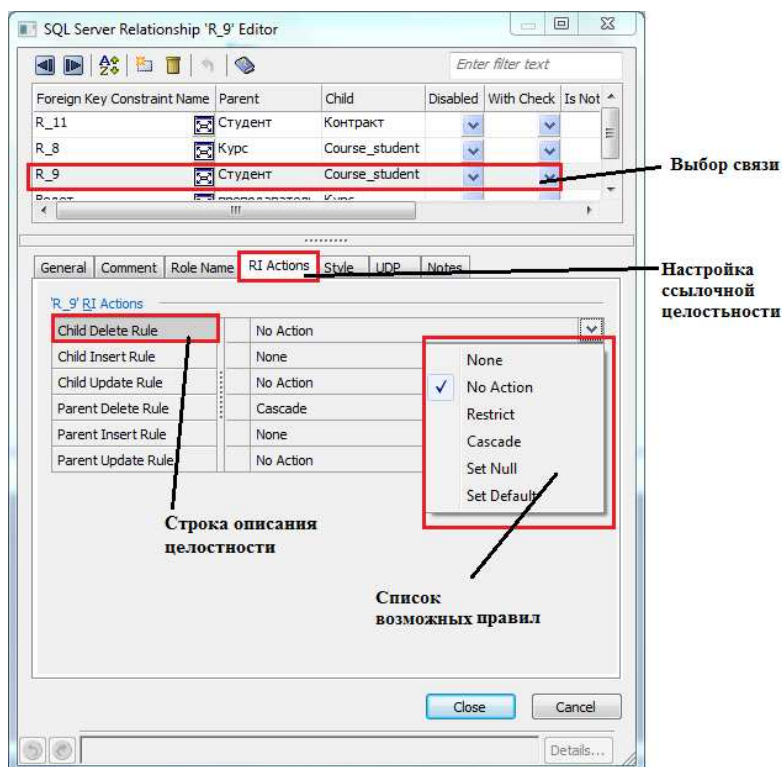


Рисунок 3.4 – Задание правил ссылочной целостности в редакторе свойств связи Erwin DM

### 3. Создание БД в среде целевой СУБД в AllFusion ERwin Data Modeler

Важным инструментом работы с физической моделью [8] **ERwin DM** является возможность перенести созданную модель в среду выбранной СУБД. **ERwin DM** поддерживает более 20 СУБД (рис. 3.5), среди них: **Oracle, Access, DB2, SQL Server, Teradata, Sybase, Informix**, однако конкретный набор поддерживаемых СУБД зависит от поставленной конфигурации. Конкретная СУБД может быть выбрана в момент создания модели или позднее. Во втором случае используется соответствующий пункт меню **Actions**.

Создание описания модели БД средствами языка **SQL** конкретной СУБД (прямое проектирование – **Forward Engineer**) запускается из пункта **Actions** меню работы при работе с моделью типа **Physical**. Результатом его работы является создание объектов БД: таблиц, тиггеров, правил. Для того чтобы посмотреть программный код описания создаваемых в СУБД объектов БД, можно сгенерировать текстовый документ на языке **SQL** выбранной СУБД.

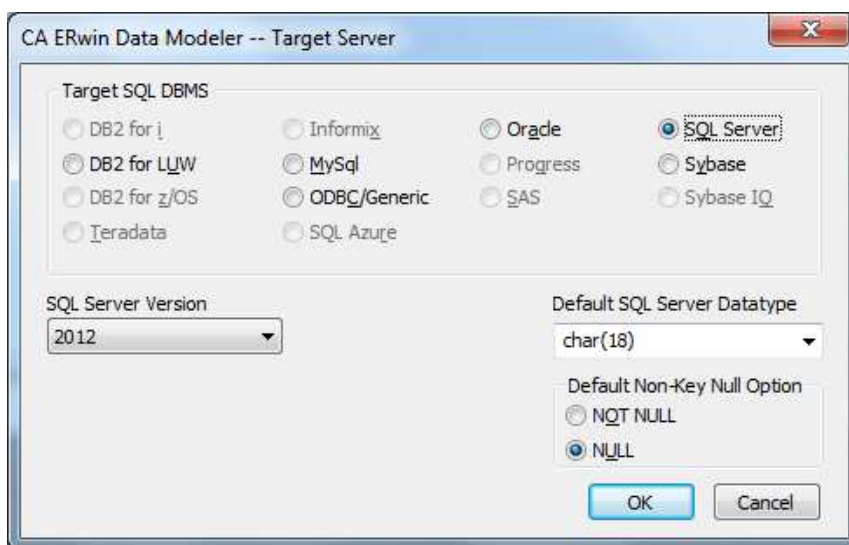


Рисунок 3.5 – Выбор СУБД

Перед созданием БД необходимо проверить соединение с СУБД. Оно позволяет в диалоговом режиме определить параметры соединения (рис. 3.6). Состав параметров зависит от используемой СУБД.



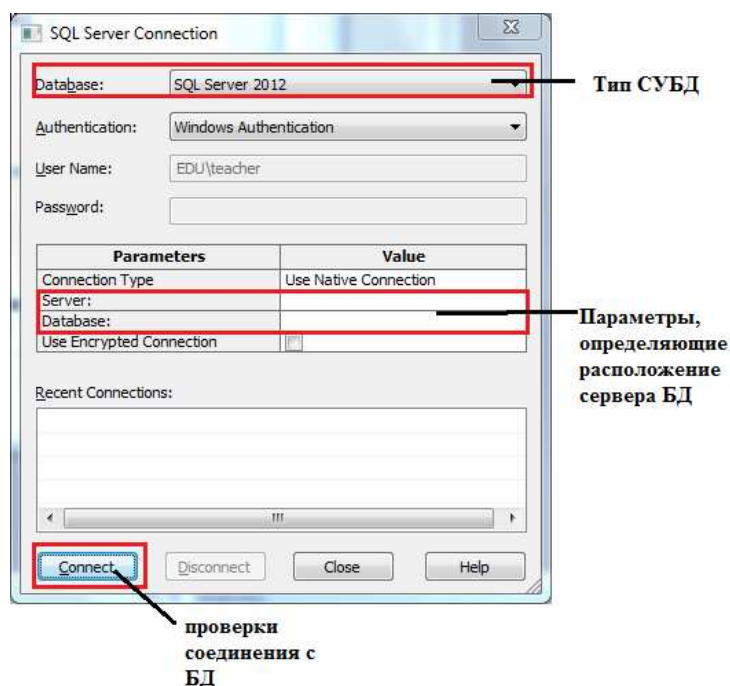


Рисунок 3.6 – Окно задания параметров соединения с БД

Для запуска процесса генерации схемы БД следует перейти по следующим пунктам основного меню **Erwin DM: Action→Forward Engineer/Schema Generation**. При этом откроется диалоговое окно (рис. 3.7), на котором настраиваются параметры процесса прямого проектирования и выбираются необходимые действия.

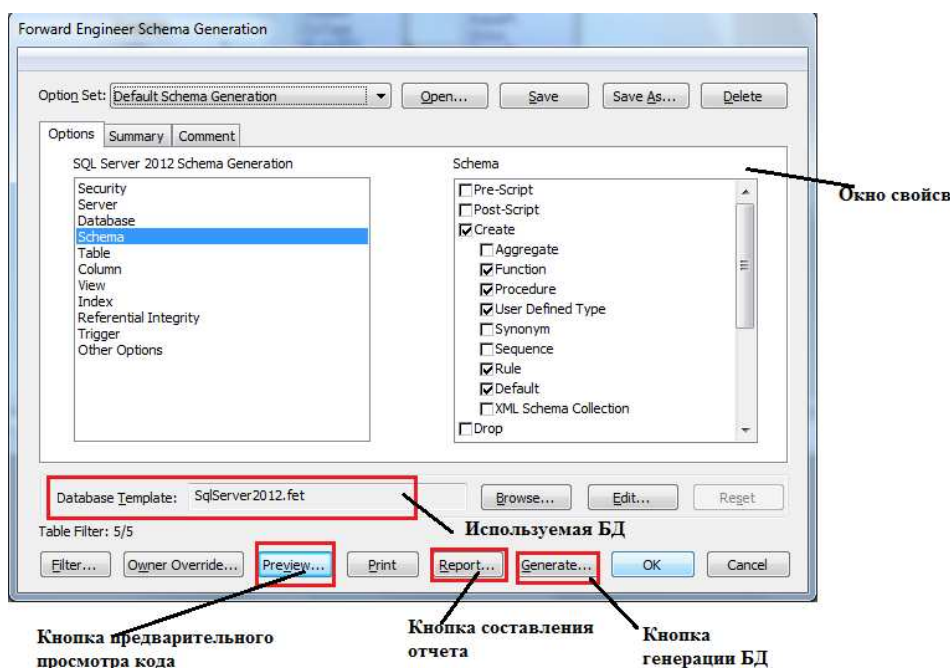


Рисунок 3.7 – Окно диалога генерации схемы БД

Результат генерации кода описания структуры таблиц БД приведён на рисунке 3.8.

```
CREATE TABLE [Курс]
(
    [IdCourse]      varchar(20) NOT NULL ,
    [Title_Cr]      char(18) NULL ,
    [Content]       nvarchar() NULL ,
    [CrsType]       tinyint NULL ,
    [Id_pr]         integer NULL
)
go

ALTER TABLE [Курс]
    ADD CONSTRAINT [XPKКурс] PRIMARY KEY CLUSTERED ([IdCourse] ASC)
go

CREATE TABLE [преподаватель]
(
    [Id_pr]         integer NOT NULL ,
    [NamePr]        varchar(20) NULL ,
    [Ststus]        nchar() NULL ,
    [Cont_Pr]       varchar(20) NULL
)
```

Рисунок 3.8 – Описания таблиц БД на языке SQL

#### 4. Последовательность выполнения практической работы

1. На основе анализа смысловых правил обработки данных в предметной области сформулировать декларативные правила целостности в модели РБД: ограничения на значения атрибутов, ограничения ссылочной целостности.

**Указания.** При задании ограничений на значения атрибутов предусмотреть различные способы задания правил.

2. Используя возможности CASE-средства, задать необходимые декларативные ограничения целостности.

3. Провести анализ правил ссылочной целостности в предметной области.



**Указания.** При задании правил ссылочной целостности определить стратегии для двух выбранных связей и операций добавления, удаления и изменения кортежей родительской и дочерней таблицы.

4. Используя возможности CASE-средства, создать описание модели РБД, включающей правила целостности.

5. При использовании **ERwin DM** создать описание БД в выбранной СУБД.

6. Оформить отчёт по практической работе.

### **5. Требования к оформлению отчета**

Отчёт по практической работе должен включать следующие разделы:

- 1) титульный лист;
- 2) графическая реляционная модель БД;
- 3) смысловое и формализованное описание правил целостности;
- 4) текст сгенерированного средствами **ERwin DM** описания БД в СУБД;
- 5) ответы на контрольные вопросы.

### **Контрольные вопросы**

1. Назовите категории целостности данных в БД.
2. Определите назначение ограничений целостности. Приведите пример смысловых правил, задающих целостность данных в БД.
3. Как задать правила целостности для значения атрибута?
4. Как будет выполняться удаление кортежа из родительской таблицы, если используется стратегия **RESTRICT**?
5. Как задать ограничения в виде диапазона допустимых значений в среде **ERWin DM**?
6. Определите стандартные стратегии поддержки ссылочной целостности.

## Практическая работа 4

### СОЗДАНИЕ ТАБЛИЦ БД В СУБД ACCESS

**Цель работы** – создание схемы БД в СУБД Access.

#### 1. Теоретические основы

БД реализуется в СУБД **Microsoft Access**. Эта СУБД широко применяется для создания персональных БД, т.к. в ней предусмотрен широкий набор простых и удобных средств, значительно упрощающих создание и использование БД даже пользователем-непрограммистом. **СУБД Access** полностью удовлетворяет требованиям реляционных СУБД. При выполнении практических работ используется **СУБД Access** версии не ниже **Access 2010**.

В БД Access можно создавать объекты разного типа: таблицы, запросы, формы, отчёты, макросы. Следует отметить, что вся БД хранится в одном файле, поэтому прежде чем их создавать, необходимо создать саму БД, т.е. файл БД, который имеет расширение **accdb**. Такой файл необходимо создать при первом запуске СУБД Access (рис. 4.1).

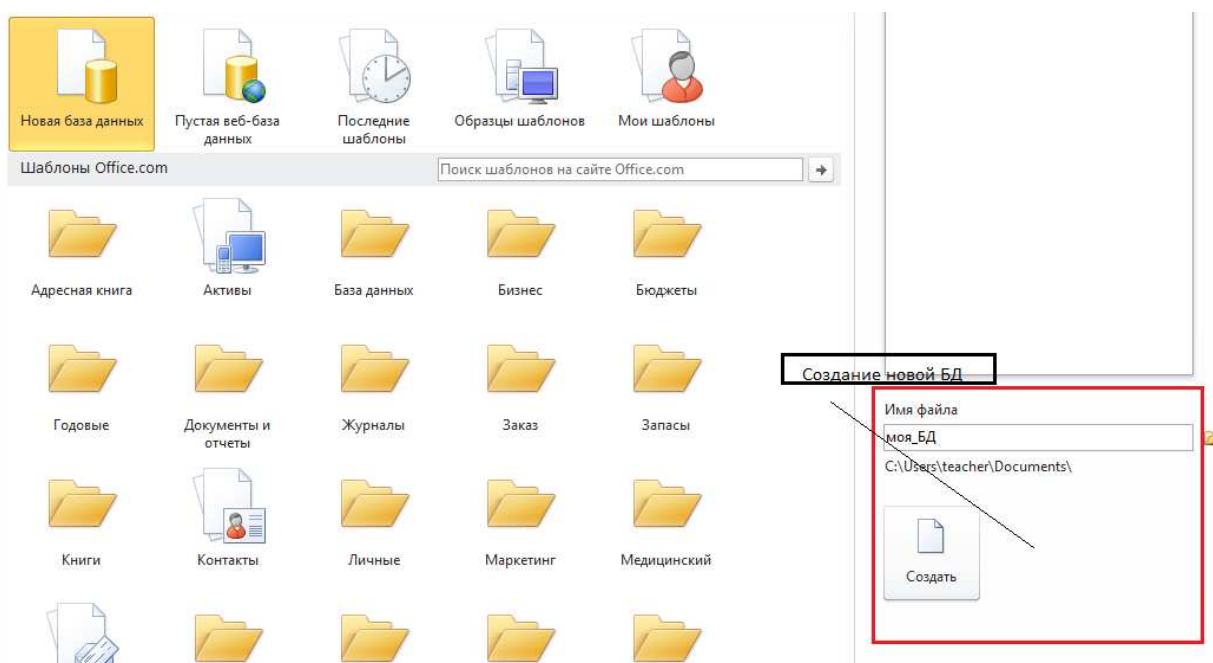


Рисунок 4.1 – Создание файла БД

В дальнейшем можно работать как с новой БД, так и открывать уже созданную.

После создания файла БД открывается основное окно **СУБД Access**, панель инструментов которой (рис. 4.2) позволяет создавать

все объекты БД. Для этого используется вкладка **Создание**, инструменты которой позволяют создавать объекты любого класса.

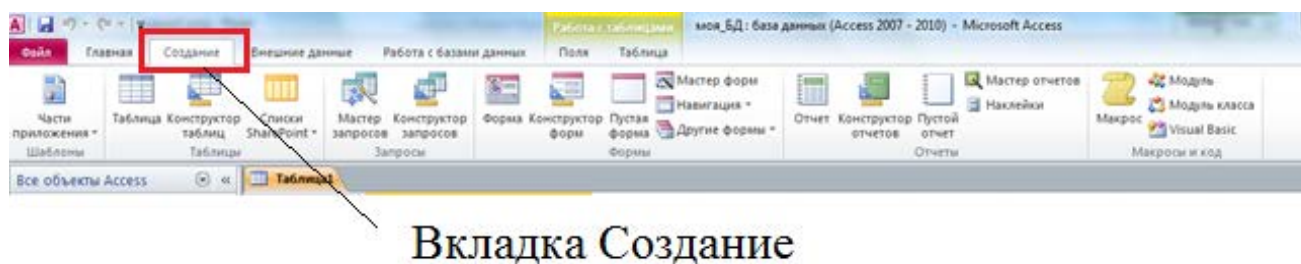


Рисунок 4.2 – Панель инструментов СУБД Access

## 2. Таблицы

Основными объектами **Access** являются таблицы. Именно они позволяют определять структуру данных и хранить данные. Остальные объекты БД создаются на основе данных таблицы.

В используемой версии **СУБД Access** таблицы создаются несколькими способами:

- на основе шаблонов, которые можно скопировать из Интернета;
- простым вводом данных в автоматически открывающуюся таблицу;
- Конструктором таблиц.

При выполнении практических заданий для определения структуры данных в таблице используется **Конструктор таблиц**, который позволяет сформировать структуру, полностью отвечающую созданной модели РБД (рис. 4.3).

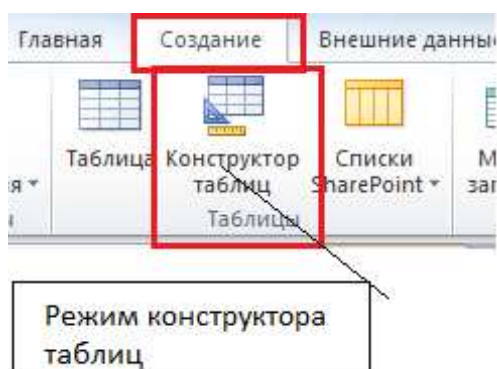


Рисунок 4.3 – Выбор режима конструктора таблиц

В открывшемся окне **Конструктора таблиц** необходимо последовательно описать все столбцы таблицы (рис. 4.4) и задать ключевое поле.

студент		
Имя поля	Тип данных	
ФИО	Текстовый	фамилия имя отчество студента
конт	Текстовый	контактные данные
скидка	Числовой	размер скидки
номст	Числовой	номер студента

Ключевое поле

Рисунок 4.4 – Описание структуры таблицы

Тип данных для каждого столбца таблицы задаётся выбором из существующих вариантов (рис. 4.5 а). Для некоторых полей целесообразно задавать дополнительные описатели (рис. 4.6 б) – размер поля, значение по умолчанию. Состав дополнительных описателей зависит от выбранного типа данных.

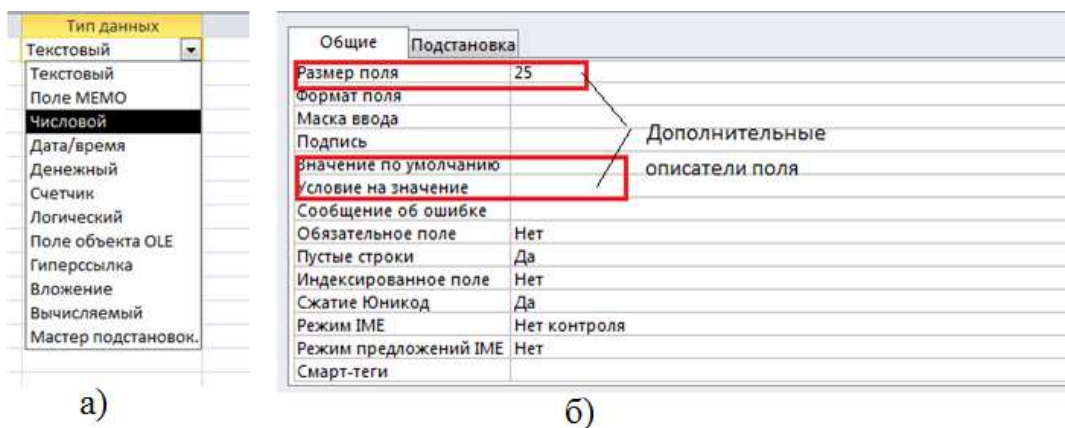


Рисунок 4.5 – Описание типа данных столбца таблицы

Задание первичного ключа поля таблицы является обязательным. Чтобы определить поле как ключевое, необходимо выделить строку с описанием этого поля и установить признак ключа (рис. 4.6). Если первичный ключ поля состоит из нескольких полей, то их следует выделить одновременно, используя левую кнопку мыши и кнопку **Shift**, и установить признак ключа.

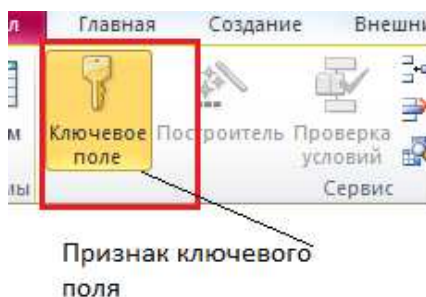


Рисунок 4.6 – Определение ключевого поля

Состав всех объектов БД можно посмотреть в **Обозревателе объектов**. Он расположен в левой части интерфейсного окна. Вид отражения объектов в **Обозревателе** можно изменять, используя контекстное меню (рис. 4.7).

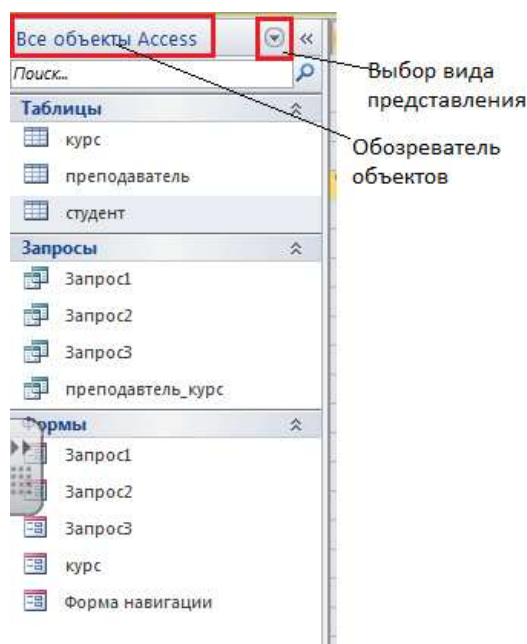


Рисунок 4.7 – Обозреватель объектов БД

Связи между таблицами БД определяют схему БД. В рамках выполняемых практических работ с **СУБД Access** схема БД создаётся на основе реляционной модели, разработанной на практическом занятии 2.

Схема **БД Access** показывает состав таблиц и связи между ними. Для создания схемы БД необходимо выбрать вкладку **Работа с базами данных** и на открывшейся панели инструментов режим **Схема базы данных** (рис. 4.8).

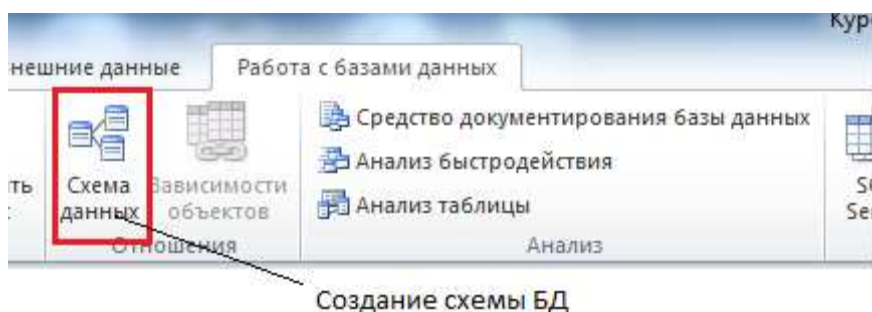


Рисунок 4.8 – Вкладка создания схемы БД

Для создания схемы БД необходимо в рабочее поле добавить таблицы и определить связи между ними. Для добавления таблицы из контекстного меню, которое вызывается нажатием правой кнопки

мышью, выбирается режим **Добавить таблицу** и последовательно добавляем созданные таблицы. После того, как добавлены все таблицы, можно определять связи между ними. Связывать между собой следует только две таблицы. Для этого кнопкой мыши фиксируем в таблице поле связи и перетаскиваем её в поле связи второй таблицы. При этом открывается **Редактор связей** (рис 4.9), в котором отображается информация о связываемых таблицах.

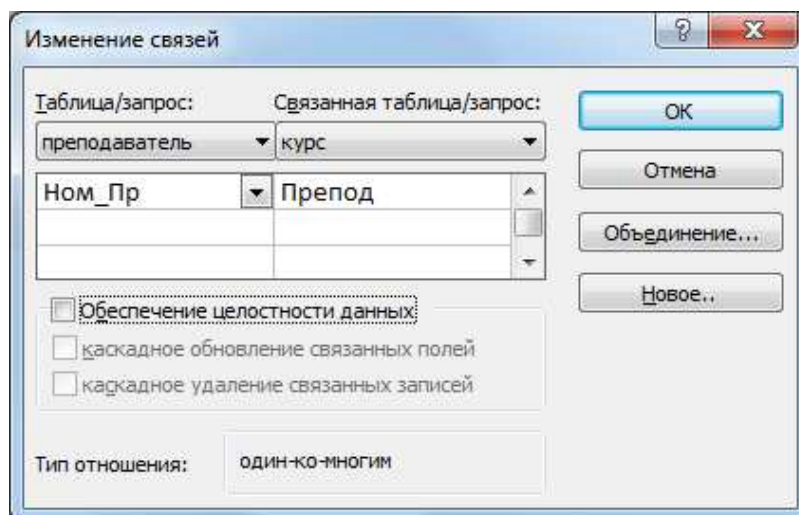


Рисунок 4.9 – Редактор связей

В результате создаётся схема БД, пример которой приведён на рисунке 4.10.

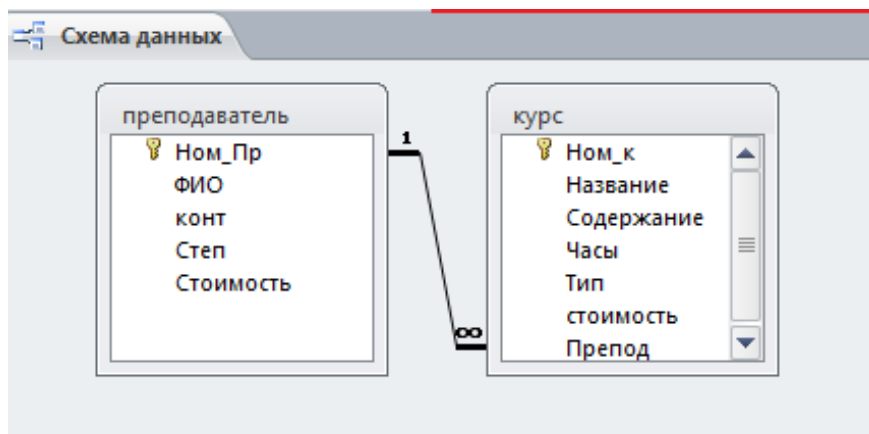


Рисунок 4.10 – Схема БД

Для ввода данных и их изменения можно двойным щелчком мыши открыть таблицу. Новые строки добавляются только в конец таблицы.



### 3. Формы

Формы служат удобным средством для работы с данными из таблицы. С их помощью можно вводить данные в таблицу, редактировать и удалять их.

Формы в **Access** можно создать для просмотра данных нескольких связанных таблиц. Тогда возможным становится с помощью формы вводить данные сразу в несколько таблиц, соблюдая условия целостности данных. Источником данных, которые представляются на форме, может быть таблица или запрос на выборку. Для объектов других типов формы создать нельзя.

В **СУБД Access** можно создавать формы различного формата и несколькими способами. На панели выделено несколько инструментов создания форм (рис. 4. 11):

- автоматическое создание (форма);
- Мастер форм;
- Конструктор формы;
- Специальные виды форм (форма навигации, другие формы).

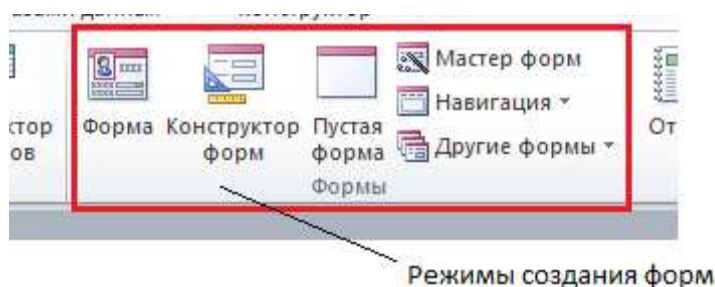


Рисунок 4.11 – Панель инструментов создания форм

Самый простор способ создания формы – автоматическое создание, для этого необходимо в обозревателе объектов выделить таблицу или запрос на выборку и нажать на панели инструментов режим **Форма**. При этом создается форма на основе шаблона, в которой включены все поля таблицы или запроса.

Для того чтобы расположить на форме только часть полей таблицы или запроса следует воспользоваться **Мастером форм**. При этом запускается диалоговый режим, в котором задается источник данных для формы – таблица или запрос, перечень включаемых в форму полей. В **Мастере форм** можно создавать формы на основе следующих шаблонов.

1. В один столбец (полноэкранная форма). При этом форме на экране будут отображаться данные одной строки таблицы (рис. 5.3 а).

Для перехода к следующей строке необходимо использовать кнопки навигации, расположенные внизу формы.

2. Ленточная. На этой форме располагается данные сразу нескольких строк таблицы. Для добавления данных выделяется специальная строка для ввода (рис. 5.3 б).

3. Табличная. Данные в такой форме представлены как в таблице, когда каждой записи соответствует одна строка таблицы, а каждому полю – один столбец.

4. Выровненный. Эта форма представляет одну запись таблицы на экране формы в виде одной строки (5.3 в).

Студенты1

Код: 2

Фамилия: Киршин

Имя: Петр

Отчество: Валерьевич

Номер группы: 151

Телефон: 110-67-82

Стипендия: ☒

Записи: 1 из 15

Навигация по строкам таблицы

а)

Студенты2

Код	Фамилия	Имя	Отчество	Номер группы	Телефон	Сти
2	Киршин	Петр	Валерьевич	151	110-67-82	<input checked="" type="checkbox"/>
3	Кривинский	Сергей	Николаевич	151	172-97-21	<input type="checkbox"/>
4	Панова	Василиса		150		<input checked="" type="checkbox"/>
5	Кульчий	Григорий	Викторович	151	269-53-75	<input checked="" type="checkbox"/>
6	Патрикеев	Олег	Борисович	252	234-11-63	<input type="checkbox"/>
7	Перлов	Кирилл	Николаевич	252	312-21-33	<input type="checkbox"/>
*						<input type="checkbox"/>

Строка для добавления новых данных

б)

Студенты3

Код	Фамилия	Имя	Отчество	Номер группы	Телефон	Стипендия
2	Киршин	Петр	Валерьевич	151	110-67-82	<input checked="" type="checkbox"/>

Записи: 1 из 15

Кнопки навигации

в)

Рисунок 5.3 – Формы для ввода и редактирования данных в таблице



Для создания формы собственного дизайна используется **Конструктор форм**. Для этого надо обладать навыками программирования, т.к. в этом случае форма создаётся как экранная форма в любой другой среде программирования (например Microsoft Visual Studio). Это требует от разработчика дополнительных знаний о проектировании и программировании экранных форм, использовании элементов ввода-вывода данных, элементов управления, таких как кнопки, списки и др. Поэтому целесообразно начальный вариант формы разрабатывать с помощью **Мастера форм**, а приводить её к желаемому виду в **Конструкторе форм**. Инструменты **Конструктора форм** включают все основные элементы управления, располагаемые на формах: кнопки, списки, меню, группа переключателей и др. Используя стандартные приёмы размещения элементов на форме, можно изменить положение полей ввода данных, изменить шрифты и цветовое решение, добавить фоновые картинки и многое другое.

Любые формы, независимо от того, каким образом они были созданы, можно доработать инструментальными средствами **Конструктора форм**.

Для разработки интерфейса с БД Access используются специальные виды форм: **Кнопочная форма** и **Форму навигации**. Как правило, при определении конфигурации СУБД Access включается только одна из специальных видов форм.

#### **4. Последовательность выполнения практической работы**

1. Запустить программу СУБД Access и изучить её интерфейс.
2. Для модели БД, разработанной на практическом занятии 2, в **Конструкторе таблиц** создать таблицы.
3. Для каждой таблицы разработать формы для ввода данных.
4. Заполнить созданные таблицы данными для выполнения запросов, приведённых в задании (Приложение). Для каждой таблицы предусмотреть ввод не менее 7–10 строк данных.
5. Подготовить отчёт по практической работе.

#### **5. Требования к оформлению отчета**

Отчёт по практической работе должен включать следующие разделы:

- титульный лист;
- графическая схема БД в СУБД Access;
- скриншот описания структуры таблицы в режиме **Конструктора таблиц**;

- скриншот формы для ввода данных в таблицы;
- скриншот таблицы с заполненными данными;
- состав объектов БД из обозревателя объектов;
- ответы на контрольные вопросы

### **Контрольные вопросы**

1. Какие группы объектов **БД Access** Вы знаете?
2. Назовите основные элементы интерфейса **СУБД Access**.
3. Какие объекты являются основными в **БД Access**?
4. Какие объекты **БД Access** предназначены для хранения данных?
5. Определите назначение формы **БД Access**?
6. Назовите способы создания форм?

## **Практическая работа 5**

### **СОЗДАНИЕ ЗАПРОСОВ В СУБД ACCESS**

**Цель работы** – создание запросов для обработки данных в СУБД Access.

#### **1. Теоретические основы**

Запросы предназначены для работы с данными. С их помощью можно выбрать необходимые данные, создать новую таблицу по результатам запроса или скопировать данные из одной таблицы в другую. Запросы позволяют удалить сразу несколько строк таблицы, удовлетворяющих определённому условию, или изменить значение данных в нескольких строках таблицы.

**СУБД Access** позволяет создавать запросы различных типов:

- запрос на выборку данных;
- запрос на обновление данных;
- запрос на удаление данных;
- запрос на добавление данных;
- запрос на создание новой таблицы;
- перекрестный запрос.

При работе с БД прежде всего нужно найти необходимые данные. Для этого используется запрос на выборку, который позволяет найти данные, удовлетворяющие заданному условию. Поиск может осуществляться в одной или нескольких таблицах, или во всей БД. Результатом выполнения этого запроса является набор строк таблицы. При формировании условия отбора можно использовать арифметические и логические операции, операции отношения и специальные функции, предусмотренные в СУБД.

Результаты поиска данных можно отсортировать по значению одного или нескольких полей, вычислить итоговые значения на данных отобранных строк.

Запрос на обновление данных позволяет изменить данные для группы строк таблицы.

Запрос на добавление данных позволяет добавить строки данных из одной таблицы в другую.

Запрос на удаление данных позволяет удалить строки, удовлетворяющие определённому критерию.

Запрос на создание таблицы позволяет сформировать новую таблицу.

Результатом перекрестного запроса является набор записей, представленных в специальном формате, напоминающем электронную таблицу.

Запросы любого типа могут использовать параметры, значения которых задаются при выполнении запроса. Введённое значение параметра может быть использовано для задания условия отбора или нового значения изменяемых данных. Это позволяет создать один объект типа запрос для выполнения нескольких вариантов его реализации.

Для создания запросов используют **Мастер запросов** или **Конструктор запросов** (рис. 5.1).

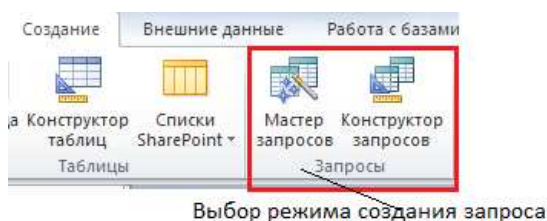


Рисунок 5.1 – Выбор режима создания запроса

**Мастер запросов** инициирует диалоговый процесс создания запроса на выборку. При его использовании нельзя задать условия отбора записей. Чтобы создать запрос любого типа и сложности используют **Конструктор запросов**.

Запросы в **Конструкторе запросов** формулируются в специальной табличной форме (рис. 5.2). В верхней части окна **Конструктора запросов** определяются источники данных – одна или несколько таблиц. Источниками данных могут быть также и другие запросы.

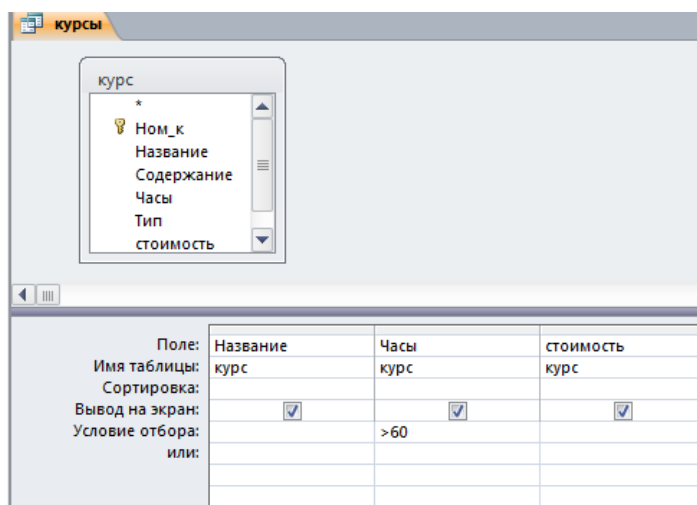


Рисунок 5.2 – Запрос на выборку в Конструкторе запросов

Для добавления нового источника данных выбирается режим **Добавление таблицы**, который вызывается из контекстного меню (рис. 5.3).

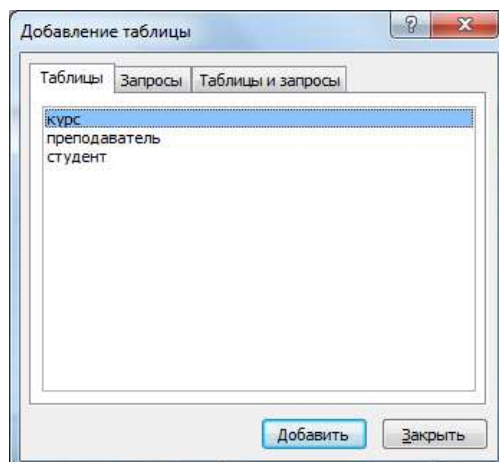


Рисунок 5.3 – Добавление источника данных для запроса

После добавления всех необходимых таблиц откроется поле, в котором можно формулировать запрос. По умолчанию СУБД Access создаёт запрос на выборку. Однако тип запроса можно изменить, выбрав на панели инструментов (рис. 5.4) запрос нужного типа.

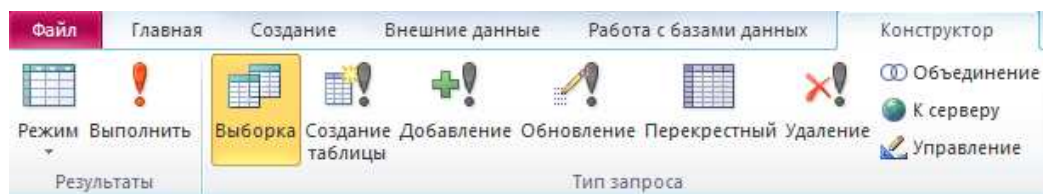


Рисунок 5.4 – Панель инструментов выбора типа запроса

В **Конструкторе запросов** запрос формулируется в табличной форме. Вид этой таблицы зависит от типа запроса.

Рассмотрим создание запроса на выборку данных из одной таблицы.


**Запрос 1.** В таблице **Курс** найти все курсы, длительность которых больше 60 часов. Для них найдём название, длительность и стоимость.

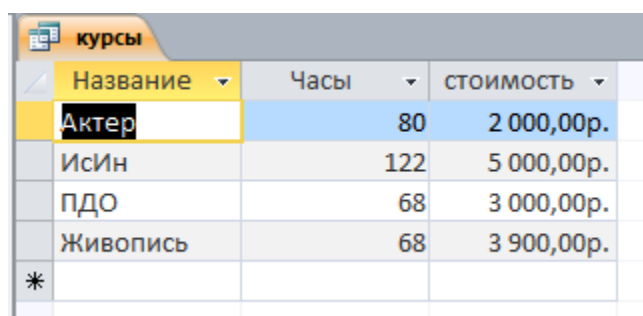
Такой запрос можно создать только с помощью **Конструктора запросов**, т.к. в формулировке задания содержится условие отбора данных. Для запуска **Конструктора запросов** выбираем вкладку **Создание**, раздел **Запросы** и выбираем режим **Конструктор запросов**.

Формирование запроса начинается с определения источника данных, которое задаётся в диалоговом окне **Добавление таблицы** (рис. 5.3). С помощью мыши в окне выберем таблицу **Курс** и добавим её к источникам данных, нажав кнопку **Добавить**. Выбранная таблица будет отображена в области источника данных запроса. Закроем окно **Добавление таблицы**.

Теперь сформируем форму запроса. Для этого определим название полей с нужными данными: с помощью мыши перетащим из списка полей таблицы **Курс** поля **Название**, **Часы**, **Стоимость** в отдельные столбцы формы запроса в строку **Поле**.

Теперь сформулируем условие отбора строк таблицы. Оно записывается как **Часы > 60**.

Это условие запишем в строке **Условие отбора** для столбца **Часы**, нажмём кнопку **Выполнить**  и получим результат выполнения запроса (рис. 5.5).



Название	Часы	стоимость
Актер	80	2 000,00р.
ИсИн	122	5 000,00р.
ПДО	68	3 000,00р.
Живопись	68	3 900,00р.
*		

Рисунок 5.5 – Результат выполнения запроса на выборку данных

Условие отбора строк таблицы может быть сложным и включать условия для значений нескольких столбцов таблицы.

Рассмотрим пример создания более сложного запроса.

**Запрос 2.** В таблице **Курс** найти все курсы, длительность которых лежит в диапазоне 60–120 и стоимость которых больше 3 000 рублей. Для них найдём название, длительность и стоимость.

В этом случае условие отбора строк таблицы определено на значениях двух столбцов и записывается:

*Часы >= 60 and Часы <=120 and Стоимость > 3000*

При записи условия отбора в форме запроса в **Конструкторе запросов** нужно определить указанные условия в двух столбцах строки **Условие отбора** (рис. 5.6).

Поле:	Название	Часы	стоимость
Имя таблицы:	курс	курс	курс
Сортировка:			
Вывод на экран:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Условие отбора:		>=60 And <=120	>3000
или:			

Рисунок 5.6 – Определение сложного условия отбора

Результат выполнения запроса приведён на рисунке 5.7.

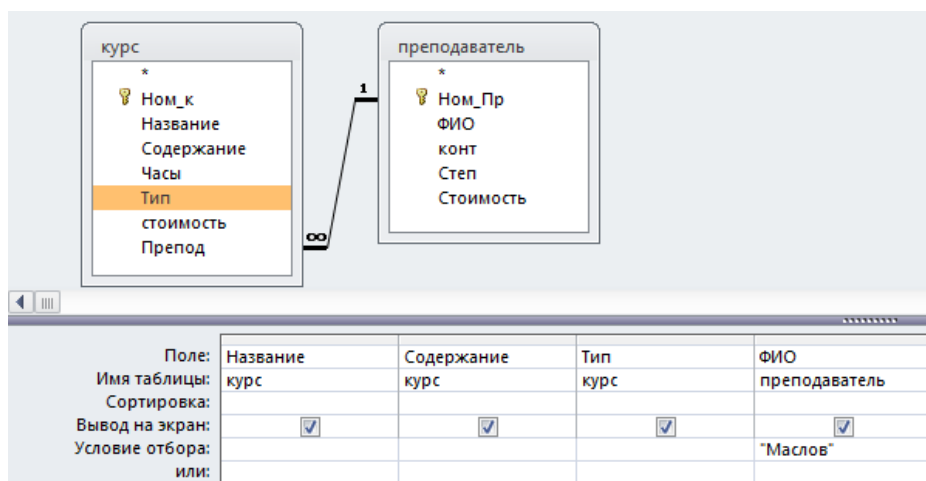
Название	Часы	стоимость
ИГ	60	3 500,00р.
Живопись	68	3 900,00р.
*		

Рисунок 5.7 – Результат выполнения запроса со сложным условием отбора

При поиске данных в запросе на выборку можно использовать данные сразу нескольких таблиц. В этом случае при определении источника данных необходимо последовательно в окне добавления таблиц (рис. 5.3) добавить все таблицы, которые содержат используемые в запросе данные. Определение условий отбора происходит так же, как в описанных выше примерах.

**Запрос 3.** Найти все курсы, которые ведёт преподаватель Маслов. Для них найдём название, длительность и тип. Для того чтобы выполнить это запрос, потребуются данные двух таблиц **Курс** и **Преподаватель**.

Форма запроса и результат его применения приведены на рисунке 5.8.



а) Форма запроса в Конструкторе запросов

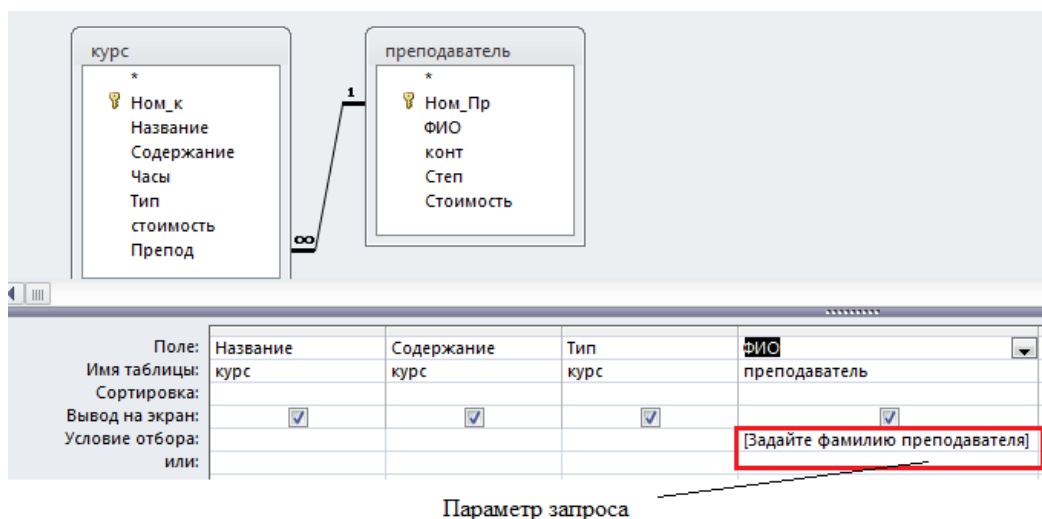
Название	Содержание	Тип	ФИО
Пс	Основы психологии	начальный	Маслов
ПДО	Психология дошкольн	продвинутый	Маслов
*			

б) результат выполнения запроса

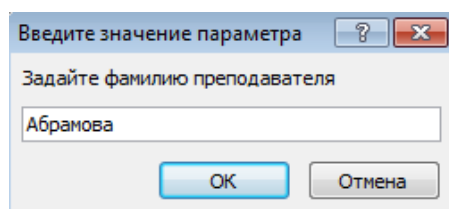
Рисунок 5.8 – Запрос на поиск данных из двух таблиц

Для того чтобы создать однотипные запросы, которые отличаются только конкретными значениями условий обработки данных, можно использовать параметры и создавать запросы с параметрами. Параметры следует задавать для запросов любого типа. Для одного запроса допускается задание нескольких параметров. Параметр запроса задаётся текстом, заключённым в квадратные скобки. Для задания параметра для условия отбора, например, следует в соответствующем столбце строки **Условие отбора** в квадратных скобках задать любой текст. Этот текст не должен совпадать с именем объекта БД (таблицы, атрибута таблицы, запроса и др.). Перед выполнением запроса на экран выводится специальное диалоговое окно с приглашением ввести значение параметра. Если в запросе определено несколько параметров, то окна ввода значений параметров будут появляться последовательно. Таким образом, выполнение одного запроса будет иметь разные результаты при определении разных значений его параметров. Пример запроса с параметром и результат его выполнения приведены на рисунке 5.9.





а) Форма запроса с параметром в Конструкторе запросов



б) Задание значения параметра

курсы	преподаватель_курс			
Название	Содержание	Тип	ФИО	
Дизайн	Промышленный дизайн	продвинутый	Абрамова	
ИИ	История искусств	начальный	Абрамова	
Живопись	Живопись	продвинутый	Абрамова	
*				

в) результат выполнения запроса

Рисунок 5.9 – Запрос на поиск данных из двух таблиц

Запросы на обновление данных и удаления данных можно создать только в **Конструкторе таблиц**. Результаты их выполнения в отличие от результатов выполнения запроса на выборку не выводятся на экран. Запрос на обновление позволяет изменить уже существующие данные только в одной таблице. Рассмотрим создание запроса на обновление для выполнения следующего запроса.

**Запрос 4.** Изменить значение скидки для всех студентов, установив её равной 3.

Первые шаги по созданию запроса выполняются так же, как и в случае создания запросов на выборку данных. После определения источника данных можно изменить тип запроса, выбрав на панели инструментов запрос на обновление (рис. 5.10).

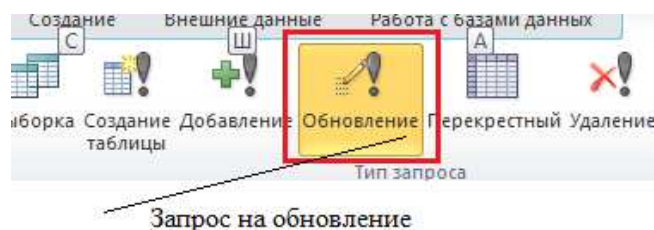
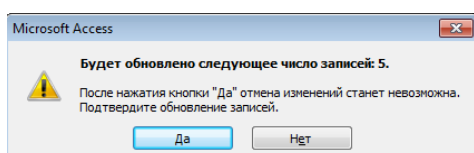


Рисунок 5.10 – Выбор типа запроса на обновление

В форме запроса на обновление определяются (рис. 5.11) имена изменяемых полей, новое значение (строка **Обновление**) и условия отбора обновляемых строк (строка **Условие отбора**). Одним запросом на обновление можно изменить значения сразу нескольких полей.

Новое значение данных в таблице может быть задано сложным выражением. Для его формирования целесообразно использовать специальный инструмент, который называется **Построитель выражений**. Он позволяет создавать сложное выражение на основе выбора необходимых имён полей из таблиц, запросов, знаков операций, функций, что позволяет избежать синтаксических ошибок в задании элементов запроса.

а) Форма запроса на обновление в Конструкторе запросов



б) Сообщение системы при выполнении запроса на обновление

ФИО	конт	скидка	номст	И
Иванов Сергей	12345	3	1	
Самойлова Ирина	23456	3	2	
Пирогов Олег	87654	3	3	
Баталов Николай		3	7	
Соколова Анна	3456	3	11	
*		0	0	

в) результат выполнения запроса на обновление

Рисунок 5.11 – Выбор типа запроса на обновление

Рассмотрим использование построителя выражений на примере создания следующего запроса.

**Запрос 5.** Увеличить на 10 % стоимость часа преподавателей, имеющих степень доктора наук.

Чтобы использовать **Построитель выражений**, следует воспользоваться контекстным меню, которое вызывается правой кнопкой мыши. В нашем примере курсор мыши должен быть установлен на строке **Обновление** поля **Стоимость**. В левой части окна **Построитель выражений** (рис. 5.12) выберем таблицу **Преподаватель**, в которой обновляются данные. Справа отобразится список полей таблицы. Выбираем нужное поле **Стоимость** и добавляем его двойным щелчком мыши в окно построителя. Знаки операций можно вводить с клавиатуры. При этом в верхней части окна сформируется выражение (рис. 5.12).

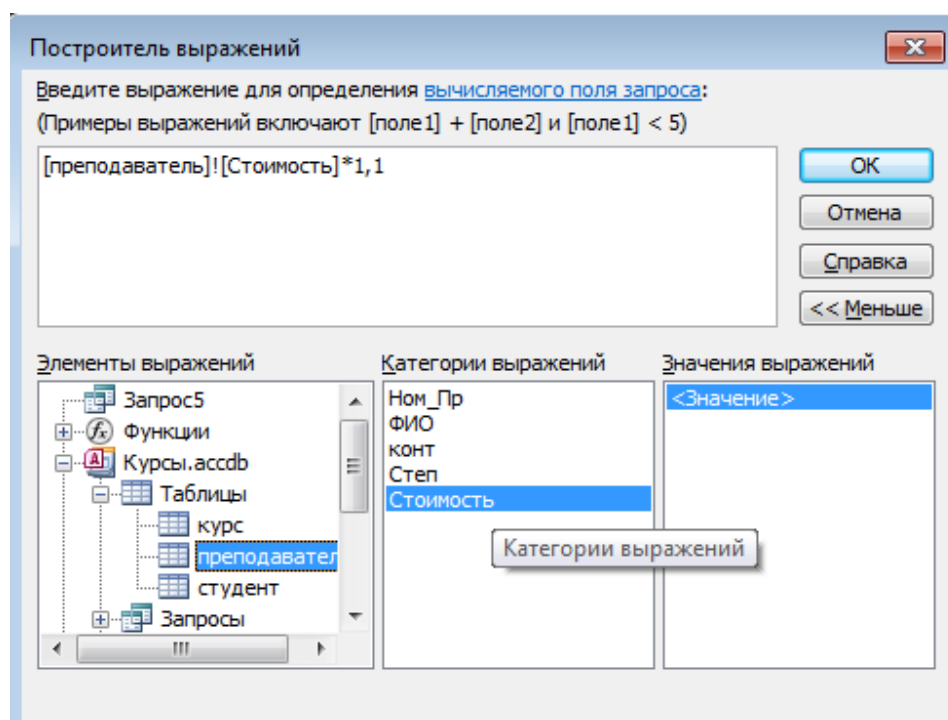


Рисунок 5.12 – Использование построителя выражений для создания сложного выражения

После этого **Построитель выражений** можно закрыть, а в форме запроса на обновление появится созданное выражение (рис. 5.13).

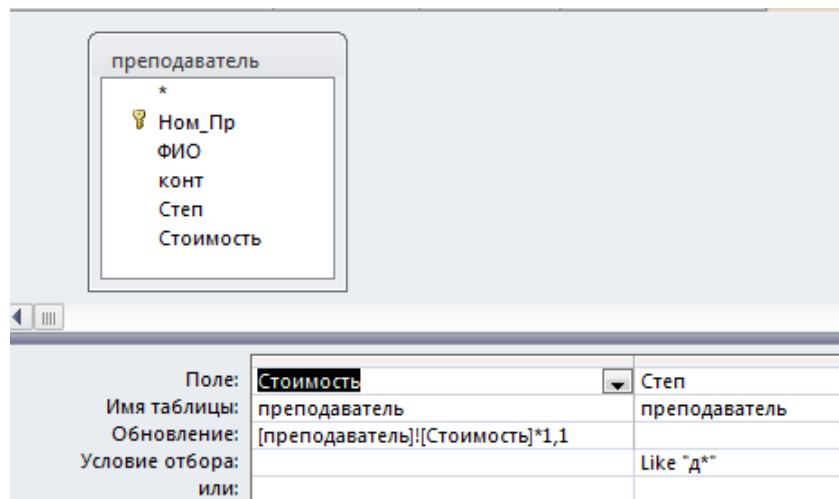
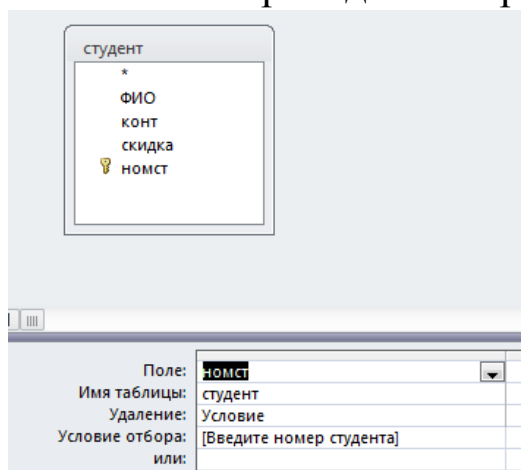


Рисунок 5.13 – Результат применения построителя выражений

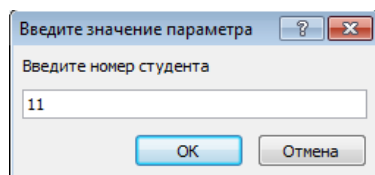
Пример создания запроса на удаление рассмотрим на примере.

**Запрос 6.** Удалить данные о студенте с заданным номером.

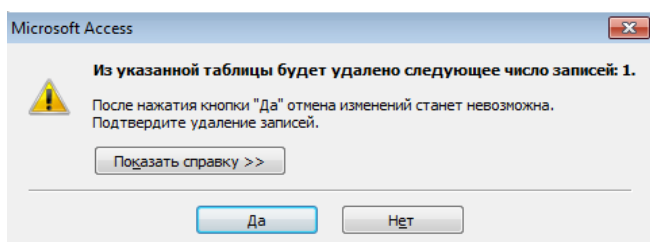
Номер студента зададим параметром. Форма запроса и результат его выполнения приведены на рисунке 5.14.



а) форма запроса на удаление в Конструкторе запросов



б) ввод значения параметра запроса



в) результат выполнения запроса на удаление

Рисунок 5.14 – Форма запрос на удаление и результат его выполнения

В общем случае для формирования запроса в режиме **Конструктора запросов** необходимо выполнить следующие шаги.

1. На ленте основного меню системы выбрать вкладку **Создать – Запрос**.

2. На вкладке **Запрос** выбрать инструмент **Конструктор запросов**.

3. Определить источник данных для запроса (одну или несколько таблиц).

4. Выбрать тип запроса.

5. В табличной форме определить поля таблиц (используемых в запросе).

6. Сформировать условие обработки.

7. Сохранить запрос.

Для определения условия отбора можно использовать специальные операторы **Between, In, Like**.

Оператор **Between** используется, если надо определить диапазон правильных значений для столбца таблицы.

Для проверки, содержится ли значение в заданном списке, можно использовать логический оператор **Or**. Однако удобнее использовать оператор **In**. Он позволяет определить список правильных значений для столбца таблицы. Если набор значений невелик (не более 7–9 значений), то можно задать условие

*In ("к.т.н"; "д.т.н."; "преподаватель").*

Для столбцов, в которых хранятся текстовые данные и данные типа дата, можно задать шаблон, которому они должны удовлетворять. Для задания шаблона используется оператор **Like**. При задании шаблона можно использовать следующие служебные символы:

- \* – заменяет любое количество любых символов,
- ? – заменяет одну любую цифру,
- \_ – заменяет один любой символ (в том числе и цифру).

## **2. Последовательность выполнения практической работы**

1. Запустить программу **Microsoft Access** и открыть созданную на практическом занятии 4 базу данных.

2. Изучить интерфейс **СУБД Access** для создания запросов.

3. С помощью **Конструктора запросов** разработать запросы для реализации основных операций с данными каждой таблицы (состав

запросов на обработку данных приведён в Приложении). При построении запросов необходимо предусмотреть возможность ввода значений его параметров.

4. Создать формы для реализованных запросов на поиск данных.

5. Подготовить отчёт по работе.

### **3. Требования к оформлению отчета**

Отчёт по практической работе должен включать следующие разделы:

1) титульный лист;

2) схема БД в **СУБД Access**;

3) скриншот инспектора объектов для объектов типа **Запросы**;

4) скриншот всех созданных объектов **БД Access**;

5) скриншоты выполненных запросов;

6) ответы на контрольные вопросы.

### **Контрольные вопросы**

1. Назовите тип объектов **БД Access**, которые используются для обработки данных.

2. Назовите средства создания запросов. В чем их отличия?

3. Определите типы запросов в **СУБД Access**.

4. Для чего используются параметры в запросе?

5. Как можно задать параметр в запросе?

6. Для чего используется **Построитель выражений**?

7. Для чего используется оператор **Like**?

## Практическая работа 6

### СОЗДАНИЕ ОТЧЕТОВ В СУБД ACCESS

**Цель работы** – получение навыков работы с фильтрами и создания отчетов по данным базы данных.

#### 1. Теоретические основы

##### 1.1 Фильтры

Фильтрация данных представляет собой удобный инструмент для отображения нужных данных. Они позволяют ограничить объем выводимых на экран строк формы, запроса или таблицы.

Для задания фильтрации необходимо открыть таблицу в обозревателе объектов, выбрать вкладку **Сортировка и фильтр** (рис. 6.1) и нажать знак включения фильтра

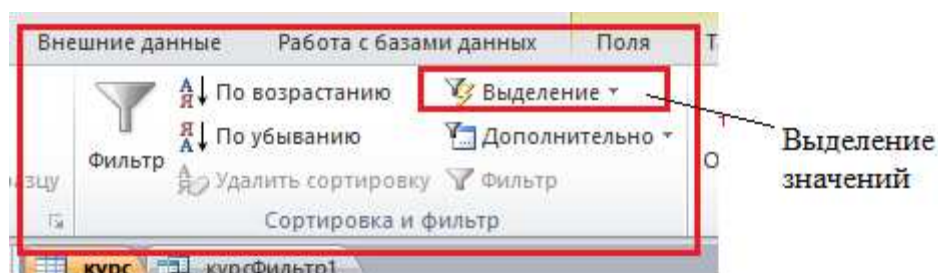


Рисунок 6.1– Режим использование фильтров

В Access допускается несколько способов задания фильтров:

- общие фильтры (рис. 6.2) всегда под рукой, с их помощью можно задать значение фильтра или применять готовые фильтры сравнения;
- фильтрация по выделению используют, если вы ищете строки, содержащие выделенное значение;
- расширенный фильтр позволяет получить данные, которые отслеживают значения в нескольких полях таблицы.

Применение общего фильтра похоже на использование фильтрации в таблицах **Excel**. Чтобы задать одно значение для выбранного столбца, следует использовать выпадающий список. Он содержит все значения выбранного столбца. В этом списке необходимо выделить (установить флажок) одно или несколько значений. Фильтр можно определить на основе шаблона, используя те же служебные символы, которые используются в операторе **Like**.

Курс	КурсФильтр1	Ном_к	Название	Содержание	Часы	Тип	стоимость
		1	БД	проектирование БД	42	начальный	1 000,00р.
		2	ИС	Основы проектирования	45	продвинутой	1 500,00р.
		3	АЯ	Разговорный английский	60	начальный	1 500,00р.
		4	Дизайн	Промышленный дизайн	60	продвинутой	
		8	Актер	Актерское мастерство	80	начальный	2 000,00р.
		9	Рисунок	Основы рисунка	40	начальный	2 500,00р.
		10	Графика	технологии графика	32	продвинутой	3 000,00р.
		11	ИИ	История искусств	60	начальный	1 500,00р.

Рисунок 6.2 – Включение общего фильтра

Для быстрой фильтрации можно выбрать команду **Выделение** (рис. 6.1). Фильтрация «по выделенному фрагменту» позволит найти те строки таблицы с таким значением в выделенном столбце, которое удовлетворяет заданному условию. Вы получите такой же результат, если щелкните нужную ячейку правой кнопкой мыши.

Например, мы можем найти все строки, в которых в поле **Часы** содержатся значения в диапазоне 45–80 (рис. 6.3). Полученный результат представлен на рисунке 6.4.

Диалог «Диапазон чисел»

Не меньше: 45

Не больше: 80

OK Отмена

Рисунок 6.3 – Задания значений для фильтрации «по выделенному»

ном_к	Название	Содержание	Часы	Тип	стоимость
2	ИС	Основы проектирования	45	продвинутой	1 500,00р.
3	АЯ	Разговорный английский	60	начальный	1 500,00р.
4	Дизайн	Промышленный дизайн	60	продвинутой	
8	Актер	Актерское мастерство	80	начальный	2 000,00р.
11	ИИ	История искусств	60	начальный	1 500,00р.
21	Пс	Основы психологии	60	начальный	2 000,00р.
22	ПДО	Психология дошкольного	68	продвинутой	3 000,00р.
31	ИДМ	История древнего мира	45	продвинутой	2 000,00р.
32	ИГ	История Греции	60	профессионал	3 500,00р.
33	Живопись	Живопись	68	продвинутой	3 900,00р.

Рисунок 6.4 – Результат применения фильтра «по выделенному»

Для задания сложного фильтра, использующего условия для выделения по значению нескольких столбцов таблицы, используется расширенный фильтр. Для применения расширенных фильтров необходимо создавать выражения в формах, которые аналогичны форме запроса в **Конструкторе запроса** (рис.6.5 а). В отличие от

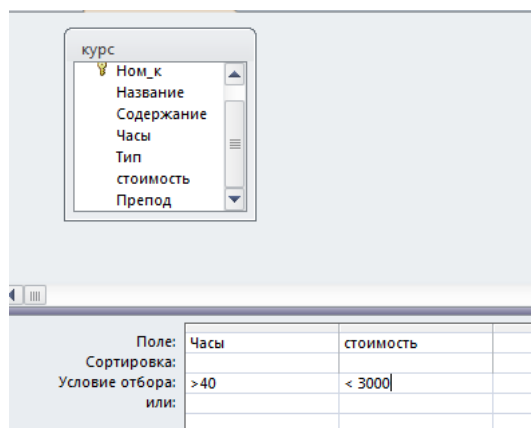


запроса, в поле данных можно определить только одну таблицу. В результате получим строки таблицы, в которых значения данных удовлетворяют заданному сложному условию (рис. 6.5 б).

В приведённом примере (рис. 6.5) осуществляется фильтрация данных по сложному условию: количество часов больше 40, стоимость меньше 3 000 руб.

Основные различия запросов и фильтров заключаются в следующем:

- фильтры позволяют работать с данными только одной таблицы;
- при задании фильтров нельзя определить столбцы, значения которых нужно выводить, всегда выводятся все столбцы таблицы;
- фильтры не являются объектами БД, их нельзя сохранить и для повторного использования придётся настроить фильтр заново;
- фильтры не позволяют проводить вычисления над данными;
- фильтры могут создаваться только при работе с таблицей или формой.



а) Формирование расширенного фильтра

ом_к	Название	Содержание	Часы	Тип	стоимость
2	ИС	Основы проектировани	45	продвинутой	1 500,00р.
3	АЯ	Разговорный английски	60	начальный	1 500,00р.
4	Дизайн	Промышленный дизай	60	продвинутый	
8	Актер	Актрское мастерство	80	начальный	2 000,00р.
11	ИИ	История искусств	60	начальный	1 500,00р.
21	Пс	Основы психологии	60	начальный	2 000,00р.
22	ПДО	Психология дошкольнс	68	продвинутый	3 000,00р.
31	ИДМ	История древнего мир	45	продвинутый	2 000,00р.
32	ИГ	История Греции	60	профессионал	3 500,00р.
33	Живопись	Живопись	68	продвинутый	3 900,00р.

б) результат применения расширенного фильтра

Рисунок 6.5 – Использование расширенного фильтра

## 1.2 Отчеты

Рассмотренные средства работы с данными БД не позволяют сохранить результаты вне БД. Для того чтобы сделать результаты работы с БД в других формах документов, например, текстовом документе, следует использовать отчёты. Отчёт формирует специальный объект, который можно экспортировать, например, в документ, который затем подлежит распечатке. Отчёты содержат данные из таблиц или запросов. В них могут содержаться дополнительные элементы: колонтитулы, как в документах Word, пояснительный текст, графики и рисунки. Данные отчёта могут быть отсортированы по одному или нескольким полям, их можно группировать и включать статистические данные, вычисленные на этих данных. В **СУБД Access** существует несколько способов создания отчётов (рис. 6.6):

- автоматическое создание отчётов;
- создание с помощью **Мастера отчетов**;
- создание с помощью **Конструктора отчетов**.

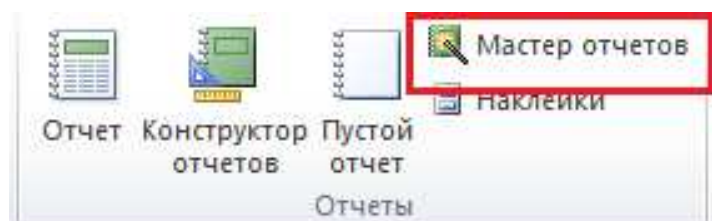
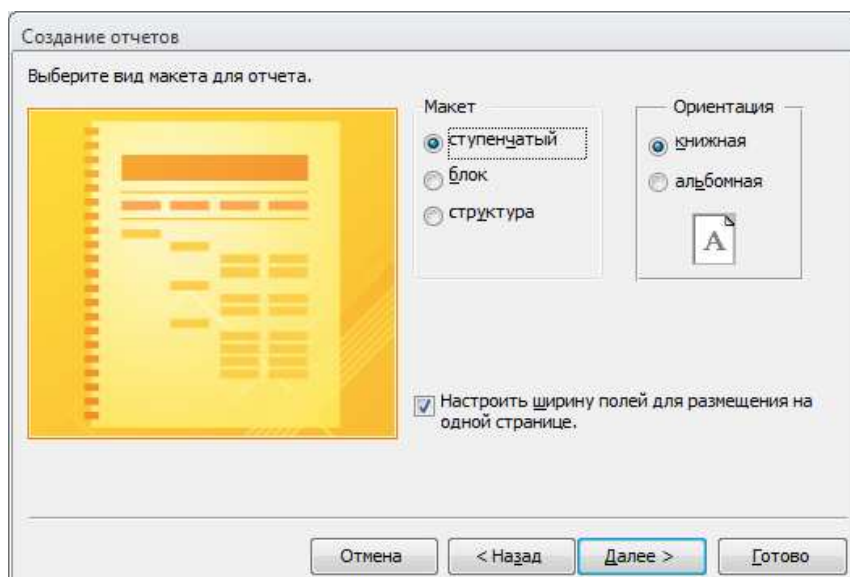


Рисунок 6.6 – Режимы создания отчётов

Самый простой способ создания отчётов – автоматическое создание. Однако в этом случае нет возможности сделать отчёт в нужной вам форме. В практической работе используются средства **Мастера отчетов** для их создания. **Мастер отчетов** позволяет создать отчёт нужного формата, который разрабатывается на основе существующих в системе шаблонов (рис 6.7): ступенчатый, блок, структура, использовать группировку и сортировку данных при формировании отчёта (рис. 6.8).



**Рисунок 6.7 – Форматы макета отчета**

Студенты			
Номер группы	Фамилия	Имя	Отчество
150	Аронова	Ольга	
	Власова	Ирина	
	Громов	Николай	
	Дунаев	Андрей	
	Панова	Василиса	
	Пирогова	Нина	
	Сомов	Олег	
151	Киршин	Петр	Валерьевич
	Кривинский	Сергей	Николаевич
	Кульчий	Григорий	Викторович
252	Патрикеев	Олег	Борисович

**Рисунок 6.8 – Фрагмент отчёта по данным таблицы Студенты**

## **2. Последовательность выполнения практической работы**

1. Запустить программу **Microsoft Access** и открыть БД, созданную на предыдущей практической работе.

2. Выполнить фильтрацию данных с применением общих фильтров.

3. Выполнить фильтрацию данных, задав диапазон значений для выделенного столбца.

4. Задать расширенный фильтр и проверить его выполнение.

5. Построить отчёты по данным таблиц и запросов на выборку данных.

6. Оформить отчёт по практической работе.

### **3. Требования к оформлению отчёта**

Отчёт по практической работе должен включать следующие разделы:

- 1) титульный лист;
- 2) схема БД;
- 3) скриншот обозревателя объектов;
- 4) описание фильтров и скриншоты результатов их применения;
- 5) сформированные отчёты;
- 6) скриншоты отчётов;
- 7) выводы по работе;
- 8) ответы на контрольные вопросы.

#### **Контрольные вопросы**

1. Для чего используются фильтры?
2. Как выполнить фильтрацию данных в **БД Access**?
3. Как задать диапазон значений при фильтрации «по выделенному»?
4. Определите Ваши действия при создании расширенного фильтра.
5. Определите ситуации, в которых полезно использовать фильтры.
6. Для чего используются отчёты в **БД Access**?
7. Определите способы создания отчётов.
8. Опишите процесс создания отчёта с использованием **Мастера отчётов**.
9. Назовите отличия фильтров от запросов.

## Практическая работа 7

### СОЗДАНИЕ ИНТЕРФЕЙСА К БД ACCESS

**Цель работы** – создание общего интерфейса с БД Access с помощью инструментальных средств.

#### 1. Основные понятия

##### 1.1 Макросы

При разработке БД можно создавать программные объекты. К ним относятся макросы и модули на **VBA**. Макросы широко применяются в программных продуктах **Microsoft Office** и представляют собой программы, которые используют набор команд из доступного каталога команд. Они применяются для того, чтобы автоматизировать выполнение часто повторяющихся действий. Макрос представляет собой набор макрокоманд, каждая из которых имеет свой набор параметров. Макрос является отдельным объектом **БД Access** и должен иметь уникальное имя. При выполнении макроса автоматически выполняется последовательность определённых в нём команд.

Макросы дают возможность разрабатывать программные объекты без написания программных кодов. Они позволяют значительно ускорить и упростить создание таких приложений для работы с БД, которые не требуют знания специальных возможностей языка программирования **VBA**. С помощью макросов можно расширить возможности функциональных интерфейсов, создаваемых инструментальными средствами **СУБД Access**. С их помощью можно настроить панели инструментов, добавить управляющие элементы на любую форму, связав кнопки формы с выполнением определённых функций.

Для создания макросов в **СУБД Access** не предусмотрено каких-либо мастеров. Все операции по определению состава и параметров команд макроса выполняются в режиме Макрос, который вызывается из вкладки **Создание** в режиме **Макросы и код** (рис. 7.1).

При выполнении практической работы макросы используются для разработки интерфейса с созданной БД.

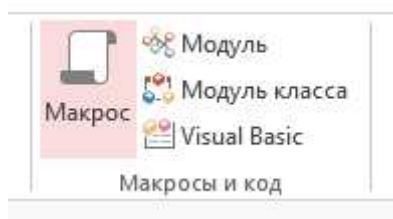


Рисунок 7.1 – Режим создания макроса

Прежде чем создать макрос, необходимо определить последовательность действий, которые он будет выполнять. Затем в диалоговом окне последовательно выбрать из каталога макрокоманд (рис. 7.2) команды макроса и сохранить его под уникальным именем.

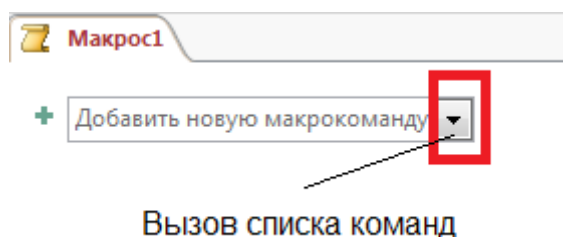


Рисунок 7.2 – Вызов каталога макрокоманд для создания макроса

Макросы могут содержать произвольное количество команд. Для каждой команды в режиме диалога задаётся набор его параметров. Например, для создания макроса для обновления данных выбирается команда **Открыть запрос** и имя запроса (рис. 7.3).

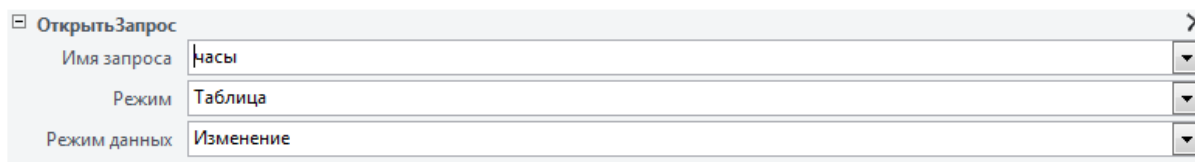


Рисунок 7.3 – Задание параметров для команды Открыть запрос

Образованный макрос можно использовать при создании интерфейса с объектами БД.

## 1.2 Кнопочная форма

Удобно иметь общий интерфейс, который позволит работать со всеми объектами БД. Такой интерфейс можно создать с помощью формы специального вида – **Главной кнопочной формы** (Main Switchboard). Главная кнопочная форма создаётся для навигации по объектам БД. Элементами **Главной кнопочной формы** являются кнопки, с помощью которых можно управлять работой с объектами

БД: формами и отчётами. Для одной БД можно создать несколько кнопочных форм, но только одна может быть определена как главная, остальные кнопочные формы будут подчинёнными. Вызвать их можно только с помощью элементов **Главной кнопочной формы**.

Любая кнопочная форма представляет собой особый вид формы, которая обеспечивает навигацию при работе с объектами БД: таблицами, формами, запросами и отчётами. Она позволяет создать меню для работы с БД с помощью кнопок, вызывающих элементы БД. Элементами, которые вызываются с помощью кнопок, могут быть только объекты БД типа формы и отчёты. Запросы и таблицы не могут быть непосредственно вызваны с помощью кнопок, это можно сделать только через формы, созданные для таблиц и запросов на выборку. Чтобы вызвать с помощью кнопок другие элементы БД, нужно использовать макросы.

Предварительно необходимо продумать дизайн **Главной кнопочной формы**, в котором определяются правила группировки действий для работы с объектами БД. Работу с объектами БД следует группировать на разных страницах кнопочной формы. Такая группировка поможет пользователю лучше ориентироваться при работе с БД. Например, можно сгруппировать работу по типам объектов БД: запросы, отчёты, ввод и редактирование данных. Для удобства работы на подчинённых кнопочных формах следует размещать кнопки возврата в **Главную кнопочную форму**. Для создания кнопочной формы можно использовать **Конструктора форм**. Однако в **СУБД Access** существует удобное и простое средство создания таких форм – (Swthboard Manager). С его помощью можно достаточно просто создать **Диспетчер кнопочных форм** основной макет интерфейса, который затем можно доработать с помощью **Конструктора форм**.

Как правило, интерфейс с БД на основе кнопочной формы содержит главную кнопочную форму и необходимое количество подчинённых кнопочных форм, в которых режимы работы с БД группируются по какому-либо принципу таким образом, чтобы обеспечить удобный интерфейс пользователя с БД. Например, можно создавать подчинённые кнопочные формы, работающие с различными типами объектов БД: таблицами, запросами, отчётами (рис. 7.4). На подчинённых кнопочных формах необходимо предусмотреть возможность возврата к главной кнопочной форме.



Рисунок 7.4 – Макет интерфейса с объектами БД

Для создания интерфейса на основе кнопочных форм используется **Диспетчер кнопочных форм**, который вызывается на вкладке **Работа с базами данных**. Первой создаётся главная кнопочная форма (рис. 7.2). Для создания интерфейса с объектами БД можно использовать следующую технологию:

- создать главную кнопочную форму;
- создать необходимое количество подчинённых кнопочных форм, для этого выбрать режим **Создать** (рис. 7.5), откроется диалоговое окно (рис. 7.3 а), в котором задаётся имя новой кнопочной формы, таким образом последовательно создаются кнопочные формы, для макета, представленного на рисунке 7.4, будет создано 3 кнопочных формы (рис. 7.6 б);

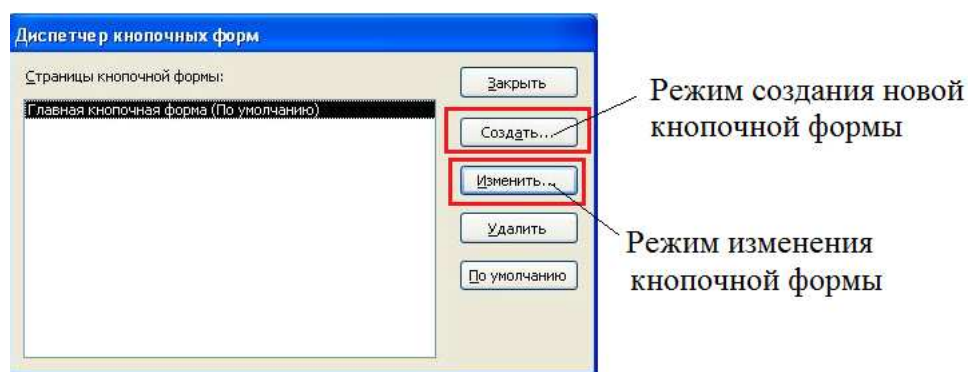


Рисунок 7.5 – Режимы работы диспетчера кнопочных форм

- создать управляющие элементы (кнопки) на главной кнопочной форме, для этого на странице главной кнопочной формы выбрать режим **Изменить** (рис. 7.5), с помощью режима **Создать** страницы **Изменение страницы кнопочной формы** (рис. 7.7) задать необходимо количество кнопок;



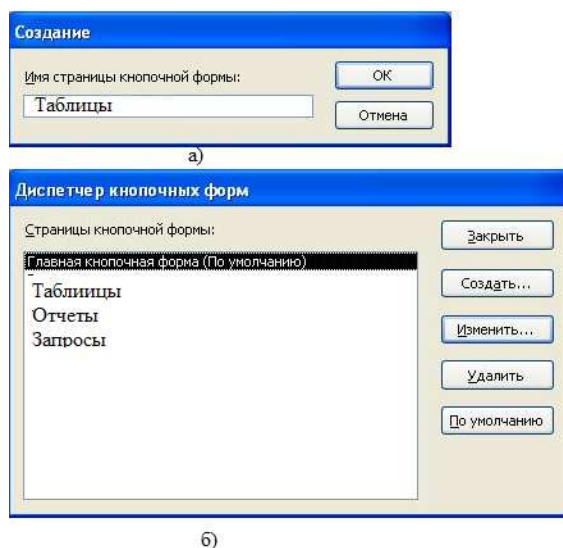


Рисунок 7.6 – Создание новой кнопочной формы

- для каждого объекта БД, который помещается на кнопочную форму, необходимо создать объекты типа форма или макрос;
- создать элементы (кнопки) на подчинённых кнопочных формах и связать их с объектами БД (формами, отчётами и макросами);
- для выхода из подчинённой кнопочной формы и перехода на главную кнопочную форму добавить элемент (кнопку) **Выход**;

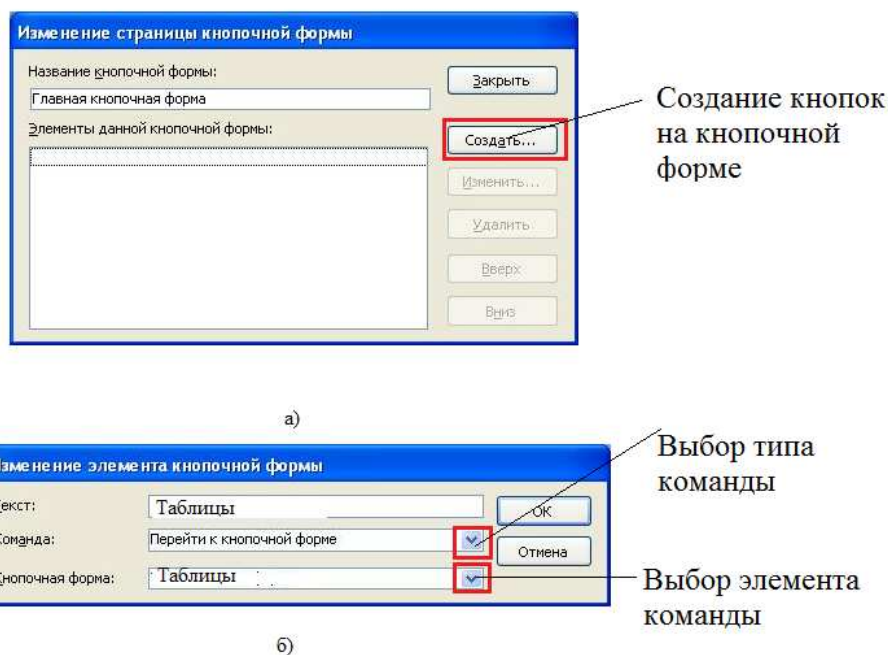


Рисунок 7.7 – Создание элементов (кнопок) на главной кнопочной форме

– для завершения работы с интерфейсом разместить на главной кнопочной форме элемент (кнопку) **Выход из БД** (рис. 7.8).

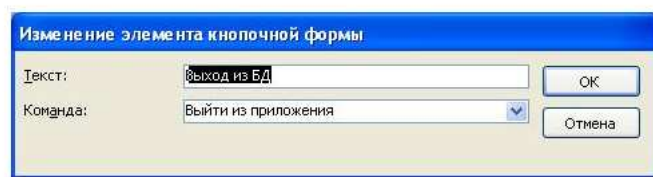


Рисунок 7.8 – Создание кнопки завершения работы с интерфейсом

После открытия главной кнопочной формы на экране откроется созданный интерфейс (рис. 7.9).

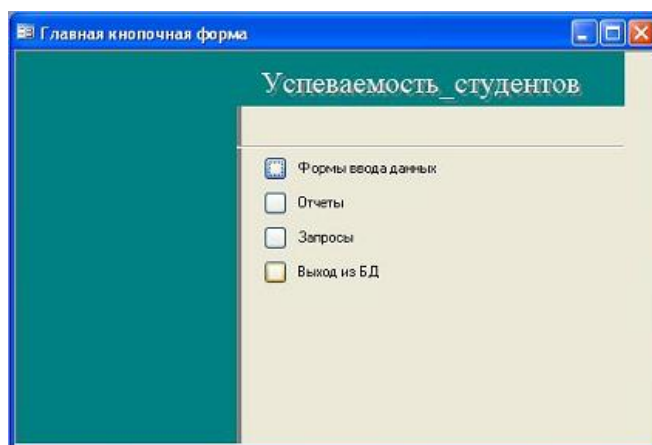


Рисунок 7.9 – Интерфейс с БД на основе Кнопочной формы

### 1.3 Форма навигации

Форма навигации является удобным и более простым средством создания интерфейса с объектами БД. При её создании используется технология «drag-and-drop», которая позволяет быстро образовывать интерфейс с БД на основе созданных форм объектов (таблиц, запросов, отчётов). Для тех объектов, для которых в **СУБД Access** не предусмотрены формы (это запросы на удаление и изменение данных), можно использовать возможности **Конструктора форм**, с помощью которого добавлять на общую форму дополнительные элементы – кнопки.

Для создания формы навигации необходимо на панели инструментов вкладка **Создание** выбрать **Навигация** и вид формы (рис. 7.10).

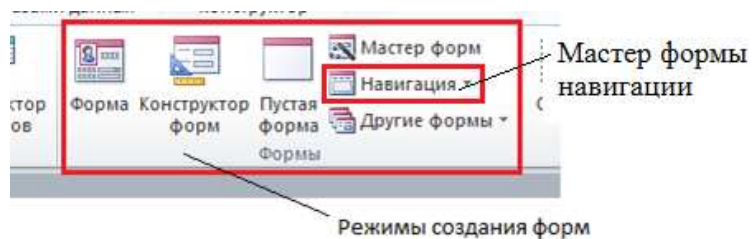


Рисунок 7.10 – Создание формы навигации

В поля формы, которое создаётся на основе выбранного шаблона (рис. 7.11), перетащить с помощью мыши в поля, которые помечены текстом «Создать», формы для объектов из обозревателя объектов БД.

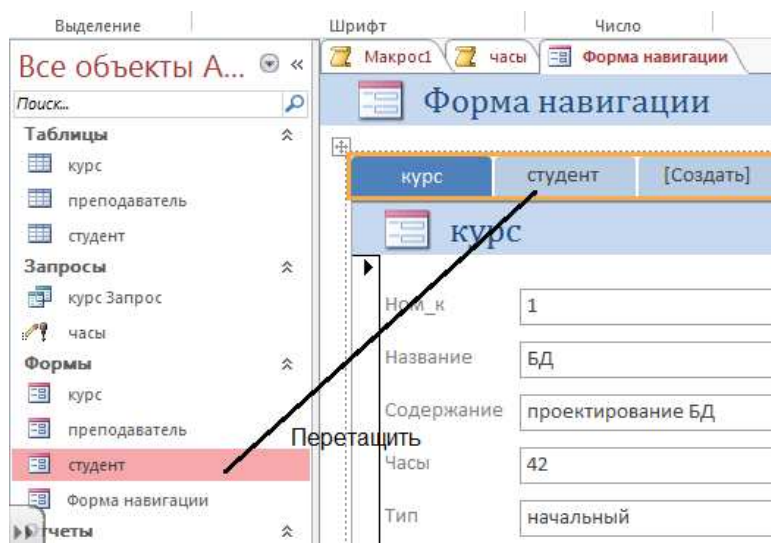


Рисунок 7.11 – Создание элементов формы навигации

Для добавления дополнительных элементов следует открыть форму навигации в режиме **Конструктора форм** и перетащить на неё необходимые элементы. Они включают поля ввода, различные списки, управляющие кнопки и др. (рис. 7.12). Добавленные элементы нужно настроить с помощью мастера настройки.

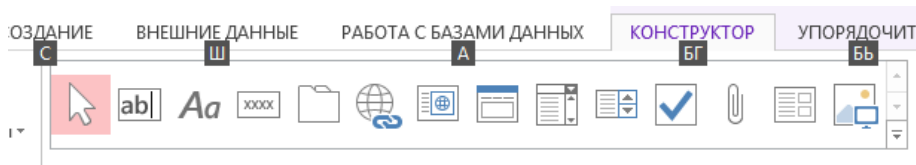



Рисунок 7.12 – Дополнительные элементы панели инструментов

Для того чтобы разместить на форме навигации кнопку вызова макроса, элемент  добавляется на форму (рис. 7.13). При этом автоматически вызывается мастер настройки элемента, который позволяет задать свойства кнопки. Для настройки кнопки вызова

макроса (рис. 7.14) последовательно выбираем команду **Выполнить макрос**, которая находится в категории **Разное**, задаём информацию, отображаемую на кнопке. В нашем случае это текст **Часы**. В результате получим интерфейсную форму, представленную на рисунке 7.15.

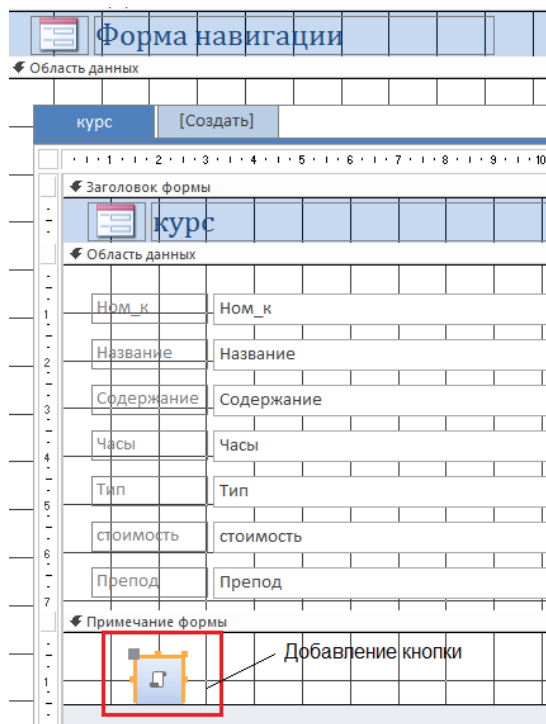


Рисунок 7.13 – Добавление кнопки вызова макроса

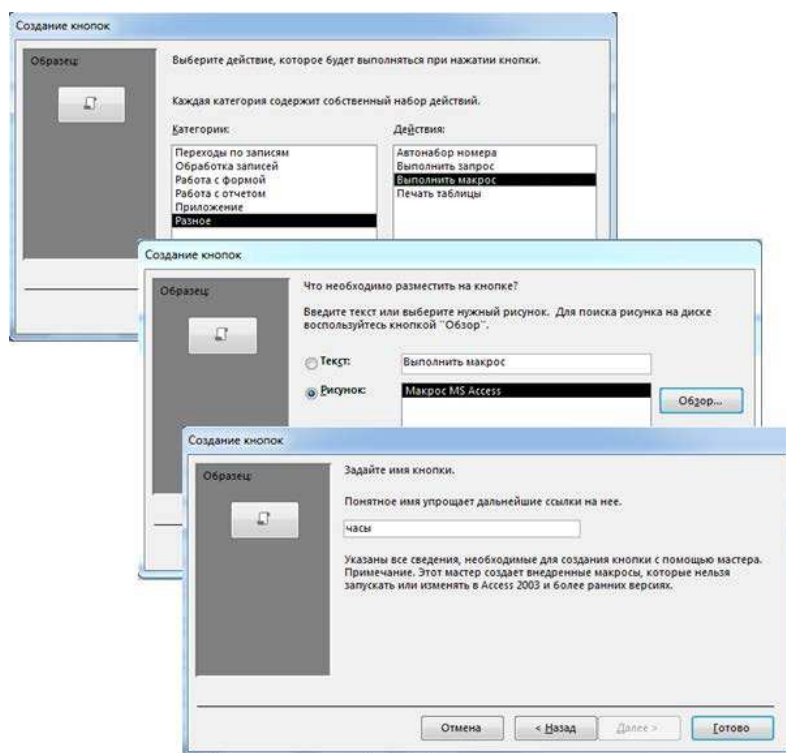


Рисунок 7.14 – Настройка свойств кнопки

курс	студент
курс	
Ном_к	1
Название	БД
Содержание	проектирование БД
Часы	42
Тип	начальный
стоимость	1 000,00р.
Препо	1

Рисунок 7.15 – Интерфейс с БД на основе Формы навигации

## 2. Последовательность выполнения практической работы

1. Запустить программу **Microsoft Access**.
2. Открыть созданную на практическом занятии 6 базу данных.
3. Изучить инструментальные средства разработки интерфейса с объектами БД.
4. Используя инструментальные средства **СУБД Access**, создать общий интерфейс для работы со всеми объектами БД.
5. С помощью **Конструктора форм** внести изменения в главную интерфейсную форму, добавив управляющие элементы.
6. Подготовить отчёт по работе.

## 3. Требования к оформлению отчета

Отчёт по практической работе должен включать следующие разделы:

- 1) титульный лист;
- 2) схема БД в **СУБД Access**;
- 3) пример макроса;
- 4) скриншот общего интерфейса с БД;
- 5) скриншоты реализации различных режимов работы с разработанным интерфейсом;

6) ответы на контрольные вопросы

### **Контрольные вопросы**

1. Что такое макросы БД Access?

2. Для чего используются макросы?

3. Как можно добавить управляющий элемент на кнопочную форму?

4. Как можно добавить управляющий элемент на Форму навигации?

5. Как изменить способ ввода связанных данных?

## Практическая работа 8

### ЦЕЛОСТНОСТЬ ДАННЫХ В БАЗЕ ДАННЫХ

**Цель работы** – приобрести навыки описания правил целостности в СУБД Access.

#### 1. Теоретические основы

При работе с БД необходимо быть уверенным в том, что изменение данных будет проходить по тем правилам, которые существуют в заданной предметной области. Это свойство БД называется целостностью. Она позволяет поддерживать соответствие данных БД предметной области. В СУБД Access для определения правил целостности используется система правил, которые позволяют определить «правильные» данные, которые могут храниться в столбце таблицы. Кроме того, они гарантируют поддержание устойчивых и корректных связей между записями в связанных таблицах, а также обеспечивающих защиту от случайного удаления или изменения данных в связанных таблицах.

Правила целостности данных на уровне столбца таблицы определяются при описании её структуры. Для этого в режиме **Конструктора таблиц** при описании свойств каждого поля таблицы можно определить ограничения на допустимые значения данных в строке **Условие на значение** (рис. 8.1). Для каждого типа значений в СУБД Access определён свой набор средств для задания правил целостности столбца.

Имя поля	Тип данных
Ном_Пр	Числовой
ФИО	Текстовый
конт	Текстовый
Степ	Текстовый
Стоимость	Денежный

Подстановка	
Формат поля	##0,00"р."; -##0,00"р."
Число десятичных знаков	Авто
Маска ввода	
Подпись	
Значение по умолчанию	
Условие на значение	Between 500 And 5000
Сообщение об ошибке	
Обязательное поле	Нет
Индексированное поле	Нет
Смарт-теги	
Выравнивание текста	Общее

Рисунок 8.1 – Задание условия на значение поля

Ограничения на значения поля можно задавать в виде условий, использующих операторы сравнения:

- > «значение»;
- <«значение»;
- <=«значение»;
- >=«значение»;
- =«значение».

Операторы сравнения могут использоваться в текстовых и числовых полях, а также в полях дат.

Условия на значения могут быть сложными, т.е. в них можно использовать логические операции **And**, **Or**, **Not**. Если ограничение определяется диапазоном значений, то можно использовать операторы сравнения и логические операторы, например,

*>=500 And <=5000.*

При использовании оператора **Between** то же условие будет иметь вид

*Between 500 And 5000.*

Для проверки, содержится ли значение в заданном списке, можно использовать оператор **In**, который позволяет определить список правильных значений для столбца таблицы. Если набор значений невелик (не более 7–9 значений), то можно задать условие

*In ("к.т.н"; "д.т.н."; "преподаватель").*

Для столбцов, в которых хранятся текстовые данные и данные типа дата, можно задать шаблон, которому они должны удовлетворять. Для задания шаблона используется оператор **Like**.

Примеры использования шаблона приведены в таблице 8.1.

Таблица 8.1 – Шаблоны значений

<i>Вид шаблона</i>	<i>Описание</i>
Like "И*в"	Текст начинается с буквы «И» и заканчивается буквой «в». Число символов в тексте может быть любое. Этому шаблону удовлетворяют следующие значения «Иванов» «Игра престолов»
Like "* игра"	Текст, включающий слово «игра» после пробела
Like "1?.03.2018"	Любая дата между 10 и 19 марта 2018 года
Like "И_анов"	Слово начинается с символа «И», второй символ может быть любой, затем обязательно следуют символы «анов». Этому шаблону удовлетворяют, например, слова «Иванов», «Иганов», «Идамов»



Для определения ограничений на значения полезно использовать **Построитель выражений**. Это позволит избежать ошибок при определении условий на значение. Примеры формирования условий на значения столбца таблицы приведены на рисунке 8.2.

Рисунок 8.2 – Задание ограничений на значение поля с помощью Построителя выражений

В СУБД Access используется система правил, которые позволяют поддерживать согласованность данных, хранящихся в двух связанных таблицах, и защищают данные в связанных таблицах от случайного удаления или изменения. Такие правила определяют ссылочную целостность данных и задаются для каждой отдельной связи при работе со **Схемой данных** (рис. 8.2).

В СУБД Access определены следующие правила:

- поле связи родительской таблицы является ключевым полем;
- поля связи связываемых таблиц имеют один тип данных.

Ссылочная целостность данных в БД Access определяется стратегиями, запрещающими появление «повисших ссылок»:

- запрещает появление в дочерней таблице строки со значением поля внешнего ключа, для которого нет строки с таким же значением первичного ключа родительской главной таблицы (например, нельзя

сохранить строку в таблице **Курс**, если ввести номер преподавателя, который не записан в таблице **Преподаватель**, при попытке сделать это СУБД выдает сообщение об ошибке (рис. 8.3));

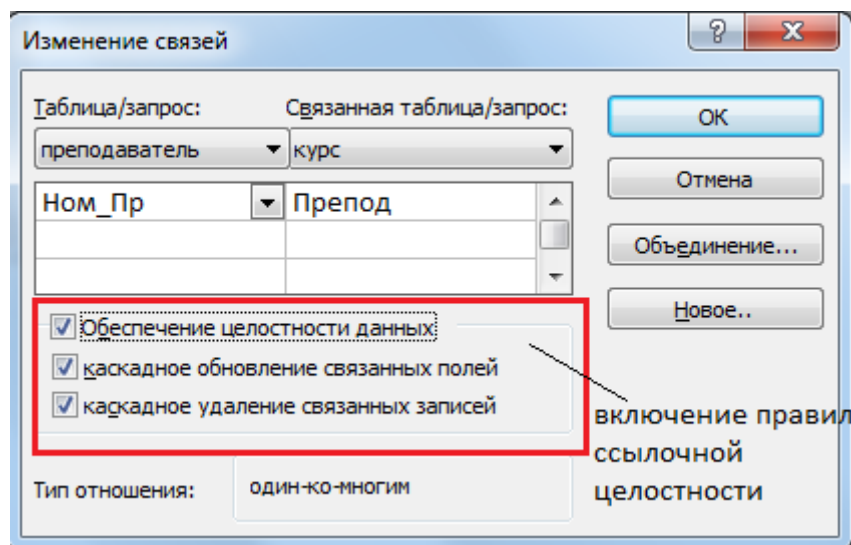


Рисунок 8.3 – Использование правил ссылочной целостности

– запрещает удаление или изменение первичного ключа строки из родительской таблицы, если существуют связанные с ней строки в дочерней таблице (например, невозможно удалить или изменить строку из таблицы **Преподаватель**, если в таблице **Курс** содержатся данные о курсах, которые он ведёт).

Однако эти правила можно изменить, разрешив изменять и удалять данные из связанной таблицы при изменении или удалении данных из главной таблицы (рис. 8.3). Такие правила называют правилами каскадного изменения и каскадного удаления данных.

Следует отметить, что такие правила надо определять для каждой отдельной связи.

## 2. Последовательность выполнения практической работы

1. Запустить программу **Microsoft Access** и открыть БД, созданную на предыдущей практической работе.

2. Задать ограничения на значения поля таблицы на основе правил целостности, разработанных на практическом занятии 3.

3. Проверить работу правил целостности, вводя ошибочные данные.

4. Задать правила целостности для связи на основе правил ссылочной целостности, разработанных на практическом занятии 3.

5. Проверить работу правила целостности путём ввода данных в таблицу.

6. Изменить правила контроля целостности для связи, добавив каскадное удаление и каскадное обновление.

7. Проверить работу правила целостности посредством ввода данных в таблицу, удалив данные из родительской таблицы.

8. Проверить состав данных в дочерней таблице.

9. Изменить данные первичного ключа в родительской таблице.

10. Проверить изменение данных в дочерней таблице.

11. Подготовить отчёт по работе.

### **3. Требования к оформлению отчёта**

Отчёт по практической работе должен включать следующие разделы:

- 1) титульный лист;
- 2) описание правил целостности в **СУБД Access**;
- 3) результаты проверки правил целостности в виде скриншотов работы с БД;
- 4) описание правил ссылочной целостности;
- 5) результаты проверки выполнения правил ссылочной целостности;
- 6) выводы по работе;
- 7) ответы на контрольные вопросы.

### **Контрольные вопросы**

1. Что такое правила целостности данных?
2. Какие виды правил целостности Вы знаете?
3. Как задать ограничение на значение столбца таблицы?
4. Какие операторы можно использовать для задания ограничений на значение столбца таблицы?
5. Для чего используется оператор **Like**?
6. Что такое ссылочная целостность?
7. Какие правила определяют ссылочную целостность данных?
8. Как задать правила ссылочной целостности?
9. Поясните работу правила «каскадное изменение данных».

## ЗАКЛЮЧЕНИЕ

В настоящем пособии приведены основополагающие подходы к проектированию реляционных БД. Выполнение практических заданий позволило не только пройти все этапы проектирования БД, но и освоить основные технологические приёмы разработки модели БД с использованием CASE-средств.

Практические задания этого пособия не ставили целью изучение всех возможностей **СУБД Access**, поэтому и для более глубокого изучения **СУБД Access** целесообразно использовать дополнительные источники, некоторые из которых приведены в списке рекомендуемой литературы. Наиболее полное описание новых возможностей этой СУБД можно найти на сайте разработчика <http://msdn.microsoft.com/ru-ru/library/>.

Практические задания, предлагаемые в данном пособии, являются только основой для дальнейшего углубленного изучения вопросов, связанных с проектированием и реализацией БД.

## ЛИТЕРАТУРА

1. Бекаревич Ю.Б., Пушкина Н.В. Microsoft Access 2013: Самоучитель. – СПб.: БХВ, 2014. – 464 с.
2. Голицына О.Л., Максимов Н.В., Попов И.И. Базы данных : учеб. пособие. – 4-е изд., перераб. и доп. – М.: ФОРУМ : ИНФРА-М, 2018. – 400 с.
3. Гринченко Н. Н. Проектирование баз данных. СУБД Microsoft Access: учеб. пособие для вузов. – М.: Горячая линия – Телеком, 2013. – 240 с.
4. Дейт К.Дж. Введение в системы баз данных. – М.: Вильямс, 2018. – 1328 с.
5. Диго С. М. Базы данных: проектирование и создание: учебник для вузов. – М.: ЕАОИ, 2008. – 171 с.
6. Конноли Т. Базы данных. Проектирование, реализация и сопровождение. Теория и практика. – М.: Вильямс, 2017. – 1440 с.
7. Кузин А.В. Базы данных: учеб. пособие для студ. – М.: Академия, 2016. – 315 с.
8. Маклаков С.В. Создание информационных систем с AllFusion Modeling Suite. – М.: ДИАЛОГ-МИФИ, 2005. – 432 с.
9. Сидорова Н.П. Проектирование реляционных баз данных: сборник лабораторных работ: метод. пособие. – М.: Изд. Дом МЭИ, 2011. – 32 с.
10. Хомоненко А.Д., Цыганков В.М., Мальцев М.Г. Базы данных. – М.: КОРОНА Принт, 2009. – 672 с.
11. Шустова Л.И., Тараканов О.В. Базы данных: учебник. – М.: НИЦ ИНФРА-М, 2017. – 304 с.

## Приложение

# ВАРИАНТЫ ПРЕДМЕТНОЙ ОБЛАСТИ ДЛЯ ПРАКТИЧЕСКИХ ЗАДАНИЙ

### Вариант 1

#### ***Предметная область***

Предприятие по реализации товаров по заказам. Предприятие располагает складом, на котором хранятся товары. Для получения товара клиент оформляет заказ. Для пополнения товаров на складе предприятие заключает договоры с поставщиками.

#### ***Бизнес-правила***

1. Организация формирует заказы из товаров, имеющихся на складе.
2. Клиент может оформить заказ на несколько товаров.
3. Формированием заказа занимается один сотрудник.

***Минимальный набор сущностей:*** товар, клиент, поставщик, сотрудник.

***Минимальный набор атрибутов:*** наименование товара, назначение товара, стоимость товара; фамилия клиента, контактные данные клиента, адрес клиента; название фирмы поставщика, фамилия руководителя фирмы, контактные данные, адрес фирмы; фамилия сотрудника, должность сотрудника, оклад сотрудника, номер служебного телефона сотрудника.

#### ***Запросы на обработку***

1. Найти фамилию и телефон клиента, оформившего заказ с заданным номером.
2. Составить список товаров, количество которых на складе меньше заданной величины.
3. Найти заказы, оформленные заданным сотрудником.
4. Найти заказы, стоимость на которые не выше заданной величины.
5. Увеличить на 5% стоимость товаров, поставляемых заданным поставщиком.

### Вариант 2

#### ***Предметная область***

Ремонтная мастерская занимается ремонтом бытовой техники. Клиент оформляет заказ на ремонт изделия. Ремонт изделий осуществляется с использованием комплектующих, которые имеются на складе мастерской. Для пополнения комплектующих на складе мастерская заключает договоры с поставщиками.

#### ***Бизнес-правила***

1. Существуют клиенты, нескольких категорий приоритетности.

2. Каждый клиент может оформить неограниченное количество заказов.

3. Клиенты оформляют один заказ на ремонт одного изделия.

4. Для получения заменяемой детали сотрудник организации оформляет отдельный заказ на получение детали со склада.

5. Один заказ выполняется одним сотрудником.

**Минимальный набор сущностей:** изделие, клиент, сотрудник, деталь.

**Минимальный набор атрибутов:** наименование изделия, вид изделия, стоимость изделия; фамилия клиента, контактные данные клиента, адрес клиента; фамилия сотрудника, должность сотрудника, оклад сотрудника, номер служебного телефона сотрудника; название детали, её стоимость, количество на складе, название фирмы производителя.

### ***Запросы на обработку***

1. Найти фамилию и телефон клиента, оформившего заказ с заданным номером.

2. Определить фамилию и телефон для клиентов заданной категории.

3. Найти заказы, оформленные заданным сотрудником.

4. Найти заказы, стоимость на которые не выше заданной величины.

5. Уменьшить на 5% стоимость заказа с заданным номером.

### **Вариант 3**

#### ***Предметная область***

Фирма по прокату автомобилей. Фирма оказывает услуги, предоставляя автомобили из своего парка. Фирма выдаёт напрокат автомобили на основе договора. Сотрудник фирмы фиксирует данные о клиенте, данные об автомобиле, дату начала проката и количество дней проката.

#### ***Бизнес-правила***

1. Клиент оформляет договор на прокат автомобиля.

2. Клиент может оформить несколько договоров.

3. Фирма ежегодно страхует автомобили в различных страховых компаниях.

4. Стоимость проката зависит от марки автомобиля.

**Минимальный набор сущностей:** автомобиль, клиент, страховая компания, сотрудник.

**Минимальный набор атрибутов:** номер автомобиля, его цвет, марка, год выпуска, сумма страховки; фамилия клиента, номер его паспорта и водительского удостоверения, контактные данные (телефон, адрес), водительский стаж; название страховой компании, номер

лицензии, адрес, фамилия руководителя; фамилия сотрудника, его должность, оклад, служебный телефон.

***Запросы на обработку***

1. Найти всех клиентов, взявших автомобиль указанной марки.
2. Найти данные о страховой компании для автомобилей, срок страховки которых заканчивается.
3. Найти контактные телефоны и фамилии всех сотрудников, принимавших заказы на прокат в указанный день.
4. Составить список автомобилей указанной марки, стоимость страховки которых менее заданной величины.
5. Уменьшить на 5% стоимость проката автомобиля заданной марки.

**Вариант 5**

***Предметная область***

Работа гостиницы.

***Бизнес-правила***

1. В гостинице имеются номера различной категории.
2. Каждый номер закреплен за одним сотрудником. Один сотрудник может обслуживать несколько номеров.
3. Гость может забронировать номер указанной категории на заданный срок.
4. Стоимость номер и зависит от его категории и перечня дополнительных услуг, которыми пользовался гость.

***Минимальный набор сущностей:*** номер, сотрудник, гость, услуга.

***Минимальный набор атрибутов:*** номер комнаты, её тип, статус (занята, забронирована, свободна), состояние (готова к проживанию, требует уборки, требует ремонта); фамилия гостя, его категория, номер паспорта, дата заселения, срок пребывания, гражданство; фамилия сотрудника, специальность, оклад, контактный телефон.

***Запросы на обработку***

1. Получить список свободных номеров заданной категории.
2. Получить список клиентов, приехавших в указанную дату с указанием их фамилии, номера и его категории.
3. Составить список номеров и их гостей, которые обслуживаются одним сотрудником.
4. Составить список забронированных номеров.
5. Уменьшить на 5% стоимость номеров заданной категории.

**Вариант 6**

***Предметная область***

Ремонт автомобиля.

***Бизнес-правила***

1. Клиент оформляет договор на ремонт автомобиля.



2. Один автомобиль ремонтирует одна бригада сотрудников.
3. Для ремонта автомобиля используются комплектующие различного типа и производителей.
4. Стоимость часа работы зависит от и категории бригады.
5. Различают несколько категорий клиентов.

**Минимальный набор сущностей:** автомобиль, клиент, бригада, деталь.

**Минимальный набор атрибутов:** номер автомобиля, его цвет, марка, год выпуска; фамилия клиента, номер его паспорта и водительского удостоверения, контактные данные (телефон, адрес); номер бригады, категория, тип (определяет, какие марки машин могут ремонтировать), фамилия бригадира, служебный телефон, стоимость часа работы; название детали, её стоимость, производитель, наличие на складе.

#### ***Запросы на обработку***

1. Определить номера договоров и марку автомобилей, ремонтом которых занимается заданная бригада.
2. Определить, какие бригады могут ремонтировать автомобили заданной марки.
3. Определить стоимость выполнения договоров, выполняемых заданной бригадой.
4. Определить список деталей, которые поставляются заданной фирмой.
5. Уменьшить на 5% стоимость деталей, поставляемых заданной фирмой.

### **Вариант 7**

#### ***Предметная область***

Работа агентства недвижимости.

#### ***Бизнес-правила***

1. Агентство занимается оформлением недвижимости в аренду.
2. Клиент может арендовать несколько объектов.
3. Для оформления одного объекта оформляется один договор.
4. С каждым объектом недвижимости работает один сотрудник.

**Минимальный набор сущностей:** клиент, сотрудник, объект недвижимости, договор.

**Минимальный набор атрибутов:** фамилия клиента, его категория, контактные данные клиента, адрес клиента, номер паспорта; фамилия сотрудника, должность сотрудника, оклад сотрудника, номер служебного телефона сотрудника; вид объекта (дом, квартира, комната и т.п.), его стоимость, адрес, состояние (новое, после ремонта, требует ремонта и т.п.), статус (занят, свободен); дата заключения договора, срок договора.

### ***Запросы на обработку***

1. Составить список договоров, оформленных в указанный день. Указать фамилию и адрес клиента, фамилию сотрудника.
2. Составить список сотрудников, заключавших договоры с заданным клиентом.
3. Составить список свободных объектов недвижимости указанного типа.
4. Составить список объектов недвижимости, договоры на которые заключал данный клиент.
5. Уменьшить на 10% стоимость объектов заданного вида.

### **Вариант 8**

#### ***Предметная область***

Отдел больницы по работе с договорами.

#### ***Бизнес-правила***

1. Пациент заключает договор на лечение в заданном отделении больницы.
2. У каждого пациента есть лечащий врач.
3. Пациент может оплачивать свой договор за счёт страховой компании.
4. Обследование пациента проводит не только лечащий врач.

***Минимальный набор сущностей:*** пациент, врач, обследование, страховая компания.

***Минимальный набор атрибутов:*** фамилия пациента, его категория, номер паспорта, номер страхового полиса, дата поступления, гражданство; фамилия врача, категория, специальность, оклад, контактный телефон; название обследования, вид обследования, дата проведения обследования, стоимость обследования; название страховой компании, номер лицензии, фамилия руководителя, контактный телефон.

### ***Запросы на обработку***

1. Найти всех пациентов, застрахованных в заданной страховой компании.
2. Найти перечень видов обследования, пройденных заданным пациентом и их стоимость.
3. Найти всех пациентов заданной категории, которые имеют заданного лечащего врача.
4. Найти все заключенные с пациентами заданного врача договоры и их стоимость.
5. Увеличить на 5% стоимость обследования заданного вида.

## **Вариант 9**

### ***Предметная область***

Типография занимается выпуском печатной продукции.

### ***Бизнес-правила***

1. Для выполнения работ типография заключает договор с клиентом.
2. Один договор выполняет одна бригада сотрудников.
3. Стоимость работ зависит от вида продукции, вида материалов и тиража издания.
4. Различают несколько категорий клиентов.
5. Для выполнения работ используются материалы.

**Минимальный набор сущностей:** клиент, издание, бригада сотрудников, материал.

**Минимальный набор атрибутов:** фамилия клиента, контактные данные клиента, адрес клиента; наименование издания, его вид, стоимость издания; номер бригады, специализация, номер служебного телефона бригадира; название материала, её стоимость, количество на складе, название фирмы производителя.

### ***Запросы на обработку***

1. Определить номера договоров и виды изданий, которые выполняет заданная бригада.
2. Определить, какие бригады могут выполнять заданный договор.
3. Определить стоимость выполнения договоров, выполняемых заданной бригадой.
4. Определить список фирм-производителей для материалов, количество которых на складе меньше заданной величины.
5. Уменьшить на 5% стоимость материалов, поставляемых заданным поставщиком.

## **Вариант 10**

### ***Предметная область***

Страховая компания.

### ***Бизнес-правила***

1. Договор страхования заключается между страховой компанией и клиентом сроком не более 1 года (но может быть и меньше).
2. При заключении договора указывается вид страхования, страховая сумма, дата начала действия договора.
3. Один клиент может заключить несколько договоров.
4. Один сотрудник работает с несколькими клиентами.
5. Страхование осуществляется на основе существующего вида услуг.

**Минимальный набор сущностей:** клиент, сотрудник, страховой продукт, договор.

**Минимальный набор атрибутов:** фамилия клиента, его категория, контактные данные клиента, адрес клиента, номер паспорта; фамилия сотрудника, должность сотрудника, оклад сотрудника, номер служебного телефона сотрудника; название страхового продукта (дом, квартира, комната, автомобиль и т.п.), его базовая стоимость; дата заключения договора, дата завершения, объект страхования, его оценочная стоимость.

***Запросы на обработку***

1. Определить номера договоров и вид страхового продукта, заключенных заданным работником.

2. Составить список клиентов, у которых срок действия договора истекает в указанную дату.

3. Составить список клиентов, сумма страхования у которых меньше указанной величины.

4. Составить список клиентов, договоры с которыми заключил заданный сотрудник.

5. Уменьшить на 5% стоимость данного страхового продукта.

**Сидорова Наталья Петровна**

**Базы данных  
Практикум по проектированию  
реляционных баз данных**

*Учебное пособие*

Ответственный редактор *Ю. Барабанищкова*  
Верстальщик *М. Глаголева*

Издательство «Директ-Медиа»  
117342, Москва, ул. Обручева, 34/63, стр. 1  
Тел/факс + 7 (495) 334-72-11  
E-mail: [manager@directmedia.ru](mailto:manager@directmedia.ru)  
[www.biblioclub.ru](http://www.biblioclub.ru)  
[www.directmedia.ru](http://www.directmedia.ru)