## Оглавление

# 1. ОСНОВНЫЕ ПОЛОЖЕНИЯ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ

## 1.1. Алгоритмизация

В базовом определении под *алгоритмизацией* понимают процесс разработки алгоритмов для решения прикладных поставленных задач. Таким образом, алгоритм — это набор инструкций, описывающий четкий порядок действий для достижения поставленной цели либо для решения поставленной задачи.

Учитывая представленные выше понятия, можно выделить основные правила по разработке алгоритмов:

− алгоритм должен состоять из ряда последовательных простых шагов, выполнение которых приведет к достижению поставленной цели;

− содержащийся в алгоритме набор инструкций должен четко и однозначно трактоваться вне зависимости от его исполнителя;

− алгоритм решения задачи должен разрабатываться в общем виде для решения класса задач.

В настоящее время выделяют следующие основные способы представления алгоритмов:

− *словесный* — представляет структуру алгоритма на естественном языке. Примером такого описания может быть инструкция по эксплуатации;

− *структурно-стилизованный*, или *псевдокод*, — описание алгоритма в таком случае происходит на естественном языке, но с использованием формализованного языка. Такой алгоритм позволяет выявить основные этапы решения задачи до представления его на языке программирования;

− *графический* — представление алгоритма в виде блок-схемы;

− *программный* — описание структуры алгоритма на языке алгоритмического программирования.

Наиболее удобная форма представления алгоритма — это блок-схема, т.е графическая структура алгоритма, в котором каждый этап обработки данных представляется в виде блоков. Правила разработки и представления блок-схем четко регламентируются ГОСТ 19.701–90 (ИСО 5807–85) «Межгосударственный стандарт. Единая система программной документации. Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения».

Выделяют следующие виды алгоритмов:

− *линейный* — список команд (шагов) для их последовательного выполнения;

− *разветвляющийся* — данный вид алгоритмов должен содержать хотя бы одну проверку условия, при котором обеспечивается переход по одному из предложенных вариантов решения;

− *циклический* — в алгоритме предусмотрено многократное повторение одной и той же последовательности действий, количество повторений определяется граничными условиями задачи.

Примеры описанных алгоритмов представлены в прил. 1.

## 1.2. Программирование

Под *программированием*, как правило, понимается процесс создания программ. Если рассматривать программирование в узком смысле, то под этим процессом принимается разработка программ-инструкций на конкретном языке программирования. При этом подразумевается, что разработчику программы предоставляется уже готовый алгоритм работы программы. В более широком смысле программирование — это вся деятельность, связанная с разработкой и поддержкой программы:

− анализ и постановка задачи;

− разработка архитектуры программы;

− проектирование программы;

− разработка алгоритмов работы программы;

− разработка структуры данных;

− написание текста программы при необходимости;

− отладка и тестирование программы;

– документирование;
– конфигурирование (настройка);
– доработка и сопровождение до вывода программы из эксплуатации.

Для исполнения программы на ЭВМ используются трансляторы — технические средства, выполняющие трансляцию программы.

В настоящее время создаются интегрированные среды разработки, которые уже содержат редактор для ввода и редактирования текстов программ, отладчики, трансляторы, компоновщики и прочие служебные модули.

Обобщенная схема классификации языков программирования представлена в прил. 2.

В настоящем учебно-методическом пособии рассмотрены основы работы с унифицированным языком программирования, имеющим практически полностью унифицированную среду разработки, Visual Basic for Application (VBA).

## 1.3. Базы данных. Основные понятия

*Базы данных (БД)* представляют собой набор сведений, которые, как правило, относятся к определенной тематике или задаче. В общем случае под БД понимается организованная специальным образом совокупность взаимосвязанных данных [1–4].

В зависимости от принципов обработки данных все БД можно разделить на два больших класса: централизованные и распределенные. Первый класс предполагает, что БД размещается на одном компьютере, при этом у него может и не быть доступа в глобальную сеть «Интернет» и тогда происходит только локальный доступ. Если же на компьютере реализован доступ в сеть, то возможен централизованный доступ. В таком случае применяются два способа обработки данных: файл – сервер и клиент – сервер.

Для реализации первого подхода необходимо, чтобы один из компьютеров, которые входят в сеть, функционировал как сервер с файлами централизованной БД. Далее по запросам пользователей информация передается на рабочие станции пользователей, где осуществляется обработка данных. Итоговые БД пользователю необходимо скопировать обратно на сервер. Пример архитектуры файл – сервер представлен на рис. 1.1.

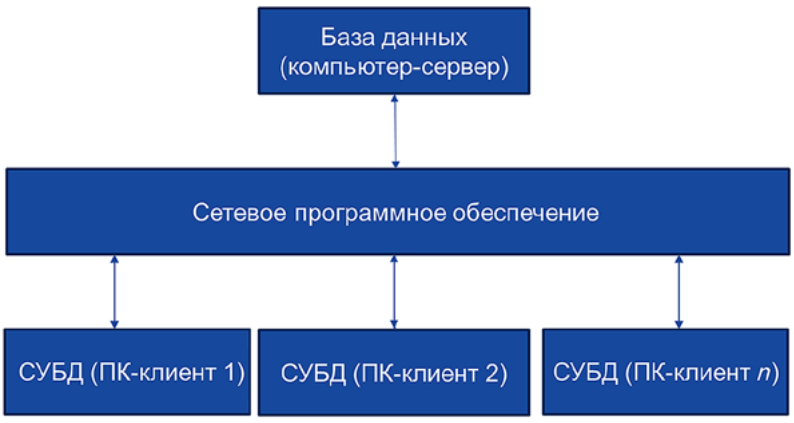


Рис. 1.1. Пример архитектуры файл – сервер
(СУБД — система управления базами данных)

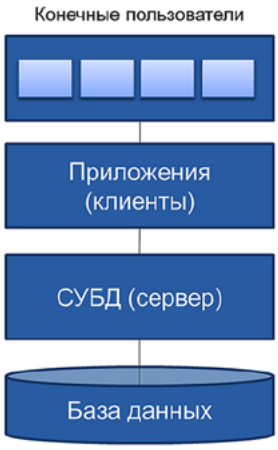


Рис. 1.2. Пример архитектуры
клиент – сервер

При регистрации пользователя в сети можно определить его функциональные возможности и задать права доступа: администратор (получает доступ к выполнению всех возможных команд) или пользователь (права доступа определяются администратором).

К основным недостаткам технологии файл – сервер отнестятся: между ПК передается значительный объем информации (соответственно, необходимо мощное оборудование); невозможность обеспечения одновременной работы с данными несколькими пользователями.

Рассмотрим реализацию технологии клиент – сервер, при использовании которой центральный сервер помимо хранения данных обеспечивает также и обработку данных (рис. 1.2). Запросы к такой БД необходимо формировать на языке структурированных запросов SQL. При поступлении запроса, который содержит описание действий и список задач, сервер инициирует процесс обработки данных. Пользователь на выходе получает уже обработанный сервером ответ — набор данных. При этом передается не весь набор данных, а только то, что необходимо пользователю, что является преимуществом по сравнению с технологией файл – сервер. Недостатком являются только значительные требования к характеристикам и мощности центрального компьютера.

В настоящее время наибольшее распространение получили централизованные БД с централизованным доступом.

При разработке баз данных используется несколько моделей баз данных: иерархическая, сетевая и реляционная.

*Иерархическая модель* с примером такой БД представлена на рис. 1.3.

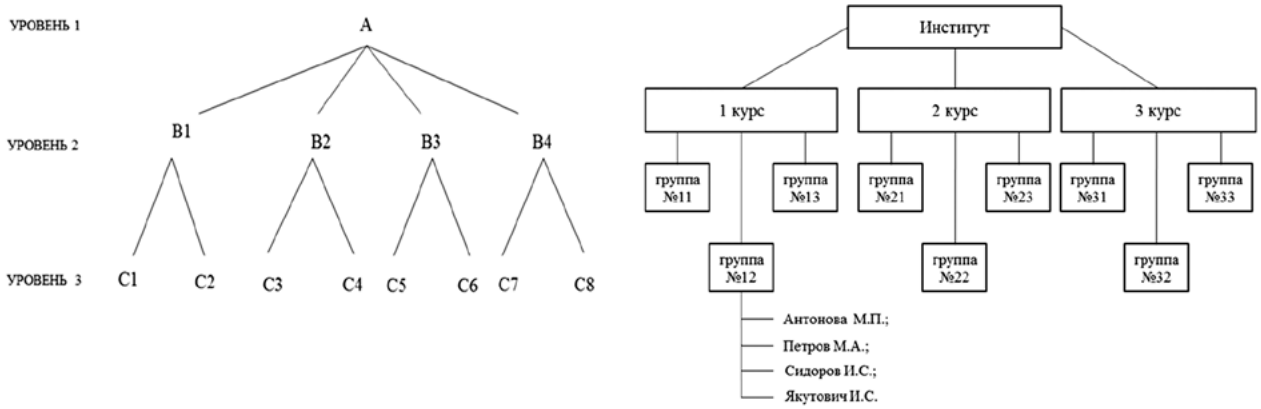


Рис. 1.3. Иерархическая модель с примером базы данных

При использовании этой модели данных сведения представляются в виде древовидной структуры, состоящей из объектов, которые можно разложить на классы, т.е. на различные уровни.

При использовании *сетевой модели* БД каждый элемент (объект) можно связать с любым другим элементом (рис. 1.4).

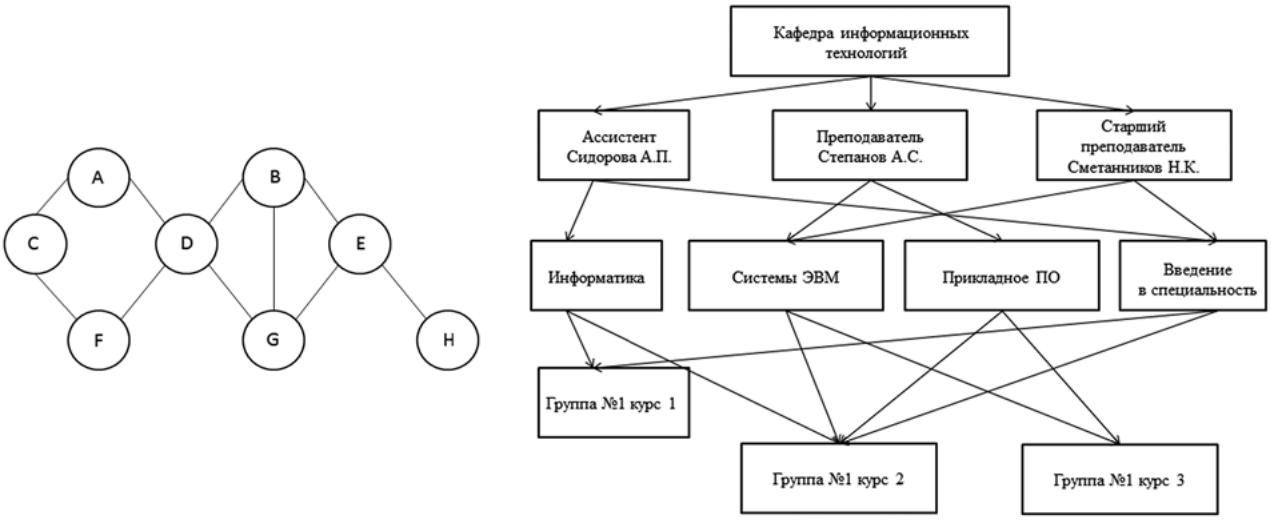


Рис. 1.4. Сетевая модель с примером базы данных

*Реляционная модель* данных была предложена еще в 1970 г. американским математиком Э. Коддом. Теоретической основой такой модели БД стала теория отношений. Ученый «показал, что любое представление данных можно свести к совокупности двумерных таблиц особого вида, называемых отношением» [1], что и послужило основой названия для данного вида БД.

В настоящем учебно-методическом пособии внимание уделено в основном реляционным базам данных, которые могут хранить информацию во взаимосвязанных двумерных таблицах с присущими им свойствами:

– каждый элемент такой таблицы является элементом данных;

– все элементы в столбце должны быть однородными, т.е. иметь одинаковый тип и длину;

– каждый столбец должен иметь уникальное имя;

– наличие одинаковых строк в таблице не допускается.

В настоящее время наиболее удобным инструментом для создания реляционных БД является Microsoft Access. В этой программе данные хранятся в таблицах, которые могут быть распределены на усмотрение разработчика по темам или задачам, но все они связаны и при наличии запросов могут быть объединены разными способами для получения необходимого результата, например аналитического отчета по конкретному вопросу.

Каждая таблица, созданная в Microsoft Access, имеет строки, которые принято называть записями, а также столбцы, которые принято называть полями (рис. 1.5).


Рис. 1.5. Пример таблицы из реляционной БД

Каждый вид данных, созданный в реляционной БД, как правило, содержится только в одном месте, но просматривать их можно относительно различных запросов пользователя.

Однако для того, чтобы установить связи между таблицами в реляционной БД, необходимо задавать в этих таблицах совпадающие значения.

Стоит также обратить внимание на то, что целесообразно избегать составных полей в таблицах. Поскольку если в одном поле будут содержаться данные по имени, отчеству и фамилии сотрудника, то отделить имена от фамилий будет крайне сложно. В рассмотренном примере целесообразнее сделать три поля: имя, отчество и фамилия.

Для того чтобы отличать записи таблицы друг от друга, необходимо в каждой таблице добавлять поле с уникальным значением — первичным ключом, с помощью которого можно связать одну таблицу с другой, обеспечивая совместное использование данных. Это позволит снизить повторения сведений в таблицах. Вместе с тем в связанных таблицах в одной из таблиц первичный ключ становится внешним. Неключевые таблицы не должны зависеть друг от друга.

В MS Access у полей может быть один из следующих типов данных:

– «Счетчик» — при добавлении каждой новой записи в таблицу программа автоматически вводит номер каждой следующей позиции;

– «Денежный» — для внесения информации о денежных значениях (причем по умолчанию в программе заложена валюта «руб.», но в случае необходимости можно выбрать и другую — «евро», «дол.» и т.д.);

– «Дата / время» — при необходимости внесения в БД дат или времени либо их комбинации;

– «Гиперссылка» — для внесения адресов гиперссылок, для перехода к web-страницам, объектам БД или другим файлам;

– «Поле MEMO» — для ввода текста неограниченной длины;

– «Поле OLE» — для внесения в БД рисунков или документов MS Word;

– «Числовой» — для введения числовых данных любого формата;

– «Текстовый» (короткий текст) — для введения текста ограниченной длины, чисел или других символов.

*Типы связей между таблицами*

Связь между таблицами в реляционной БД устанавливается по совпадающим значениям первичного ключа одной таблицы и внешнего ключа другой таблицы. Как правило, такие поля имеют одинаковые имена в обеих таблицах. Рассмотрим типы связей.

Наиболее распространенным видом считается связь «один ко многим». Связь **один ко многим** в реляционных базах данных реализуется тогда, когда объекту А может принадлежать или же соответствовать несколько объектов Б, но объекту Б может соответствовать только один объект А. Например, у одного поставщика может быть много различных товаров, однако у каждого товара может быть только один поставщик (рис. 1.6).

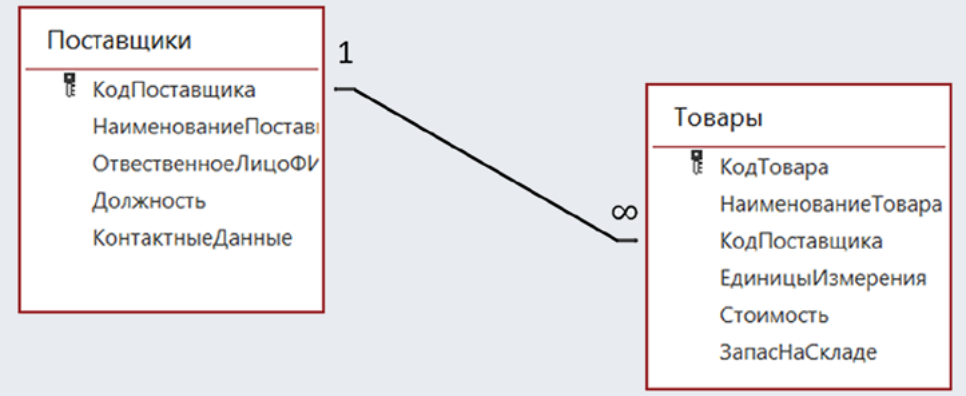


Рис. 1.6. Пример связи «один ко многим»

Связь **многие ко многим** реализуется в том случае, когда нескольким объектам из таблицы А может соответствовать несколько объектов из таблицы Б, и в тоже время нескольким объектам из таблицы Б соответствует несколько объектов из таблицы А (рис. 1.7).

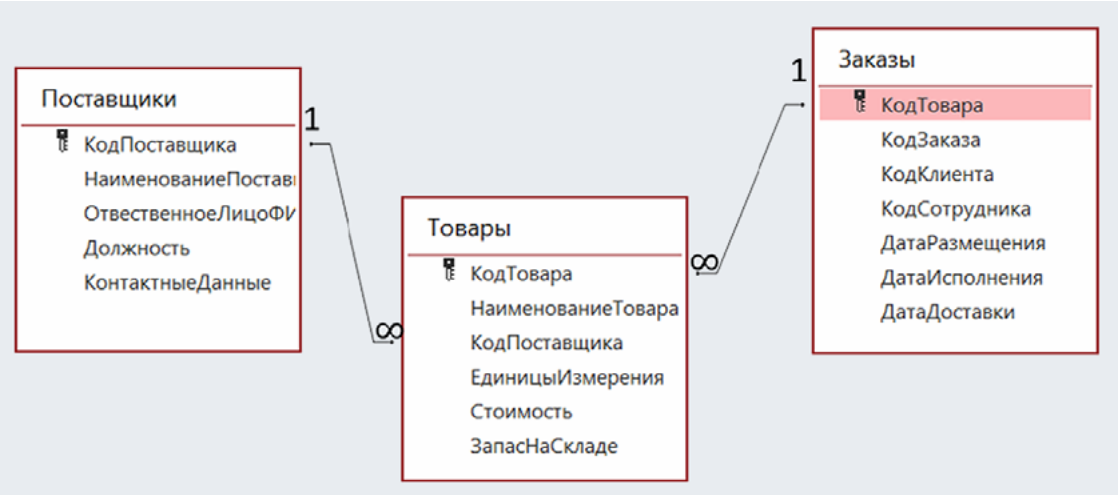


Рис. 1.7. Пример связи «многие ко многим»

Связь **один к одному** — самая редко встречающаяся связь между таблицами. Таблицы будут связаны один к одному тогда, когда одному объекту таблицы А соответствует один объект таблицы Б, и одному объекту таблицы Б соответствует один объект таблицы А. Как правило, такие таблицы целесообразно объединить в одну (рис. 1.8).

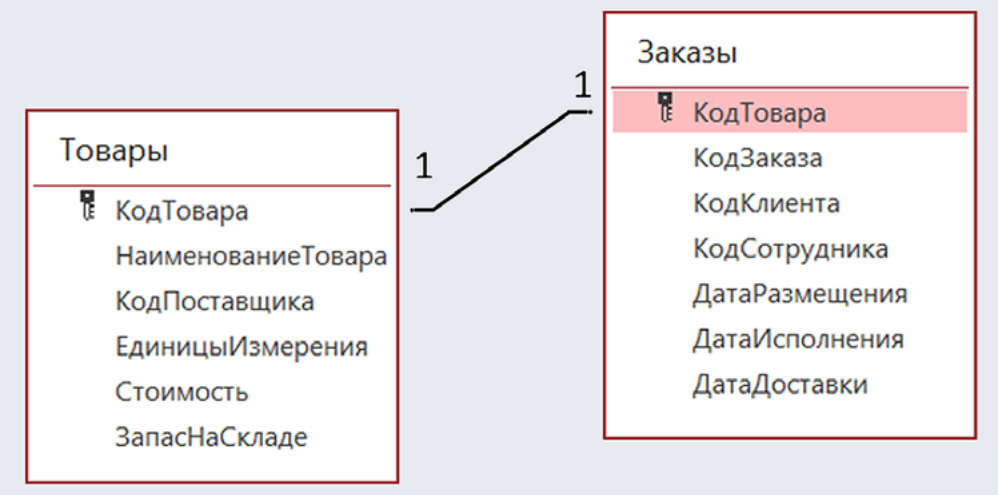


Рис. 1.8. Пример связи «один к одному»

## 1.4. Создание таблиц базы данных и определение связей между таблицами в базе данных Microsoft Access

*Цель:* освоение приемов работы с Microsoft Access 2013–2019. Формирование навыков создания и редактирования, заполнение данными и создание ключевых полей таблиц. Создание связей между ними.

*Задание:* создать пустую БД под названием «фамилия и инициалы студента.accdb». Создать таблицу «Учащиеся» и в ней прописать следующие столбцы (в скобках указан тип поля, который необходимо ему присвоить):

1) ИД (тип «Счетчик»);
2) фамилия (тип «Короткий текст»);
3) имя (тип «Короткий текст»);
4) отчество (тип «Короткий текст»);
5) пол (тип «Логический»);
6) номер зачетки (тип «Короткий текст»);
7) факультет (тип «Короткий текст»);
8) курс (тип «Числовой»);
9) группа (тип «Числовой»);
10) дата рождения (тип «Дата и время»);
11) телефон (тип «Короткий текст с маской»);
12) электронная почта (тип «Короткий текст»);
13) гражданство (тип «Короткий текст»);
14) адрес проживания (тип «Короткий текст»);
15) заметки (тип «Длинный текст»).

Создать таблицу «Ведомость» и в ней прописать следующие столбцы:

1) ИД (тип «Счетчик»);
2) учащийся (тип «Числовой»);
3) предмет (тип «Короткий текст»);
4) оценка (тип «Числовой»).

Заполнить таблицы данными. Создать ключевые поля с первичным ключом в таблицах «Учащиеся» и «Ведомость». В таблице «Ведомость» создать поле «Учащийся» с внешним ключом. Связать первичный ключ таблицы «Учащиеся» с внешним ключом таблицы «Ведомость».

**Ход работы**

*Создание базы*

Создать БД можно несколькими способами: с нуля или с использованием готового шаблона. На рис. 1.9 показано главное окно программы MS Access.

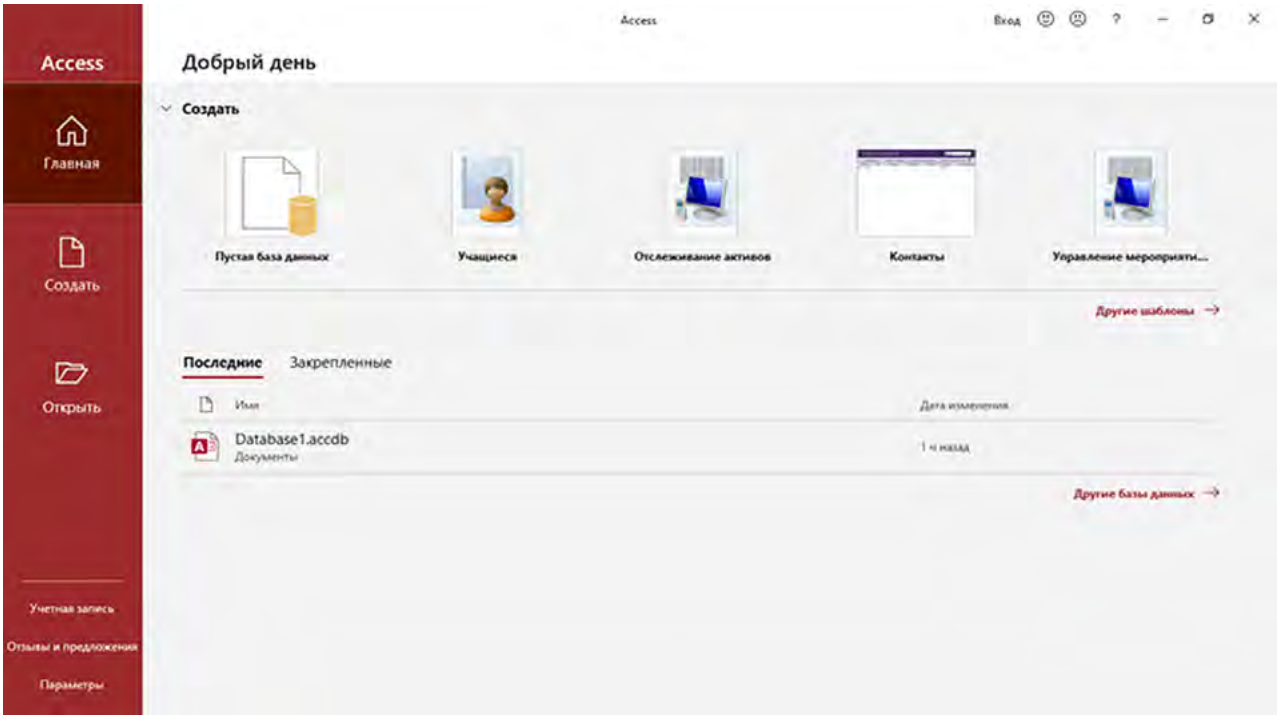


Рис. 1.9. Главное окно программы MS Access

В первом случае пользователю необходимо самостоятельно создавать элементы и объекты БД, во втором варианте все значительно проще — надо всего лишь запустить шаблон и вписать нужную информацию.

*Первый вариант работы с базами.* Создание новой базы начинается с нажатия кнопки «Пустая база данных» и последующим наполнением ее информацией.

В появившемся окне необходимо ввести название файла новой БД и нажать кнопку «Создать» (рис. 1.10).


Рис. 1.10. Диалоговое окно создания пустой БД

После создания файла откроется основное окно, где будет присутствовать одна таблица и можно приступать к наполнению базы.

*Для создания БД из шаблона* надо выбрать пользовательский шаблон. Выбираем один из требуемых объектов, например «Контакты» (рис. 1.11).

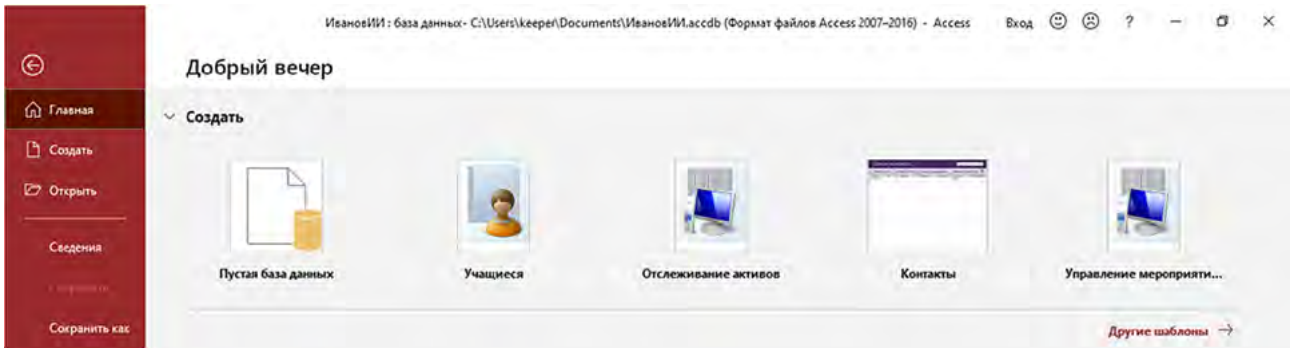


Рис. 1.11. Диалоговое окно создания БД

Появится окно, где задается имя файла и можно нажать кнопку «Создать», после чего появится готовая БД (рис. 1.12).

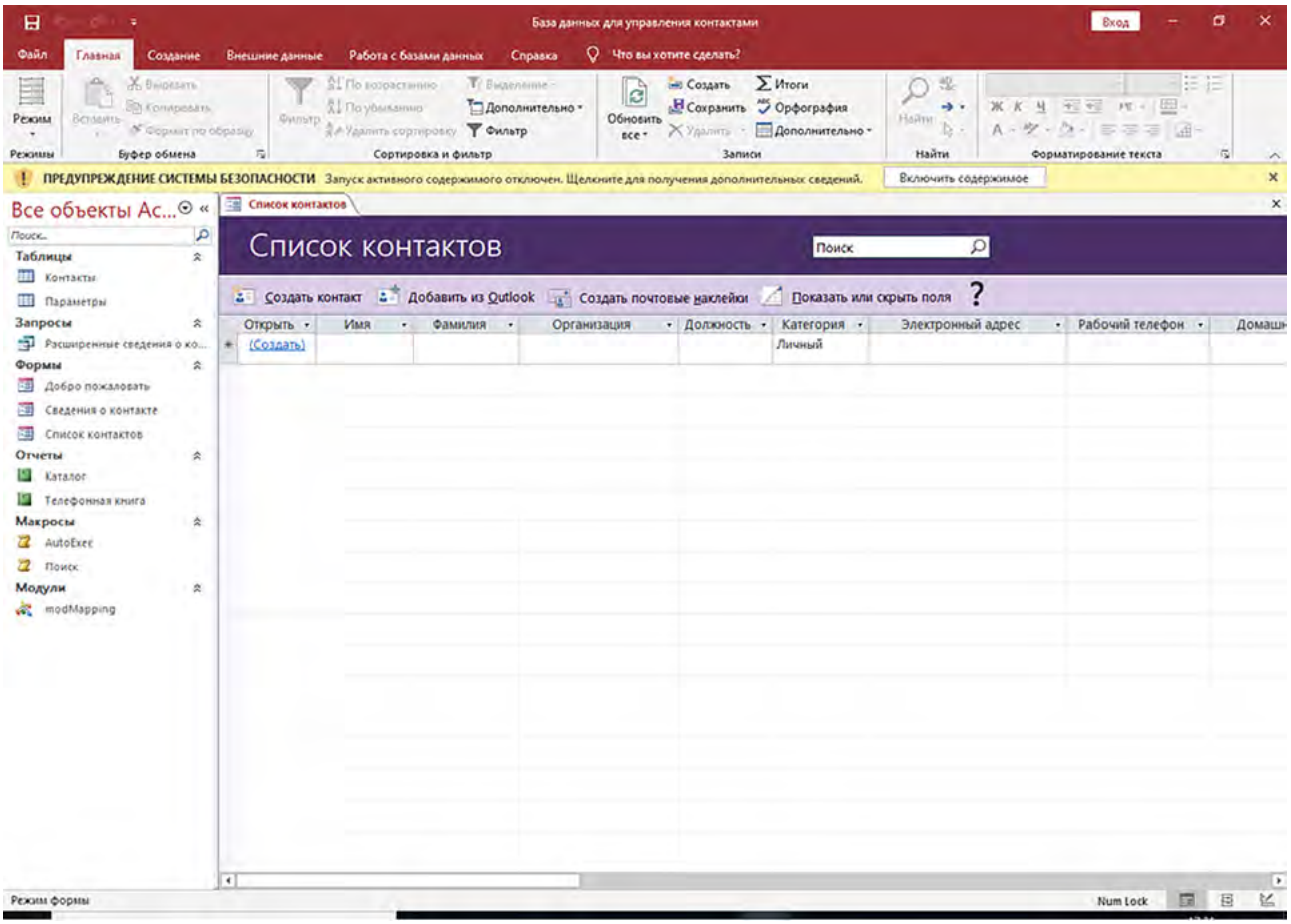


Рис. 1.12. Пример шаблона БД

*Наполнение базы*

Наполнение БД заключается в создании таблиц / таблицы и внесении в них / неё информации. Необходимо отметить, что в MS Access есть функция импорта, которую также можно использовать для ввода данных в случае наличия внешней БД.

После создания БД откроется пустая «Таблица1» с двумя пустыми столбцами (рис. 1.13).

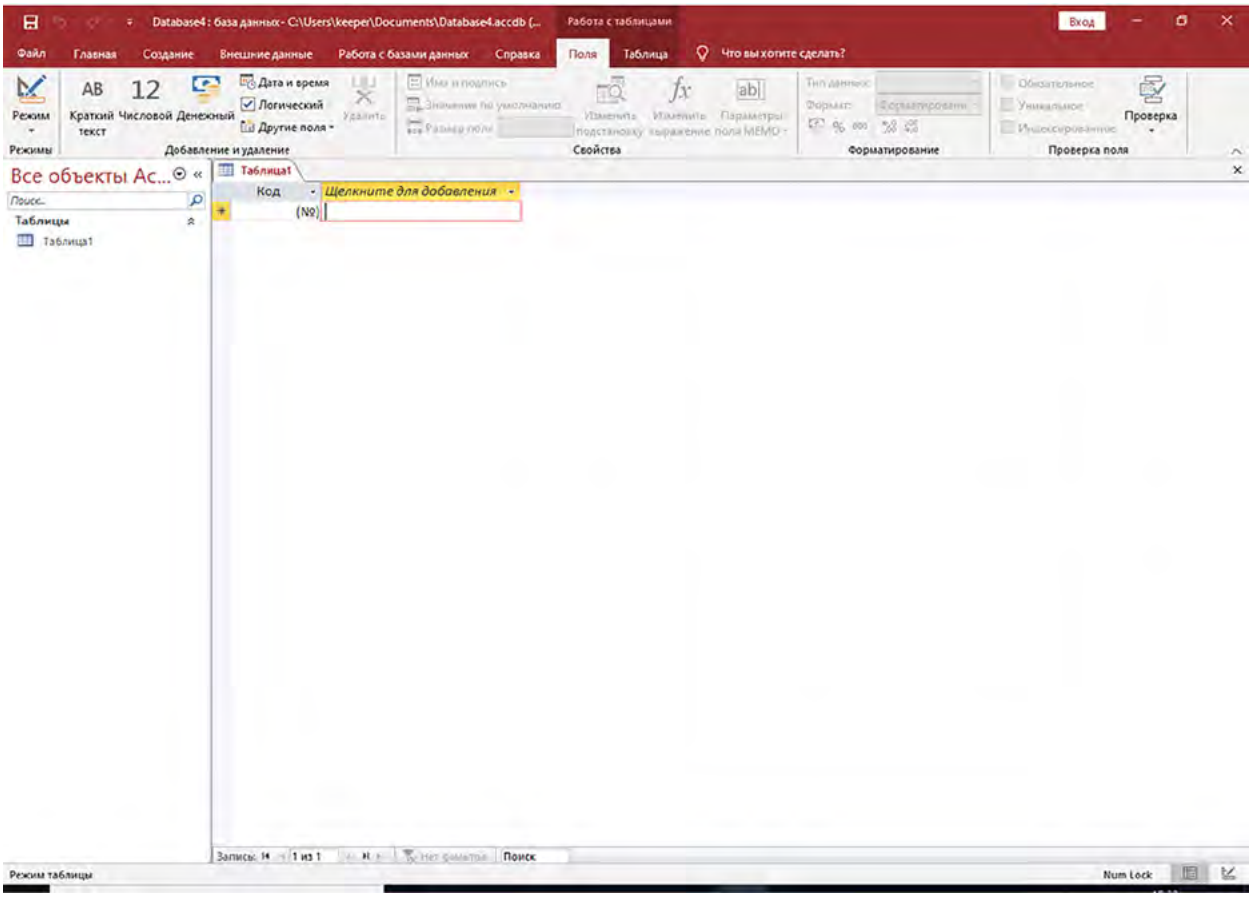


Рис. 1.13. Диалоговое окно «Создание таблиц»

Теперь надо дать таблице название. Это можно сделать, нажав на кнопку «Режим». Появится окно «Сохранение», где можно ввести новое название таблицы (рис. 1.14).

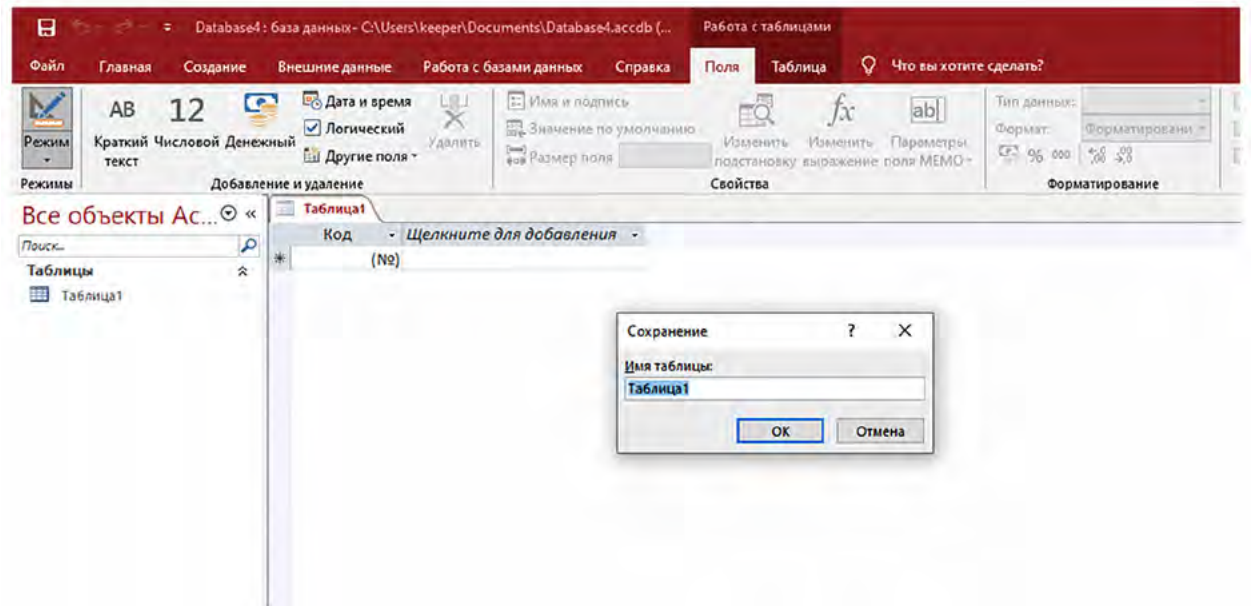


Рис. 1.14. Редактирование названия таблицы

Для того чтобы заполнить таблицу данными, необходимо сначала создать столбцы. Для этого переходим в режим «Конструктор» и начинаем заполнять столбцы «Имя поля» и «Тип данных», как описано в задании и изображено на рис. 1.15. В столбце «Тип данных» указываем соответствующие типы данных из задания.

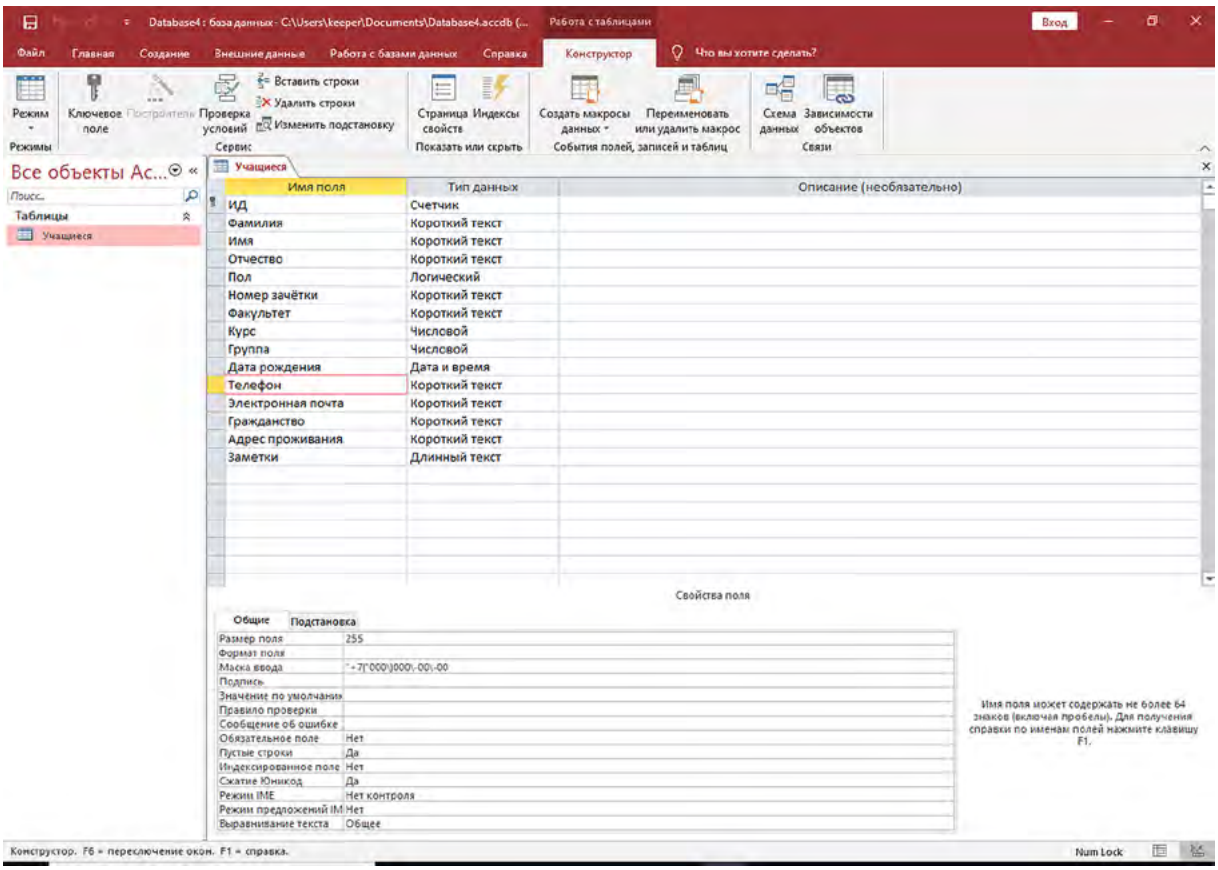


Рис. 1.15. Пример заполненной таблицы

Со второй таблицей «Ведомость» поступаем аналогично, в результате получится таблица, представленная на рис. 1.16.

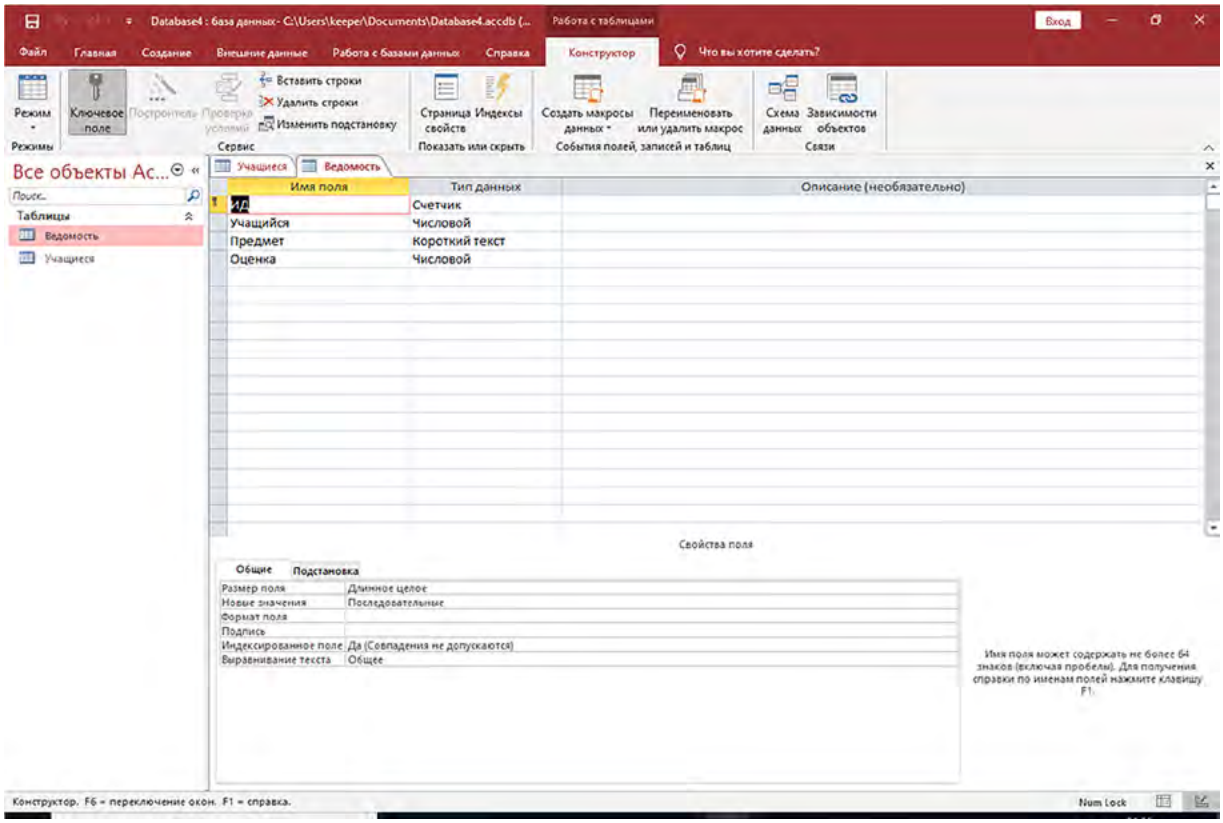


Рис. 1.16. Пример заполненной таблицы «Ведомость»

После того как поля созданы, необходимо сохранить данные, внесенные в таблицы. Для этого можно правой кнопкой мыши нажать на название таблицы и из выпадающего списка выбрать опцию «Сохранить». Теперь можно приступать к заполнению таблицы данными. Для этого ее переводим в режим «Таблица», нажав на соответствующую кнопку (рис. 1.17).

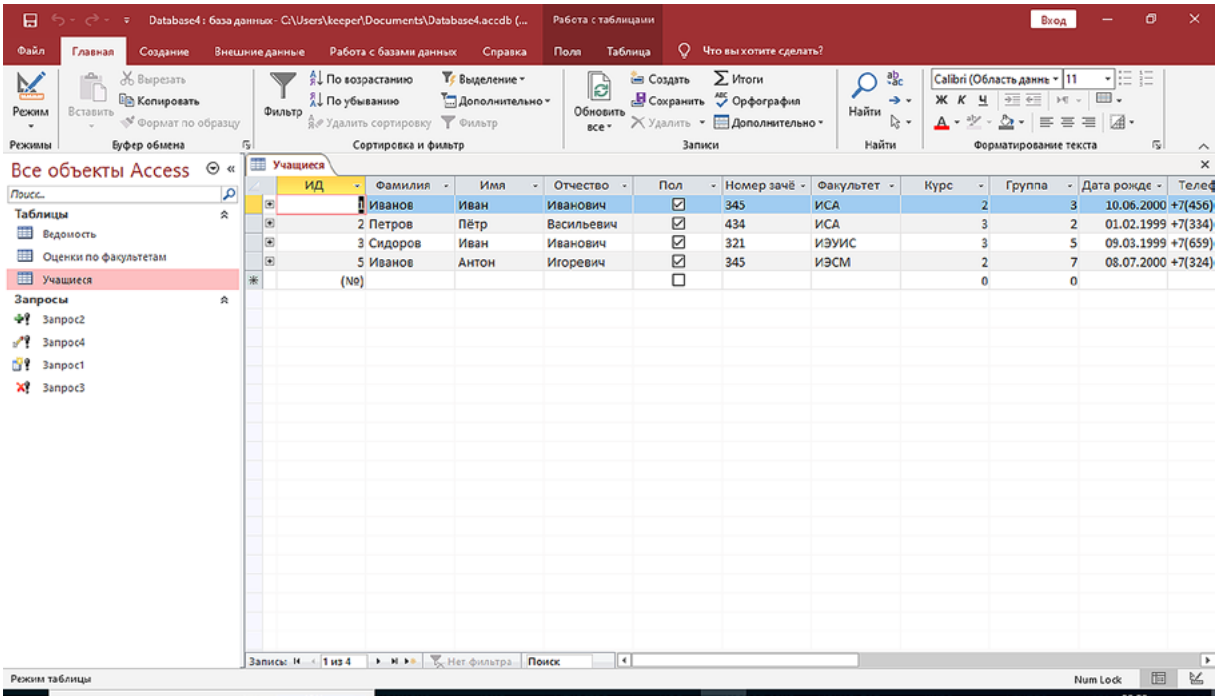


Рис. 1.17. Полученная для заполнения таблица

*Создание первичного ключа*

При работе с таблицами неизбежно возникает дублирование данных, так как некоторые из них указываются в разных таблицах, но являются связанными. Для того чтобы данные были связаны из одной таблицы с данными в других таблицах, необходимо задать параметр «Ключевое поле». Для этого перейдем в режим «Конструктор» и выделим поле, например «ИД», в контекстном меню выберем пункт «Ключевое поле», как показано на рис. 1.18. После этого сохраняем получившийся результат. Теперь можно продолжить ввод данных.

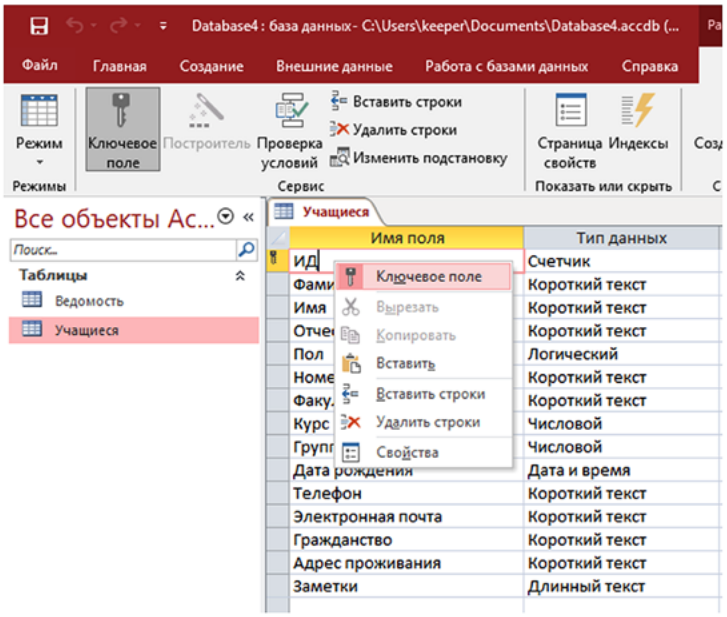


Рис. 1.18. Определение «Ключевого поля» таблицы

Конец ознакомительного фрагмента.
Приобрести книгу можно
в интернет-магазине
«Электронный универс»
e-Univers.ru