# Основные понятия теории баз данных

## 1. Базы данных и системы управления базами данных. Модели данных

**База данных (БД)** – это поименованная совокупность структурированных данных, относящихся к определенной предметной области. С понятием базы данных тесно связано понятие **системы управления базами данных** (СУБД). **СУБД** – это комплекс программных средств, предназначенных для создания структуры новой базы данных, наполнения ее содержимым, редактирования содержимого и создания различных пользовательских представлений информации (отбор, сортировка, фильтрация и т.д.). Существует множество различных систем управления базами данных, однако большинство СУБД опираются на единый комплекс основных понятий. Рассмотрим эти понятия.

**Информационный объект** – это описание некоторой **сущности** (реального объекта, явления, процесса, события) в виде совокупности логически связанных **реквизитов** (информационных элементов). Такими сущностями для информационных объектов могут служить: студент, вуз, сдача экзаменов, магазин, склад, и т.д. Например, информационный объект СТУДЕНТ может быть представлен в базе данных с помощью следующих реквизитов: **Номер зачетной книжки, Фамилия, Имя, Отчество, Год рождения**. Это можно записать так:

СТУДЕНТ (**Номер зачетной книжки, Фамилия, Имя, Отчество, Год рождения**)

Ядром любой базы данных является модель данных. **Модель данных** – это совокупность структур данных и операций их обработки. С помощью модели данных могут быть представлены информационные объекты и взаимосвязи между ними. Рассмотрим три основных типа моделей данных: иерархическую, сетевую и реляционную.

**Иерархическая модель данных** представляет собой совокупность элементов данных, расположенных в порядке их подчинения и образующих по структуре перевернутое дерево (рис. 1). К основным понятиям иерархической структуры относятся: уровень, узел и связь. Узел – это совокупность реквизитов данных, описывающих информационный объект.

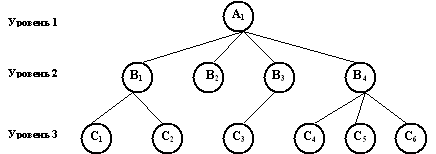


Рис. 1. Графическое изображение иерархической структуры данных.

Иерархическая структура должна удовлетворять следующим требованиям:

* каждый узел на более низком уровне связан только с одним узлом, находящимся на более высоком уровне;
* иерархическое дерево имеет только один корневой узел, не подчиненный никакому другому узлу и находящийся на самом верхнем уровне;
* к каждому узлу существует только один путь от корневого узла;

Пример, представленный на рисунке 2, иллюстрирует использование иерархической модели для построения базы данных «Институт». Информация базы данных структурирована в виде деревьев, количество которых равно количеству специальностей в институте. Информационные объекты первого уровня описываются реквизитами **номер специальности, название специальности, ФИО декана**. Информационные объекты второго уровня – реквизитами **шифр группы, ФИО старосты**, третьего уровня – реквизитами **номер зачетки, фамилия, имя, отчество**.

Подчеркиванием выделяется реквизит, который однозначно определяет описываемый информационный объект. Например, реквизит **номер специальности** однозначно определяет конкретную специальность. Для рассматриваемого примера иерархическая структура организации данных применима, так как каждый студент учится в определенной и только одной группе, которая относится к определенной и только одной специальности.

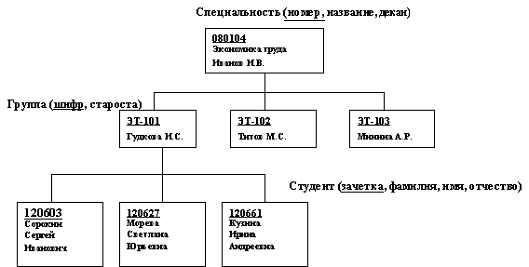


Рис. 2. Пример иерархической структуры данных.

**Сетевая модель данных** основана на тех же основных понятиях (уровень, узел, связь), что и иерархическая модель, но в сетевой модели каждый узел может быть связан с любым другим узлом. На рисунке 3 схематически изображена сетевая структура организации данных.

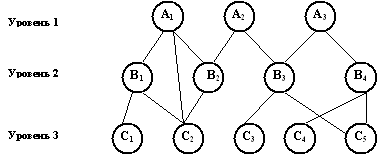


Рис. 3. Графическое изображение сетевой структуры данных.

Примером сетевой структуры может служить структура базы данных, содержащей сведения о студентах, занимающихся в спортивных секциях. Возможно участие одного студента в нескольких секциях, а также участие нескольких студентов в одной секции. Графическое изображение описанной в примере сетевой структуры, состоящей из двух типов информационных объектов, показано на рисунке 4.

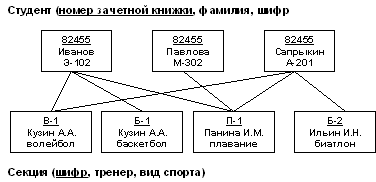


Рис. 4. Пример сетевой структуры данных.

**Реляционная модель данных** использует организацию данных в виде двумерных таблиц. Каждая **реляционная таблица** представляет собой двумерный массив и обладает следующими свойствами:

* все столбцы в таблице однородные, т.е. все элементы в одном столбце имеют одинаковый тип (числовой, символьный и т.д.) и максимально допустимый размер;
* каждый столбец имеет уникальное имя;
* одинаковые строки в таблице отсутствуют;
* порядок следования строк и столбцов в таблице может быть произвольным.

Объектами обработки реляционной БД являются (рис. 5):

**Поле** – элементарная единица логической организации данных, которая соответствует конкретному реквизиту информационного объекта (**столбец** реляционной таблицы).

**Запись** – совокупность логически связанных полей (**строка** реляционной таблицы).

**Таблица** – заданная структура полей, состоящая из конечного набора однотипных записей.

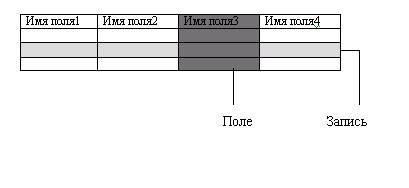


Рис. 5. Основные структурные элементы реляционной таблицы.

Например, в виде реляционной таблицы можно представить информацию о студентах, обучающихся в вузе:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Номер личного дела** | **Фамилия** | **Имя** | **Отчество** | **Пол** | **Дата рождения** | **Специальность** |
| 16493 | Сергеев | Петр | Михайлович | м | 01.01.86 | 080104 |
| 16593 | Петрова | Анна | Владимировна | ж | 15.03.85 | 080102 |
| 16693 | Анохин | Андрей | Борисович | м | 14.04.86 | 080104 |

## 2. Основы проектирования реляционных баз данных

Базы данных могут содержать различные объекты, но основными объектами любой базы данных являются таблицы. Простейшая база данных имеет хотя бы одну таблицу. Даже если таблица не содержит данных, она уже содержит информацию – это информация о структуре таблицы. **Структура таблицы** определяет методы занесения данных и хранения их в базе данных. Таблицы базы данных создаются таким образом, чтобы каждая из них содержала информацию об одном информационном объекте. Не следует сводить в одну таблицу данные, относящиеся к разным информационным объектам.

После создания различных таблиц, содержащих данные, относящиеся к различным информационным объектам базы данных, между этими таблицами должны быть установлены реляционные связи. Установка таких связей делает возможным выполнение одновременной обработки данных из нескольких таблиц. Для установки связей обычно используют ключевые поля таблиц.

**Первичный ключ** реляционной таблицы – это поле или группа полей, которые позволяют однозначно определить каждую строку в таблице. Первичный ключ должен обладать двумя свойствами:

* **однозначная идентификация записи** – запись должна однозначно определяться значением ключа;
* **отсутствие избыточности** – никакое поле нельзя удалить из ключа, не нарушая при этом свойства однозначной идентификации записи.

Если первичный ключ состоит из одного поля, то он называется **простым ключом** или **ключевым полем**. Если первичный ключ состоит из нескольких полей, то такая таблица базы данных имеет **составной ключ**.

Чтобы связать две реляционные таблицы, необходимо ключ одной связываемой таблицы ввести в состав ключа другой связываемой таблицы (возможно совпадение ключей) или ввести в структуру одной таблицы **внешний ключ**, т.е. поле, не являющееся ключевым в первой таблице, но являющееся ключевым во второй.

Пример. Рассмотрим базу данных «Деканат», состоящую из трех таблиц: СТУДЕНТ, СЕССИЯ, СТИПЕНДИЯ, имеющих показанную ниже структуру.

Таблица СТУДЕНТ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Номер личного дела** | Фамилия | Имя | Отчество | Группа |
| 120603 | Свиридова | Анна | Алексеевна | ЭТ-101 |
| 120608 | Голубев | Игорь | Андреевич | ЭТ-102 |
| … | … | … | … | … |

В этой таблице ключевым полем является поле **Номер личного дела**, оно однозначно определяет каждую запись (строку) таблицы, т.к. не существует двух или более личных дел с одинаковыми номерами.

Таблица СЕССИЯ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Номер личного дела** | Экзамен 1 | Экзамен 2 | Экзамен 3 | Результат |
| 120603 | 3 | 4 | 4 | 0 |
| 120608 | 5 | 5 | 5 | 2 |
| 120634 | 4 | 5 | 4 | 1 |
| … | … | … | … |  |

В столбце **Результат** содержатся числа 0, 1 или 2 в зависимости от того, получает ли студент стипендию по результатам сессии (0 – студент не получает стипендию, 1 – обычная стипендия, 2 – увеличенная в 2 раза стипендия). В этой таблице ключевым полем также является поле **Номер личного дела**.

Таблица СТИПЕНДИЯ

|  |  |
| --- | --- |
| **Результат** | Процент |
| 0 | 0 |
| 1 | 100 |
| 2 | 200 |

Эта таблица содержит информацию о проценте начисляемой студенту стипендии в зависимости от результата сдачи сессии. В этой таблице ключевым является поле **Результат**.

Для наглядности представления связей между таблицами перейдем к представлению таблиц в виде структур этих таблиц, т.е. будем указывать только имена полей таблиц. Итак, наша база данных содержит три таблицы (рис. 6).

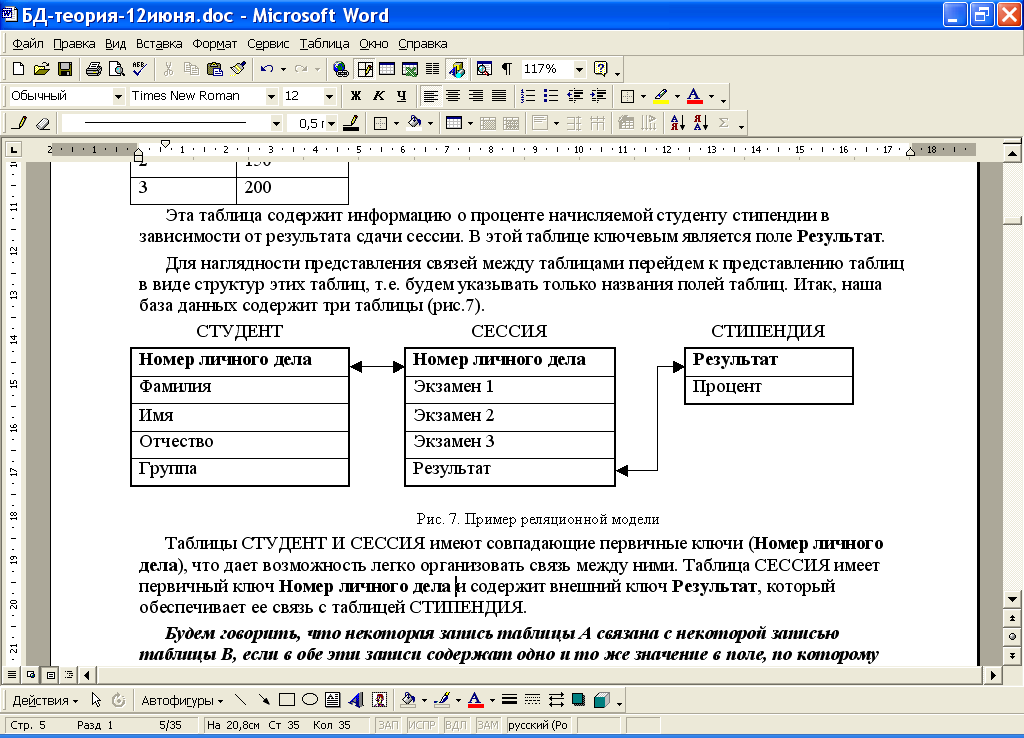


Рис. 6. Пример реляционной структуры данных.

Таблицы СТУДЕНТ и СЕССИЯ имеют совпадающие первичные ключи (**Номер личного дела**), что дает возможность легко организовать связь между ними. Таблица СЕССИЯ имеет первичный ключ **Номер личного дела** и содержит внешний ключ **Результат**, который обеспечивает ее связь с таблицей СТИПЕНДИЯ.

Будем говорить, что ***некоторая запись таблицы А связана с некоторой записью таблицы В, если в обеих таблицах эти записи содержат одно и то же значение в поле, по которому установлена связь между таблицами***.

Различают **три типа связей (отношений) между таблицами** реляционной базы данных:

* отношение «один-к-одному» (1 — 1)
* отношение «один-ко-многим» (1 — М)
* отношение «многие-ко-многим» (М — М)

Говорят, что ***таблицы А и В находятся в отношении «один-к-одному», если каждая запись в таблице А может иметь не более одной связанной с ней записи в таблице В и наоборот, каждая запись в таблице В может иметь не более одной связанной с ней записи в таблице А***.

В этом случае для связи используются ключевые поля связываемых таблиц. Этот тип связи используется достаточно редко, т.к. в этом случае данные двух таблиц могут быть объединены в одну таблицу. Можно указать следующие причины, по которым все-таки выполняется разбиение одной таблицы на несколько таблиц, связанных отношением «один-к-одному»:

* разделение очень «широкой» таблицы на несколько таблиц с целью облегчения работы;
* отделение части таблицы по соображениям защиты ее от несанкционированного доступа.

Рассмотренные ранее таблицы СТУДЕНТ и СЕССИЯ находятся в отношении «один-к-одному» (иначе говоря, между таблицами установлена связь типа «один-к-одному») (рис. 7).

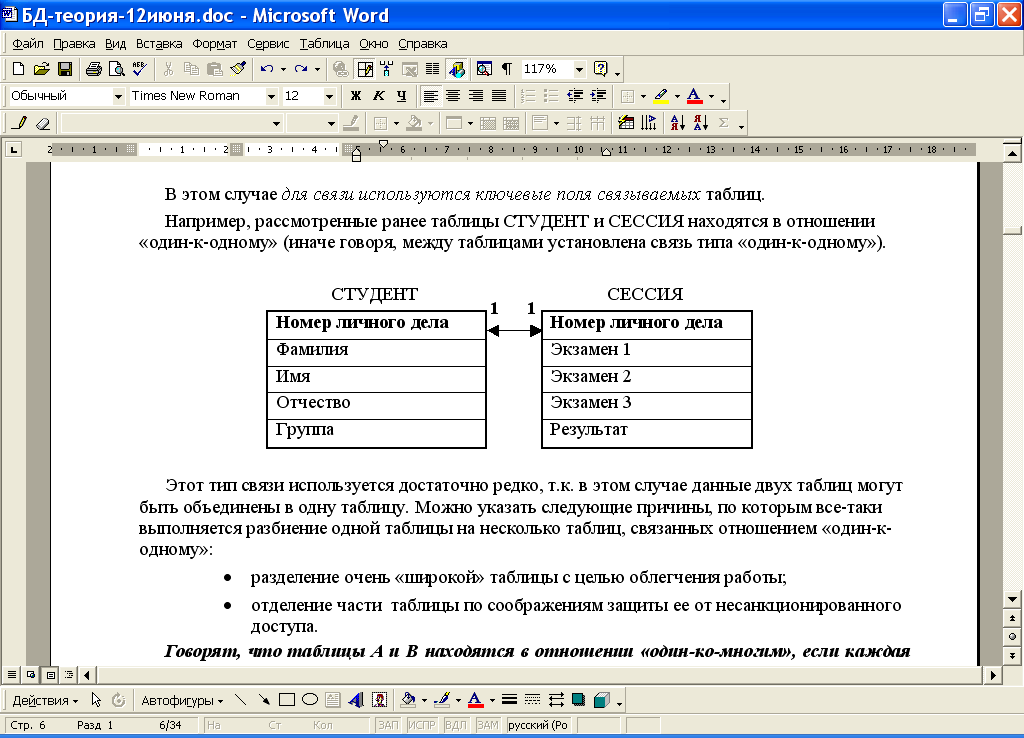


Рис. 7. Пример таблиц, находящихся в отношении «один-к-одному».

Говорят, что ***таблицы А и В находятся в отношении «один-ко-многим», если каждая запись в таблице А может быть связана с несколькими записями таблицы В, но каждая запись в таблице В не может быть связана более чем с одной записью таблицы А. Таблица А в этом случае называется главной таблицей, а таблица В – подчиненной***.

В этом случае для связи используется поле, которое является первичным ключом таблицы, находящейся на стороне отношения «один» (таблица А), и являющееся внешним ключом в таблице, находящейся на стороне отношения «многие» (таблица В).

Например, рассмотренные ранее таблицы СТИПЕНДИЯ и СЕССИЯ находятся в отношении «один-ко-многим» (рис. 8). При этом на стороне «один» находится таблица СТИПЕНДИЯ, а на стороне «многие» – таблица СЕССИЯ. Связь устанавливается по полю **Результат**. Каждая запись таблицы СТИПЕНДИЯ может иметь много связанных с ней записей в таблице СЕССИЯ, иначе говоря, в таблице СЕССИЯ может быть много строк с заданным значением в поле **Результат** (например, со значением 1). В то же время, если взять любую строку в таблице СЕССИЯ, то для нее найдется не более одной строки в таблице СТИПЕНДИЯ с таким же значением в поле **Результат**.

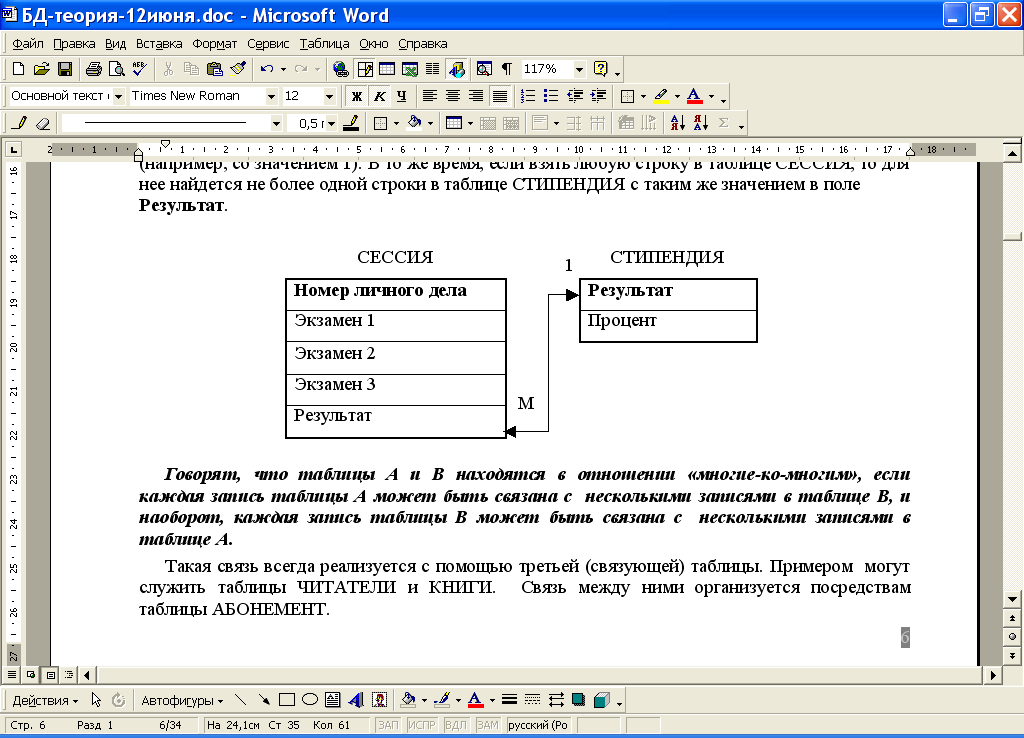


Рис. 8. Пример таблиц, находящихся в отношении «один-ко-многим».

Говорят, что ***таблицы А и В находятся в отношении «многие-ко-многим», если каждая запись таблицы А может быть связана с несколькими записями в таблице В, и наоборот, каждая запись таблицы В может быть связана с несколькими записями в таблице А***.

Такая связь всегда реализуется с помощью третьей (связующей) таблицы. Примером могут служить таблицы Читатели и Книги (рис. 9). Связь между ними организуется посредствам таблицы Абонемент.

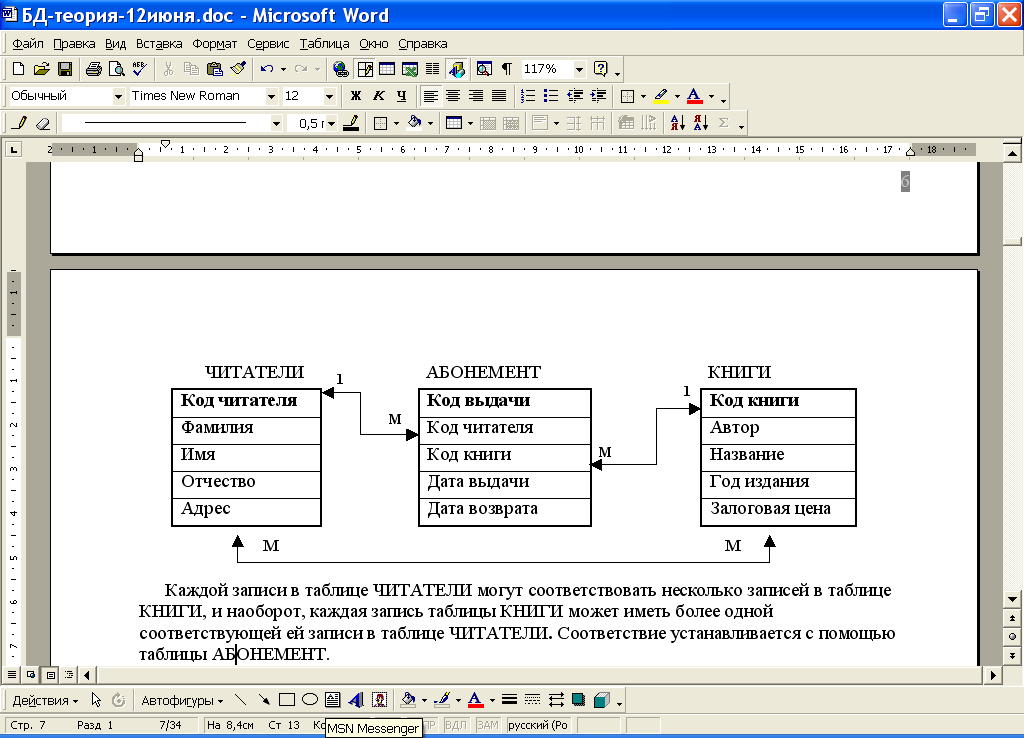


Рис. 9. Пример таблиц, находящихся в отношении «многие-ко-многим».

Каждой записи в таблице Читатели могут соответствовать несколько записей в таблице Книги, и наоборот, каждая запись таблицы Книги может иметь более одной соответствующей ей записи в таблице Читатели. Соответствие устанавливается с помощью таблицы абонемент. При этом между таблицами ЧИТАТЕЛИ и АБОНЕМЕНТ установлено отношение «один-ко-многим», в котором таблица ЧИТАТЕЛИ является главной, а таблица АБОНЕМЕНТ – подчиненной. Аналогично между таблицами КНИГИ и АБОНЕМЕНТ установлено отношение «один-ко-многим», в которой таблица КНИГИ является главной.

## 3. Этапы проектирования реляционной базы данных

В процессе разработки базы данных можно выделить следующие этапы:

**1. Анализ предметной области**

На этом этапе формируется задание по созданию БД. В нем подробно описывается состав базы данных, назначение и цели ее создания, а также перечисляется, какие виды работ предполагается осуществлять в этой базе данных (отбор, изменение данных, печать или вывод отчета и т. д.).

**2. Разработка схемы данных БД.** На этом этапе рассматривается, из каких информационных объектов должна состоять БД, какими реквизитами описывается каждый объект. Затем определяется структура реляционных таблиц БД с указанием свойств полей и связей между таблицами.

**3. Синтез компьютерной модели объекта**, предполагающий выполнение следующих типовых операций:

1. Создание файла базы данных.
2. Создание исходной таблицы или таблиц.
3. Создание экранных форм.

Экранные формы создаются с целью облегчения процедуры ввода, редактирования и просмотра данных таблицы. На основе одной таблицы можно создать несколько форм, которые могут отличаться видом или количеством используемых полей данной таблицы.

1. Заполнение данными таблиц БД.
2. Работа с созданной базой данных:

* сортировка, фильтрация и поиск записей в таблице;
* отбор данных из таблиц в соответствии с заданными критериями отбора;
* выполнение обработки данных (удаление, добавление, изменение данных, выполнение вычислений);
* подготовка отчетов.

Рассмотрим подробно этап **разработки схемы данных**. Разработку схемы данных обычно выполняют в следующей последовательности:

1. Составляется общий список полей, отражающий реквизиты информационных объектов БД.
2. В соответствии с типом данных, размещаемых в каждом поле, определяют тип и другие свойства каждого поля.
3. Поля общего списка распределяются по базовым таблицам. Критерии распределения полей по таблицам могут быть различными. Например, сведения о сотрудниках организации могут храниться в одной таблице или могут быть созданы две таблицы, одна из которых хранит информацию служебного назначения (СОТРУДНИКИ), а другая – информацию личного плана (ЛИЧНЫЕ ДАННЫЕ) (рис. 10).

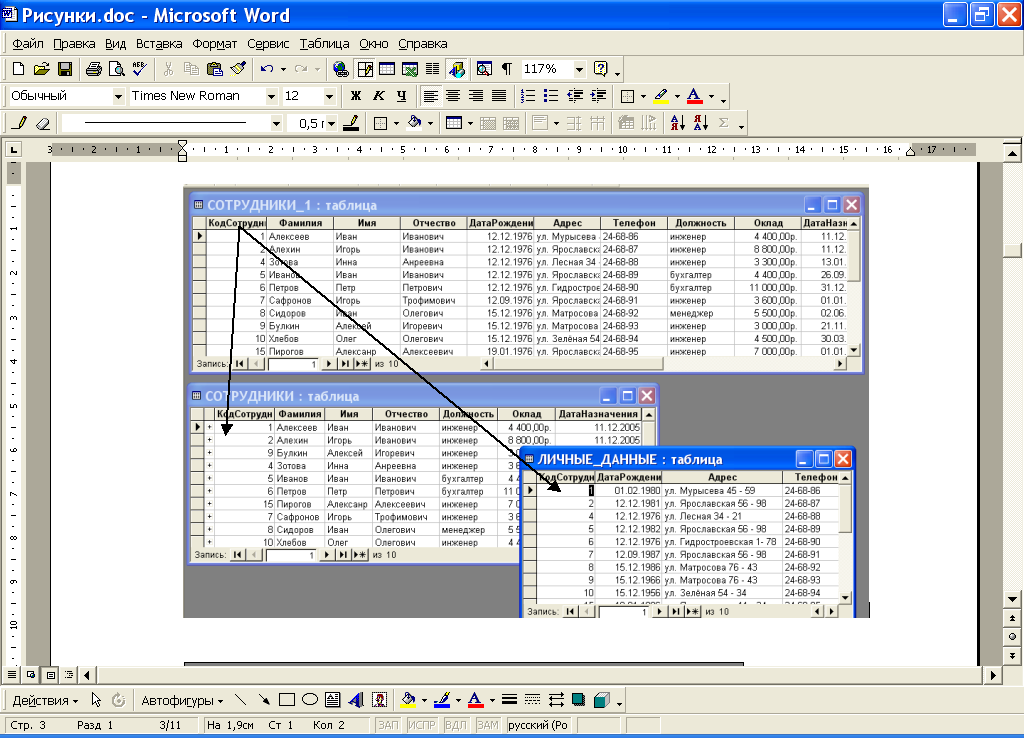


Рис. 10. Пример разделения одной таблицы на две по признаку разделения доступа к данным разных таблиц.

Однако чаще критерием выделения таблицы является факт повторения данных в таблице. Например, в таблице СОТРУДНИКИ многократно повторяются названия должностей и оклады. Возможно, было бы целесообразно создать таблицу ДОЛЖНОСТИ, в которой хранить названия и характеристики должностей (рис.11). Тогда данные будут храниться в двух таблицах: например, это могут быть таблицы СОТРУДНИКИ\_2 и ДОЛЖНОСТИ. В таблицу СОТРУДНИКИ\_2 включен внешний ключ таблицы ДОЛЖНОСТИ – поле **Код должности**. По этому полю устанавливается связь между таблицами.

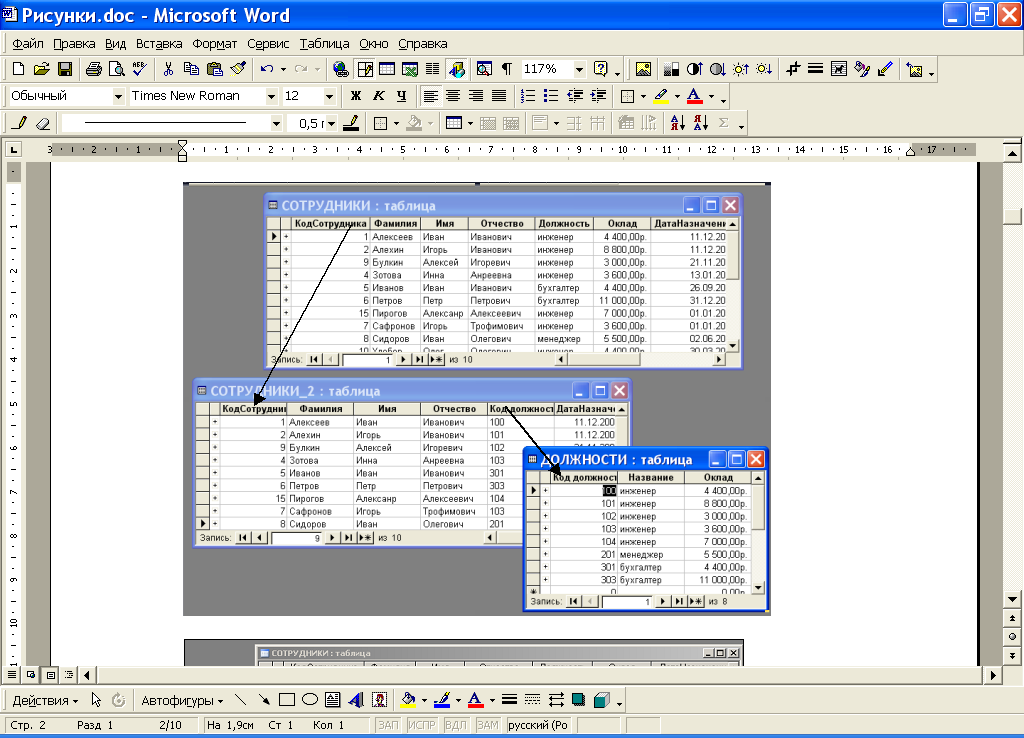


Рис. 11. Пример разделения одной таблицы на две по признаку устранения повторяемости данных.

1. Обычно в каждой из таблиц намечают **ключевое поле**. В качестве ключевого выбирают поле, данные в котором повторяться не могут и которое однозначно определяет каждую запись таблицы. Например, для таблицы СОТРУДНИКИ\_2 ключевым может быть поле, содержащее уникальный код каждого сотрудника (поле **КодСотрудника**).
2. Определяются связи между таблицами. На рисунке 12 показан пример установки связей между таблицами, составляющими одну базу данных. Такой чертеж называется **схемой данных**.

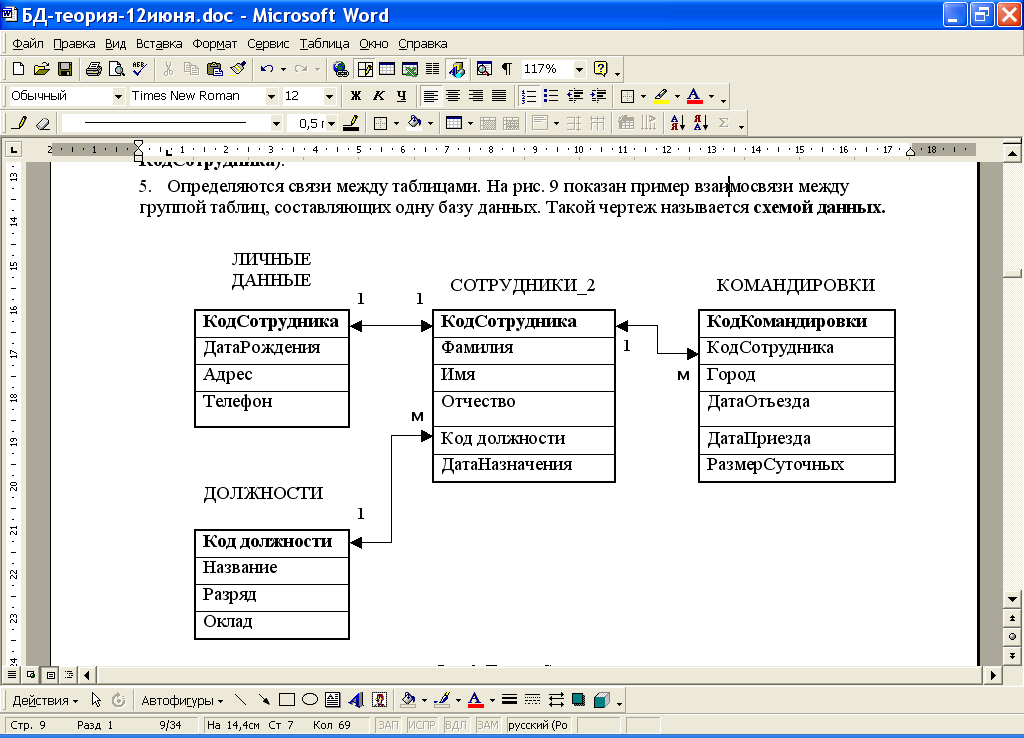


Рис. 12. Пример схемы данных.

6. Разработкой схемы данных заканчивается «бумажный» этап работы проектирования базы данных. Далее следует этап компьютерной реализации базы данных, который в значительной мере определяется выбором конкретной СУБД для реализации базы данных.

## 4. Вопросы

1. Что такое база данных (БД), система управления базами данных (СУБД)?
2. Что такое информационный объект?
3. Что такое модель данных?
4. Укажите основные понятия и характеристики иерархической и сетевой моделей данных.
5. Укажите характеристики и объекты обработки реляционной модели данных.
6. Что такое первичный ключ реляционной таблицы базы данных?
7. Что такое внешний ключ реляционной таблицы базы данных?
8. Какого типа связи (отношения) могут быть установлены между таблицами реляционной базы данных? Охарактеризуйте каждый тип связи.
9. Что такое «схема данных» реляционной базы данных?