**2. ПРИМЕР ПРОЕКТИРОВАНИЯ РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ**

Составитель: канд. техн. наук И.П. Карпова

В качестве примера возьмем базу данных компании, которая занимается издательской деятельностью.

**2.1. Инфологическое проектирование**

**2.1.1. Анализ предметной области**

База данных создаётся для информационного обслуживания редакторов, менеджеров и других сотрудников компании. БД должна содержать данные о сотрудниках компании, книгах, авторах, финансовом состоянии компании и предоставлять возможность получать разнообразные отчёты.

В соответствии с предметной областью система строится с учётом следующих особенностей:

* каждая книга издаётся в рамках контракта;
* книга может быть написана несколькими авторами;
* контракт подписывается одним менеджером и всеми авторами книги;
* каждый автор может написать несколько книг (по разным контрактам);
* порядок, в котором авторы указаны на обложке, влияет на размер гонорара;
* если сотрудник является редактором, то он может работать одновременно над несколькими книгами;
* у каждой книги может быть несколько редакторов, один из них – ответственный редактор;
* каждый заказ оформляется на одного заказчика;
* в заказе на покупку может быть перечислено несколько книг.

Выделим базовые сущности этой предметной области:

* **Сотрудники** компании. Атрибуты сотрудников – ФИО, табельный номер, пол, дата рождения, паспортные данные, ИНН, должность, оклад, домашний адрес и телефоны. Для редакторов необходимо хранить сведения о редактируемых книгах; для менеджеров – сведения о подписанных контрактах.
* **Авторы**. Атрибуты авторов – ФИО, ИНН (индивидуальный номер налогоплательщика), паспортные данные, домашний адрес, телефоны. Для авторов необходимо хранить сведения о написанных книгах.
* **Книги**. Атрибуты книги – авторы, название, тираж, дата выхода, цена одного экземпляра, общие затраты на издание, авторский гонорар.

**Контракты** будем рассматривать как связь между авторами, книгами и менеджерами. Атрибуты контракта – номер, дата подписания и участники.

Для отражения финансового положения компании в системе нужно учитывать **заказы** на книги. Для заказа необходимо хранить номер заказа, заказчика, адрес заказчика, дату поступления заказа, дату его выполнения, список заказанных книг с указанием количества экземпляров.

ER–диаграмма издательской компании приведена на рис. 3 (базовые сущности на рисунках выделены полужирным шрифтом).

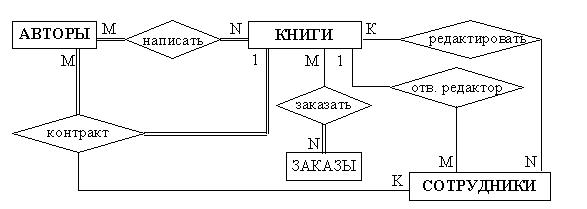


Рис.3. ER–диаграмма издательской компании

**2.1.2. Анализ информационных задач и круга пользователей системы**

Система создаётся для обслуживания следующих групп пользователей:

* администрация (дирекция);
* менеджеры;
* редакторы;
* сотрудники компании, обслуживающие заказы.

Определим границы информационной поддержки пользователей:

**1) Функциональные возможности:**

* ведение БД (запись, чтение, модификация, удаление в архив);
* обеспечение логической непротиворечивости БД;
* обеспечение защиты данных от несанкционированного или случайного доступа (определение прав доступа);
* реализация наиболее часто встречающихся запросов в готовом виде;
* предоставление возможности сформировать произвольный запрос на языке манипулирования данными.

**2) Готовые запросы:**

* получение списка всех текущих проектов (книг, находящихся в печати и в продаже);
* получение списка редакторов, работающих над книгами;
* получение полной информации о книге (проекте);
* получение сведений о конкретном авторе (с перечнем всех книг);
* получение информации о продажах (по одному или по всем проектам);
* определение общей прибыли от продаж по текущим проектам;
* определение размера гонорара автора по конкретному проекту.

**2.2. Определение требований к операционной обстановке**

Для выполнения этого этапа необходимо знать (хотя бы ориентировочно) объём работы издательства (т.е. количество книг, авторов и заказчиков), а также иметь представление о характере и интенсивности запросов.

Объём внешней памяти, необходимый для функционирования системы, складывается из двух составляющих: память, занимаемая модулями СУБД (ядро, утилиты, вспомогательные программы), и память, отводимая под данные (МД). Наиболее существенным обычно является МД. Объём памяти МД, требуемый для хранения данных, можно приблизительно оценить по формуле

http://rema44.ru/resurs/study/dbprj/Image5.gif,

где li – длина записи в i-й таблице (в байтах), Ni – примерное (максимально возможное) количество записей в i-й таблице, Na – количество записей в архиве i-й таблицы. Коэффициент 2 перед суммой нужен для того, чтобы выделить память для хранения индексов, промежуточных данных, для выполнения объёмных операций (например, сортировки) и т.п.

Посчитаем приблизительно, какой объём внешней памяти потребуется для хранения данных. Примем ориентировочно, что:

* одновременно осуществляется около пятидесяти проектов, работа над проектом продолжается в среднем два месяца (по 0,3К);
* в компании работает 100 сотрудников (по 0,2К на каждого сотрудника);
* издательство сотрудничает с тридцатью авторами (по 0,2К);
* в день обслуживается порядка двадцати заявок (по 0,1К);
* устаревшие данные переводятся в архив.

Тогда объём памяти для хранения данных за первый год примерно составит:

Mc = 2(100\*0,2+6(50\*0,3)+30\*0,2+250(20\*0,1)) = 1232 К  1,2 М,

где 250 – количество рабочих дней в году, а 12 мес./2 мес. = 6. Объём памяти будет увеличиваться ежегодно на столько же при сохранении объёма работы.

Объём памяти, занимаемый программными модулями пользователя, обычно невелик по сравнению с объёмом самих данных, поэтому может не учитываться. Требуемый объём оперативной памяти определяется на основании анализа интенсивности запросов и объёма результирующих данных.

**2.3. Выбор СУБД и других программных средств**

Анализ информационных задач показывает, что для реализации требуемых функций подходят почти все СУБД для ПЭВМ (FoxPro, Clipper, MS Access и др.). Все они поддерживают реляционную модель данных и предоставляют разнообразные возможности для работы с данными.

Объём внешней и оперативной памяти, требующийся для функционирования СУБД, обычно указывается в сопроводительной документации.

Для того чтобы в учебном примере не привязываться к конкретной СУБД, выполним описание логической схемы БД на SQL-92.

**2.4. Логическое проектирование реляционной БД**

**2.4.1. Преобразование ER–диаграммы в схему базы данных**

База данных создаётся на основании схемы базы данных. Для преобразования ER–диаграммы в схему БД приведём уточнённую ER–диаграмму, содержащая атрибуты сущностей (рис. 4).

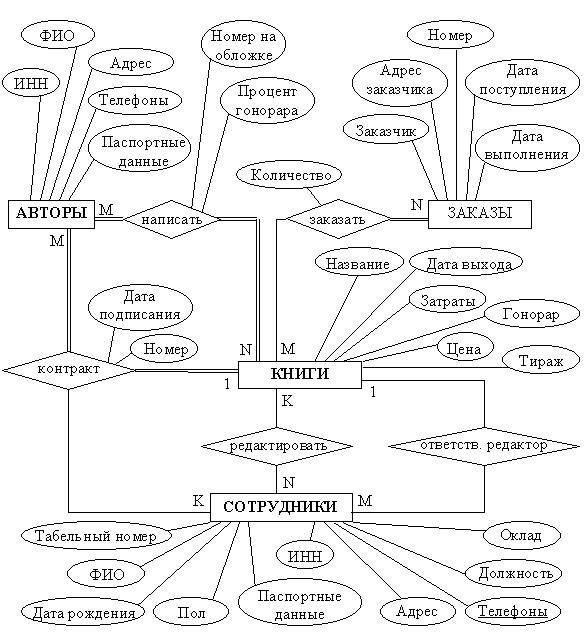


Рис.4. Уточнённая ER–диаграмма издательской компании

**Примечание:** многозначные атрибуты на рисунке выделены подчеркиванием.

Преобразование ER–диаграммы в схему БД выполняется путем сопоставления каждой сущности и каждой связи, имеющей атрибуты, отношения (таблицы БД). Будем использовать обозначения, представленные на рис. 5.

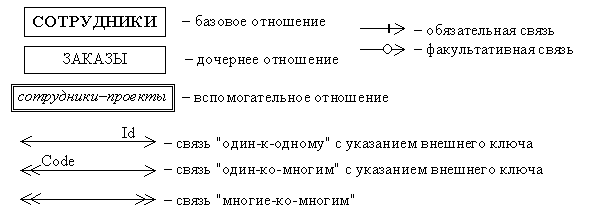


Рис.5. Обозначения, используемые на схеме базы данных

Полученная схема реляционной базы данных (РБД) приведена на рис. 6.

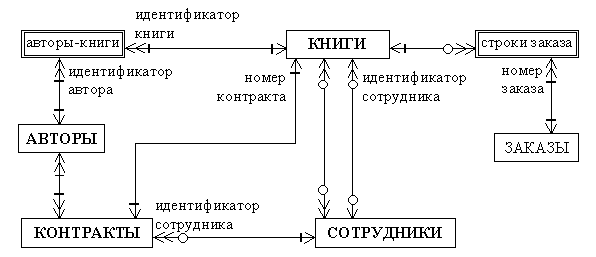


Рис.6. Схема РБД, полученная из ER–диаграммы издательской компании

На схеме (рис. 6) есть связь типа 1:1 – обязательная связь между *КНИГАМИ* и *КОНТРАКТАМИ*. Такие отношения следует объединять в одно. Дополнительный эффект от объединения этих отношений – слияние связей авторы–контракты и авторы–книги: ведь в нашем случае контракт заключается именно для написания книги.

**Примечание:** исключение для связи типа 1:1 составляют ситуации, когда для увеличения производительности системы в отдельную таблицу выделяются редко используемые данные большого объёма.

Связь типа 1:n (один-ко-многим) между отношениями реализуется через внешний ключ. Ключ вводится для того отношения, к которому осуществляется множественная связь (*КНИГИ*).

Связь *редактировать* между отношениями *КНИГИ* и *СОТРУДНИКИ* принадлежит к типу n:m (многие-ко-многим). Этот тип связи реализуется через вспомогательное отношение, которое является соединением первичных ключей соответствующих отношений.

Бинарная связь между отношениями не может быть обязательной для обоих отношений. После объединения сущностей *КНИГИ* и *КОНТРАКТЫ* остаётся три связи, обязательные для всех участников: между авторами и книгами и между заказами и строками заказов. Такой тип связи означает, что, например, прежде чем добавить новый заказ в отношение *ЗАКАЗЫ*, нужно добавить новую строку в отношение *СТРОКИ ЗАКАЗА*, и наоборот. Поэтому для такой связи необходимо снять с одной стороны условие обязательности. Так как все эти связи будут реализованы с помощью внешнего ключа, снимем условие обязательности связей для отношений, содержащих первичные ключи.

Уточнённая схема РБД издательской компании приведена на рис. 7.

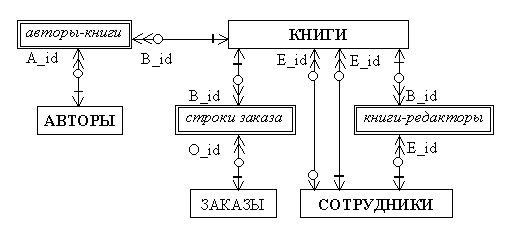


Рис.7. Уточнённая схема РБД издательской компании

Схема на рис. 7 содержит цикл "сотрудники–книги–сотрудники". Цикл допустим только в том случае, если связи, входящие в него, независимы друг от друга. Примем для нашей ПО, что ответственный редактор книги может являться также просто редактором этой же книги или не входить в число редакторов. При этом цикл не приводит к нарушению логической целостности данных.

**Примечание.** Существует несколько подходов для разрешения ситуаций, в которых связи, входящие в цикл, зависят друг от друга. Рассмотрим пример цикла "отделы–проекты–сотрудники–отделы" (рис. 8,а). Будем считать, что в выполнении проекта могут участвовать только сотрудники, работающие в том же отделе, к которому относится проект. При циклической схеме СУБД не сможет гарантировать логическую целостность данных без использования дополнительных средств.

Один из способов – разорвать цикл, исключив одну из связей (рис. 8,б) или введя промежуточное отношение (рис. 8,в). В нашем случае можно было бы разорвать связь "сотрудники–проекты", если бы каждый сотрудник участвовал во всех проектах своего отдела. Промежуточное отношение можно было бы использовать, если бы существовала общая связь между сущностями, входящими в цикл. Например, если бы каждый сотрудник заключал договор с отделом на выполнение работ в рамках проекта. Тогда сущность *ДОГОВОРЫ* отражала бы связь между отделами, сотрудниками и проектами.

Другой способ разрешения цикла заключается в том, что в промежуточное отношение *СОТРУДНИКИ– ПРОЕКТЫ*, которое реализует связь многие-ко-многим, добавляются (мигрируют) внешние ключи *Код отдела* (*D\_id*) из отношений *СОТРУДНИКИ* и *ПРОЕКТЫ* (рис. 8,г). Эти ключи проверяются на равенство друг другу с помощью соответствующего ограничения целостности. Использование этого способа возможно в том случае, когда соответствующие связи (*отдел–проект* и *отдел–сотрудник*) имеют тип один-ко-многим и являются обязательными.

В тех ситуациях, когда все эти способы не пригодны, логическая целостность контролируется программно или вручную.

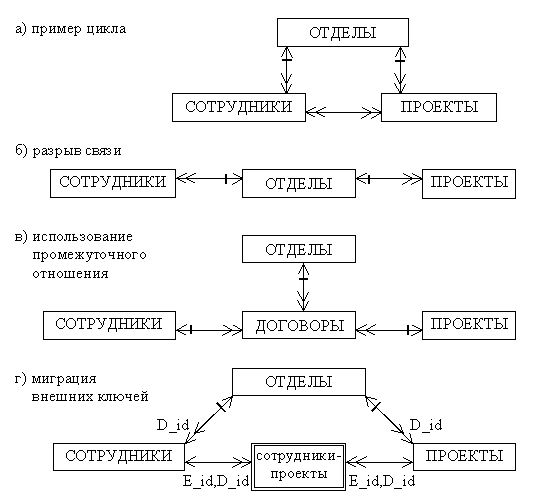


Рис.8. Некоторые способы разрешения циклов в схеме базы данных

**2.4.2. Составление реляционных отношений**

Каждое реляционное отношение соответствует одной сущности (объекту ПО) и в него вносятся все атрибуты сущности. Для каждого отношения необходимо определить первичный ключ и внешние ключи (если они есть). В том случае, если базовое отношение не имеет потенциальных ключей, вводится *суррогатный первичный ключ*, который не несёт смысловой нагрузки и служит только для идентификации записей.

**Примечание:** суррогатный первичный ключ также может вводиться в тех случаях, когда потенциальный ключ имеет большой размер (например, длинная символьная строка) или является составным (не менее трёх атрибутов).

Потенциальными ключами отношения *АВТОРЫ* являются атрибуты  
*Паспортные данные* и *ИНН*. Первый хранится как длинная строка, а последний по условиям предметной области не является обязательным. Поэтому для авторов необходимо ввести суррогатный ключ – *A\_id*. Книги можно идентифицировать по атрибуту *Контракт*: его номер обязателен и уникален. Потенциальные ключи отношения *СОТРУДНИКИ* – атрибуты *ИНН*, *Паспортные данные*, *Табельный номер*, причём все они обязательные. Табельный номер занимает меньше памяти, чем ИНН, поэтому он и будет первичным ключом. Кортежи отношения *ЗАКАЗЫ* можно идентифицировать ключом *Номер заказа*.

Потенциальными ключами вспомогательных отношений являются комбинации первичных ключей соответствующих базовых отношений.

Отношения приведены в табл. 1-7. Для каждого отношения указаны атрибуты с их внутренним названием, типом и длиной. Типы данных обозначаются так: N – числовой, C – символьный, D – дата (последний имеет стандартную длину, зависящую от СУБД, поэтому она не указывается).

Таблица 1. Схема отношения *СОТРУДНИКИ* (Employees)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Табельный номер | E\_ID | N(4) | первичный ключ |
| Фамилия, имя, отчество | E\_NAME | C(50) | обязательное поле |
| Дата рождения | E\_BORN | D |  |
| Пол | E\_SEX | C(1) | обязательное поле |
| Паспортные данные | E\_PASSP | C(50) | обязательное поле |
| ИНН | E\_INN | N(12) | обязательное уникальное поле |
| Должность | E\_POST | C(30) | обязательное поле |
| Оклад | E\_SALARY | N(8,2) | обязательное поле |
| Адрес | E\_ADDR | C(50) |  |
| Телефоны | E\_TEL | C(30) | многозначное поле |

Таблица 2. Схема отношения *КНИГИ* (Books)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Номер контракта | B\_CONTRACT | N(6) | первичный ключ |
| Дата подписания контракта | B\_DATE | D | обязательное поле |
| Менеджер | B\_MAN | N(4) | внешний ключ (к Employees) |
| Название книги | B\_TITLE | N(40) | обязательное поле |
| Цена | B\_PRICE | N(6,2) | цена экземпляра книги |
| Затраты | B\_ADVANCE | N(10,2) | общая сумма затрат на книгу |
| Авторский гонорар | B\_FEE | N(8,2) | общая сумма гонорара |
| Дата выхода | B\_PUBL | D |  |
| Тираж | B\_CIRCUL | N(5) |  |
| Ответственный редактор | B\_EDIT | N(4) | внешний ключ (к Employees) |

Таблица 3. Схема отношения *АВТОРЫ* (Authors)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Код автора | A\_ID | N(4) | суррогатный первичный ключ |
| Фамилия, имя, отчество | A\_NAME | C(50) | обязательное поле |
| Паспортные данные | A\_PASSP | C(50) | обязательное поле |
| ИНН | A\_INN | N(12) | уникальное поле |
| Адрес | A\_ADDR | C(50) | обязательное поле |
| Телефоны | A\_TEL | C(30) | многозначное поле |

Таблица 4. Схема отношения *ЗАКАЗЫ* (Orders)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Номер заказа | O\_ID | N(6) | первичный ключ |
| Заказчик | O\_COMPANY | С(40) | обязательное поле |
| Дата поступления заказа | O\_DATE | D | обязательное поле |
| Адрес заказчика | O\_ADDR | C(50) | обязательное поле |
| Дата выполнения заказа | O\_READY | D |  |

Таблица 5. Схема отношения *КНИГИ–АВТОРЫ* (Titles)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Код книги (№ контракта) | B\_ID | N(6) | внешний ключ (к Books) |
| Код автора | A\_ID | N(4) | внешний ключ (к Authors) |
| Номер в списке | A\_NO | N(1) | обязательное поле |
| Гонорар | A\_FEE | N(3) | процент от общего гонорара |

Таблица 6. Схема отношения *КНИГИ–РЕДАКТОРЫ* (Editors)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Код книги (№ контракта) | B\_ID | N(6) | внешний ключ (к Books) |
| Код редактора | E\_ID | N(4) | внешний ключ (к Employees) |

Таблица 7. Схема отношения *СТРОКИ ЗАКАЗА* (Items)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Номер заказа | O\_ID | N(6) | внешний ключ (к Orders) |
| Код книги (№ контракта) | B\_ID | N(6) | внешний ключ (к Books) |
| Количество | B\_COUNT | N(4) | обязательное поле |

**2.4.3. Нормализация полученных отношений (до 4НФ)**

**1НФ.**Для приведения таблиц к 1НФ требуется составить прямоугольные таблицы (один атрибут – один столбец) и разбить сложные атрибуты на простые, а многозначные атрибуты вынести в отдельные отношения.

**Примечание.** В реальных БД сложные атрибуты разбиваются на простые, если:

а) этого требует внешнее представление данных;

б) в запросах поиск может осуществляться по отдельной части атрибута.

Разделим атрибуты *Фамилия, имя, отчество* на два атрибута *Фамилия* и *Имя, отчество* и *Паспортные данные* на атрибуты *Номер паспорта* (уникальный), *Дата выдачи* и *Кем выдан*.

Многозначный атрибут *Телефоны* для сотрудников компании следует сначала разделить на два – *Домашние телефоны* и *Рабочие телефоны*. (Для авторов мы не будем различать домашние и рабочие телефоны). Затем нужно создать отдельные отношения с (нерабочими) телефонами для сотрудников (*ТЕЛЕФОНЫ СОТРУДНИКОВ*) и для авторов (*ТЕЛЕФОНЫ АВТОРОВ*).

Атрибут *Рабочие телефоны* отношения *СОТРУДНИКИ* имеет неоднородные значения. Один из номеров телефонов – основной – определяется рабочим местом сотрудника (рассматриваются только стационарные телефоны). Наличие других номеров зависит от того, есть ли в том же помещении (комнате) другие сотрудники, имеющие стационарные телефоны. Можно добавить в отношение *СОТРУДНИКИ* атрибут *Номер комнаты*, а в атрибуте   
*Рабочие телефоны* хранить номер того телефона, который стоит на рабочем месте сотрудника. Дополнительные номера телефонов можно будет вычислить из других кортежей с таким же номером комнаты. Но в случае увольнения сотрудника мы потеряем сведения о номере рабочего телефона.

Поэтому создадим новое отношение *КОМНАТЫ* и включим в него атрибуты *Номер комнаты* и *Телефон*. Так как в комнате может не быть телефона, первичный ключ нового отношения не определен (ПК не может содержать null–значения), но на этих атрибутах можно определить составной уникальный ключ. Связь между отношениями *СОТРУДНИКИ* и *КОМНАТЫ* реализуем через составной внешний ключ (*Номер комнаты*, *Телефон*). Значение внешнего ключа для каждого сотрудника будем брать из того кортежа, в котором хранится основной рабочий телефон этого сотрудника.

**2НФ.**В нашем случае составные первичные ключи имеют отношения *СТРОКИ ЗАКАЗА*, *КНИГИ–АВТОРЫ* и *КНИГИ–РЕДАКТОРЫ*. Неключевые атрибуты этих отношений функционально полно зависят от первичных ключей.

**3НФ.** В отношении *ЗАКАЗЫ* атрибут *Адрес заказчика* зависит от атрибута *Заказчик*, а не от первичного ключа, поэтому адрес следует вынести в отдельное отношение *ЗАКАЗЧИКИ*. Но при этом первичным ключом нового отношения станет атрибут *Заказчик*, т.е. длинная символьная строка. Целесообразнее перенести в новое отношение атрибуты *Заказчик* и *Адрес заказчика* и ввести для него суррогатный ПК. Так как каждый заказчик может сделать несколько заказов, связь между отношениями *ЗАКАЗЧИКИ* и *ЗАКАЗЫ* будет 1:n и суррогатный ПК станет внешним ключом для отношения *ЗАКАЗЫ* .

В отношении *СОТРУДНИКИ* атрибут *Оклад* зависит от атрибута *Должность*. Поступим с этой транзитивной зависимостью так же, как в предыдущем случае: создадим новое отношение *ДОЛЖНОСТИ*, перенесём в него атрибуты *Должность* и *Оклад* и введём суррогатный первичный ключ.

В отношениях *СОТРУДНИКИ* и *АВТОРЫ* атрибуты *Дата выдачи* и *Кем выдан* зависят от атрибута *Номер паспорта*, а не от первичного ключа. Но если мы выделим их в отдельное отношение, то получившиеся связи будут иметь тип 1:1. Следовательно, декомпозиция нецелесообразна.

**4НФ.** Отношения данного примера не нарушают 4НФ, т.к. не содержат нетривиальных многозначных зависимостей.

В реальных базах данных после нормализации может проводиться денормализация. Она проводится с одной целью – повышение производительности БД. Рассмотрим некоторые запросы к нашей базе данных.

Например, запрос на получение списка телефонов авторов или домашних телефонов сотрудников потребует в нормализованной БД соединения отношений. Пользователю безразлична форма представления этого списка: номера телефонов через запятую или в столбец. Поэтому мы откажемся от создания отдельных отношений с номерами телефонов, и вернёмся к варианту с многозначными полями. (Это не касается рабочих телефонов сотрудников).

Другой запрос: как определяется, можно ли выполнить очередной заказ? Для каждой позиции заказа нужно просуммировать количество книг по выполненным заказам, получить остаток (тираж минус полученная сумма) и сравнить остаток с объёмом заказа. Такой расчёт может потребовать много времени, поэтому предлагается добавить в отношение *КНИГИ* производный атрибут *Остаток тиража*. Значение этого атрибута должно автоматически пересчитываться при установлении даты выполнения заказа.

После проведённых преобразований схема БД выглядит так (рис. 9):

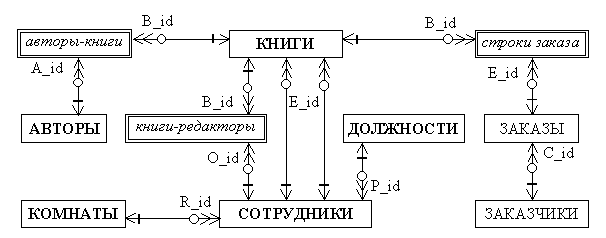


Рис.9. Окончательная схема РБД издательской компании

Окончательные схемы отношений базы данных с указанием ключей и других ограничений целостности приведены в табл. 8–17.

Таблица 8. Схема отношения *ДОЛЖНОСТИ* (Posts)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Код должности | P\_ID | N(3) | суррогатный первичный ключ |
| Название должности | P\_POST | C(30) | обязательное поле |
| Оклад | P\_SAL | N(8,2) | обязательное поле |

Таблица 9. Схема отношения *КОМНАТЫ* (Rooms)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Номер комнаты | R\_NO | N(3) | обязательное поле |
| Номер телефона | R\_TEL | C(10) |  |

Таблица 10. Схема отношения *СОТРУДНИКИ* (Employees)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Табельный номер | E\_ID | N(4) | первичный ключ |
| Фамилия | E\_FNAME | C(20) | обязательное поле |
| Имя, отчество | E\_LNAME | С(30) | обязательное поле |
| Дата рождения | E\_BORN | D |  |
| Пол | E\_SEX | C(1) | обязательное поле |
| Код должности | E\_POST | N(3) | внешний ключ (к Posts) |
| Номер комнаты | E\_ROOM | N(3) | составной внешний ключ  (к Rooms) |
| Номер телефона | E\_TEL | C(10) |
| ИНН | E\_INN | С(12) | обязательное поле |
| Номер паспорта | E\_PASSP | C(12) | обязательное поле |
| Кем выдан паспорт | E\_ORG | С(30) | обязательное поле |
| Дата выдачи паспорта | E\_PDATE | D | обязательное поле |
| Адрес | E\_ADDR | C(50) |  |

Таблица 11. Схема отношения *ЗАКАЗЧИКИ* (Customers)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Код заказчика | C\_ID | N(4) | суррогатный первичный ключ |
| Заказчик | C\_NAME | C(30) | обязательное поле |
| Адрес заказчика | C\_ADDR | C(50) | обязательное поле |

Таблица 12. Схема отношения *АВТОРЫ* (Authors)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Код автора | A\_ID | N(4) | суррогатный ключ |
| Фамилия | A\_FNAME | C(20) | обязательное поле |
| Имя, отчество | A\_LNAME | С(30) | обязательное поле |
| ИНН | A\_INN | С(12) |  |
| Номер паспорта | A\_PASSP | C(12) | обязательное поле |
| Кем выдан паспорт | A\_ORG | С(30) | обязательное поле |
| Дата выдачи паспорта | A\_PDATE | D | обязательное поле |
| Адрес | A\_ADDR | C(50) | обязательное поле |
| Телефоны | A\_TEL | C(30) | многозначное поле |

Таблица 13. Схема отношения *КНИГИ* (Books)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Номер контракта | B\_CONTRACT | N(6) | первичный ключ |
| Дата подписания контракта | B\_DATE | D | обязательное поле |
| Менеджер | B\_MAN | N(4) | внешний ключ (к Employees) |
| Название книги | B\_TITLE | N(40) | обязательное поле |
| Цена | B\_PRICE | N(6,2) | цена экземпляра книги |
| Затраты | B\_ADVANCE | N(10,2) | общая сумма затрат на книгу |
| Авторский гонорар | B\_FEE | N(8,2) | общая сумма гонорара |
| Дата выхода | B\_PUBL | D |  |
| Тираж | B\_CIRCUL | N(5) |  |
| Ответственный редактор | B\_EDIT | N(4) | внешний ключ (к Employees) |
| Остаток тиража | B\_REST | N(5) | производное поле |

Таблица 14. Схема отношения *ЗАКАЗЫ* (Orders)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Номер заказа | O\_ID | N(6) | первичный ключ |
| Код заказчика | O\_COMPANY | N(4) | внешний ключ (к Customers) |
| Дата поступления заказа | O\_DATE | D | обязательное поле |
| Дата выполнения заказа | O\_READY | D |  |

Таблица 15. Схема отношения *КНИГИ–АВТОРЫ* (Titles)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Код книги (№ контракта) | B\_ID | N(6) | внешний ключ (к Books) |
| Код автора | A\_ID | N(4) | внешний ключ (к Authors) |
| Номер в списке | A\_NO | N(1) | обязательное поле |
| Гонорар | A\_FEE | N(3) | процент от общего гонорара |

Таблица 16. Схема отношения *СТРОКИ ЗАКАЗА* (Items)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Номер заказа | O\_ID | N(6) | внешний ключ (к Orders) |
| Код книги (№ контракта) | B\_ID | N(6) | внешний ключ (к Books) |
| Количество | B\_COUNT | N(4) | обязательное поле |

Таблица 17. Схема отношения *КНИГИ–РЕДАКТОРЫ* (Editors)

|  |  |  |  |
| --- | --- | --- | --- |
| *Содержание поля* | *Имя поля* | *Тип, длина* | *Примечания* |
| Код книги (№ контракта) | B\_ID | N(6) | внешний ключ (к Books) |
| Код редактора | E\_ID | N(4) | внешний ключ (к Employees) |

**2.4.3. Определение дополнительных ограничений целостности**

Перечислим ограничения целостности, которые не указаны в табл. 8–17.

1. Значения всех числовых атрибутов – больше 0 (или null, если атрибут необязателен).
2. Область значений атрибута *Sex* отношения *EMPLOYEES* – символы 'м' и 'ж'.
3. Отношение *ROOMS* не имеет первичного ключа, но комбинация значений (*R\_no*, *Tel*) уникальна.
4. В отношении *TITLES* порядковые номера авторов на обложке одной книги должны идти подряд, начиная с 1.
5. В отношении *TITLES* сумма процентов гонорара по одной книге равна 100.

Ограничения (4,5) нельзя реализовать в схеме отношения. В реальных БД подобные ограничения целостности реализуются программно (через внешнее приложение или специальную процедуру контроля данных).

**2.4.4. Описание групп пользователей и прав доступа**

Опишем для каждой группы пользователей права доступа к каждой таблице и к каждому полю (атрибуту).

1. Администратор БД: имеет доступ ко всем данным (по записи), может изменять структуру базы данных и связи между отношениями. Устанавливает права доступа для всех остальных групп.
2. Представители администрации компании: имеют доступ по чтению ко всем данным и доступ по записи к отношениям *POSTS*, *ROOMS* и*EMPLOYEES*.
3. Менеджеры: имеет доступ по чтению ко всем данным, кроме отношения *POSTS*. Имеют доступ по записи к отношениям *AUTHORS*, *CUSTOMERS*,*BOOKS*, *EDITORS*, *TITLES*, *ORDERS*, *ITEMS*.
4. Редакторы: имеют доступ по чтению к следующим отношениям:
   * *AUTHORS*, кроме полей *A\_passp*, *A\_org*, *A\_pdate*, *A\_INN*(паспортные данные и ИНН).
   * *BOOKS*, кроме полей *B\_advance*, *B\_fee* (затраты и гонорар).

* + *EDITORS*.
  + *TITLES*.

1. Сотрудники, принимающие и выполняющие заказы: имеют доступ по записи к отношениям *CUSTOMERS*, *ORDERS*, *ITEMS* и по чтению к полям *B\_title*, *B\_circul*, *B\_price*и *B\_rest* отношения *BOOKS* (название, тираж, цена, непроданный остаток тиража).

**2.5. Физическое проектирование БД**

Мы условились не привязываться к конкретной СУБД и выполнять описание логической схемы БД на SQL-92.

Приведём фрагмент описания схемы БД на DDL:

1. Отношение *POSTS* (должности):

create table posts (

p\_id numeric(3) primary key,

p\_post varchar(30) not null,

p\_salary numeric(8,2) not null check(p\_salary > 0));

1. Отношение *ROOMS* (комнаты):

create table rooms (

r\_no numeric(3) primary key,

r\_tel varchar(10),

unique(r\_no, r\_tel));

1. Отношение *EMPLOYEES* (сотрудники):

create table employees (

e\_id numeric(4) primary key,

e\_fname varchar(20) not null,

e\_lname varchar(30) not null,

e\_born date,

e\_sex char(1) not null check(e\_sex in ('ж','м')),

e\_post numeric(3) references posts,

e\_room numeric(3),

e\_tel varchar(10),

e\_inn char(12) not null,

e\_passp char(12) not null,

e\_org varchar(30) not null,

e\_pdate date not null,

e\_addr varchar(50),

foreign key(e\_room,e\_tel) references rooms(r\_no,r\_tel));

Другие отношения описываются аналогично.

Права доступа пользователей, описанные в п. 2.4.4, предоставляются с помощью команды GRANT. Рассмотрим для примера права сотрудника компании ***user1***, который принимает и обслуживает заказы. Права доступа к отношениям *CUSTOMERS*, *ORDERS*, *ITEMS* могут быть описаны следующим образом:

grant insert, update on customers to user1;

grant insert, update, delete on orders to user1;

grant insert, update, delete on items to user1;

Для реализации частичного доступа к отношению *BOOKS* следует создать соответствующее представление и предоставить доступ к этому представлению:

create view goods (id, title, circul, price, rest)

as select b\_contract, b\_title, b\_circul, b\_price, b\_rest

from books;

grant select on goods to user1;

Приведём примеры нескольких готовых запросов:

1. Список всех текущих проектов (книг, находящихся в печати и в продаже):

create view edits as

select b\_contract, b\_title, b\_date

from books

where b\_rest is null or b\_rest > 0;

1. Список редакторов, работающих над книгами:

create view edits (title, fname, lname) as

select b\_title, e\_fname, e\_lname /\*ответственный редактор\*/

from books, employees e

where b\_edit=e\_id and

(b\_publ is null or b\_publ > sysdate);

union /\*sysdate – текущая дата\*/

select b\_title, a\_fname, a\_lname

from books, employees e, editors d

where b.b\_contract=d.b\_id and d.e\_id=a.e\_id and

(b\_publ is null or b\_publ > sysdate)

order by 1;

1. Определение общей прибыли от продаж по текущим проектам:

create view edits (title, total) as

select b\_title, (circul–rest)\*price–advance

from books

where b\_rest is null or b\_rest > 0;

Анализ готовых запросов показывает, что для повышения эффективности работы с данными необходимо создать индексы для всех внешних ключей (и всех первичных ключей, если выбранная СУБД не создаёт их автоматически). Приведём примеры создания индексов:

create index e\_posts on employees(e\_post);

create index b\_editors on books(b\_edit);

create unique index r\_tel on rooms(r\_no,r\_tel);