**§ 6  
Проектирование многотабличной базы данных**

Рассмотрим на конкретном примере методику проектирования многотабличной базы данных. Для этого снова вернемся к задаче моделирования работы с информацией, выполняемой приемной комиссией при поступлении абитуриентов в университет (см. § 3).  
**Табличная форма модели данных**  
В § 3 была построена модель данных, состоящая из трех взаимосвязанных таблиц. Воспроизведем ее еще раз.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **ФАКУЛЬТЕТЫ** | | **Название факультета** | | **Экзамен 1** | | **Экзамен 2** | | **Экзамен 3** | | |  | | --- | | **СПЕЦИАЛЬНОСТИ** | | **Название специальности** | | **Название факультета** | | **План приема** | |
| |  | | --- | | **АБИТУРИЕНТЫ** | | Регистрационный номер | | Фамилия | | Имя | | Отчество | | Дата рождения | | Город | | Законченное учебное заведение | | Название специальности | | Производственный стаж | | Медаль | | Оценка за экзамен 1 | | Оценка за экзамен 2 | | Оценка за экзамен 3 | | Зачисление | | |

Эти три таблицы можно рассматривать как модель данных в реляционной СУБД. Но работать с БД в таком виде неудобно. Помимо того что реляционная БД должна состоять из таблиц, к ней предъявляется еще ряд требований.  
Одним из главных требований является требование отсутствия избыточности (или минимизация избыточности) данных. Избыточность приводит к лишнему расходу памяти. Память нужно экономить. Это не только увеличивает информационную плотность базы данных, но и сокращает время поиска и обработки данных.  
Очевидный недостаток описанных таблиц — многократное повторение длинных значений полей в разных записях. Например, название специальности «Радиофизика и электроника» будет повторяться в 100 записях для 100 абитуриентов, которые на нее поступают. Проще сделать так. В таблице СПЕЦИАЛЬНОСТИ для каждой специальности ввести свой короткий код. Тогда полное название запишется в БД только один раз, а в анкетах абитуриентов будет указываться только код. Точно так же можно закодировать названия факультетов.  
Внесем изменения в таблицы ФАКУЛЬТЕТЫ и СПЕЦИАЛЬНОСТИ.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **ФАКУЛЬТЕТЫ** | | **КОД ФАКУЛЬТЕТА** | | **Название факультета** | | **Экзамен 1** | | **Экзамен 2** | | **Экзамен 3** | | |  | | --- | | **СПЕЦИАЛЬНОСТИ** | | **КОД СПЕЦИАЛЬНОСТИ** | | **Название специальности** | | **Название факультета** | | **План приема** | |

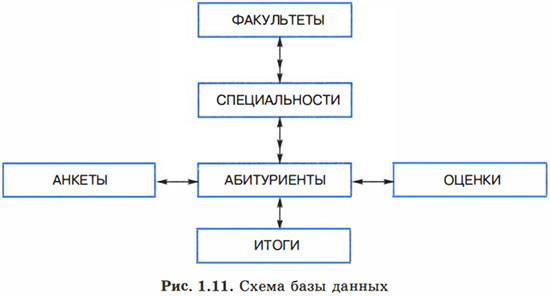
Здесь предполагаются два упрощающих допущения: пусть на разных специальностях одного факультета сдаются одни и те же экзамены, а число экзаменов на всех факультетах равно трем (это вполне разумно).  
Очень неудобной для работы является таблица АБИТУРИЕНТЫ. В ней слишком много полей. В частности, такую таблицу неудобно будет просматривать на экране, легко запутаться в полях.  
Поступим следующим образом. Разделим «большую» таблицу АБИТУРИЕНТЫ на четыре таблицы поменьше:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **АНКЕТЫ** | | Регистрационный номер | | Фамилия | | Имя | | Отчество | | Дата рождения | | Город | | Законченное учебное заведение | | |  | | --- | | **АБИТУРИЕНТЫ** | | Регистрационный номер | | Код специальности | | Медаль | | Производственный стаж | | |  | | --- | | **ОЦЕНКИ** | | Регистрационный номер | | Оценка за экзамен 1 | | Оценка за экзамен 2 | | Оценка за экзамен 3 | | |  | | --- | | **ИТОГИ** | | Регистрационный номер | | Зачисление | |

С такими таблицами работать гораздо проще. На разных этапах работы приемной комиссии каждая из этих таблиц будет иметь самостоятельное значение.  
Таблица АНКЕТЫ содержит анкетные данные, не влияющие на зачисление абитуриента в вуз. В таблице АБИТУРИЕНТЫ содержатся сведения, определяющие, куда поступает абитуриент, а также данные, которые могут повлиять на его зачисление (предположим, что это может быть производственный стаж и наличие медали). Таблица ОЦЕНКИ — это ведомость, которая будет заполняться для всех абитуриентов в процессе приема экзаменов. Таблица ИТОГИ будет содержать результаты зачисления всех абитуриентов.

**Отношения и связи**  
Каждая из спроектированных выше таблиц будет представлена в БД отдельным отношением. Опишем все их в строчной форме, дав в некоторых случаях полям сокращенные имена и подчеркнув главные ключи.  
**ФАКУЛЬТЕТЫ** (КОД ФКТ, ФАКУЛЬТЕТ, ЭКЗАМЕН\_1, ЭКАМЕН 2, ЭКЗАМЕН З)  
**СПЕЦИАЛЬНОСТИ** (КОД СПЕЦ, СПЕЦИАЛЬНОСТЬ, КОД\_ФКТ, ПЛАН)  
**АБИТУРИЕНТЫ** (РЕГНОМ, КОД\_СПЕЦ, МЕДАЛЬ, СТАЖ)  
**АНКЕТЫ** (РЕГ\_НОМ, ФАМИЛИЯ, ИМЯ, ОТЧЕСТВО, ДАТА РОЖД, ГОРОД, УЧ ЗАВЕДЕНИЕ)  
**ОЦЕНКИ** (РЕГНОМ, ОЦЕНКАХ, ОЦЕНКА\_2, ОЦЕНКА\_3)  
**ИТОГИ** (РЕГНОМ, ЗАЧИСЛЕНИЕ)  
*Чтобы эти шесть таблиц представляли собой систему, между ними должны быть установлены связи.*  
Фактически связи уже имеются через общие имена полей. Первые два отношения связаны между собой кодом факультета, второе и третье — кодом специальности, а четыре последних — регистрационным номером. Связи позволяют определить соответствия между любыми данными в этих таблицах, например между фамилией некоторого абитуриента и его оценкой по математике; между названием города и результатами экзамена по русскому языку выпускников школ этого города и пр. Благодаря этим связям возможно получение ответов на запросы, требующие поиска информации в нескольких таблицах одновременно.

**Схема базы данных**  
Для явного указания связей между таблицами должна быть построена схема базы данных. В схеме указывается наличие связей между таблицами и типы связей. Схема для нашей системы представлена на рис. 1.11.



В схеме использованы два типа связей: один к одному и один ко многим- Первый обозначен двунаправленной одинарной стрелкой, второй — одинарной стрелкой в одну сторону и двойной в другую. При связи «один к одному» с одной записью в таблице связана одна запись в другой таблице. Например, одна запись об абитуриенте связана с одним списком оценок. При наличии связи «один ко многим» одна запись в некоторой таблице связана с множеством записей в другой таблице. Например, с одним факультетом связано множество специальностей, а с одной специальностью — множество абитуриентов, поступающих на эту специальность.  
Связь «один ко многим» — это связь между двумя соседними уровнями иерархической структуры. А таблицы, связанные отношениями «один к одному», находятся на одном уровне иерархии. В принципе все они могут быть объединены в одну таблицу, поскольку главный ключ у них один — РЕГ\_НОМ. Но чем это неудобно, было объяснено выше.

**Что такое целостность данных**  
СУБД поддерживает организацию связей между таблицами БД, обеспечивающую одно важное свойство базы данных, которое называется целостностью данных.  
Система не допустит, чтобы одноименные поля в разных связанных между собой таблицах имели разные значения. Согласно этому принципу, будет автоматически контролироваться ввод данных. В связанных таблицах может быть установлен режим каскадной замены: если в одной из таблиц изменяется значение поля, по которому установлена связь, то в других таблицах одноименные поля автоматически изменят свои значения. Аналогично действует режим каскадного удаления: достаточно удалить запись из одной таблицы, чтобы связанные записи исчезли из всех остальных таблиц. Это естественно, поскольку, например, если закрывается какой-то факультет, то исчезают и все его специальности. Или если у абитуриента изменяют регистрационный номер в таблице АБИТУРИЕНТЫ, то автоматически номер должен обновиться и в других таблицах.  
На этом проектирование базы данных завершается. Это был теоретический этап. Практическая работа по созданию базы данных будет проходить в рамках компьютерного практикума.

**Система основных понятий**

|  |  |  |
| --- | --- | --- |
| **Проектирование многотабличной базы данных** | | |
| 1-й этап: анализ предметной области Результат: построение структуры данных — информационной модели предметной области | | |
| 2-й этап: построение модели данных для будущей БД | | |
| Реляционная модель данных (система таблиц) | | |
| Типы связей | Схема | Целостность |
| Один к одному, один ко многим | Граф, отражающий структуру данных и связей В БД | Свойство согласованности действий с повторяющимися данными (поддерживается СУБД) |

**Вопросы и задания**  
1. а) Перечислите задачи, которые должна решать проектируемая информационная система «Приемная комиссия».  
б) Какие информационные процессы происходят на различных этапах приемной кампании в вузе?  
в) Какая информация добавляется к базе данных на каждом этапе?  
2. а) В чем заключается построение модели данных?  
б) Что означает свойство целостности БД?  
в) Какие данные следует добавить в БД приемной комиссии, если требуется учитывать преподавателей, принимающих экзамены, и деление абитуриентов на экзаменационные группы?  
г) Какие данные следует добавить в БД приемной комиссии, если дополнительно к требованиям предыдущего задания нужно учитывать расписание экзаменов, т. е. сведения о том, где, когда и какому преподавателю сдает экзамен данная группа?  
д) Постройте схему БД с учетом выполнения заданий 2, в и 2, г.  
3. а) При проектировании БД были определены следующие отношения: МАГАЗИН(НОМЕР МАГ, ТИП, АДРЕС, ДИРЕКТОР, ТЕЛЕФОН) ОТДЕЛ (НАЗВАНИЕ ОТД, ЗАВЕДУЮЩИЙ, ТЕЛЕФОН) ПРОДАВЕЦ (ТАБЕЛЬНЫЙНОМ, ФАМИЛИЯ, ИМЯ, ОТЧЕСТВО, КАТЕГОРИЯ)  
Являются ли эти отношения связанными? Добавьте всё, что необходимо для их связи; изобразите схему БД в графическом виде.  
б) Спроектируйте базу данных для информационной системы «Наша школа», содержащей сведения об учителях, учениках, классах, изучаемых предметах. Замечание: данное задание носит творческий характер и может быть выполнено во многих вариантах. Устройте конкурс на лучшее решение этой задачи.