

**АНАТОЛИЙ ХОМОНЕНКО
ВЛАДИМИР ГОФМАН**

РАБОТА С БАЗАМИ ДАННЫХ В DELPHI

3-Е ИЗДАНИЕ



**ПРОЕКТИРОВАНИЕ
РЕЛЯЦИОННЫХ БАЗ
ДАННЫХ**

**ТЕХНОЛОГИИ BDE, ADO,
dbExpress И INTERBASE
EXPRESS**

**РАБОТА С ЛОКАЛЬНЫМИ
И УДАЛЕННЫМИ БАЗАМИ
ДАННЫХ**

**ПРОГРАММИРОВАНИЕ
НА SQL**

PRO

**ПРОФЕССИОНАЛЬНОЕ
ПРОГРАММИРОВАНИЕ**

УДК 681.3.06
ББК 32.973.26-18.2
X76

Хомоненко А. Д., Гофман В. Э.

X76 Работа с базами данных в Delphi. — 3-е изд., перераб. и доп. —
СПб.: БХВ-Петербург, 2005. — 640 с.: ил.

ISBN 5-94157-361-8

Рассматривается использование средств Delphi 7 для разработки приложений баз данных. Даются понятия баз данных, характеризуются элементы и описываются этапы проектирования реляционных баз данных, изложена технология разработки информационных систем, освещаются приемы работы с данными, создание таблиц и приложений баз данных, подготовка отчетов. Рассматриваются навигационный и реляционный способы доступа к данным с помощью технологий BDE, ADO, dbExpress и Interbase Express, основы программирования на SQL. Показывается использование локальных и удаленных баз данных, включая создание многоуровневых информационных систем и публикацию баз данных в Интернете. Также рассматриваются возможности Delphi 2005 по работе с базами данных. Благодаря подробному изложению тем и большому числу примеров книга может служить практическим руководством по работе с базами данных.

Для программистов

УДК 681.3.06
ББК 32.973.26-18.2

Группа подготовки издания:

Главный редактор	<i>Екатерина Кондукова</i>
Зам. главного редактора	<i>Игорь Шишигин</i>
Зав. редакцией	<i>Григорий Добин</i>
Компьютерная верстка	<i>Натальи Смирновой</i>
Корректор	<i>Наталья Першакова</i>
Дизайн серии	<i>Инны Тачиной</i>
Оформление обложки	<i>Игоря Цырульникова</i>
Зав. производством	<i>Николай Тверских</i>

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 01.02.05.

Формат 70×100^{1/16}. Печать офсетная. Усл. печ. л. 51,6.

Тираж 4000 экз. Заказ №

"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Санитарно-эпидемиологическое заключение на продукцию № 77.99.02.953.Д.006421.11.04
от 11.11.2004 г. выдано Федеральной службой по надзору
в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"
199034, Санкт-Петербург, 9 линия, 12

ISBN 5-94157-361-8

© Хомоненко А. Д., Гофман В. Э., 2005
© Оформление, издательство "БХВ-Петербург", 2005

Оглавление

Предисловие	1
Часть I. Основы работы с базами данных	5
Глава 1. Основные понятия баз данных	7
Банки данных	7
Модели данных.....	8
Базы данных и приложения.....	9
BDE.....	10
ADO	11
dbExpress.....	11
Варианты архитектуры для BDE	11
Глава 2. Реляционные базы данных и средства работы с ними	16
Элементы реляционной базы данных.....	16
Таблицы баз данных.....	16
Ключи и индексы.....	19
Методы и способы доступа к данным	21
Связь между таблицами	23
Механизм транзакций.....	27
Бизнес-правила.....	28
Словарь данных	29
Таблицы форматов dBase и Paradox.....	29
Средства для работы с реляционными базами данных	34
Инструменты.....	34
Компоненты.....	35
Исключения баз данных.....	40
Глава 3. Проектирование баз данных.....	43
Нормализация базы данных.....	43
Избыточность данных и аномалии.....	44
Приведение к нормальным формам.....	46
Первая нормальная форма	47
Вторая нормальная форма	48
Третья нормальная форма	49
Средства CASE	51

Глава 4. Технология создания информационной системы	55
Создание таблиц базы данных.....	55
Описание полей.....	58
Задание индексов	60
Задание ограничений на значения полей	62
Задание ссылочной целостности	65
Задание паролей	67
Задание языкового драйвера	69
Задание таблицы для выбора значений	70
Просмотр списка подчиненных таблиц.....	74
Изменение структуры таблицы.....	74
Создание приложения BDE	75
Использование модуля данных.....	77
Глава 5. Компоненты доступа к данным	81
Наборы данных	81
Состояния наборов данных.....	84
Режимы наборов данных	88
Доступ к полям.....	90
Особенности набора данных <i>Table</i>	92
Особенности набора данных <i>Query</i>	100
Объекты поля.....	105
Редактор полей	107
Операции с полями.....	115
Доступ к значению поля.....	116
Проверка типа и значения поля	119
Форматирование отображаемого значения поля	124
Источник данных	127
Часть II. ТЕХНОЛОГИИ ДОСТУПА К ДАННЫМ	131
Глава 6. Визуальные компоненты для работы с данными	133
Отображение и редактирование значения логического поля	135
Отображение и выбор значения поля.....	136
Отображение и выбор значения поля в списке.....	138
Простой и комбинированный списки	138
Списки, сформированные по значениям поля набора данных	139
Представление записей в табличном виде	139
Характеристики сетки.....	140
Столбцы сетки	143
Использование модифицированной сетки	149

Использование навигационного интерфейса.....	152
Вывод графических изображений	154
Построение диаграмм	158
Глава 7. Навигационный доступ к данным с помощью BDE	164
Операции с таблицей БД.....	165
Создание, удаление и переименование.....	165
Установка уровня доступа	168
Сортировка набора данных	169
Навигация по набору данных	171
Перемещение по записям.....	172
Переход по закладкам.....	181
Фильтрация записей	184
Фильтрация по выражению	185
Фильтрация по диапазону.....	192
Навигация с псевдофильтрацией.....	196
Поиск записей	197
Поиск в наборах данных	197
Поиск по индексным полям	204
Модификация набора данных	206
Редактирование записей	208
Добавление записей	214
Удаление записей	216
Пример формы приложения	217
Работа со связанными таблицами	224
Пример приложения	225
Использование механизма транзакций.....	234
Глава 8. Реляционный доступ к данным с помощью BDE.....	236
Основные сведения о языке SQL.....	237
Функции языка.....	239
Определение данных.....	239
Создание и удаление таблицы	240
Изменение состава полей таблицы	242
Создание и удаление индекса	243
Отбор данных из таблиц.....	244
Описание инструкции <i>SELECT</i>	244
Управление полями.....	245
Простое условие отбора записей.....	249
Сложные критерии отбора записей.....	252
Группирование записей	253
Сортировка записей	254
Соединение таблиц	257

Модификация записей	259
Редактирование записей	259
Вставка записей	260
Удаление записей	262
Статический и динамический запросы.....	262
Глава 9. Технология dbExpress.....	266
Общая характеристика.....	266
Установление соединения с сервером	267
Компоненты доступа к данным.....	271
Универсальный доступ к данным.....	272
Просмотр таблиц.....	277
Выполнение SQL-запроса.....	277
Выполнение хранимых процедур.....	278
Компонент редактирования набора данных.....	279
Отладка соединения с сервером	282
Глава 10. Технология ADO.....	284
Общая характеристика.....	284
Установление соединения	286
Управление соединением и транзакциями.....	289
Компоненты доступа к данным.....	291
Доступ к таблицам	293
Выполнение запросов.....	294
Вызов хранимых процедур.....	294
Компонент <i>ADODataset</i>	295
Команды ADO	295
Пример приложения.....	297
Глава 11. Создание и просмотр отчетов с помощью Rave Reports	301
Характеристика генератора отчетов	301
Визуальное конструирование отчетов.....	302
Интерфейс визуального конструктора.....	302
Состав проекта отчетов.....	303
Редактор событий.....	306
Компоненты, представленные на многостраничной панели инструментов	307
Компоненты отображения данных.....	309
Компоненты управления отчетом	312
Компонент-проект отчета	312
Компонент управления отчетом	313

Компоненты установления соединения.....	313
Схема управления отчетом и подсоединения данных.....	314
Примеры создания и просмотра отчетов.....	316
Предварительный просмотр отчета.....	317
Простой отчет приложения базы данных.....	319
Глава 12. Инструменты	322
Программа BDE Administrator.....	322
Работа с псевдонимами.....	323
Параметры драйвера.....	326
Системные установки.....	329
Использование конфигурационных файлов.....	330
Программа Database Desktop.....	331
Редактирование записей таблиц.....	332
Работа с псевдонимами.....	333
Работа с SQL-запросами.....	333
Визуальное конструирование запросов.....	335
Отбор записей из таблицы.....	336
Редактирование записей.....	338
Вставка и удаление записей.....	339
Связывание таблиц.....	341
Программа SQL Builder.....	342
Программа SQL Explorer.....	349
Программа Data Pump.....	351
Часть III. Удаленные базы данных	355
Глава 13. Введение в работу с удаленными базами данных	357
Основные понятия.....	357
Архитектура "клиент-сервер".....	358
Сервер и удаленная БД.....	359
Средства работы с удаленными БД.....	359
Сервер InterBase.....	361
Бизнес-правила.....	362
Организация данных.....	363
Запуск сервера.....	364
Особенности приложения.....	365
Соединение с базой данных.....	365
Соединение с базой из программы IBConsole.....	366
Компонент <i>Database</i>	366
Компонент <i>Session</i>	371
Соединение с базой данных из приложения.....	374

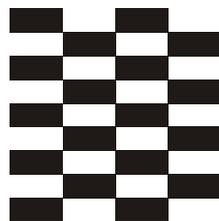
Глава 14. Работа с удаленными базами данных	377
Создание базы данных.....	377
Управление структурой таблиц.....	380
Описание столбца.....	383
Ограничения столбца.....	383
Описание ключей.....	387
Определение ограничений ссылочной целостности	388
Использование индексов.....	390
Использование доменов	391
Использование просмотров	392
Хранимые процедуры	393
Создание и удаление хранимых процедур.....	394
Язык хранимых процедур.....	395
Виды хранимых процедур	401
Вызов хранимой процедуры выбора	401
Вызов хранимой процедуры действия	403
Использование триггеров.....	405
Создание и изменение триггера	406
Примеры использования триггера.....	407
Создание генераторов.....	409
Использование функций, определяемых пользователем.....	410
Реализация механизма транзакций	412
Использование механизма кэшированных изменений.....	414
Использование компонентов <i>Database</i> , <i>Table</i> и <i>Query</i>	415
Использование компонента <i>UpdateSQL</i>	419
Использование механизма событий сервера.....	422
Управление привилегиями	425
Манипулирование данными	427
Глава 15. Технология InterBase Express	430
Общая характеристика.....	430
Установление соединения с сервером	431
Управление транзакциями	432
Компоненты доступа к данным.....	434
Генераторы для автоинкрементных полей	435
Доступ к таблицам	436
Выполнение запросов.....	436
Получение и редактирование данных.....	437
Компонент <i>IBSQL</i>	441
Пример приложения.....	442
Глава 16. Инструменты для работы с удаленными базами данных	446
Программа <i>IBConsole</i>	446
Управление сервером.....	447

Подключение к серверу	447
Регистрация сервера	449
Просмотр протокола работы сервера	450
Операции с сертификатами	450
Управление пользователями	451
Управление БД	451
Регистрация базы данных	452
Подключение базы данных	452
Создание базы данных	453
Просмотр метаданных	453
Сбор мусора	454
Проверка состояния базы данных	454
Анализ статистики	455
Сохранение и восстановление базы данных	456
Интерактивное выполнение SQL-запросов	460
Программа SQL Monitor	463
Глава 17. Трехуровневые приложения	466
Принципы построения трехуровневых приложений	466
Сервер приложений	468
Приложение клиента	473
Часть IV. Базы данных и Интернет	483
Глава 18. Основные элементы интернет-технологий	485
Сценарии JavaScript, JScript и VBScript	486
Элементы управления ActiveX	487
Апплеты и сервлеты Java	488
Интерфейсы CGI и WinCGI	488
Интерфейсы ISAPI/NSAPI	489
Страницы ASP, PHP и IDC/HTX	489
Формирование Web-страниц	490
Интерфейсы OLE DB, ADO, ODBC	491
Статическая публикация БД	492
Динамическая публикация БД	492
Web-приложения	493
Протоколы передачи данных	494
Универсальный указатель ресурсов (URL)	495
Язык разметки гипертекста HTML	495
Расширяемый язык разметки XML	496
Программа XML Mapper	504

Глава 19. Web-приложения и интерфейсы	508
Характеристика Web-приложений	508
Web-приложения в сетях интранет	509
Web-приложения с модулями расширения серверной части	511
Web-приложения с модулями расширения клиентской части	513
Архитектура Web-приложений, публикующих БД	514
Двухуровневые Web-приложения	515
Трехуровневые Web-приложения	516
Интерфейсы Web-приложений	517
Общий интерфейс взаимодействия CGI	517
Переменные окружения	520
Стандартный вывод	522
Интерфейс программирования серверных приложений ISAPI	522
Глава 20. Публикация баз данных средствами Delphi	526
Компоненты, используемые при разработке Web-приложений	526
Статическая публикация	528
Компоненты генерации HTML-страниц	533
Компонент <i>PageProducer</i>	533
Компонент <i>Data.SetPageProducer</i>	534
Компонент <i>Data.SetTableProducer</i>	534
Компонент <i>QueryTableProducer</i>	537
Пример генератора HTML-страниц	537
Динамическая публикация	545
Создание модуля CGI	545
Создание ISAPI-модуля расширения сервера	554
Обработка пользовательского ввода в модуле ISAPI	560
Публикация графики	562
Использование технологии ADO	569
Глава 21. Работа с электронной почтой	572
Использование функции <i>ShellExecute</i>	572
Использование интерфейса MAPI	573
Глава 22. Характеристика Web-служб	579
Основные понятия	579
Документ WSDL	580
Вызываемый интерфейс	584
Страница <i>Web.Services</i> Палитры компонентов	584
Схема взаимодействия клиента и сервера	588
Разработка клиента для Web-службы	589
Импортирование документа WSDL	589
Обращение к вызываемому интерфейсу	591
Пример создания клиента Web-службы	594

Часть V. Delphi 2005.....	597
Глава 23. Работа с БД в Delphi 2005	599
Общая характеристика Delphi 2005.....	599
Характеристика VCL для .NET.....	601
Разработка приложения VCL Forms	602
Характеристика способов	602
Приложения БД VCL Forms для .NET	603
Приложения БД VCL Forms для Win32.....	604
Технология ADO.NET	604
Общая характеристика.....	604
Схема доступа к данным	607
ADO.NET в Delphi 2005.....	607
Провайдеры данных для .NET.....	608
Приложение Windows Forms с ADO.NET и BDP.NET.....	611
Приложение.	
Фрагменты иерархии классов VCL.....	616
Предметный указатель	619

Глава 2



Реляционные базы данных и средства работы с ними

Большинство современных баз данных для персональных компьютеров являются реляционными. Достоинства реляционной модели организации данных — простота, гибкость структуры, удобство реализации на компьютере, наличие теоретического описания.

Элементы реляционной базы данных

Реляционная база данных (БД) состоит из взаимосвязанных таблиц. Каждая таблица содержит информацию об объектах одного типа, а совокупность всех таблиц образует единую БД.

Таблицы баз данных

Таблицы, образующие БД, находятся в каталоге (папке) на жестком диске. Таблицы хранятся в файлах и похожи на отдельные документы или электронные таблицы (например, табличного процессора Microsoft Excel), их можно перемещать и копировать обычным способом, скажем, с помощью Проводника Windows. Однако в отличие от документов, таблицы БД поддерживают *многопользовательский* режим доступа, это означает, что их могут одновременно использовать несколько приложений.

Для одной таблицы создается несколько файлов, содержащих данные, индексы, ключи и т. п. Главным из них является файл с данными, имя этого файла совпадает с именем таблицы, которое задается при ее создании. В некотором смысле понятия таблицы и ее главного файла являются синонимами, при выборе таблицы выбирается именно ее главный файл: для таблицы dBase это файл с расширением dbf, а для таблицы Paradox — файл с расширением db. Имена остальных файлов таблицы назначаются автоматически — все файлы имеют одинаковые имена, совпадающие с именами таблиц, и разные расширения, указывающие на содержимое соответствующего файла. Расширения файлов приведены ниже в данной главе *в разд. "Таблицы форматов dBase и Paradox"*.

Каждая таблица БД состоит из строк и столбцов и предназначена для хранения данных об однотипных объектах информационной системы. Строка таблицы называется *записью*, столбец таблицы — *полем* (рис. 2.1). Каждое поле должно иметь уникальное в пределах таблицы имя.

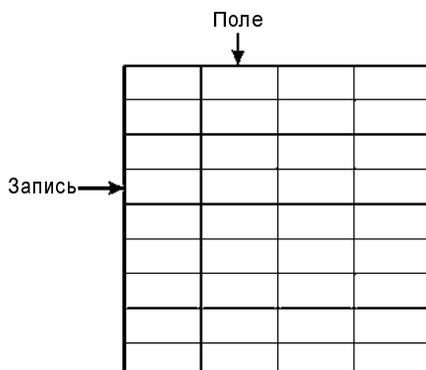


Рис. 2.1. Схема таблицы базы данных

Поле содержит данные одного из допустимых типов, например, строкового, целочисленного или типа "дата". При вводе значения в поле таблицы автоматически производится проверка соответствия типа значения и типа поля. В случае, когда эти типы не совпадают, а преобразование типа значения невозможно, генерируется исключение.

Особенности организации таблиц зависят от конкретной СУБД, используемой для создания и ведения БД. Например, в локальной таблице dBase и в таблице сервера InterBase нет поля автоинкрементного типа (с автоматически нарастаемым значением), а в таблице dBase нельзя определить ключ. Подобные особенности необходимо учитывать при выборе типа (формата) таблицы, т. к. они влияют не только на организацию БД, но и на построение приложения для работы с этой БД. Однако, несмотря на все различия таблиц, существуют общие правила создания и ведения БД, а также разработки приложений, которые и будут далее рассмотрены.

Замечание

В зависимости от типа таблиц и системы разработки приложений может использоваться различная терминология. Например, в InterBase поле таблицы называется столбцом.

Основу таблицы составляет описание ее полей, каждая таблица должна иметь хотя бы одно поле. Понятие структуры таблицы является более широким и включает:

- описание полей;
- ключ;

- индексы;
- ограничения на значения полей;
- ограничения ссылочной целостности между таблицами;
- пароли.

Иногда ограничения на значения полей, ограничения ссылочной целостности между таблицами, а также права доступа называют одним общим термином "ограничения".

Отметим, что отдельные элементы структуры зависят от формата таблиц, например, для таблиц dBase нельзя задать ограничения ссылочной целостности (т. к. у них нет ключей). Все элементы структуры задаются на физическом уровне (уровне таблицы) и действуют для всех программ, выполняющих операции с БД, включая средства разработки и ведения БД (например, программу Database Desktop). Многие из этих элементов (например, ограничения на значения полей или поля просмотра) можно также реализовать в приложении программно, однако в этом случае они действуют только в пределах своего приложения.

С таблицей в целом можно выполнять следующие операции:

- создание (определение структуры);
- изменение структуры (реструктуризация);
- переименование;
- удаление.

При *создании* таблицы задаются структура и имя таблицы. При сохранении на диске создаются все необходимые файлы, относящиеся к таблице. Их имена совпадают с именем таблицы.

При *изменении структуры* таблицы в ней могут измениться имена и характеристики полей, состав и наименования ключа и индексов, ограничения. Однако имена таблицы и ее файлов остаются прежними.

При *переименовании* таблица получает новое имя, в результате чего новое имя также получают все ее файлы. Для этого используются соответствующие программы (утилиты), предназначенные для работы с таблицами БД, например, Database Desktop или Data Pump.

Замечание

Таблицу нельзя переименовать, просто изменив названия всех ее файлов, например, с помощью Проводника Windows.

При *удалении* таблицы с диска удаляются все ее файлы. В отличие от переименования, удаление таблицы можно выполнить посредством любой программы (в том числе и с помощью Проводника Windows).

Ключи и индексы

Ключ представляет собой комбинацию полей, данные в которых однозначно определяют каждую запись в таблице. Простой ключ состоит из одного поля, а составной (сложный) — из нескольких полей. Поля, по которым построен ключ, называют *ключевыми*. В таблице может быть определен только один ключ. Ключ обеспечивает:

- однозначную идентификацию записей таблицы;
- ускорение выполнения запросов к БД;
- установление связи между отдельными таблицами БД;
- использование ограничений ссылочной целостности.

Ключ также называют *первичным ключом* или *первичным (главным) индексом*.

Информация о ключе может храниться в отдельном файле или совместно с данными таблицы. Например, в БД Paradox для этой цели используется отдельный файл (ключевой файл или файл главного индекса) с расширением px. В БД Access вся информация содержится в одном общем файле с расширением mdb. Значения ключа располагаются в определенном порядке. Для каждого значения ключа имеется уникальная ссылка, указывающая на расположение соответствующей записи в таблице (в главном ее файле). Поэтому при поиске записи выполняется не последовательный просмотр всей таблицы, а прямой доступ к записи на основании упорядоченных значений ключа.

Ценой, которую разработчик и пользователь платят за использование такой технологии, является увеличение размера БД вследствие необходимости хранения значений ключа, например, в отдельном файле. Размер этого файла зависит не только от числа записей таблицы (что достаточно очевидно), но и от полей, составляющих ключ. В ключевом файле, кроме ссылок на соответствующие записи таблицы, сохраняются и значения самих ключевых полей. Поэтому при вхождении в состав ключа длинных строковых полей размер ключевого файла может оказаться соизмеримым с размером файла с данными таблицы.

Таблицы различных форматов имеют свои особенности построения ключей. Вместе с тем существуют и общие правила.

- Ключ должен быть уникальным. У составного ключа значения отдельных полей (но не всех одновременно) могут повторяться.
- Ключ должен быть достаточным и не избыточным, т. е. не содержать поля, которые можно удалить без нарушения уникальности ключа.
- В состав ключа не могут входить поля некоторых типов, например, графическое поле или поле комментария.

Выбор ключевых полей не всегда является простой и очевидной задачей, особенно для таблиц с большим количеством полей. Нежелательно выбирать в качестве ключевых поля, содержащие фамилии людей в таблице сотрудников организации или названия товаров в таблице данных склада. В этом случае высока

вероятность существования двух и более однофамильцев, а также товаров с одинаковыми названиями, которые различаются, к примеру, цветом (значение другого поля). Для указанных таблиц можно использовать, например, поле кода сотрудника и поле артикула товара. При этом предполагается, что указанные значения являются уникальными.

Удобным вариантом создания ключа будет использование для него поля соответствующего типа, которое автоматически обеспечивает поддержку уникальности значений. Для таблиц Paradox таким является поле автоинкрементного типа, еще одним достоинством которого является небольшой размер (4 байта). В то же время в таблицах dBase и InterBase поле подобного типа отсутствует, и программист должен обеспечивать уникальность значений ключа самостоятельно, например, используя специальные генераторы.

Отметим, что при создании и ведении БД правильным подходом считается задание в каждой таблице ключа даже в случае, если на первый взгляд он не нужен. В примерах таблиц, которые приводятся при изложении материала, как правило, ключ создается, и для него вводится специальное автоинкрементное поле с именем Code или Number.

Индекс, как и ключ, строится по полям таблицы, однако он может допускать повторение значений составляющих его полей — в этом и состоит его основное отличие от ключа. Поля, по которым построен индекс, называют *индексными*. Простой индекс состоит из одного поля, а составной (сложный) — из нескольких полей.

Индексы при их создании именовуются. Как и в случае с ключом, в зависимости от СУБД индексы могут храниться в отдельных файлах или совместно с данными. Создание индекса называют *индексированием таблицы*.

Использование индекса обеспечивает:

- увеличение скорости доступа к данным (поиска);
- сортировку записей;
- установление связи между отдельными таблицами БД;
- использование ограничений ссылочной целостности.

В двух последних случаях индекс применяется совместно с ключом второй таблицы.

Как и ключ, индекс представляет собой своеобразное оглавление таблицы, просмотр которого выполняется перед обращением к ее записям. Таким образом, использование индекса повышает *скорость доступа* к данным в таблице за счет того, что доступ выполняется не последовательным, а индексно-последовательным методом.

Сортировка представляет собой упорядочивание записей по полю или группе полей в порядке возрастания или убывания их значений. Можно сказать, что индекс служит для сортировки таблиц по индексным полям. В частности, в Delphi записи набора Table можно сортировать только по индексным полям.

Набор данных *Query* позволяет выполнить средствами SQL сортировку по любым полям, однако и в этом случае для индексированных полей упорядочивание записей выполняется быстрее.

Для одной таблицы можно создать несколько индексов. В каждый момент времени один из них можно сделать текущим, т. е. активным. Даже при существовании нескольких индексов таблица может не иметь текущего индекса (текущий индекс важен, например, при выполнении поиска и сортировки записей набора данных *Table*).

Ключевые поля обычно автоматически индексируются. В таблицах *Paradox* ключ также является главным (первичным) индексом, который не именуется. Для таблиц *dBase* ключ не создается, и его роль выполняет один из индексов.

Замечание

Создание ключа может привести к побочным эффектам. Так, если в таблице *Paradox* определить ключ, то записи автоматически упорядочиваются по его значениям, что в ряде случаев является нежелательным.

Таким образом, использование ключей и индексов позволяет:

- однозначно идентифицировать записи;
- избегать дублирования значений в ключевых полях;
- выполнять сортировку таблиц;
- ускорять операции поиска в таблицах;
- устанавливать связи между отдельными таблицами БД;
- использовать ограничения ссылочной целостности.

Одной из основных задач БД является обеспечение *быстрого доступа к данным* (поиска данных). Время доступа к данным в значительной степени зависит от используемых для поиска данных методов и способов.

Методы и способы доступа к данным

Выделяют следующие методы доступа к данным таблиц:

- последовательный;
- прямой;
- индексно-последовательный.

При *последовательном* методе выполняется последовательный просмотр всех записей таблицы и поиск нужных из них. Этот метод доступа является крайне неэффективным и приводит к значительным временным затратам на поиск, прямо пропорциональным размеру таблицы (числу ее записей). Поэтому его рекомендуется использовать только для относительно небольших таблиц.

При *прямом* доступе нужная запись выбирается в таблице на основании ключа или индекса. При этом просмотр других записей не выполняется. Напомним, что значения ключей и индексов располагаются в упорядоченном виде и содержат ссылку, указывающую на расположение соответствующей записи в таблице. При поиске записи выполняется не последовательный просмотр всей таблицы, а непосредственный доступ к записи на основании ссылки.

Индексно-последовательный метод доступа включает в себя элементы последовательного и прямого методов доступа и используется при поиске группы записей. Этот метод реализуется только при наличии индекса, построенного по полям, значения которых должны быть найдены. Суть его заключается в том, что находится индекс первой записи, удовлетворяющей заданным условиям, и соответствующая запись выбирается из таблицы на основании ссылки. Это является прямым доступом к данным. После обработки первой найденной записи осуществляется переход к следующему значению индекса, и в таблице выбирается запись, соответствующая значению этого индекса. Так последовательно перебираются индексы всех записей, удовлетворяющих заданным условиям, что является последовательным доступом.

Достоинством прямого и индексно-последовательного методов является максимальная возможная скорость доступа к данным, плата за которую — расход памяти на хранение информации о ключах и индексах.

Указанные методы доступа реализуются СУБД и не требуют специального программирования. Задачей разработчика является определение соответствующей структуры БД, в данном случае — определение ключей и индексов. Так, если для поля создан индекс, то при поиске записей по этому полю автоматически используется индексно-последовательный метод доступа, в противном случае — последовательный метод.

Замечание

При создании составного индекса важен порядок составляющих его полей. Например, если индекс создан по полям "фамилия", "номер отдела" и "дата рождения", а поиск ведется одновременно по полям "фамилия", "дата рождения" и "номер отдела", то такой индекс использован не будет. В результате доступ к таблице осуществляется последовательным методом. В подобной ситуации (для таблицы с большим числом записей) разработчик должен создать также индекс, построенный по полям "фамилия", "дата рождения" и "номер отдела".

При выполнении операций с таблицами используется один из следующих *способов доступа к данным*:

- навигационный;
- реляционный.

Навигационный способ доступа заключается в обработке каждой отдельной записи таблицы. Этот способ обычно используется в локальных БД или в удаленных БД небольшого размера. Если необходимо обработать несколько записей, то все они обрабатываются поочередно.

Реляционный способ доступа основан на обработке сразу *группы записей*, при этом если необходимо обработать одну запись, то обрабатывается группа, состоящая из одной записи. Так как реляционный способ доступа основывается на SQL-запросах, его также называют *SQL-ориентированным*. Этот способ доступа ориентирован на выполнение операций с удаленными БД и является предпочтительным при работе с ними, хотя его можно использовать и для локальных БД.

Способ доступа к данным выбирается программистом и зависит от средств доступа к БД, используемых при разработке приложения. Например, в приложениях, создаваемых в Delphi, реализацию навигационного способа доступа можно осуществить посредством компонентов Table или Query, а реляционного — с помощью компонента Query.

Таким образом, методы доступа к данным определяются структурой БД, а способы доступа — приложением.

Связь между таблицами

В частном случае БД может состоять из одной таблицы, содержащей, например, дни рождения сотрудников организации. Однако обычно реляционная БД состоит из набора взаимосвязанных таблиц. Организация связи (отношений) между таблицами называется *связыванием* или *соединением таблиц*.

Связи между таблицами можно устанавливать как при создании БД, так и при выполнении приложения, используя средства, предоставляемые СУБД. Связывать можно две или несколько таблиц. В реляционной БД, кроме связанных таблиц, могут быть и отдельные таблицы, не соединенные ни с одной другой таблицей. Это не меняет сути реляционной БД, которая содержит единую информацию об информационной системе, связанную не в буквальном смысле (связь между таблицами), а в функциональном смысле — вся информация относится к одной системе.

Для связывания таблиц используются *поля связи* (иногда применяется термин "совпадающие поля"). Поля связи обязательно должны быть индексированными. В подчиненной таблице для связи с главной таблицей задается индекс, который также называется *внешним ключом*. Состав полей этого индекса должен полностью или частично совпадать с составом полей индекса главной таблицы.

Особенности использования индексов зависят от формата связываемых таблиц. Так, для таблиц dBase индексы строятся по одному полю и нет деления на ключ (главный или первичный индекс) и индексы. Для организации связи в главной и подчиненной таблицах выбираются индексы, составленные по полям совпадающего типа, например, целочисленного.

Для таблиц Paradox в качестве полей связи главной таблицы должны использоваться поля ключа, а для подчиненной таблицы — поля индекса. Кроме того, в подчиненной таблице обязательно должен быть определен ключ. На рис. 2.2 показана схема связи между таблицами БД Paradox.

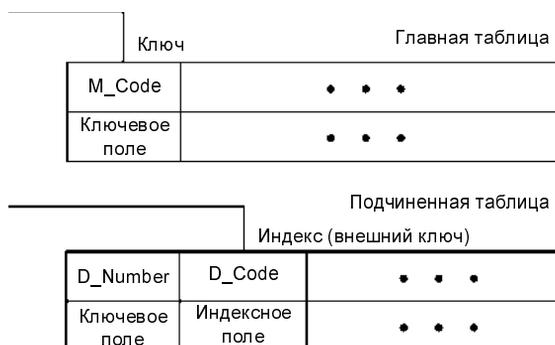


Рис. 2.2. Схема связи между таблицами базы данных Paradox

В главной таблице определен ключ, построенный по полю `M_Code` автоинкрементного типа. В подчиненной таблице определен ключ по полю `D_Number` также автоинкрементного типа и индекс, построенный по полю `D_Code` целочисленного типа. Связь между таблицами устанавливается по полям `D_Code` и `M_Code`. Индекс по полю `D_Code` является внешним ключом. В названиях полей включены префиксы, указывающие на принадлежность поля соответствующей таблице. Так, названия полей главной таблицы начинаются с буквы `M` (Master), а названия полей подчиненной таблицы начинаются с буквы `D` (Detail). Подобное именование полей облегчает ориентацию в их названиях, особенно при большом количестве таблиц.

Замечание

Как отмечалось, поля связи должны быть индексированными, хотя, строго говоря, это требование не всегда является обязательным. При доступе к данным средствами языка SQL можно связать (соединить) между собой таблицы и по неиндексированным полям. Однако в этом случае скорость выполнения операций будет низкой.

Связь между таблицами определяет отношение подчиненности, при котором одна таблица является *главной* (родительской, или мастером — Master), а вторая — *подчиненной* (дочерней, или детальной — Detail). Саму связь (отношение) называют связь "главный-подчиненный", "родительский-дочерний" или "мастер-детальный". Существуют следующие виды связи:

- отношение "один-к-одному";
- отношение "один-ко-многим";
- отношение "много-к-одному";
- отношение "много-ко-многим".

Отношение "*один-к-одному*" означает, что одной записи в главной таблице соответствует одна запись в подчиненной таблице. При этом возможны два варианта:

- для каждой записи главной таблицы есть запись в подчиненной таблице;
- в подчиненной таблице может не быть записей, соответствующих записям главной таблицы.