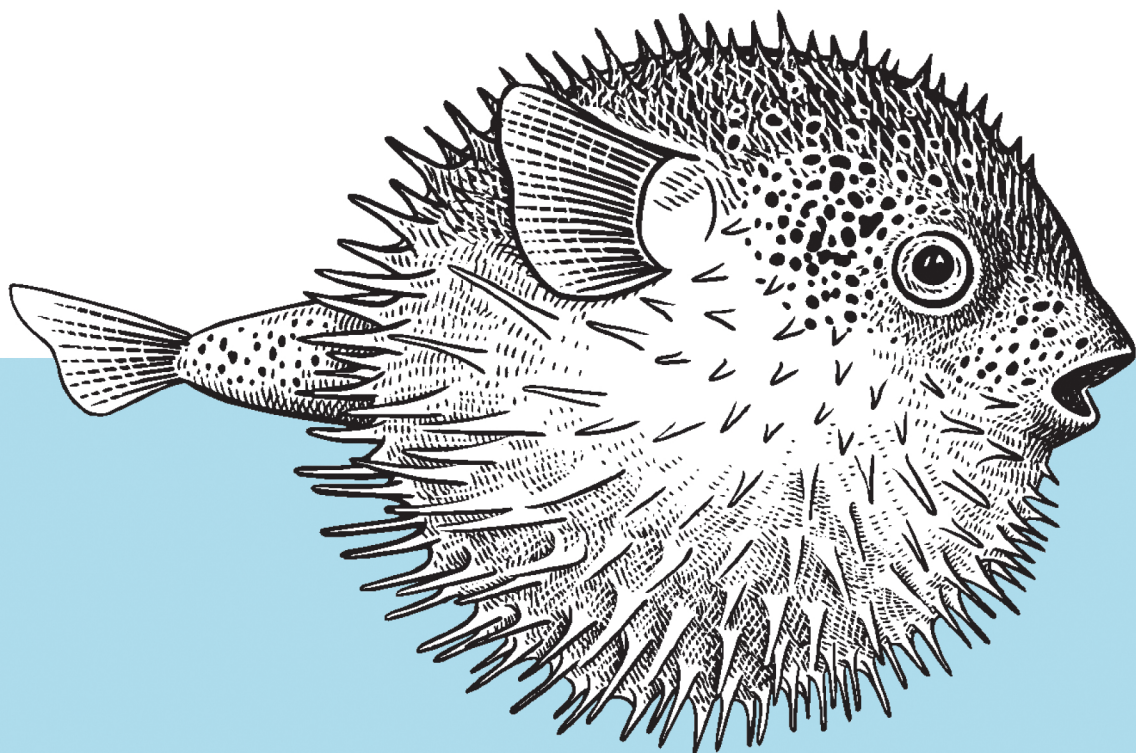




БИБЛИОТЕКА  
ПРОГРАММИСТА

Уолтер Шилдс

# SQL



БЫСТРОЕ ПОГРУЖЕНИЕ



ББК 32.973.233-018.2  
УДК 004.65  
Ш57

## Шилдс Уолтер

Ш57 SQL: быстрое погружение. — СПб.: Питер, 2022. — 224 с.: ил. — (Серия «Библиотека программиста»).

ISBN 978-5-4461-1835-9

Что общего между самыми востребованными профессиями и стремительным увеличением количества информации в мире? Ответ: язык структурированных запросов (SQL). SQL — рабочая лошадка среди языков программирования, основа основ для современного анализа и управления данными.

Книга «SQL: быстрое погружение» идеальна для всех, кто ищет новые перспективы карьерного роста; для разработчиков, которые хотят расширить свои навыки и знания в программировании; для любого человека, даже без опыта, кто хочет воспользоваться возможностями будущего, в котором будут править данные.

**16+** (В соответствии с Федеральным законом от 29 декабря 2010 г. № 436-ФЗ.)

ББК 32.973.233-018.2  
УДК 004.65

Права на издание получены по соглашению с ClydeBank Media LLC при содействии RussoRights LLC. Все права защищены. Никакая часть данной книги не может быть воспроизведена в какой бы то ни было форме без письменного разрешения владельцев авторских прав.

Информация, содержащаяся в данной книге, получена из источников, рассматриваемых издательством как надежные. Тем не менее, имея в виду возможные человеческие или технические ошибки, издательство не может гарантировать абсолютную точность и полноту приводимых сведений и не несет ответственности за возможные ошибки, связанные с использованием книги. Издательство не несет ответственности за доступность материалов, ссылки на которые вы можете найти в этой книге. На момент подготовки книги к изданию все ссылки на интернет-ресурсы были действующими.

ISBN 978-1945051753 англ.

© Translated and published with permission from ClydeBank Media LLC.  
This translated work is based on SQL QuickStart Guide: The Simplified  
Beginner's Guide to Managing, Analyzing, and Manipulating Data with SQL  
by Walter Shields.

© 2019 by ClydeBank Media LLC. All rights reserved.

ClydeBank Media LLC is not responsible for the quality of this translated work

ISBN 978-5-4461-1835-9

© Перевод на русский язык ООО «Прогресс книга», 2022

© Издание на русском языке, оформление ООО «Прогресс книга», 2022

© Серия «Библиотека программиста», 2022

# Оглавление

<b>Введение .....</b>	<b>11</b>
Почему я написал эту книгу.....	13
Поддержка для новичка.....	14
Охват и цель книги .....	14
SQL и ваша карьера.....	15
Как организована книга .....	16

## **ЧАСТЬ I. СОЗДАНИЕ СРЕДЫ ОБУЧЕНИЯ SQL**

<b>Глава 1. Структура базы данных.....</b>	<b>20</b>
Основная терминология.....	20
Основные элементы реляционных баз данных .....	24
Типы данных .....	32
Системы управления реляционными базами данных .....	36
Оператор SELECT .....	37
Запросы, операторы, условия и ключевые слова .....	38
Введение в SQLite.....	39
Резюме .....	40
<b>Глава 2. Инструменты и стратегии SQL.....</b>	<b>41</b>
База данных sTunes.....	41
Браузер базы данных для SQLite.....	42
Установка браузера базы данных для SQLite .....	42
Как проверить свои знания в SQL.....	43
Стратегии успеха .....	43
Резюме .....	44
<b>Глава 3. Работа с базой данных в SQLite .....</b>	<b>45</b>
Программное окружение .....	45
Открытие базы данных sTunes.....	46
Структура базы данных .....	47
Просмотр индивидуальных записей .....	49
Вкладка Execute SQL .....	50
Контрольные вопросы.....	53
Резюме .....	53

## ЧАСТЬ II ОПЕРАТОРЫ SQL

<b>Глава 4. Работа с запросами .....</b>	<b>56</b>
Добавление комментариев к запросам .....	56
Общая структура запроса.....	58
Пишем свой первый запрос .....	58
Синтаксис и соглашение о кодировании .....	61
Использование псевдонима.....	62
Условие ORDER BY.....	64
Получение ограниченного числа записей с помощью условия LIMIT .....	67
Контрольные вопросы.....	69
Резюме .....	69
<b>Глава 5. Преобразование данных в информацию .....</b>	<b>70</b>
Операторы сравнения, логические и арифметические операторы.....	71
Фильтрация данных (WHERE).....	73
Фильтрация строк.....	78
Использование оператора LIKE для поиска подстановочных знаков .....	80
Фильтрация записей по дате .....	84
Функция DATE() .....	85
Использование операторов AND и OR с двумя отдельными полями .....	86
Оператор OR .....	87
Использование круглых скобок с операторами AND и OR для указания порядка операций.....	88
Оператор CASE.....	91
Контрольные вопросы.....	96
Резюме .....	96
<b>Глава 6. Работа с несколькими таблицами .....</b>	<b>97</b>
Что такое соединение.....	97
Соединения и структура реляционной базы данных.....	101
Псевдонимы соединяемых таблиц.....	102
Типы соединений.....	105
Внутренние соединения для случаев соединения двух и более таблиц.....	113
Использование левых внешних соединений с операторами NULL, IS и NOT.....	116
Преобразование правого соединения в левое.....	119
Контрольные вопросы.....	122
Резюме .....	122
<b>Глава 7. Функции языка SQL .....</b>	<b>123</b>
Добавление вычислений к запросам.....	124
Типы функций в SQL .....	124
Управление текстовыми данными с помощью строковых функций .....	127

Конкатенация строк .....	128
Обрезка строки.....	130
Дополнительные строковые функции .....	134
Функции даты и времени .....	135
Агрегатные функции .....	140
Вложенные функции на примере ROUND() .....	141
Использование агрегатных функций и условия GROUP BY.....	142
Использование условий WHERE и HAVING со сгруппированными запросами.....	145
Условия WHERE и HAVING .....	147
Группировка по нескольким столбцам .....	148
Несколько заключительных слов о функциях .....	149
Контрольные вопросы.....	150
Резюме .....	150

### **ЧАСТЬ III**

## **РАСШИРЕННЫЕ ВОЗМОЖНОСТИ ЯЗЫКА SQL**

<b>Глава 8. Подзапросы.....</b>	<b>152</b>
Использование агрегатных функций в подзапросах .....	153
Использование подзапроса в операторе SELECT .....	155
Использование подзапроса с условием WHERE .....	156
Подзапросы без агрегатных функций.....	157
Возврат нескольких значений из подзапроса.....	158
Подзапросы и условие DISTINCT .....	160
Контрольные вопросы.....	163
Резюме .....	163
<b>Глава 9. Представления .....</b>	<b>164</b>
Работа с представлениями .....	164
Использование представлений.....	166
Изменения представлений.....	167
Соединенные представления.....	168
Удаление представлений с помощью оператора DROP .....	171
Контрольные вопросы.....	172
Резюме .....	172
<b>Глава 10. DML – язык управления данными .....</b>	<b>173</b>
Чем различаются анализ данных и управление базами данных.....	173
Добавление данных в БД.....	175
Обновление данных и ключевое слово SET.....	177
Удаление данных .....	178
Контрольные вопросы.....	180
Резюме .....	180

<b>Заключение .....</b>	<b>181</b>
Главное – задавать правильные вопросы .....	181
Как найти свой путь .....	181
Выбор специальности для работы с базами данных.....	182
Все ли дело в деньгах? .....	182
SQL – это универсальный язык .....	183
Смена карьеры.....	184
Как продавать свои навыки .....	184
Визуализация данных.....	185
Советы для успешного собеседования.....	185
Сертификация по SQL.....	186
Напутственные слова .....	186
<b>Приложение I. Контрольные вопросы и ответы на них.....</b>	<b>188</b>
Глава 3. Контрольные вопросы.....	188
Глава 4. Контрольные вопросы.....	190
Глава 5. Контрольные вопросы.....	193
Глава 6. Контрольные вопросы.....	195
Глава 7. Контрольные вопросы.....	197
Глава 8. Контрольные вопросы.....	199
Глава 9. Контрольные вопросы.....	202
Глава 10. Контрольные вопросы.....	205
<b>Приложение II. Список ключевых слов SQL по главам .....</b>	<b>207</b>
Глава 4. Ключевые слова.....	207
Глава 5. Ключевые слова.....	208
Глава 6. Ключевые слова.....	210
Глава 7. Ключевые слова.....	212
Глава 8. Ключевые слова.....	213
Глава 9. Ключевые слова.....	214
Глава 10. Ключевые слова.....	214
<b>Об авторе .....</b>	<b>215</b>
<b>Глоссарий .....</b>	<b>216</b>
<b>Библиография.....</b>	<b>222</b>

# ЧАСТЬ I

СОЗДАНИЕ СРЕДЫ ОБУЧЕНИЯ SQL

# глава 1

## Структура базы данных

### Краткое содержание

- ✓ Понимание использования языков баз данных
- ✓ Как работает реляционная база данных
- ✓ Типы данных
- ✓ Системы управления реляционными базами данных (СУБД)
- ✓ SQLite

При обучении новой технической дисциплине начинать стоит с базового словаря. Мы постарались выдержать баланс: изложить основные термины и концепции, избегая жаргона или сложных правил. В этой главе мы познакомим вас с концепцией реляционной базы данных и продемонстрируем типы данных, с которыми вы будете иметь дело в обычной базе данных. Мы также познакомим вас с основным оператором SQL — `SELECT`.

### Основная терминология

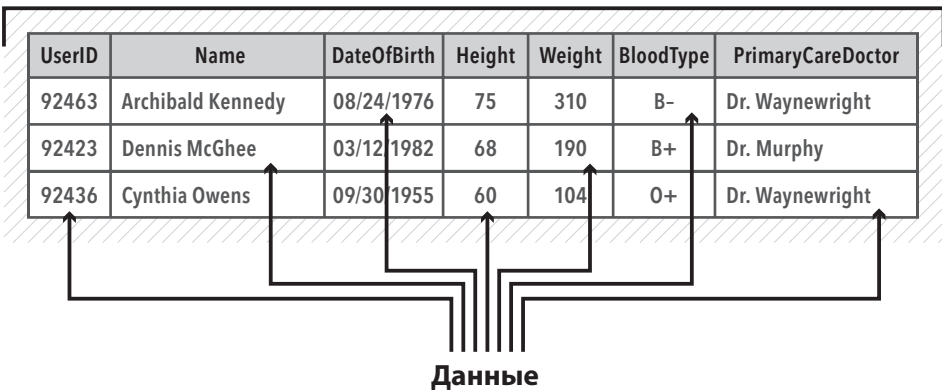
Данные — это часть информации [7]. Данные находятся повсюду и содержатся везде, но на практике термин «данные» обычно относится к информации уже записанной или той, которую можно записать. *Таблица* — один из самых простых инструментов, используемых для записи и визуализации данных. Таблица — это двухмерная сетка, состоящая из строк и столбцов.

#### ПРИМЕЧАНИЕ

При работе с базами данных таблица также может называться «базовой относительной переменной», хотя в этой книге мы будем придерживаться термина «таблица». Сводная терминология представлена на рис. 5.



Таблица



The diagram illustrates a table with patient data. Arrows point from the word 'Данные' (Data) to various cells in the table, indicating that the content within these cells represents data. The cells pointed to include 'UserID', 'Name', 'DateOfBirth', 'Height', 'Weight', 'BloodType', and 'PrimaryCareDoctor' across different rows.

UserID	Name	DateOfBirth	Height	Weight	BloodType	PrimaryCareDoctor
92463	Archibald Kennedy	08/24/1976	75	310	B-	Dr. Waynewright
92423	Dennis McGhee	03/12/1982	68	190	B+	Dr. Murphy
92436	Cynthia Owens	09/30/1955	60	104	O+	Dr. Waynewright

Рис. 2

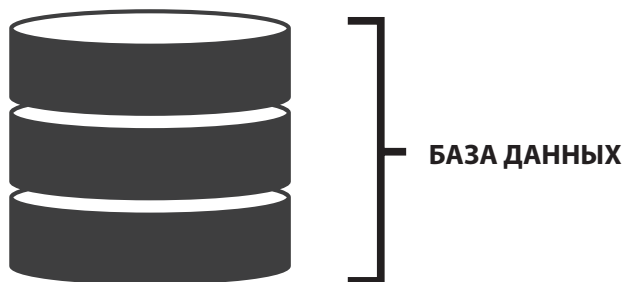
На рис. 2 представлена таблица, содержащая различные типы данных. К данным могут относиться имена, числа, даты, символы (например, «+» или «-») или любые другие форматы. Данные — это просто информация. Следовательно, при обработке данных нам следует соответствующим образом ограничить их. На рис. 2 показана таблица, в которой хранится основная информация о пациентах. Данные о пациентах заданы в различных форматах — числа, имена и даты, а в поле **BloodType** (группа крови) представлена строка из двух символов (буква и символ «+» или «-»). Форматы, используемые для визуализации данных, не случайны. Все базы данных содержат *метаданные*, то есть данные, описывающие структуру и форматирование самих данных, часто называемые «данными о данных». Например, поле **DateOfBirth** (дата рождения) может содержать метаданные, которые преобразовывают информацию к формату мм/дд/гггг. Метаданные, содержащиеся в поле **Height** (рост), могут ограничивать длину данных двумя знаками, если рост измеряется в дюймах.

Термин «база данных» можно определить как совокупность данных, упорядоченных для упрощения и скорости поиска и извлечения с помощью компьютера. База данных, как правило, изображается графически в виде цилиндров, расположенных один над другим (рис. 3), что символизирует несколько жестких дисков, используемых при создании центра хранения данных большой емкости.

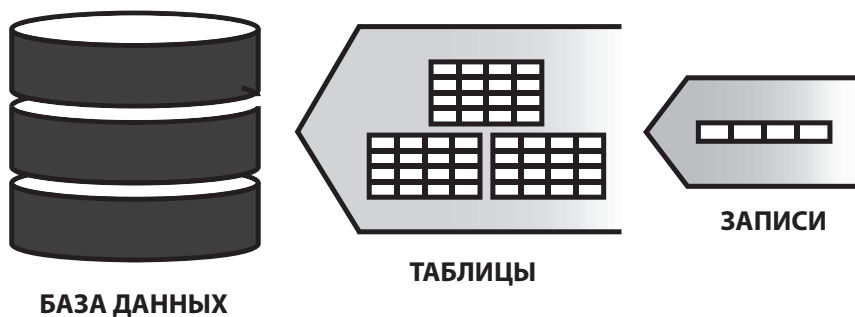
Как правило, информация внутри базы данных хранится в виде набора таблиц. Каждая таблица содержит определенные наборы данных, которые могут быть взаимосвязаны с другими данными из других таблиц и ссылаться на них.

**ПРИМЕР**

Таблица данных пациентов (рис. 2) – это просто таблица, а не база данных. Однако эту таблицу можно было бы включить в базу данных вместе с другими таблицами, такими как информация о лабораторных тестах, выписанных рецептах, историях посещений, персонале больницы, о врачах, их специализации и времени приема.

**Рис. 3**

Роль базы данных — упростить взаимодействие, организацию и анализ связанных данных из различных источников. Данные сохраняются во взаимосвязанных таблицах, что дает возможность манипулировать ими более гибко.



База данных состоит из таблиц. Таблицы состоят из записей.

**Рис. 4**

*Строки* в таблице называются *записями*. Также их можно называть *кортежами*. Столбцы в таблице, как правило, называются *полями*. Также их можно назвать *атрибутами*. Поля/атрибуты — это категории, используемые для определения данных в записи (строке).

ПРИМЕЧАНИЕ

В этой книге для описания строк в таблице мы будем использовать термин «записи», а для описания столбцов – «поля». См. рис. 5 – основная терминология.

ОСНОВНАЯ ТЕРМИНОЛОГИЯ

Терминология, используемая в этой книге	Другое название
Запись, строка	Кортеж
Поле, столбец	Атрибут
Таблица	Связи, базовая относительная переменная

Рис. 5

Каждая запись разбита на несколько полей, представляющих собой отдельные элементы данных, которые описывают конкретный элемент. Например, в таблице на рис. 6 хранятся сведения о пациентах определенной больницы, медицинского учреждения или страхового фонда. В данном случае база данных, скорее всего, будет состоять из нескольких таблиц. Понимание того, как таблицы связаны друг с другом, — это и есть ключ к пониманию основной архитектуры базы данных.

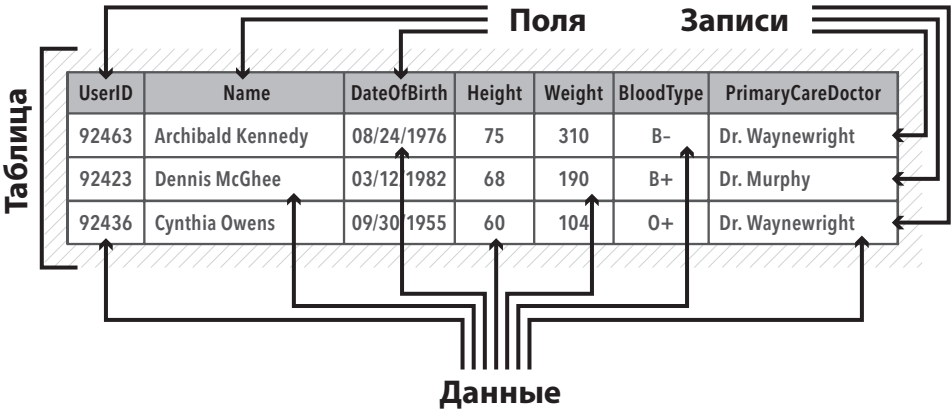


Рис. 6

## Основные элементы реляционных баз данных

*Реляционная база данных* — это конструкция базы данных, которая в 1969 году была официально утверждена ученым из IBM Эдгаром Франком Коддом. В следующем году Кодд опубликовал статью под названием «Реляционная модель данных для больших, совместно используемых банков данных» [8]. Девять лет спустя несколько крупных игроков в сфере технологий, в том числе IBM и Relational Software Inc. (позже ставшая Oracle), начали использовать реляционные базы данных в коммерческих целях. Четыре десятилетия спустя реляционная модель становится наиболее распространенной формой проектирования баз данных.

Чтобы получить элементарное представление о работе реляционных баз данных, важно понимать роль ключевых полей.

Реляционная база данных будет содержать множество таблиц, аналогичных таблице `patient_info` (рис. 7). Таблицы связаны друг с другом с помощью ключевых полей. Как вы видите на рисунке, таблица `patient_info` содержит поля, являющиеся первичным ключом и внешним ключом. Каждая таблица в реляционной базе данных должна содержать первичный ключ. Первичный ключ — это уникальный идентификатор записи в таблице. *Первичный ключ* каждой записи должен быть уникальным и не должен быть нулевым (пустым). Обратите внимание на поле `PatientID` в таблице `patient_info`. Поскольку это поле используется как первичный ключ, у каждой записи в таблице значение данного поля должно быть уникальным. Иными словами, никакие две записи не могут иметь один и тот же `PatientID`.

Хотя первичный ключ (в данном случае `PatientID`) должен быть уникальным, другие поля могут содержать данные, повторяющиеся более чем в одной записи. Рассмотрим поле `PrimaryCareDoctorID`. Например, если Dr. Waynewright, ID 106547 (см. первую строку рис. 7), лечит нескольких пациентов из базы данных, то его имя и идентификатор могут встречаться в нескольких записях таблицы.

### ПРИМЕЧАНИЕ

В реляционной базе данных таблицы часто называют «связями», так как они содержат набор записей (строк), связанных с различными полями (столбцами). Однако в этой книге мы будем использовать термин «таблица». См. рис. 5 — «Основная терминология».

*Внешний ключ* — это поле в таблице, значение которого соответствует первичному ключу в другой таблице. Предположим, что в дополнение к нашей

таблице `patient_info` в базе данных существует еще одна таблица с именем `primary_care_doctors`, в которой в качестве первичного ключа используется поле `PrimaryCareDoctorID`. В таблице `primary_care_doctors` строка Dr. Waynewright с ID 106547 появится только в одной записи. Именно совпадение различных ключевых полей в разных таблицах обеспечивает исключительно важную взаимосвязь в реляционной базе данных. Как правило, эти связи отображаются в виде *схемы* базы данных, также известной как *диаграмма «сущность — связь»* (ERD, *Entity Relationship Diagram*), которая служит своего рода эскизом для базы данных.

**ПЕРВИЧНЫЙ КЛЮЧ**

**ВНЕШНИЙ КЛЮЧ**

PatientID	PatientName	PrimaryCareDoctorID	PrimaryCareDoctorName	DateOfBirth	Height	Weight	BloodType
92463	Archibald Kennedy	106547	Dr. Waynewright	8/24/1976	75	310	B–
92425	Dennis McGhee	106474	Dr. Murphy	3/12/1982	68	190	B+
92443	Cynthis Owens	106547	Dr. Waynewright	9/30/1955	60	104	O+
92478	William Hampton	106437	Dr. Salazar	6/5/1973	73	175	AB–
92392	Hilda Bass	106783	Dr. Dean	6/10/1997	68	152	B+
92436	Frankie Stone	106437	Dr. Salazar	5/28/1979	68	106	O+
92403	Verna Sullivan	106984	Dr. Conner	7/17/2010	66	125	O+
92398	Merle Doyle	106439	Dr. Frank	1/8/1962	65	143	B–
92442	Ruth Swanson	106954	Dr. Hines	2/15/1970	61	160	O–
92384	Johnathan Singleton	106474	Dr. Murphy	6/2/1970	61	232	AB+
92405	WM Patrick	106439	Dr. Frank	6/11/1955	69	196	O+
92376	Mona Norris	106984	Dr. Conner	10/15/1932	60	98	B+
92399	Rick Gordon	106366	Dr. Hart	1/25/2002	68	149	B+
92408	Don Rivera	106437	Dr. Salazar	7/26/1954	72	185	A–
92389	Sheri Griffin	106211	Dr. Harvey	12/16/1987	78	132	AB–
92466	Guillermo Lawrence	106954	Dr. Hines	2/8/1978	60	219	O+
92310	Felipe Parker	106474	Dr. Murphy	12/10/1998	61	165	O–
92413	Brandi Carlson	106399	Dr. Flowers	11/20/2000	66	112	B+
92398	Floyd Casey	106783	Dr. Dean	12/14/1986	61	203	A–
92439	Patrick Walton	106366	Dr. Hart	8/11/1973	76	189	O+
92421	Vicki Klein	106954	Dr. Hines	11/28/1980	65	98	O+
92381	Cathy Harrison	106474	Dr. Murphy	11/16/1946	78	203	AB–
92393	Ann Guerrero	106783	Dr. Dean	6/25/1974	61	142	B–
92437	Gustavo Bates	106399	Dr. Flowers	2/25/2001	78	165	A–

Рис. 7

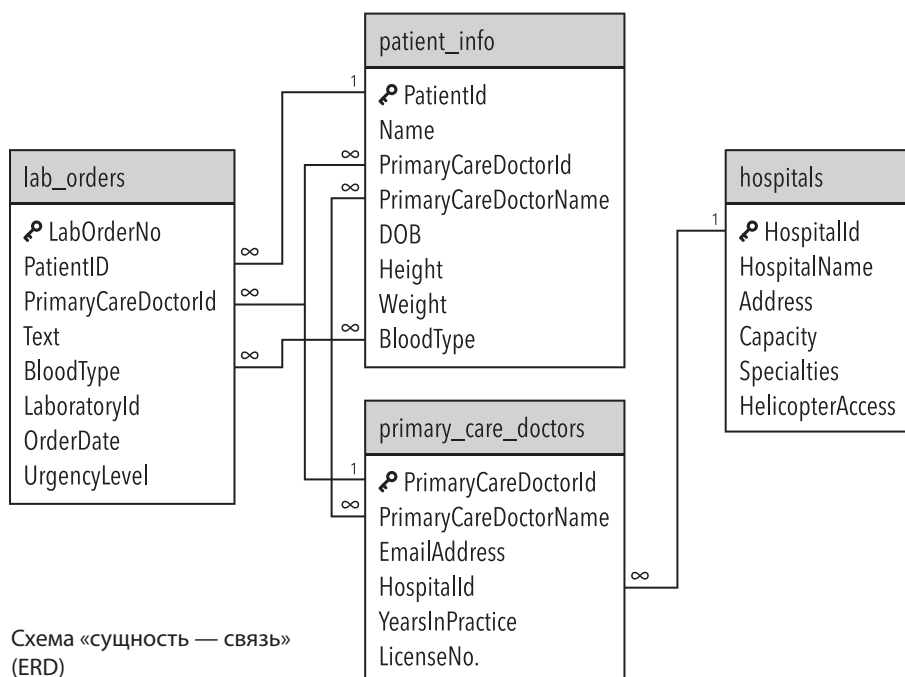


Рис. 8

Пока не беспокойтесь, что означают символы 1 и ∞. Очень скоро мы к ним вернемся. А сейчас рассмотрим схему и ее взаимосвязи. В данной схеме всего четыре таблицы, и таблицы связаны друг с другом посредством одного или нескольких общих полей. Поле **PatientID** — это первичный ключ для таблицы **patient\_info** и одновременно внешний ключ для таблицы **lab\_orders**. Точно так же поле **HospitalId** — первичный ключ для таблицы **hospitals**, но это внешний ключ для таблицы **primary\_care\_doctors**. Очень просто, правда? Давайте рассмотрим другую схему.

Схема на рис. 9 описывает базу данных для обработки заказов и доставки товаров клиентам. И здесь пора поговорить о символах 1 и ∞, которые находятся на схеме на концах соединительных линий. Эти символы описывают связи между таблицами. Когда на одном конце соединительной линии находится символ 1, а на другом — символ ∞, это означает связь между полями таблиц «один-ко-многим».

Рассмотрим подробнее таблицу **products** (рис. 10). В ней представлены данные о различных товарах и их атрибутах.

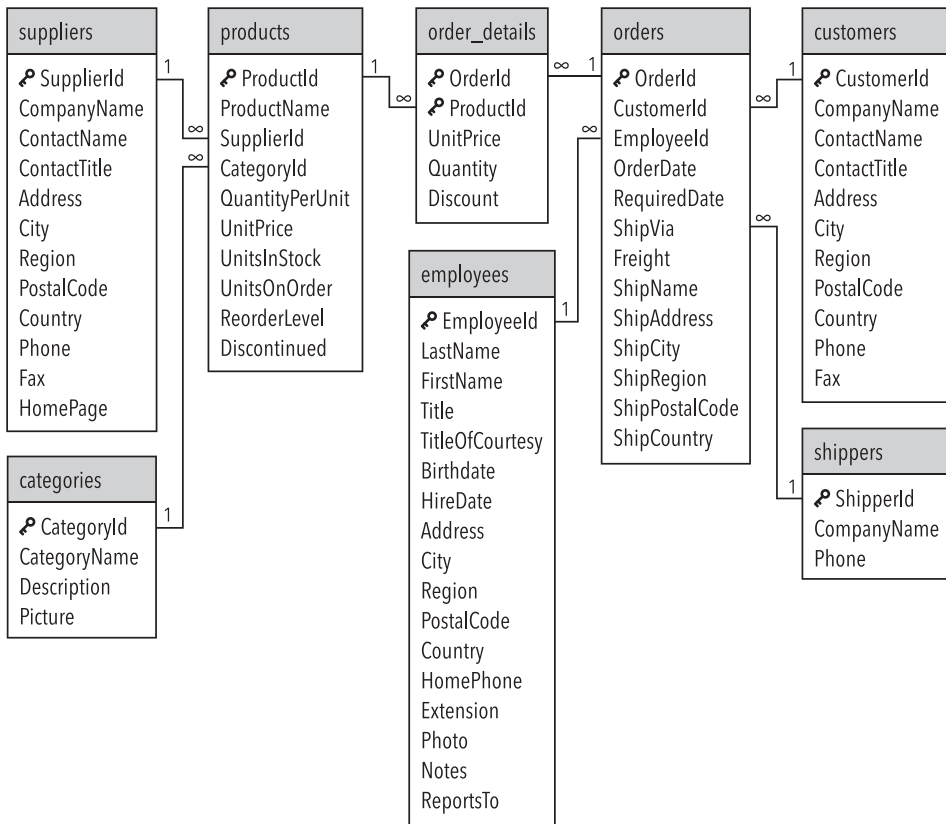


Рис. 9

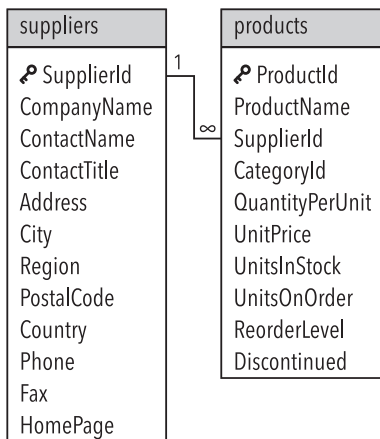


Рис. 10

Поле **ProductId** — первичный ключ для этой таблицы (обозначено значком ключа). Каждая запись в таблице будет содержать уникальный идентификационный номер товара. Фактически основная цель этой таблицы — каталогизировать атрибуты различных товаров базы данных. Теперь давайте проанализируем связи между таблицами **products** и **suppliers**.



## SUPPLIERS

SupplierId	CompanyName	ContactName	ContactTitle	Address	City	etc.
S-101	Van Eck Industries	Bruce Davidson	Vp Operations	2158 Del Dew Drive	Temple Hills	...
S-102	Wright & Gate Co	Wilma Joy	Supply Chain Supervisor	291 Creekside Lane	Ventura	...
S-103	Olivias Supply	Brad Pence	Site Manager, Baton Rouge	4353 Locust View Drive	Baton Rouge	...
S-104	Cantor Corporation	Orville Bedford	President	2811 West Drive	Chicago	...
S-105	Bellagio Finland	Wallace Grim	Distributions Supervisor	4939 Breezewood Court	Chanute	...
S-106	Decks Materials	John Tuck	VP Operations	4529 Counts Lane	Lexington	...
S-107	Lennox Co	Rachel Durst	Site Manager, Jackson	2216 Rhapsody Street	Gainesville	...

Рис. 11



## PRODUCTS

ProductId	ProductName	SupplierId	CategoryId	QuantityPerUnit	UnitPrice	etc.
P001	Welding goggles	S102	SA-432	1	\$12.99	...
P002	Welding helmet	S102	SA-432	1	\$41.49	...
P003	Stick electrodes	S104	WE-214	40	\$7.00	...
P004	Magnetic clamp	S101	WE-220	1	\$11.86	...
P005	Heat resistant blanket	S104	WE-212	1	\$3.73	...
P006	Work table	S105	GE-100	1	\$1,386.67	...
P007	Replacement plates	S105	GE-100	1	\$396.00	...
P008	Welding wire	S104	WE-214	1	\$112.86	...
P009	Welding coveralls	S102	SA-435	1	\$60.27	...
P010	Welding nozzle	S103	WE-214	1	\$141.65	...
P011	Gas regulator	S106	AU-100	1	\$166.25	...
P012	Welding hoods	S102	SA-432	1	\$42.37	...
P013	Spot welding electrode	S104	WE-212	1	\$2.35	...
P014	Plasma cutter	S107	PL-100	1	\$1,645.91	...
P015	Plasma cutter cutting tip	S107	PL-100	1	\$9.27	...

Рис. 12



Между таблицами `suppliers` и `products` существует связь «один-ко-многим», основанная на данных поля `SupplierId`. В таблице `suppliers` каждая запись будет иметь уникальный идентификационный номер для каждого поставщика. В таблице `products` может быть несколько записей с одним и тем же идентификационным номером поставщика.

Значок ключа, расположенный рядом с полем `SupplierId` в таблице `suppliers`, означает, что `SupplierId` — это первичный ключ для этой таблицы. В базе данных, безусловно, может содержаться большое количество разных товаров (каждый будет иметь уникальный идентификационный номер), поступающих от одного поставщика и каталогизированных в таблице `products`. Взгляните на таблицу `suppliers`, где должен быть только один уникальный идентификационный номер поставщика для каждой записи.

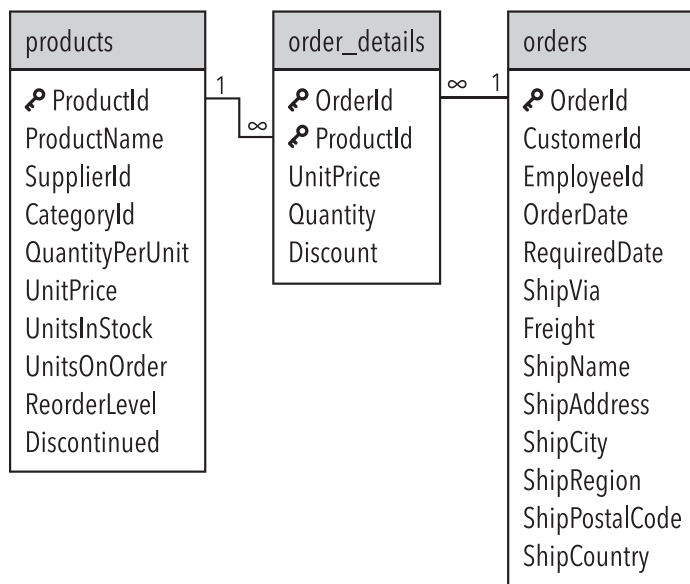


Рис. 13

**ПРИМЕЧАНИЕ**

Данные в поле `supplierId` могут повторяться в нескольких записях в таблице `products`, но не в таблице `suppliers`.

Давайте проанализируем взаимосвязь между таблицами `products`, `order_details` и `orders` (рис. 14).



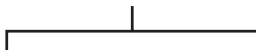
**Рис. 14**

Таблица `order_details` имеет два первичных ключа (обозначено значками ключей). Можно трактовать это как *составной ключ*, то есть для определения первичного ключа используются два или более поля. Хотя технически в примере задействовано два ключа, стоит рассматривать их как единый элемент — собственно первичный ключ.

Комбинация данных в полях, используемая для формирования составного ключа, работает как уникальный идентификатор для любой записи в таблице. Другими словами, если запись `OrderId` в таблице `order_details` равна 101, а `ProductId` той же записи — P006, то мы можем предположить, что никакая другая запись в таблице не будет иметь такую же комбинацию данных этих двух полей. Могут существовать одна или несколько других записей с `OrderId`, равным 101, а также одна или несколько других записей с `ProductId`, равным P006, но только одна

запись может иметь и 101 в качестве своего `OrderId`, и P006 в качестве своего `ProductId`. Эта комбинация данных работает как составной ключ, который, как и любой первичный ключ, обеспечивает уникальный идентификатор для каждой записи в таблице.

### СОСТАВНОЙ КЛЮЧ



OrderId	ProductId	UnitPrice	Quantity	Discount
101	P006	\$1,386.67	1	NULL
101	P003	\$7.00	3	NULL
101	P005	\$3.73	1	10%
102	P011	\$166.23	1	NULL
102	P013	\$2.35	1	NULL
103	P014	\$1,645.91	1	NULL
104	P001	\$12.99	3	NULL
104	P012	\$42.37	3	NULL
104	P011	\$166.23	2	10%
104	P003	\$7.00	5	NULL
105	P010	\$141.65	1	NULL
105	P004	\$11.86	3	NULL
105	P003	\$7.00	2	NULL
106	P014	\$1,645.91	1	NULL

Рис. 15

В связи «один-ко-многим» стандартный первичный ключ находится на стороне «один». Например, в таблице `orders` мы видим, что первичный ключ, поле `OrderId`, предоставляет уникальный идентификатор для каждой записи в таблице. Сторона связи «ко-многим» находится в таблице `order_details`. Как вы думаете, почему?

Давайте рассуждать логически. Можно сделать вывод, что роль таблицы `order_details` — предоставить информацию о различных заказанных товарах. Также допустимо, что любой товар может быть заказан несколько раз несколькими заказчиками при самых разных обстоятельствах, с разными ценами и т. д. Следовательно, поле `ProductId` не может использоваться само по себе в качестве первичного ключа для таблицы `order_details`. Кроме того, любой заказ может включать в себя несколько товаров, и если мы проанализируем другие поля,

используемые в таблице `order_details` — `UnitPrice`, `Quantity` и `Discount`, тогда мы четко поймем, что эти поля обозначают свойства определенного товара, а не заказ в целом. Также поле `OrderId` не может использоваться само по себе в качестве первичного ключа для таблицы `order_details`. Решение состоит в том, чтобы объединить `ProductId` и `OrderId` в составной ключ, тем самым гарантируя, что данные, содержащиеся в столбцах `UnitPrice`, `Quantity` и `Discount`, будут соответствовать уникальному и определенному заказу и уникальному и определенному товару в этом заказе.

## Типы данных

Ранее в этой главе мы описывали концепцию метаданных, которые представляют собой детальную информацию обо всех объектах системы. При разработке базы данных с использованием SQL для каждого используемого столбца должен быть назначен определенный *тип данных*. Типы данных могут незначительно отличаться в зависимости от используемой версии SQL. Однако, как правило, у вас будут числовые, символьные или текстовые типы данных, дата и время, а также логические значения. Рассмотрим каждый из них.

### Числовые типы данных

В состав числовых данных входят целочисленные данные, которые служат для отображения целых чисел. Обычно, когда используется целочисленный тип данных, он имеет ограничения на длину. Напомним, что изображенная на рис. 7 таблица содержала сведения о пациентах. Для столбца `weight` (Вес) разумно использовать целочисленный тип данных с ограничением до трех цифр. Почему?

1. Вполне достаточно округлить значение в большую и меньшую сторону до ближайшего значения фунта или килограмма и не использовать десятичные знаки.
2. Вряд ли нам понадобится более трех цифр для указания значения веса в фунтах.

Когда целочисленные данные не подходят и возникает необходимость в более точном числовом формате, мы можем использовать формат чисел с плавающей запятой. Как и целочисленные данные, они могут быть ограничены по длине.