Boxoft Image To PDF Demo. Purchase from www.Boxoft.com to remove the watermark

Занимательное программирование

БАЗЫ ДАННЫХ

Манга

マンガでわかる

データベース

高橋 麻奈／著　あづま 笙子／作画　トレンド・プロ／制作

Ohmsha

ОБРАЗОВАТЕЛЬНАЯ МАНГА

ЗАНИМАТЕЛЬНОЕ ПРОГРАММИРОВАНИЕ

БАЗЫ ДАННЫХ

Мана Такахаси
Сёко Адзума

Перевод
Сенниковой Т. И.

ДМК ПРЕСС
ИЗДАТЕЛЬСТВО
ДОДЭКА

Москва
Додэка, ДМК Пресс, 2014

УДК 004.6
ББК 32.972.34
Т15

Такахаси, Мана.
Т15 Занимательное программирование. Базы данных. Манга / Мана Такахаси (автор), Сёко Адзума (худож.) ; пер. Сенниковой Т. И. — М. : ДМК Пресс, 2014. — 238 с. : ил. — (Серия «Образовательная манга»). — Доп. тит. л. яп. — ISBN 978-5-97060-044-3.

У принцессы Руруны и Кейна возникла проблема: в их торгующей фруктами империи царит неразбериха из-за противоречивых данных, и поэтому дыни подменяются яблоками и клубникой, что вызывает большие трудности в работе. И что же им делать? Конечно же, построить реляционную базу данных, и поможет им в этом Тико, чудесная фея баз данных. Она покажет Руруне и Кейну, как создать базу данных, которая поможет управлять продажами, реализацией товара и его экспортом. Вы узнаете, как работает база данных, и поймёте значение таких терминов, как схемы, ключи, нормализация и транзакции.

Если у вас голова идёт кругом, когда речь заходит о базах данных, или же вы просто заплутали в лабиринте чисел и данных, которые, как вам кажется, неподвластны контролю, присоединяйтесь к Руруне и Кейну.

УДК 004.6
ББК 32.972.34

Original Japanese edition
Manga de Wakaru Database (Manga Guide: Databases)
By Mana Takahashi (Author), Shoko Azuma (Illustrator) and
Trend-Pro Co., Ltd. (Producer)
Published by Ohmsha, Ltd.
3-1 Kanda Nishikicho, Chiyodaku, Tokyo, Japan
Russian language edition copyright © 2014 by DMK Press
Translation rights arranged with Ohmsha, Ltd.

Все права защищены. Никакая часть этого издания не может быть воспроизведена в любой форме или любыми средствами, электронными или механическими, включая фотографирование, ксерокопирование или иные средства копирования или сохранения информации, без письменного разрешения издательства.

ISBN 978-4-274-06631-3 (яп.) Copyright © 2005 by Mana Takahashi and Trend-Pro Co., Ltd.
ISBN 978-5-94120-263-8 (Додэка) © Перевод, Издательский дом «Додэка-XXI», 2013
ISBN 978-5-97060-044-3 (ДМК Пресс) © Редактура, издание, ДМК Пресс, 2014

ПРЕДИСЛОВИЕ

Базы данных являются крайне важной частью практически всех автоматизированных коммерческих систем. Возможно, некоторые из читателей этой книги задумываются о том, чтобы внедрить базы данных в их повседневную работу. Другим, возможно, нужно разработать базу данных для решения реальных коммерческих задач. База данных — это технология, которая поддерживает работу таких бизнес-систем и обеспечивает техническую составляющую, и её истинную сущность понять непросто.

Эта книга построена так, чтобы читатель мог овладеть основами проектирования баз данных с помощью истории в стиле манга. В конце каждой главы даются практические задания с целью укрепления и расширения полученных знаний. Каждая глава построена так, чтобы читатель достиг понимания технологии баз данных, оценивая при этом, насколько хорошо он усвоил прочитанное.

Построение материала выглядит следующим образом:

В Главе 1 рассказывается, для чего мы используем базы данных. Зачем нужны базы данных? Какие могут возникать проблемы, если их не использовать? Вы получите обзорную информацию, которая необходима при использовании баз данных.

В Главе 2 даётся основная терминология. Вы узнаете о различных моделях баз данных и других относящихся к ним понятиях.

В Главе 3 объясняется, как разработать базу данных, а в частности реляционную базу данных – наиболее распространённый вид БД.

Глава 4 посвящена языку SQL, который применяется для работы с реляционными базами данных. С помощью этого языка вы можете легко управлять данными.

В Главе 5 объясняется структура системы баз данных. Так как БД — это система, посредством которой данными пользуется большое количество людей, вы научитесь тому, как сделать этот процесс безопасным.

В Главе 6 приводятся описания приложений для работы с базами данных. Вы узнаете, как используются веб-приложения и другие виды систем баз данных.

Многие внесли свой вклад в издание этой книги: Шоко Азума (рисунки), компания TREND-PRO (издание), издательство Ohmsha (планирование, редактирование, маркетинг). Выражаю свою глубокую признательность всем участникам проекта.

Надеюсь, эта книга будет полезна всем её читателям.

Мана Такахаси

СОДЕРЖАНИЕ

1. ЧТО ТАКОЕ БАЗА ДАННЫХ 1
Зачем нам базы данных? 2
Что случилось в королевстве? 16
Данные дублируются..... 16
Данные могут противоречить друг другу 17
Данные трудно обновлять..... 18
Наш ответ — база данных!..... 19
Как пользоваться базой данных?..... 19
Итоги..... 21
Использование программного обеспечения для управления базами данных..... 21

2. ЧТО ТАКОЕ РЕЛЯЦИОННАЯ БАЗА ДАННЫХ 23
Термины, используемые в базах данных..... 24
Реляционные базы данных 34
Какие бывают модели данных 39
Операции извлечения данных..... 39
Теоретико-множественные операции..... 39
Специальные реляционные операции 43
Вопросы и задания 46
Да здравствует реляционная база данных! 48
Итоги..... 48
Ответы 48

3. ДАВАЙТЕ СПРОЕКТИРУЕМ БАЗУ ДАННЫХ! 49
Модель сущность–связь (E–R-модель)..... 50

Нормализация таблицы **56**
Что такое модель сущность–связь (E–R-модель) .. **74**
Как анализировать модель сущность–связь **74**
Пример 1. Связь «один к одному» 75
Пример 2. Связь «один ко многим» 75
Пример 3. Связь «многие ко многим» 76
Вопросы и задания 76
Нормализация таблицы **78**
Вопросы и задания 79
Стадии разработки базы данных **81**
Итоги **82**
Ответы **82**
Проектирование базы данных **84**

4. ДАВАЙТЕ ИЗУЧАТЬ SQL! **85**
Применение SQL **86**
Поиск данных с помощью команды SELECT **93**
Применение агрегатных функций **98**
Соединение таблиц **101**
Создание таблиц **103**
Обзор языка SQL **106**
Поиск данных с помощью команды SELECT **106**
Создание условий **107**
Операторы сравнения 107
Логические операторы **107**
Шаблоны **108**
Поиск **109**
Вопросы и задания **109**
Агрегатные функции **110**
Агрегирование данных. Группирование 111
Вопросы и задания **112**
Поиск данных **113**

Использование подзапроса 114
Коррелированный подзапрос 115
Вопросы и задания 116
Соединение таблиц 116
Создание таблиц 117
Вставка, обновление и удаление строк 118
Представление 119
Вопросы и задания 120
Использование SQL из прикладного ПО 121
Перемещение по записям с использованием курсора 124
Итоги 124
Ответы 125
Стандартизация SQL 128

5. ДАВАЙТЕ УПРАВЛЯТЬ БАЗОЙ ДАННЫХ! ..129
Что такое транзакция? 130
Что такое блокировка (lock)? 135
Защита базы данных 142
Как всё ускорить с помощью индексирования 147
Аварийное восстановление 152
Свойства транзакций 157
Атомарность (Atomicity) 158
Вопросы и задания 159
Согласованность (Consistency) 159
Изоляция (isolation) 160
Вопросы и задания 161
Двухфазное блокирование (two-phase locking) ... 161
Детализация блокировок 162
Вопросы и задания 163
Другие виды управления параллелизмом (параллельным доступом) 163

управление меткой времени (Timestamp control) 163
Оптимистическое управление параллелизмом (Optimistic control) 163
Уровни изоляции 164
Устойчивость 165
Вопросы и задания 166
Индексы (Index) 167
Вопросы и задания 169
Оптимизация запроса 169
Соединение вложенных циклов (Nested Loop Join) 171
Соединение сортировка—слияние (Sort Merge Join) 171
Хэш-соединение (Hash Join) 172
Оптимизатор (optimizer) 172
На базе правил (rule based) 172
По стоимости выполнения (cost based) 172
Когда наступает катастрофа 172
Виды сбоев 173
Контрольные точки (Checkpoints) 173
Вопросы и задания 174
Итоги 174
Ответы 174

6. КРУГОМ БАЗЫ ДАННЫХ 177
Применение баз данных 183
Базы данных и "Всемирная паутина" 185
Распределённые базы данных 191
Хранимые процедуры и триггеры 193
Базы данных в Интернете 202
Использование хранимых процедур 205
Вопросы и задания 206
Что такое распределённая база данных (Distributed Database)? 206

Горизонтальное распределение (Horizontal Distribution) 206
Вертикальное распределение (Vertical Distribution) ... 207
Декомпозиция данных (Data partitioning) 208
Горизонтальная декомпозиция (Horizontal Partition) .. 208
Вертикальная декомпозиция (Vertical Partition) 209
Предотвращение несогласованности при двухфазной фиксации транзакций 209
Вопросы и задания 211
Связанные таблицы в распределённых БД 211
Вложенные циклы (nested loop) 212
Сортировка слиянием (sort merge) 212
Полуслияние (semi join) 213
Хэш-полуслияние (hash semi join) 214
Репликация баз данных (Database replication) 215
Только чтение (read-only) 215
Репликация, доступная для всех серверов 216
Дальнейшее применение баз данных 217
XML 217
Объектно-ориентированные базы данных (OODB) 217
Итоги 219
Ответы 219
Подведение итогов 220

ЧАСТО ИСПОЛЬЗУЕМЫЕ SQL-КОМАНДЫ ... 221

СПРАВОЧНАЯ ЛИТЕРАТУРА 222

ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ 223

1.
ЧТО ТАКОЕ БАЗА ДАННЫХ

ЗАЧЕМ НАМ БАЗЫ ДАННЫХ?
БЛА, БЛА, БЛА
Фру
Сметана
Клубника нового урожая
Дыни
Яблоки
КЛАЦ, КЛАЦ
Королевство Ямс
ВАШЕ ВЫСОЧЕСТВО, ПРИНЦЕССА РУРУНА...
НЕ БУДЬТЕ ТАКОЙ ГРУСТНОЙ!
НО, КОЛОНЭ, ДОРОГАЯ МОЯ,...

Целая гора
МНЕ ТАК МНОГО НАДО СДЕЛАТЬ!

КОРОЛЕВСТВО ЯМС — "СТРАНА ФРУКТОВ".
Благословенная земля, ля-ля
Какой чудесный день!
ЗНАТНЫМ БУДЕТ УРОЖАЙ!

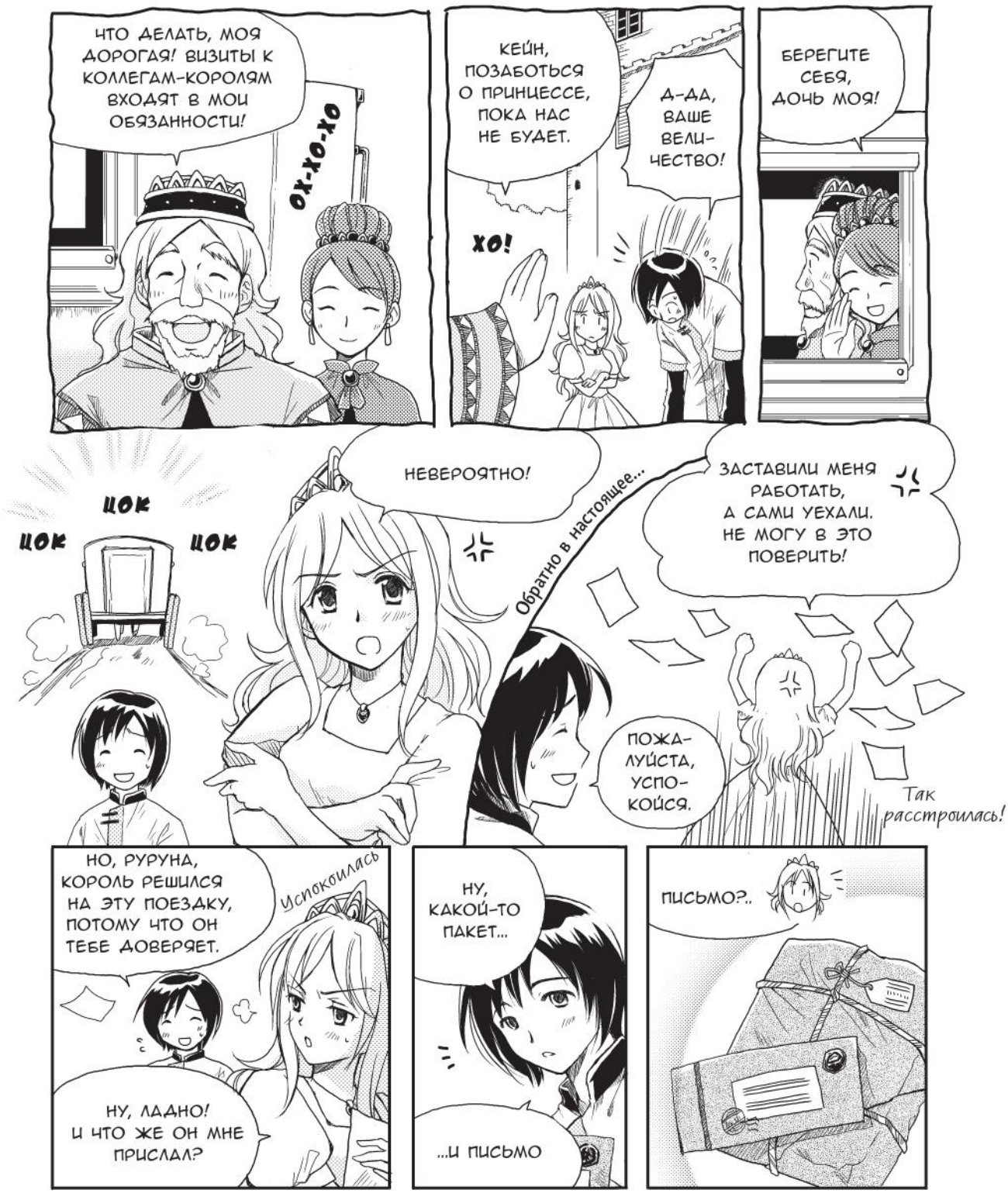
ВАМ БЫ РАДОВАТЬСЯ, ЧТО ТАК МНОГО ДЕЛ!
Бр...
Вот, возьмите ещё.

Ох...
ЖАЛЬ, ЧТО Я НЕ МОГУ РАБОТАТЬ БЫСТРЕЕ.

НУ, ЧТО ВЫ, ВАШЕ ВЫСОЧЕСТВО, Я РЕГУЛЯРНО ПРИНОШУ ВАМ ДОКУМЕНТЫ,
отдел продаж
отдел развития
отдел экспорта
СОСТАВЛЕННЫЕ ОТДЕЛОМ ПРОДАЖ, ОТДЕЛОМ РАЗВИТИЯ И ОТДЕЛОМ ЭКСПОРТА, ТАК ВЕДЬ? И ВЫ ПРЕКРАСНО СПРАВЛЯЕТЕСЬ!
НУ, ДА.

У НАС ЧРЕЗВЫЧАЙНО ЭФФЕКТИВНАЯ СИСТЕМА!
А ТЕПЕРЬ ЗА РАБОТУ!
Ла-ди-дада, ла-ди-дум
БАХ!

ГММ...
КАК-ТО НЕ ОЧЕНЬ УДОБНО РАБОТАТЬ ОТДЕЛЬНО С ДАННЫМИ ПО КАЖДОМУ ОТДЕЛУ.
ЦЕЛАЯ МОРОКА БЫЛА, КОГДА ЦЕНЫ НА ЯБЛОКИ ПРИШЛИ ТОЛЬКО НА СЛЕДУЮЩИЙ ДЕНЬ.
ВАШЕ ВЫСОЧЕСТВО, ПРИНЦЕССА РУРУНА!
БАХ!
А, ЭТО ТЫ, КЕЙН. ЧТО У ТЕБЯ?
У МЕНЯ ПОДАРОК ОТ КОРОЛЯ.
ОТ МОЕГО ОТЦА?..
ЕСЛИ БЫ МОИ РОДИТЕЛИ ВСЁ ЕЩЁ БЫЛИ В ЗАМКЕ, ЭТОЙ НЕРАЗБЕРИХИ С ЦЕНАМИ НЕ ПРОИЗОШЛО БЫ!..
ДРОЖЬ ДРОЖЬ
Прин-цесса?
В недалёком прошлом...
ВАМ ОБЯЗАТЕЛЬНО УЕЗЖАТЬ?


Что делать, моя дорогая! Визиты к коллегам-королям входят в мои обязанности!
Ох-хо-хо
Кейн, позаботься о принцессе, пока нас не будет.
Д-да, ваше величество!
Хо!
Берегите себя, дочь моя!
Цок
Цок
Цок
Невероятно!
Обратно в настоящее...
Заставили меня работать, а сами уехали. Не могу в это поверить!
Пожалуйста, успокойся.
Так расстроилась!
Но, Руруна, король решился на эту поездку, потому что он тебе доверяет.
Успокоилась
Ну, ладно! И что же он мне прислал?
Ну, какой-то пакет...
...и письмо
Письмо?..

"МЫ ПОБЫВАЛИ В ДАЛЁКОЙ СТРАНЕ И НАШЛИ ТАМ КНИГУ ОБ ОДНОЙ ПЕРЕДОВОЙ ТЕХНОЛОГИИ.
ТОТ, КТО ОТДАЛ ЕЁ НАМ, СКАЗАЛ, ЧТО В КНИГЕ РАССКАЗЫВАЕТСЯ О СЕКРЕТНОЙ ТЕХНОЛОГИИ ПОД НАЗВАНИЕМ "БАЗА ДАННЫХ".
МЫ СЛЫШАЛИ, ЧТО БАЗА ДАННЫХ — ЭТО СИСТЕМА, ПОЗВОЛЯЮЩАЯ ЛЮДЯМ ИСПОЛЬЗОВАТЬ ДАННЫЕ, ОБМЕНИВАТЬСЯ И УПРАВЛЯТЬ ИМИ.
НО ПОЛЬЗОВАТЬСЯ КНИГОЙ МОЖНО ПО-РАЗНОМУ, ВСЁ ЗАВИСИТ ОТ ТОГО, КТО ЕЁ ЧИТАЕТ.
ТОТ ЧЕЛОВЕК ОТДАЛ НАМ КНИГУ В НАДЕЖДЕ, ЧТО КОРОЛЕВСТВО ЯМС ИСПОЛЬЗУЕТ ЕЁ ВО БЛАГО.
РУРУНА...

ШАРАХ!
О, ТЫ ДАЖЕ НЕ ПРЕДСТАВЛЯЕШЬ, КАК МЕНЯ СЕЙЧАС ВСЁ БЕСИТ!
КЛОЧЬЯ КЛОЧЬЯ
ОТКРОЙ ЕЁ И ИСПОЛЬЗУЙ ВО БЛАГО НАШЕЙ СТРАНЫ".
ЧТО ЕЩЁ ЗА ФИГНЯ?
Ш... Ш...
НАДО ЖЕ, ТАКАЯ СТАРИННАЯ...
ШЕЛЕСТ
ЗАПЕРТА.
Не открывается
О, В КОНВЕРТЕ КЛЮЧ. МОЖЕТ, ОТ НЕЁ?
подо-шел
ЩЁЛК..
БА-БАХ!
А-А-А-АЙ!
!!!
?!
Апчхи!
Кхе, кхе!

Свобода-а-а!!!
НУ, И...
...ГДЕ ЭТО Я?
И КТО ВЫ ТАКИЕ?
ТЫ В КОРОЛЕВСКОМ ЗАМКЕ.
ЗДЕСЬ НЕПЛОХО!
МЕНЯ ЗОВУТ КЕЙН, Я ПОМОЩНИК РУРУНЫ — ПРИНЦЕССЫ КОРОЛЕВСТВА ЯМС.
А ТЫ КТО ТАКАЯ?
Летает?!
ВХИХ!
ПРИВЕДЕНИЕ?!
Только не это!

НЕДОВОЛЬНА!

ОБИЖАЕШЬ! МЕНЯ ЗОВУТ ТИКО.

Я ФЕЯ.

ФЕЯ?

...Я НЕ ПРИВЕДЕНИЕ!

МЕНЯ МОГУТ ВИДЕТЬ ТОЛЬКО ТЕ, КТО ОТКРЫЛ ЭТУ КНИГУ.

ШАРАХ!

Эээ!

Хи-хи!

ТАК ТЫ ВЫЛЕТЕЛА ИЗ ЭТОЙ КНИЖКИ?

АГА!

ЭТА КНИГА ОБЛАДАЕТ СВЕРХСИЛОЙ И ПОМОГАЕТ ЛЮДЯМ, ОТКРЫВШИМ ЕЁ, ВОСПОЛЬЗОВАТЬСЯ ЗНАНИЯМИ...

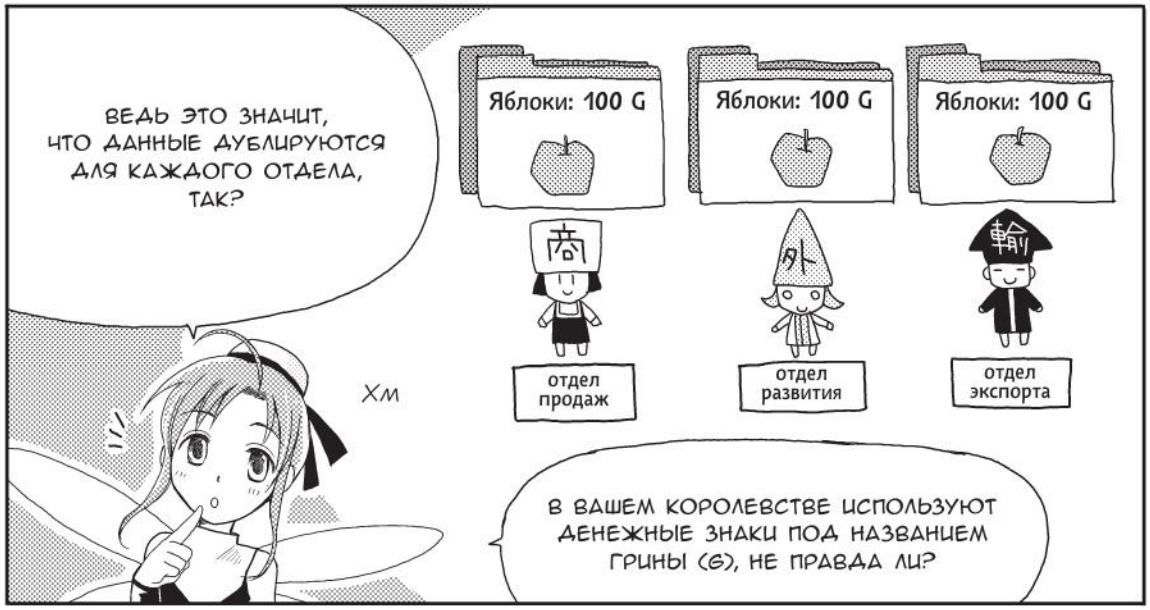
...НАДЛЕЖАЩИМ ОБРАЗОМ

ЗНАЧИТ... ТЫ БУДЕШЬ НАМ ПОМОГАТЬ?

ТОЧНО!

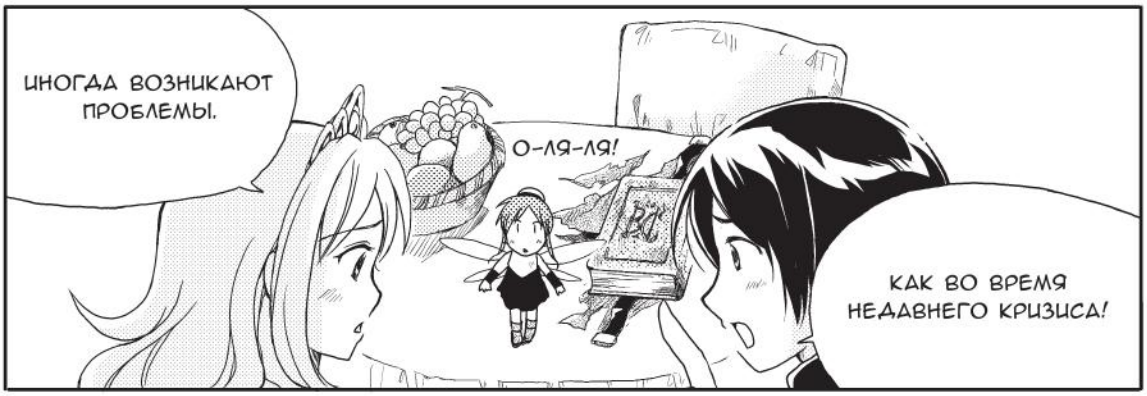
ГМ...

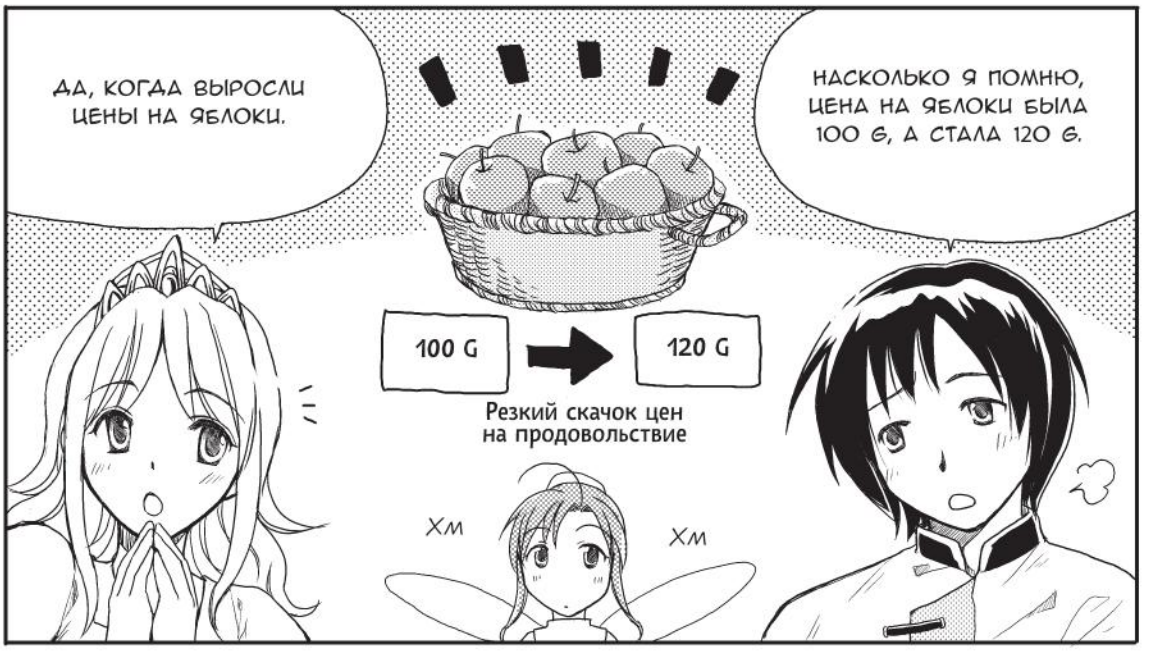
ПОКА ЧТО ОНА КАЖЕТСЯ БЕЗОБИДНОЙ...
АГА...
НУ, ХВАТИТ ОБО МНЕ! Я ПОЛАГАЮ, ВЫ ОТКРЫЛИ КНИГУ, ЧТОБЫ УЗНАТЬ ПРО...
...БАЗЫ ДАННЫХ?
ПОЖАЛУЙ, ЧТО ДА...
ТОГДА НАЧНЁМ.
ЧТОБЫ СОЗДАТЬ БАЗУ ДАННЫХ...
МИНУТОЧКУ!
ВОПРОС, КОНЕЧНО, ЭЛЕМЕНТАРНЫЙ...
...НО ЧТО ТАКОЕ БАЗА ДАННЫХ?
ТАК ВЫ НЕ ЗНАЕТЕ?
ВАМ ПРИХОДИТСЯ ИМЕТЬ ДЕЛО С ЧИСЛАМИ И ЗНАЧЕНИЯМИ, ТАК?
ДА, И ВОЗНИКАЕТ КУЧА ПРОБЛЕМ...
Я РАБОТАЮ С ЧИСЛАМИ И ЗНАЧЕНИЯМИ, КОТОРЫЕ ОТНОСЯТСЯ К ТОВАРАМ,
КЛИЕНТАМ И ПРОДАЖАМ, И СОЗДАЮ ФАЙЛЫ ПО ОТДЕЛАМ.
Товар
Клиент
Про-дажи
О, ТАК ТЫ РАБОТАЕШЬ С ДАННЫМИ В РАЗРОЗНЕННОМ ВИДЕ, ПО ОТДЕЛАМ. БЕДНЯЖКА!
Хм

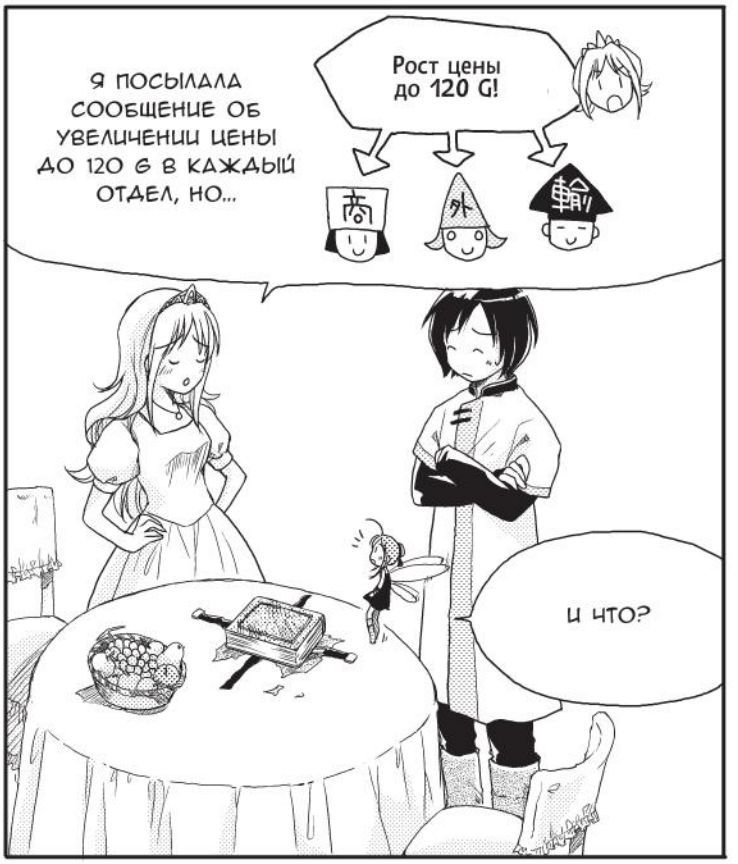

ВЕДЬ ЭТО ЗНАЧИТ, ЧТО ДАННЫЕ ДУБЛИРУЮТСЯ ДЛЯ КАЖДОГО ОТДЕЛА, ТАК?
Яблоки: 100 G
Яблоки: 100 G
Яблоки: 100 G
商
外
輸
отдел продаж
отдел развития
отдел экспорта
Хм
В ВАШЕМ КОРОЛЕВСТВЕ ИСПОЛЬЗУЮТ ДЕНЕЖНЫЕ ЗНАКИ ПОД НАЗВАНИЕМ ГРИНЫ (G), НЕ ПРАВДА ЛИ?

ВЕРНО.
ЯБЛОКИ 100G
ЯБЛОКИ 100G
ЯБЛОКИ
И В КАЖДОМ ОТДЕЛЕ СВОИ ОТДЕЛЬНЫЕ ДАННЫЕ.

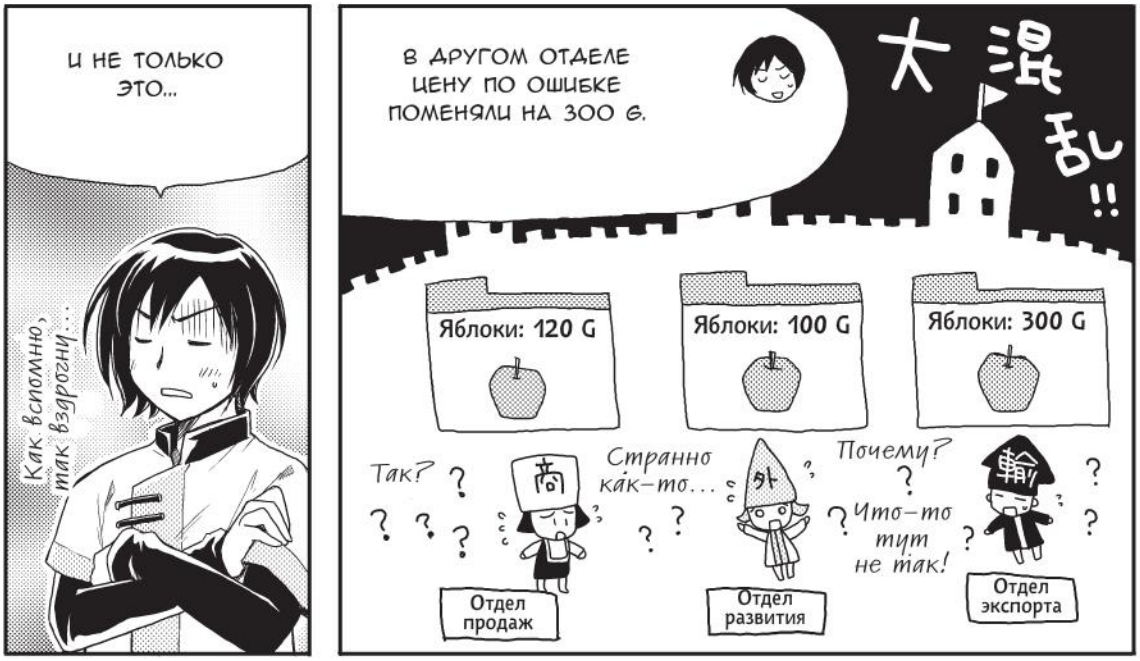
КОЛОНЭ ГОВОРИТ, ЧТО...
..."ЭТО ОЧЕНЬ ЭФФЕКТИВНАЯ СИСТЕМА",
НО...


ИНОГДА ВОЗНИКАЮТ ПРОБЛЕМЫ.
О-ЛЯ-ЛЯ!
КАК ВО ВРЕМЯ НЕДАВНЕГО КРИЗИСА!


Да, когда выросли цены на яблоки.
Насколько я помню, цена на яблоки была 100 G, а стала 120 G.
100 G
120 G
Резкий скачок цен на продовольствие
Хм
Хм


Я посылала сообщение об увеличении цены до 120 G в каждый отдел, но...
Рост цены до 120 G!
商
外
輸
И что?

...в одном отделе забыли изменить цену.
ШОК!
Я не получал от вас сообщения ...
Я спал ...
Отдел развития
Яблоки 100 G
Цена осталась прежней!


И НЕ ТОЛЬКО ЭТО...
Как вспомню, так вздрогну...
В ДРУГОМ ОТДЕЛЕ ЦЕНУ ПО ОШИБКЕ ПОМЕНЯЛИ НА 300 G.
大混乱!!
Яблоки: 120 G
Яблоки: 100 G
Яблоки: 300 G
Так?
Странно как-то...
Почему?
Что-то тут не так!
Отдел продаж
Отдел развития
Отдел экспорта

ПОЛУЧИЛОСЬ, ЧТО ДАННЫЕ В РАЗНЫХ ОТДЕЛАХ НЕ СОГЛАСУЮТСЯ, ТАК?
ОХ!
Вздох
ЭХ!
ВОТ ИМЕННО.

ОСОБЕННО ТЯЖЕЛО БЫЛО КЕЙНУ! ЕМУ ПРИШЛОСЬ БЕГАТЬ ПО ВСЕМУ КОРОЛЕВСТВУ...
...И ИСПРАВЛЯТЬ ОШИБКИ.
ДА УЖ...
Яблоки 120 G
Цена на яблоки другая...
Исправьте цену, пожалуйста!
Ого.
До сих пор ещё не отошёл...

МОЙ ОТЕЦ КАК-ТО СКАЗАЛ: "ДАВАЙ КОГДА-НИБУДЬ В БУДУЩЕМ ОРГАНИЗУЕМ ТУР ПО МЕСТАМ СБОРА ФРУКТОВ!"
НО, КАК МНЕ КАЖЕТСЯ, МЫ ЕЩЁ ОЧЕНЬ ДАЛЕКИ ОТ ЭТОГО.
Он свободный и беспечный.
Добро пожаловать в Ямс!
фруктовую страну
Давайте начнём новое дело, используя уже существующую систему. Хе-хе-хе!

ДАЖЕ ЕСЛИ МЫ ВСЁ-ТАКИ НАЧНЁМ НОВОЕ ДЕЛО, ПОХОЖЕ, ЧТО ИСПОЛЬЗОВАТЬ НАКОПЛЕННЫЕ В НЫНЕШНЕМ ВИДЕ ДАННЫЕ БУДЕТ НЕВОЗМОЖНО.
Как-то всё нескладно
НАЧИНАЯ НОВОЕ ДЕЛО, ВАМ ПРИДЁТСЯ СОЗДАТЬ НОВЫЕ ДОКУМЕНТЫ ДЛЯ НОВОГО ОТДЕЛА.

ЗНАЧИТ, МОИ ОБЯЗАННОСТИ НИ НА ЙОТУ НЕ УМЕНЬШАТСЯ!
РАЗДРАЖЕНА
Успокойся, пожалуйста!

ПОХОЖЕ, ТЕБЕ ПРИХОДИТСЯ ВНОСИТЬ ДАННЫЕ И ДЕЛАТЬ ПОДТВЕРЖДЕНИЯ КАЖДЫЙ РАЗ, КОГДА ЧТО-ТО МЕНЯЕТСЯ, А ЭТО ТЯЖЕЛАЯ РАБОТЁНКА.
Тебя ведь измучит такая работа с данными, даже если будешь стараться изо всех сил?
Кипа документов

СИСТЕМА, В КОТОРОЙ ДАННЫМИ МОЖЕТ ПОЛЬЗОВАТЬСЯ КАЖДЫЙ, НАЗЫВАЕТСЯ **БАЗОЙ ДАННЫХ**.
ЕСЛИ БЫ ТЫ УМЕЛА ЕЙ ПОЛЬЗОВАТЬСЯ, ТЕБЕ БЫ НЕ ПРИШЛОСЬ ХРАНИТЬ БЕСПОЛЕЗНЫЕ ДАННЫЕ.
Гениально!

ПРАВДА?
ТЫ ХОЧЕШЬ СКАЗАТЬ, У НАС МОЖЕТ БЫТЬ СИСТЕМА ГОРАЗДО ЭФФЕКТИВНЕЕ СУЩЕСТВУЮЩЕЙ?

БЕЗ СОМНЕНИЯ! ТЕБЕ ХОТЕЛОСЬ БЫ ИМЕТЬ ТАКУЮ СИСТЕМУ?
АГА!

Ты такая умная...
ТЫ НАМ ЕЁ ПОДАРИШЬ?
УВЫ, НЕТ.
Хи-хи

У МЕНЯ НЕТ ФИЗИЧЕСКОЙ ОБОЛОЧКИ, ПОЭТОМУ В РЕАЛЬНОМ МИРЕ Я НЕ МОГУ ПОЛЬЗОВАТЬСЯ КОМПЬЮТЕРОМ.
Мне жаль.
Ясно.
НО В БЛАГОДАРНОСТЬ ЗА ТО, ЧТО ВЫ ВЫПУСТИЛИ МЕНЯ ИЗ КНИГИ...

...Я ВАС ВСЕМУ НАУЧУ.
Опять за парту?!
РАДИ МОЕЙ СТРАНЫ...
...Я ПОЙДУ НА ЭТО!
О, принцесса!

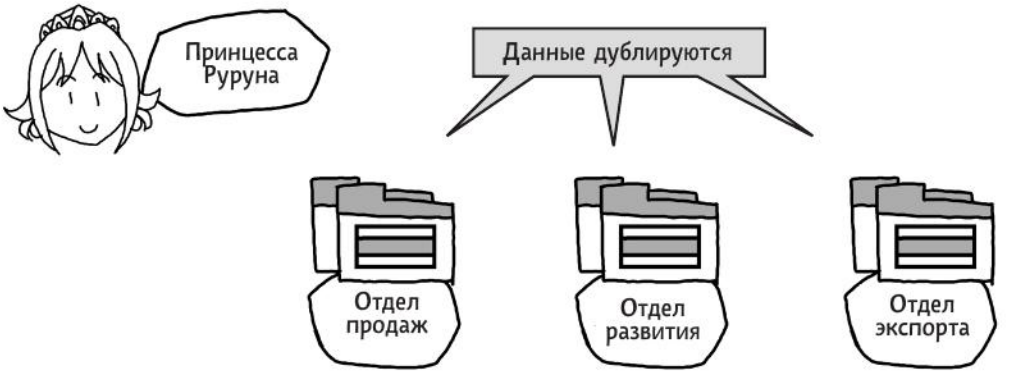
Что случилось в королевстве?

В королевстве Ямс при работе с данными сложилась система, основанная на документах. Но, похоже, такая система вызывает некоторые проблемы. Что же это за проблемы? Давайте рассмотрим подробно.

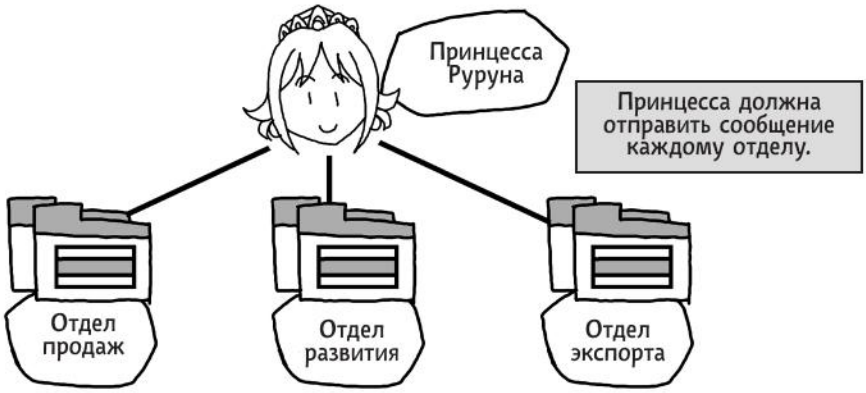
Сейчас в королевстве есть три отдела: отдел продаж, отдел развития и отдел экспорта. Отдел продаж отслеживает все продажи фруктов, выращенных в стране, отдел развития работает со странами — деловыми партнёрами королевства, а отдел экспорта ведает отправкой фруктов за границу.

Данные дублируются

Принцессу Руруну не устраивает нынешняя система. Но почему? Каждый отдел в королевстве работает с данными сам по себе. Например, отдел продаж и отдел экспорта по отдельности создают документы для работы с данными. Значит, во всех отделах данные дублируются без надобности. Каждый отдел должен внести данные, сохранить их, затем напечатать документ для подтверждения, и всё это — пустая трата времени. Вдобавок данные, попавшие в один отдел, никогда эффективным образом не будут использованы другим отделом.



Но и это не всё. Такая система создаёт проблемы, когда данные необходимо изменить. Например, предположим, что цена на яблоки изменилась. Чтобы всем стало об этом известно, принцесса Руруна должна оповестить каждый отдел индивидуально. Разве это удобно?



ДАННЫЕ МОГУТ ПРОТИВОРЕЧИТЬ ДРУГ ДРУГУ

Если каждый отдел оповещён об изменении цены, то это может показаться вполне достаточным. Однако тут могут возникнуть новые проблемы. Предположим, что принцесса Руруна благополучно оповестила все три отдела об изменении цены на яблоки. Но отдел развития мог забыть поменять цену, или же отдел экспорта поставил цену 300 G, а не 120 G. Такие ошибки приводят к разночтению данных при обмене ими между отделами, вследствие чего содержание документов отличается от реального положения вещей.

■ ОТДЕЛ ПРОДАЖ

Название товара	Цена
Дыни	800 G
Клубника	150 G
Яблоки	120 G
Лимоны	200 G

■ ОТДЕЛ РАЗВИТИЯ

Название товара	Цена
Дыни	800 G
Клубника	150 G
Яблоки	100 G
Лимоны	200 G

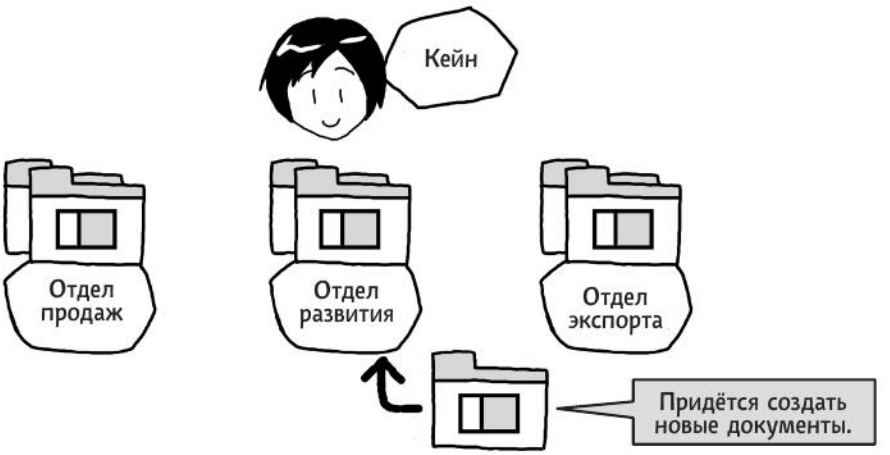
■ ОТДЕЛ ЭКСПОРТА

Название товара	Цена
Дыни	800 G
Клубника	150 G
Яблоки	300 G
Лимоны	200 G

ДАННЫЕ ТРУДНО ОБНОВЛЯТЬ

Существующая система не только приводит к противоречивым данным, но также затрудняет ответную реакцию на изменения в бизнесе. Например, представим себе, что король решил сформировать новый отдел — туризма. Когда гид будет проводить тур по фруктовым садам и рассказывать про торговлю фруктами в королевстве, он захочет привести самые последние данные по продажам.

Но, к несчастью, существующая система не всегда может позволить одному отделу воспользоваться данными других отделов, ведь документы ведутся независимо друг от друга. Чтобы управлять работой нового туристического направления, принцессе Руруне придётся сделать копии всех документов, относящихся к работе отдела туризма!



■ ДОКУМЕНТЫ ДЛЯ ОТДЕЛА ПРОДАЖ

Название товара	Цена	
Дыни	800 G	
Клубника	150 G	
Яблоки	120 G	
Лимоны	200 G	

■ ДОКУМЕНТЫ ДЛЯ ОТДЕЛА ТУРИЗМА

Название товара	Цена	
Дыни	800 G	
Клубника	150 G	
Яблоки	120 G	
Лимоны	200 G	

Это, в свою очередь, увеличит число дублируемых данных, образующихся при формировании нового отдела. Учитывая эти недостатки, существующую систему нельзя назвать эффективной. Она затрудняет создание новых проектов и реакцию на изменения.

НАШ ОТВЕТ — БАЗА ДАННЫХ!

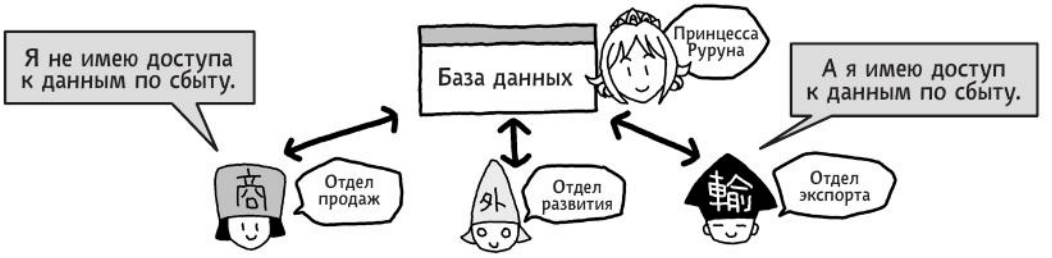
Так почему же такая система неэффективна? Всё дело в том, что каждый отдел работает с данными по отдельности и независимо друг от друга. Что же делать Руруне и Кейну? Им нужно создать базу данных! Они должны унифицировать (привести к единообразию) работу с данными для всего королевства. В следующей главе я расскажу вам, как это сделать.

Единая система управления данными гарантирует, что все отделы обладают корректной информацией, ведь каждый из них запрашивает данные из единого источника. Какая эффективная система получается! Она предотвращает появление противоречивых данных, а также исключает их дублирование, позволяя упростить процедуру формирования и интегрирования новых отделов.

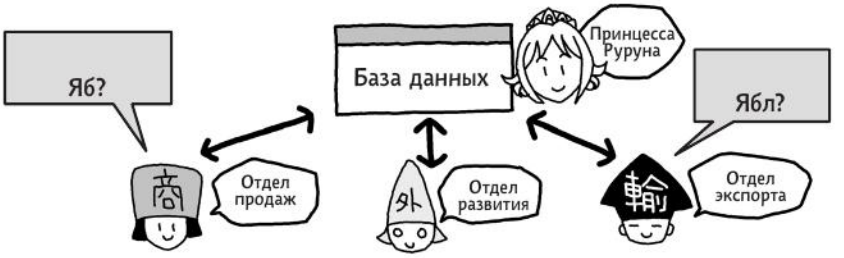
КАК ПОЛЬЗОВАТЬСЯ БАЗОЙ ДАННЫХ?

Для внедрения и управления базой данных вам нужно понять её уникальные задачи. Во-первых, ею будет пользоваться много людей, поэтому вам нужно определить метод, упрощающий ввод и извлечение данных. Этот метод должен быть простым в использовании.

Конечно, новая система несёт в себе некоторые риски. К примеру, может оказаться так, что какие-то пользователи украдут или переиначат важную информацию, такую как данные о продажах, а ведь это конфиденциальная информация, и для её защиты доступ к ней должен быть ограничен. Или, скажем, к данным по продажам в королевстве Моро должен иметь доступ только отдел экспорта. Организация безопасности и прав доступа очень важна при создании такой системы.



Новая система может иметь и другие проблемы. Базой данных могут одновременно пользоваться много людей. Предположим, что кто-то из отдела развития и кто-то из отдела экспорта одновремнно пытаются изменить название товара: первый — с «Яблоки» на «Яб», а второй — с «Яблоки» на «Ябл». И что же тогда произойдёт? Если этой базой будут пользоваться многие, то такую проблему обязательно надо учитывать.



Кроме того, нужно следить за тем, чтобы не потерять какие-нибудь данные. Например, может «зависнуть» система или откажет жёсткий диск, и это вызовет искажение данных. В базе должен быть механизм восстановления данных после таких распространённых сбоев.

Хотелось бы добавить, что в базе хранится большое количество данных, поэтому надо вести поиск довольно быстро. Для этого новая система должна быть достаточно мощной.

Давайте же приступим к изучению баз данных вместе с принцессой Руруной и Кейном и узнаем, как разрешить эти проблемы. Вперёд, к главе 2!

ИТОГИ

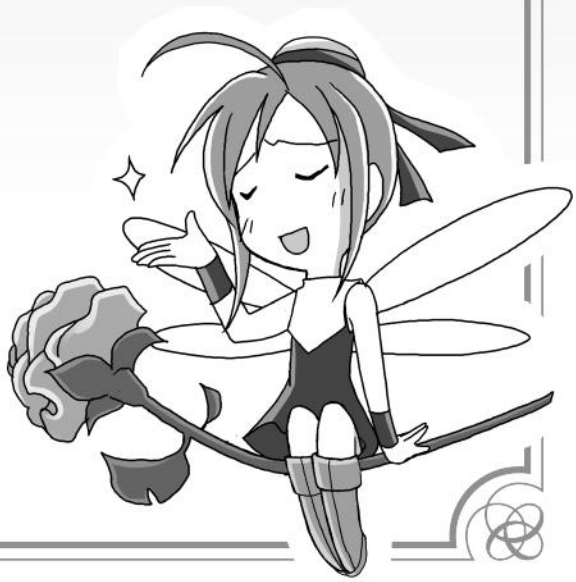
- Работа с данными в виде отдельных разрозненных документов может вызывать искажение данных и их дублирование.
- База данных позволяет легко обмениваться информацией, избегая ее искажения и дублирования.

ИСПОЛЬЗОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ

База данных, которую мы собираемся рассматривать, управляется программой, называемой СУБД (система управления базами данных, DBMS — Database Management System). У СУБД есть много полезных функций — она позволяет вводить данные в базу, предотвращать искажение и извлекать большое количество информации с большой скоростью. Благодаря СУБД базой данных могут одновременно пользоваться много людей. Плюс к этому СУБД может обеспечить безопасность базы данных — к примеру, она позволяет базе работать корректно даже при возникновении сбоя. К тому же СУБД предоставляет удобный в работе пользовательский интерфейс. В следующей главе мы рассмотрим базы данных и функции СУБД.

2.

ЧТО ТАКОЕ РЕЛЯЦИОННАЯ БАЗА ДАННЫХ



ТЕРМИНЫ, ИСПОЛЬЗУЕМЫЕ В БАЗАХ ДАННЫХ

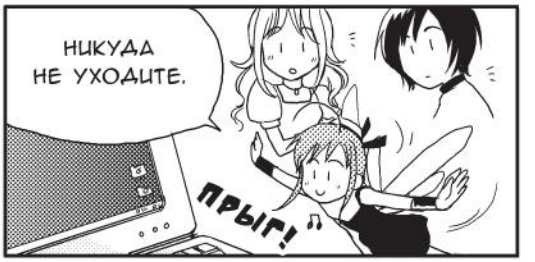
Попался!
А это ещё кто?
ТЫ ЖЕ ОБЕЩАЛ ПРИЙТИ КО МНЕ СЕГОДНЯ НА СВИДАНИЕ.
НУ... Я, ХМ...
ЭТО ДЛЯ ТЕБЯ.
???
ТАЩИТ ЗА СОБОЙ...
Ририна, я ещё вернусь.
Как насчёт моих цве-тов?
Фух!
ВСЁ-ТАКИ КАКОЙ ОН НАДОЕДЛИВЫЙ!
ПЛЮХ
НО РАМИНЕС — ПРИНЦ ИЗ СОСЕДНЕГО КОРОЛЕВСТВА.
ТЫ НЕ ДОЛЖНА ОТНОСИТЬСЯ К НЕМУ ТАК ПРЕНЕБРЕ-ЖИТЕЛЬНО.
А то я не знаю...
ДОБРОЕ УТРО!

А-А-А!
А, ЭТО ТЫ! И ДАВНО ТЫ ТУТ?
С ТЕХ ПОР, КАК ВЫ НЕ ТАК ДАВНО ОТКРЫЛИ КНИГУ.
Привет!

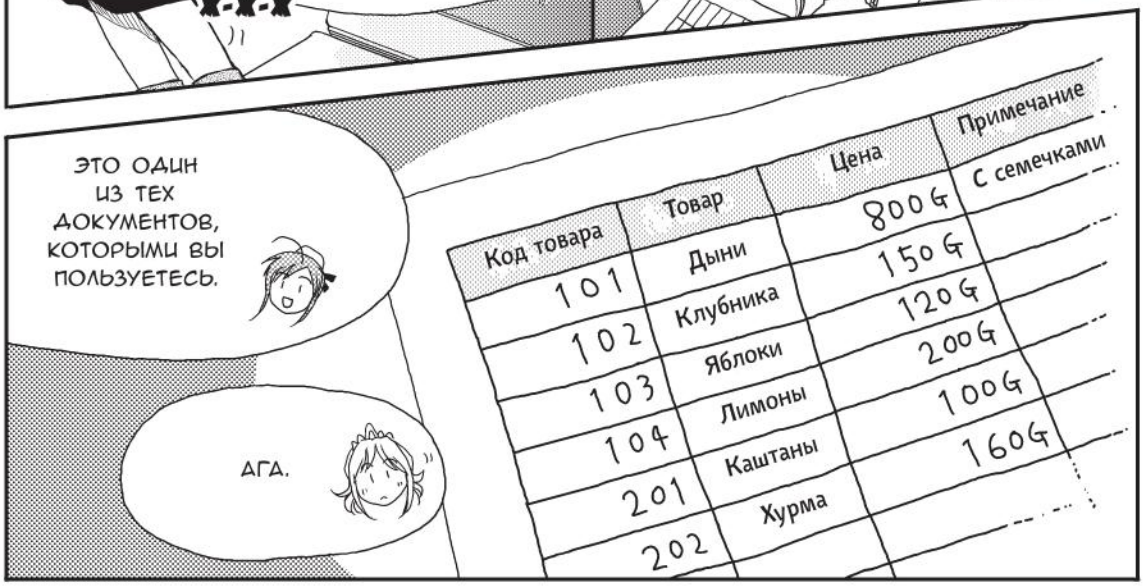
КОМУ ДОБРОЕ, А КОМУ И НЕТ. ТЫ ТАКАЯ БЕЗЗАБОТНАЯ.
Ты не перестаёшь меня удивлять.
НУ, ДА ЛАДНО. ВО ВСЯКОМ СЛУЧАЕ РАМИНЕСА ЗДЕСЬ УЖЕ НЕТ.

ДАВАЙТЕ, НАКОНЕЦ, НАЧНЁМ ПРОЕКТИРОВАТЬ БАЗУ ДАННЫХ.
Стоп
ЖИМ ЖИМ
МИНУТОЧКУ!

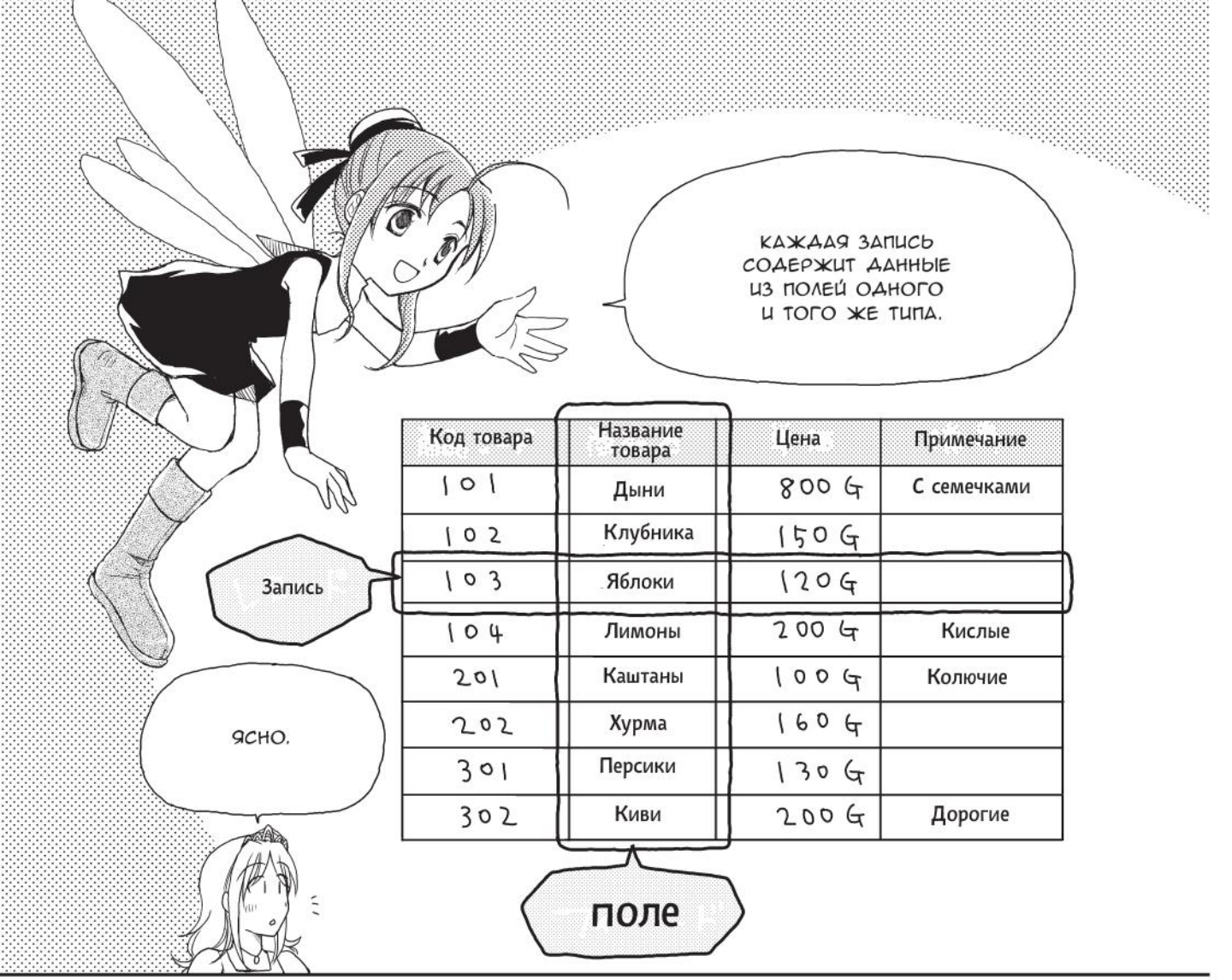
РАЗВЕ МОЖНО ПРОЕКТИРОВАТЬ БАЗУ ДАННЫХ, НЕ ИМЕЯ НИКАКИХ ЗНАНИЙ?
Хорошая основа — это важно.
ДА...
Надо постараться.
У-у-у...
ПРЕЖДЕ ВСЕГО ДАВАЙТЕ ВЫУЧИМ ТЕРМИНЫ.


НИКУДА НЕ УХОДИТЕ.
ПРЫГ!

ВО ДАЁТ!
НЫРЯЕТ ВНУТРЬ



Код товара	Товар	Цена	Примечание
101	Дыни	800 G	С семечками
102	Клубника	150 G	
103	Яблоки	120 G	
104	Лимоны	200 G	
201	Каштаны	100 G	
202	Хурма	160 G	

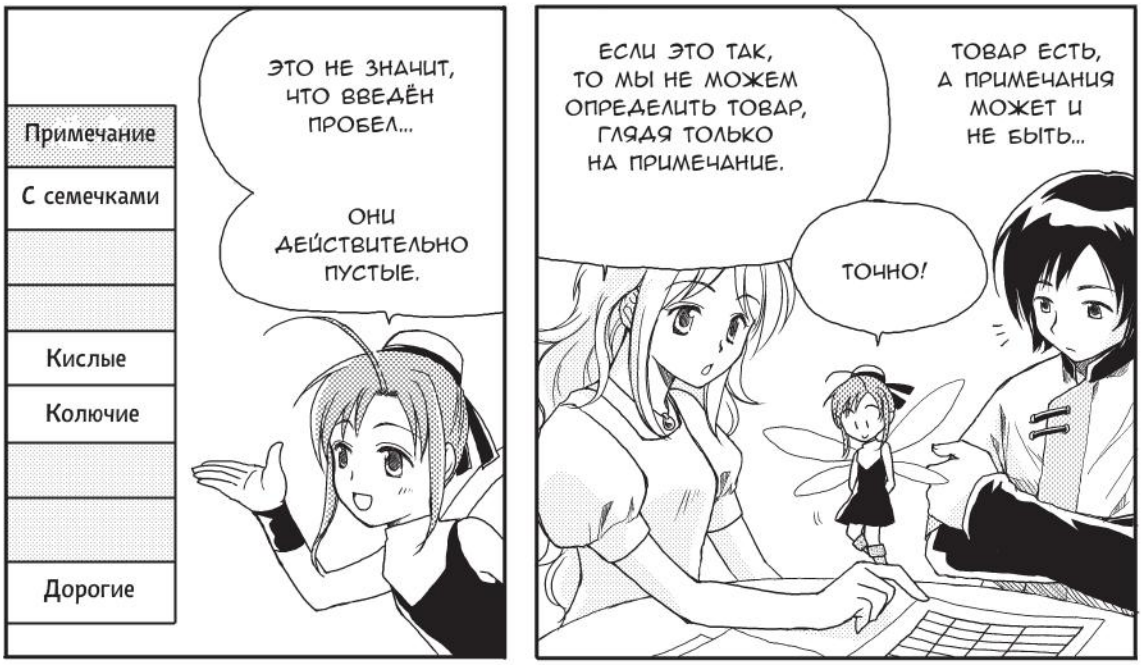


Код товара	Название товара	Цена	Примечание
101	Дыни	800 G	С семечками
102	Клубника	150 G	
103	Яблоки	120 G	
104	Лимоны	200 G	Кислые
201	Каштаны	100 G	Колючие
202	Хурма	160 G	
301	Персики	130 G	
302	Киви	200 G	Дорогие

ОДИНАКОВЫХ КОДОВ БЫТЬ НЕ МОЖЕТ.
НУ, ЭТО ОЧЕВИДНО!
И НЕ БЫВАЕТ ДУБЛИРУЮЩИХ ЗАПИСЕЙ. ПОЭТОМУ, ЕСЛИ ВЫ ЗНАЕТЕ, ЧТО КОД ТОВАРА 101,
103
102
101
ВЫ МОЖЕТЕ ОПРЕДЕЛИТЬ, ЧТО ЭТО ДЫНИ.
ЛОГИЧНО.
Я поняла,
НО ДЛЯ ЦЕНЫ ЭТО НЕ ТАК.
ХЛОП!
Цена
800 G
150 G
120 G
200 G
ЧТО ВЫ МОЖЕТЕ СКАЗАТЬ, ГЛЯДЯ НА ЦЕНУ?
НА ЦЕНУ?
О, Я НАШЛА ОДИНАКОВЫЕ ЧИСЛА.
Цена
С семечк
00 G
150 G
120 G
200 G
100 G
160 G
130 G
200 G
Кислые
Колючие
Дорогие
103 Лимоны
104 Каштаны
201 Хурма
202 Персики
301 Киви
302
ДАЖЕ ЕСЛИ ТЫ ЗНАЕШЬ, ЧТО ЦЕНА 200 G...
БЕЗ ДРУГИХ ДАННЫХ ТЕБЕ НЕИЗВЕСТНО, К ЛИМОНУ ОНА ОТНОСИТСЯ ИЛИ К КИВИ.
?
200 G
200 G

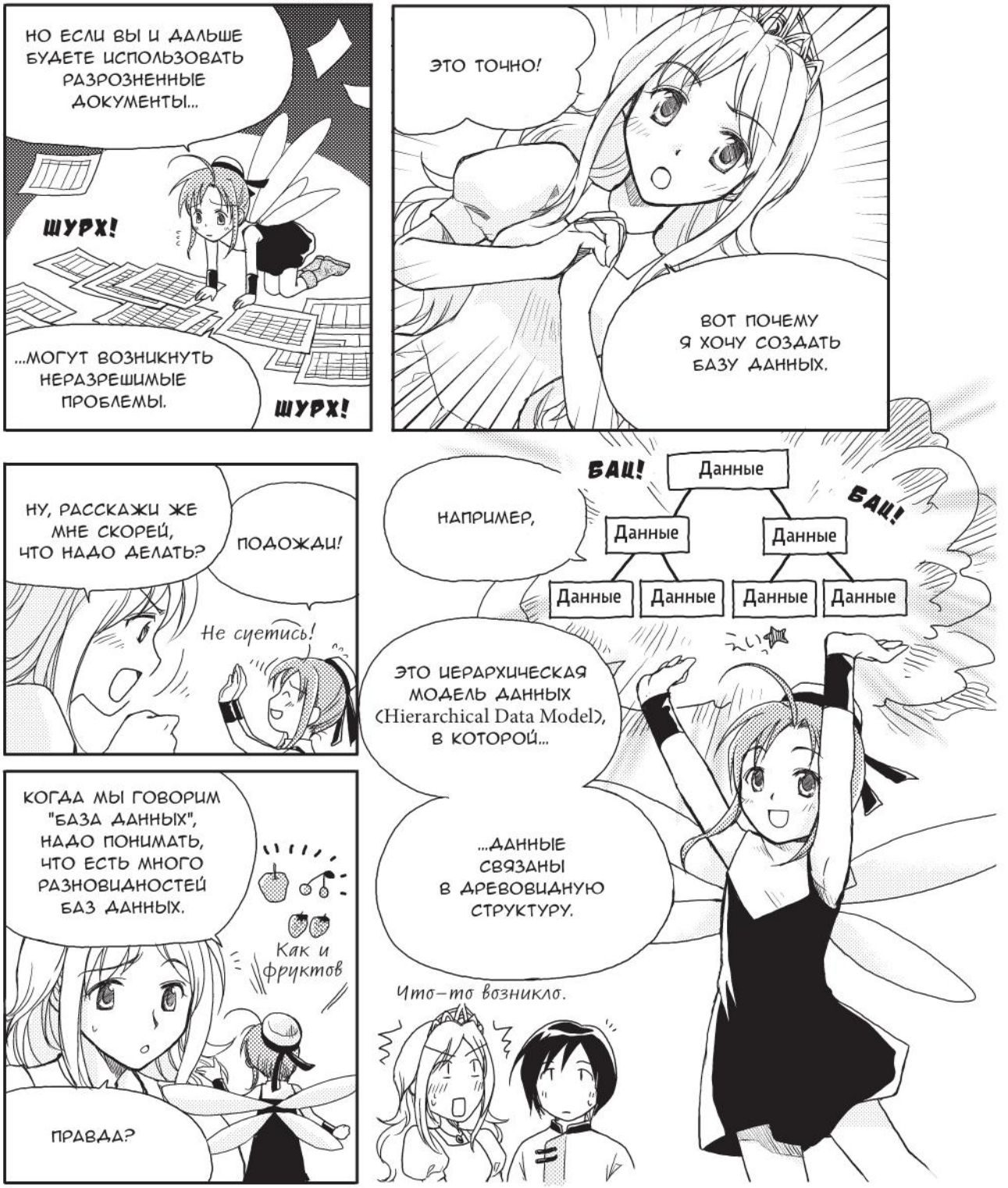
Сотрудник	Примечание
Руруна	Блондинка деятельная
Кэйн	Брюнет уравновешенный

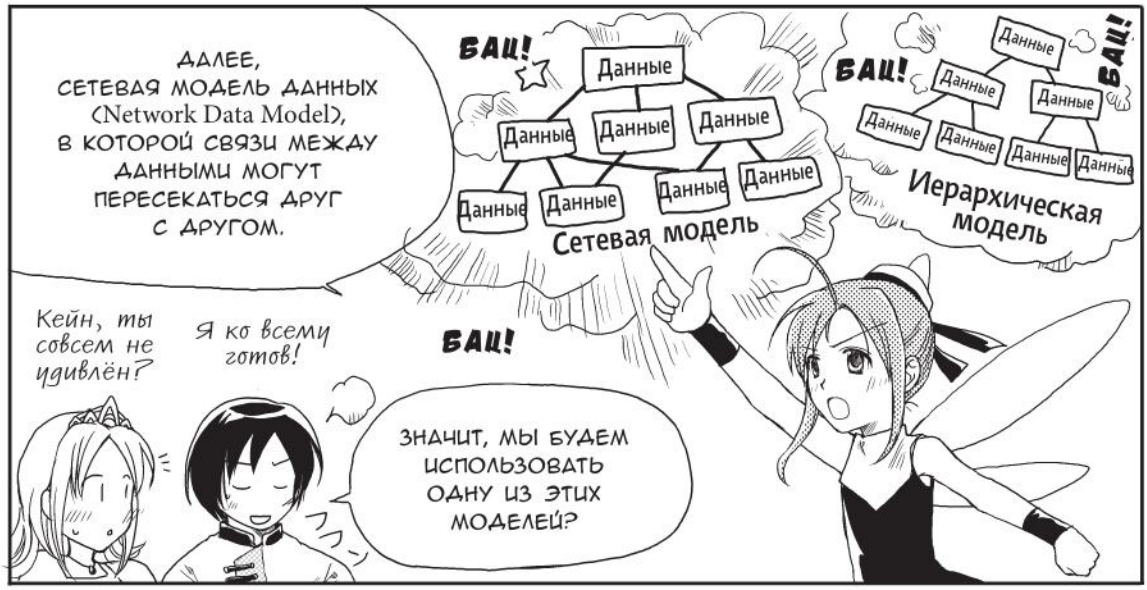
	Примечание
G	с семечками
0G	
120G	
200G	Кислые
100G	Колючие

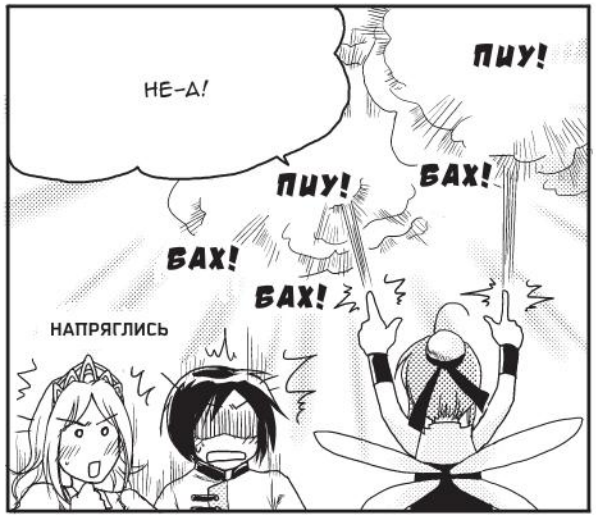

Примечание
С семечками
Кислые
Колючие
Дорогие
ЭТО НЕ ЗНАЧИТ, ЧТО ВВЕДЁН ПРОБЕЛ...
ОНИ ДЕЙСТВИТЕЛЬНО ПУСТЫЕ.
ЕСЛИ ЭТО ТАК, ТО МЫ НЕ МОЖЕМ ОПРЕДЕЛИТЬ ТОВАР, ГЛЯДЯ ТОЛЬКО НА ПРИМЕЧАНИЕ.
ТОВАР ЕСТЬ, А ПРИМЕЧАНИЯ МОЖЕТ И НЕ БЫТЬ...
ТОЧНО!

Не может быть NULL.
Код товара | Название товара | Цена | Примечание
101 | Дыни | 800 G | С семечками
102 | Клубника | 150 G |
103 | Яблоки | 120 G |
104 | Лимоны | 200 G | Кислые
201 | Каштаны | 100 G | Колючие
202 | Хурма | 160 G |
301 | Персики | 130 G |
302 | Киви | 200 G |
NULL
В МИРЕ БАЗ ДАННЫХ ОТСУТСТВИЕ ЗНАЧЕНИЯ НАЗЫВАЕТСЯ NULL.
ЗНАЧЕНИЕ "NULL" ПРИМЕНИМО ДЛЯ ПОЛЯ "ПРИМЕЧАНИЕ", НО НИКОГДА — ДЛЯ ПОЛЯ "КОД ТОВАРА", КОТОРОЕ ОПРЕДЕЛЯЕТ ДАННЫЕ.

ВОТ И ВСЁ ПРО ТЕРМИНЫ БАЗ ДАННЫХ.
ВАМ ВСЁ ПОНЯТНО?
АГА...
В общих чертах...
NULL? Пустой? Уникальный?
БОРМОТАНИЕ...
БОРМОТАНИЕ...


НО ЕСЛИ ВЫ И ДАЛЬШЕ БУДЕТЕ ИСПОЛЬЗОВАТЬ РАЗРОЗНЕННЫЕ ДОКУМЕНТЫ...
ШУРХ!
...МОГУТ ВОЗНИКНУТЬ НЕРАЗРЕШИМЫЕ ПРОБЛЕМЫ.
ШУРХ!
ЭТО ТОЧНО!
ВОТ ПОЧЕМУ Я ХОЧУ СОЗДАТЬ БАЗУ ДАННЫХ.
НУ, РАССКАЖИ ЖЕ МНЕ СКОРЕЙ, ЧТО НАДО ДЕЛАТЬ?
ПОДОЖДИ!
Не суетись!
КОГДА МЫ ГОВОРИМ "БАЗА ДАННЫХ", НАДО ПОНИМАТЬ, ЧТО ЕСТЬ МНОГО РАЗНОВИДНОСТЕЙ БАЗ ДАННЫХ.
Как и фруктов
ПРАВДА?
НАПРИМЕР,
БАЦ!
БАЦ!
Данные
Данные
Данные
Данные
Данные
Данные
Данные
ЭТО ИЕРАРХИЧЕСКАЯ МОДЕЛЬ ДАННЫХ (Hierarchical Data Model), В КОТОРОЙ...
...ДАННЫЕ СВЯЗАНЫ В ДРЕВОВИДНУЮ СТРУКТУРУ.
Что-то возникло.

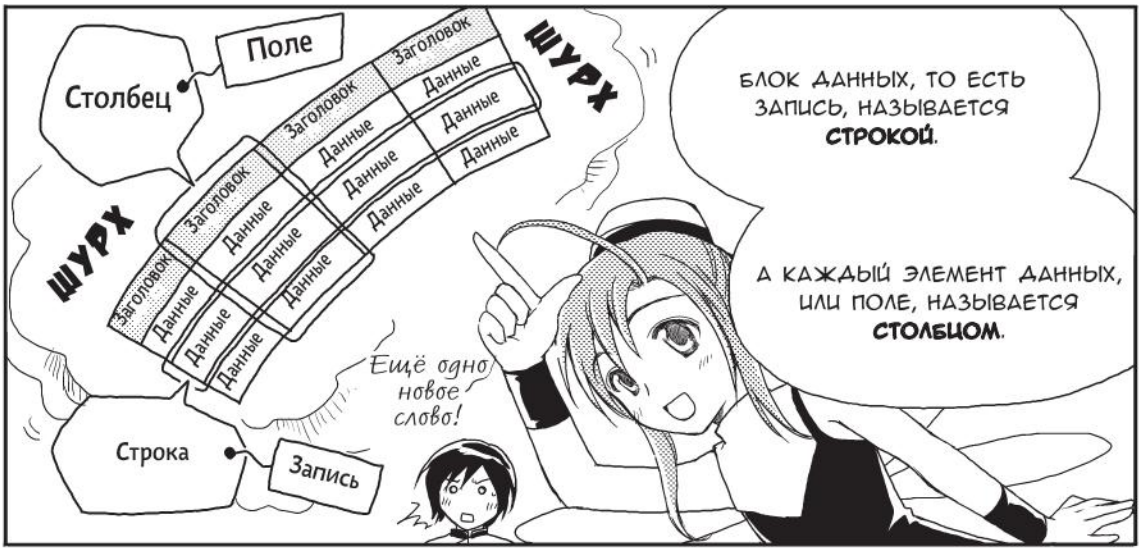

ДАЛЕЕ, СЕТЕВАЯ МОДЕЛЬ ДАННЫХ (Network Data Model), В КОТОРОЙ СВЯЗИ МЕЖДУ ДАННЫМИ МОГУТ ПЕРЕСЕКАТЬСЯ ДРУГ С ДРУГОМ.
БАЦ!
Данные
Сетевая модель
Иерархическая модель
Кейн, ты совсем не удивлён?
Я ко всему готов!
ЗНАЧИТ, МЫ БУДЕМ ИСПОЛЬЗОВАТЬ ОДНУ ИЗ ЭТИХ МОДЕЛЕЙ?


НЕ-А!
ПИУ!
ПИУ!
БАХ!
БАХ!
БАХ!
НАПРЯГЛИСЬ

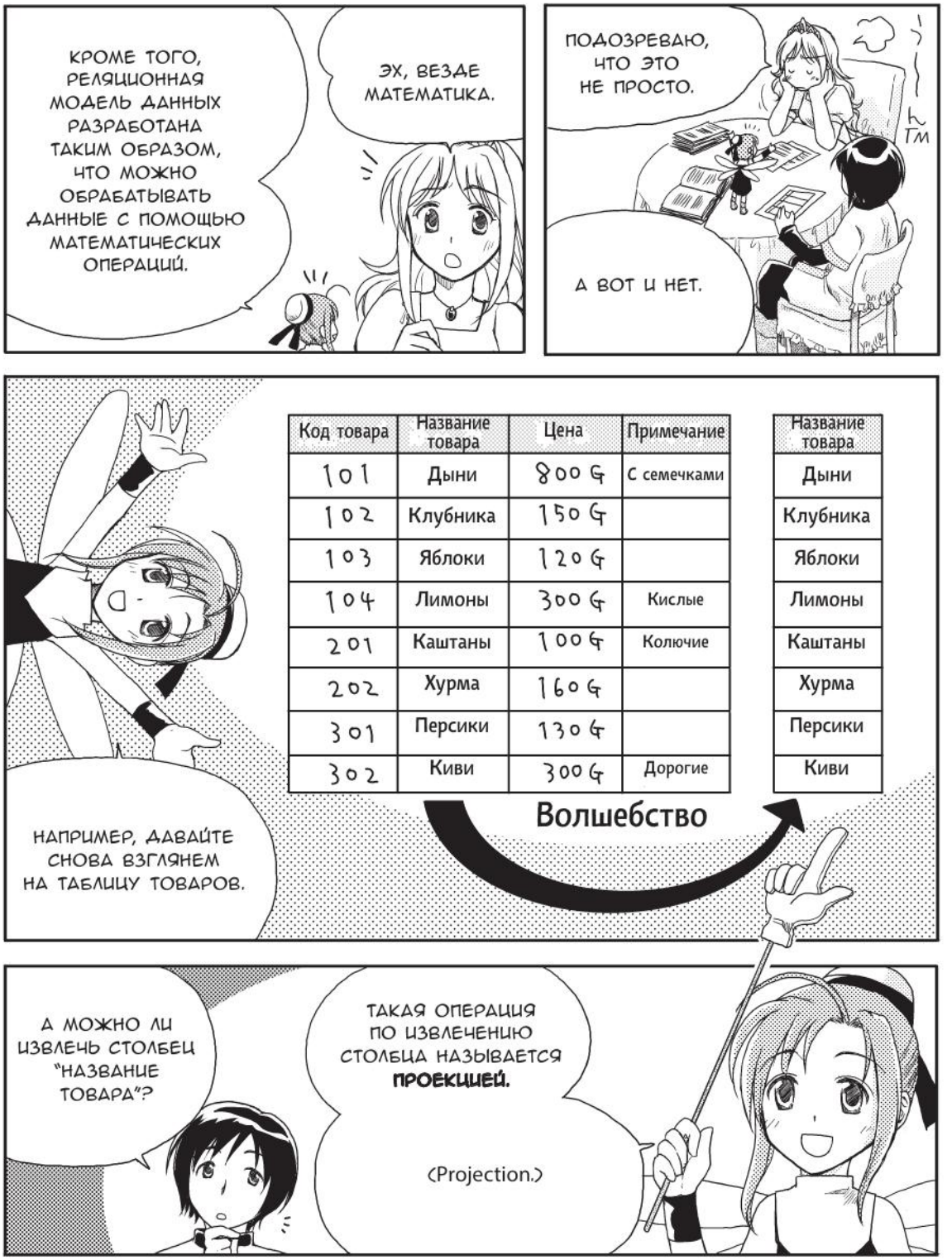
МЕЖДУ ПРОЧИМ, ЕСТЬ ЕЩЁ ОДНА РАЗНОВИДНОСТЬ, ГОРАЗДО БОЛЕЕ ПРОСТАЯ В ИСПОЛЬЗОВАНИИ, ЧЕМ ЭТИ ДВЕ.
Ага, испугался?
Вовсе нет.
ОНА НАЗЫВАЕТСЯ...

РЕЛЯЦИОННАЯ МОДЕЛЬ ДАННЫХ (Relational Data Model).
РЕЛЯЦИОННАЯ? ЧТО ЭТО?
ЧПОК!

РЕЛЯЦИОННЫЕ БАЗЫ_ДАННЫХ

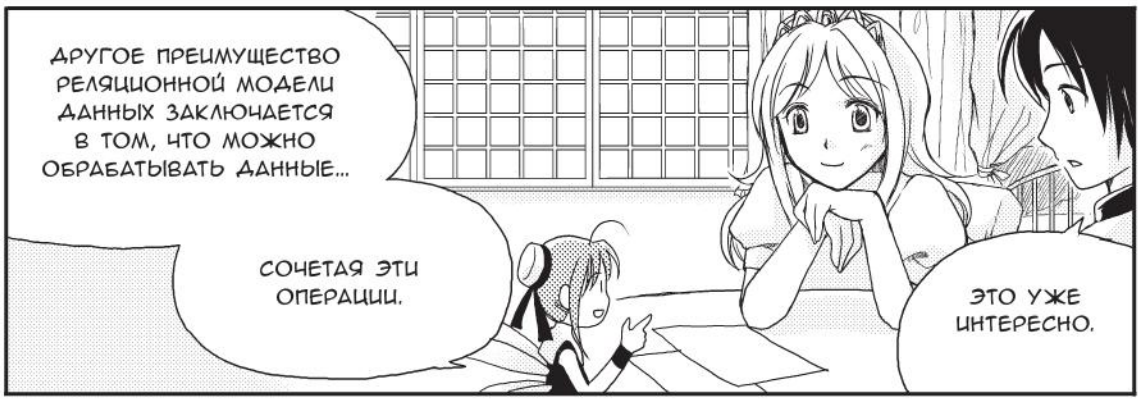


Ключ
Одно поле играет в базе данных особую роль.
Такое особое поле называется **ключом**.
Ключ — это особое поле?
Да, например...
...поле — код товара в документе, который мы недавно смотрели.
Это поле выполняет важную роль: указывает на коды товаров.
Первичный ключ
Код товара
101
102
103
104
201
202
301
302
Этот код товара называется **первичным ключом**.
Я и не думал, что будет так много терминов.
Ну, с таблицами я постоянно имею дело.
Всё просто, если умеешь работать с данными с помощью таблиц.
Это уже один плюс в пользу реляционной модели данных.
Обрабатывать данные могут даже те, кто не очень много знает о базах данных.


КРОМЕ ТОГО, РЕЛЯЦИОННАЯ МОДЕЛЬ ДАННЫХ РАЗРАБОТАНА ТАКИМ ОБРАЗОМ, ЧТО МОЖНО ОБРАБАТЫВАТЬ ДАННЫЕ С ПОМОЩЬЮ МАТЕМАТИЧЕСКИХ ОПЕРАЦИЙ.
ЭХ, ВЕЗДЕ МАТЕМАТИКА.
ПОДОЗРЕВАЮ, ЧТО ЭТО НЕ ПРОСТО.
А ВОТ И НЕТ.
НАПРИМЕР, ДАВАЙТЕ СНОВА ВЗГЛЯНЕМ НА ТАБЛИЦУ ТОВАРОВ.
Волшебство
Код товара | Название товара | Цена | Примечание
101 | Дыни | 800 G | С семечками
102 | Клубника | 150 G |
103 | Яблоки | 120 G |
104 | Лимоны | 300 G | Кислые
201 | Каштаны | 100 G | Колючие
202 | Хурма | 160 G |
301 | Персики | 130 G |
302 | Киви | 300 G | Дорогие
Название товара
Дыни
Клубника
Яблоки
Лимоны
Каштаны
Хурма
Персики
Киви
А МОЖНО ЛИ ИЗВЛЕЧЬ СТОЛБЕЦ "НАЗВАНИЕ ТОВАРА"?
ТАКАЯ ОПЕРАЦИЯ ПО ИЗВЛЕЧЕНИЮ СТОЛБЦА НАЗЫВАЕТСЯ ПРОЕКЦИЕЙ.
(Projection.)

ЗНАЧИТ, ИЗВЛЕЧЕНИЕ СТОЛБЦА "НАЗВАНИЕ ТОВАРА" — ЭТО ПРОЕКЦИЯ?
ДА, НИЧЕГО СЛОЖНОГО, НЕ ПУГАЙТЕСЬ.

ЕСТЬ ЕЩЁ НЕСКОЛЬКО ДРУГИХ ОПЕРАЦИЙ. ВСЕГО ИХ ВОСЕМЬ.
8 операций!
Объединение (UNION)
Вычитание (DIFFERENCE) (в некоторых источниках разность)
Декартово произведение (CARTESIAN PRODUCT)
Деление (DIVISION)
Проекция (PROJECTION)
Пересечение (INTER-SECTION)
Выборка (SELECT)
Соединение (JOIN)
ТАК МНОГО! А ГОВОРИЛА НЕ ПУГАЙТЕСЬ.

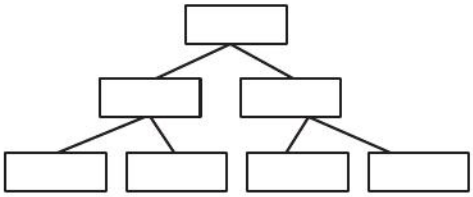

ДРУГОЕ ПРЕИМУЩЕСТВО РЕЛЯЦИОННОЙ МОДЕЛИ ДАННЫХ ЗАКЛЮЧАЕТСЯ В ТОМ, ЧТО МОЖНО ОБРАБАТЫВАТЬ ДАННЫЕ...
СОЧЕТАЯ ЭТИ ОПЕРАЦИИ.
ЭТО УЖЕ ИНТЕРЕСНО.

ИТАК, ДЛЯ КОРОЛЕВСТВА ЯМС МЫ СОЗДАДИМ РЕЛЯЦИОННУЮ БАЗУ ДАННЫХ, ВЕРНО?
Дай пять!
В ТОЧКУ!
ТОП, ТОП, ТОП... БАХ!
РАМИНЕС У ВАС?!
КТО ЭТО ?
?
?
ПРИНЦ РАМИНЕС УШЁЛ СОВСЕМ НЕДАВНО...
...ВОТ ТЕ НА!
ИЗВИНИТЕ.
Хи-хи-хи
Раминес!
БАХ!
СКОЛЬКО ЖЕ ЖЕНЩИН ОН?..
ха-ха
Обалдеть

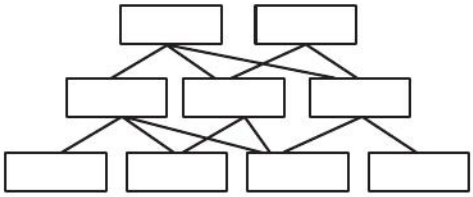
КАКИЕ БЫВАЮТ МОДЕЛИ ДАННЫХ

Когда вы говорите «база данных» (сокращённо БД), какую базу вы имеете в виду? Существует много видов БД, подходящих для управления данными. Идентификация данных и операционные методы, которые используются в базе, называются её моделью данных. Наиболее распростанены три модели данных.

Как я уже рассказывала Руруне и Кейну, первый тип — это иерархическая модель данных. В этой модели данных элемент-«потомок» имеет только одного «предка». Второй тип — это сетевая модель данных. В отличие от иерархической, в сетевой модели «потомок» может иметь сколько угодно «предков».



Иерархическая модель данных



Сетевая модель данных

Чтобы использовать одну из этих моделей, вы должны управлять данными, держа в уме физическое расположение и порядок данных. Следовательно, при использовании иерархической или сетевой модели данных сложно выполнять гибкий и быстрый поиск данных.

Третий тип модели — реляционная модель данных. Реляционная БД обрабатывает данные с помощью таблицы — простой для понимания концепции. Давайте обсудим эту модель подробнее.

■ РЕЛЯЦИОННАЯ МОДЕЛЬ ДАННЫХ

ОПЕРАЦИИ ИЗВЛЕЧЕНИЯ ДАННЫХ

Как извлекать данные в реляционной БД? Обрабатывать и извлекать данные в реляционной БД можно, выполняя строго определённые математические операции. Существует восемь основных операций, и они разделяются на две категории — **теоретико-множественные** и **специальные реляционные**.

ТЕОРЕТИКО-МНОЖЕСТВЕННЫЕ ОПЕРАЦИИ

Теоретико-множественными операциями называют операции **объединения** (union), **вычитания** (difference), **пересечения** (intersection) и **декартова произведения** (Cartesian product). Эти операции работают с одной или несколькими группами строк для создания новой группы строк. В общем, они определяют, какие строки из входных данных появятся в выходных данных. Давайте посмотрим на примере таблиц товаров 1 и 2.

■ ТОВАРЫ 1

Название товара	Цена
Дыни	800 G
Клубника	150 G
Яблоки	120 G
Лимоны	200 G

■ ТОВАРЫ 2

Название товара	Цена
Дыни	800 G
Клубника	150 G
Каштаны	200 G
Хурма	350 G

ОБЪЕДИНЕНИЕ (UNION)

Операция **Объединение** (UNION) позволяет выбрать все товары, входящие в таблицу товаров 1 и таблицу товаров 2. В результате получается таблица, приведённая ниже.

Название товара	Цена
Дыни	800 G
Клубника	150 G
Яблоки	120 G
Лимоны	200 G
Каштаны	200 G
Хурма	350 G

Операция объединение извлекает все строки из обеих таблиц и комбинирует их. На нижнем рисунке показано, как выглядят данные из двух таблиц после выполнения операции UNION. Все строки таблиц товаров 1 и 2 были извлечены.

ВЫЧИТАНИЕ (DIFFERENCE)

Операция **Вычитание** (DIFFERENCE) извлекает строки только из одной таблицы. Например, с помощью этой операции можно извлечь все товары из первой таблицы, которые не входят во вторую таблицу. Результаты зависят от того, в какой таблице находятся извлекаемые строки и в какой таблице находятся исключаемые строки.

Название товара	Цена
Яблоки	120 G
Лимоны	200 G

Название товара	Цена
Каштаны	200 G
Хурма	350 G

ПЕРЕСЕЧЕНИЕ (INTERSECTION)

Можно также извлекать товары, которые содержатся в обеих таблицах, — Товары 1 и Товары 2. Эта операция называется **Пересечение** (INTERSECTION). Ниже приведён результат пересечения таблиц 1 и 2.

Название товара	Цена
Дыни	800 G
Клубника	150 G

ДЕКАРТОВО ПРОИЗВЕДЕНИЕ (CARTESIAN PRODUCT)

Операция CARTESIAN PRODUCT комбинирует все строки двух таблиц. В этом примере мы получаем 3 × 3 = 9 строк. Заметим, что, в отличие от предыдущих примеров, названия столбцов (или полей) в этих двух таблицах не одинаковы.

■ ТОВАРЫ

Код товара	Название товара	Цена
101	Дыни	800 G
102	Клубника	150 G
103	Яблоки	120 G

3 строки

■ ПОКУПАТЕЛИ

Код покупателя	Покупатель
12	Королевство Моро
23	Империя Фарайя
25	Королевство Рамбутан

3 строки

■ ДЕКАРТОВО ПРОИЗВЕДЕНИЕ

Код товара	Название товара	Цена	Код покупателя	Покупатель
101	Дыни	800 G	12	Королевств Моро
101	Дыни	800 G	23	Империя Фарайя
101	Дыни	800 G	25	Королевство Рамбутан
102	Клубника	150 G	12	Королевство Моро
102	Клубника	150 G	23	Империя Фарайя
102	Клубника	150 G	25	Королевство Рамбутан
103	Яблоки	120 G	12	Королевство Моро
103	Яблоки	120 G	23	Империя Фарайя
103	Яблоки	120 G	25	Королевство Рамбутан

3 × 3 = 9 строк

СПЕЦИАЛЬНЫЕ РЕЛЯЦИОННЫЕ ОПЕРАЦИИ

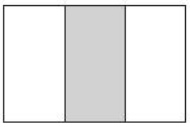
Как было сказано выше, реляционная база данных выстроена так, что данные можно извлекать теоретико-множественными операциями и специальными реляционными операциями. Давайте посмотрим на оставшиеся четыре операции, характерные для реляционных БД. Это **специальные реляционные операции** — **Проекция** (Projection), **Выборка** (Selection), **Соединение** (Join) и **Деление** (Division).

ПРОЕКЦИЯ (PROJECTION)

Операция **Проекция** (PROJECTION) извлекает столбцы из таблицы. В приведённом ниже примере эта оперция используется для извлечения только названий товаров, входящих в таблицу товаров.

Название товара
Дыни
Клубника
Яблоки
Лимоны

Представьте себе, что проекция — это извлечение по вертикали, как показано ниже.

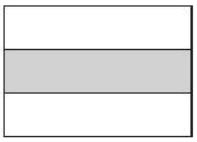


ВЫБОРКА (SELECTION)

Операция **Выборка** (SELECTION) извлекает две строки из таблицы.

Название товара	Цена
Дыни	800 G
Клубника	150 G

Выборка — это как проекция, но она извлекает строки, а не столбцы. При выборке данные извлекаются по горизонтали.



СОЕДИНЕНИЕ (JOIN)

Операция **Соединение** (JOIN) обладает широкими возможностями. Эта операция относится к работе по соединению таблиц. В качестве примера давайте взглянем на приведённые ниже таблицы.

■ ТОВАРЫ

Код товара	Название товара	Цена
101	Дыни	800 G
102	Клубника	150 G
103	Яблоки	120 G
104	Лимоны	200 G

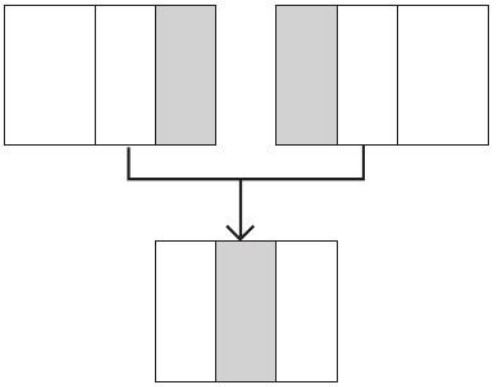
■ ПРОДАЖИ

Дата	Код товара	Количество
11/1	102	1100
11/1	101	300
11/5	103	1700
11/8	101	500

В этих двух таблицах столбцы «Код товара» дают нам информацию, о каком товаре идёт речь. Первого ноября было продано 1100 кг клубники (код товара 102). Таблица «Продажи» не содержит информации о названии товара, но в ней есть столбец с кодом товара. Другими словами, благодаря ссылке на код товара таблица «Продажи» позволяет нам понять, какой товар был продан. В данном случае «Код товара» — это первичный ключ в таблице «Товары», и внешний ключ в таблице «Продажи». В результате, объединяя две таблицы таким образом, чтобы внешний ключ ссылался на первичный ключ, получаем следующую таблицу.

Дата	Код товара	Название товара	Цена	Количество
11/1	102	Клубника	150 G	1100
11/1	101	Дыни	800 G	300
11/5	103	Яблоки	120 G	1700
11/8	101	Дыни	800 G	500

Таким образом мы получаем новую динамическую таблицу данных о продажах, куда входят столбцы: «дата», «код товара», «название товара», «цена» и «количество». На рисунке внизу показано это соединение, где закрашенная область обозначает столбец, который имеется в обеих первоначальных таблицах.



ДЕЛЕНИЕ (DIVISION)

И, наконец, давайте рассмотрим операцию **Деление** (DIVISION). Эта операция извлекает строки, где значения в столбцах совпадают со значениями в столбцах во второй таблице, но при этом возвращает столбцы, которых нет во второй таблице.

■ ПРОДАЖИ

Код покупателя	Покупатель	Дата
12	Королевство Моро	3/5
12	Королевство Моро	3/10
23	Империя Фарайя	3/5
25	Королевство Рамбутан	3/21
30	Королевство Кивано	3/25

■ ПОКУПАТЕЛИ

Код покупателя	Покупатель
12	Королевство Моро
23	Империя Фарайя

В результате деления таблицы «Продажи» на таблицу «Покупатели» получаем таблицу, которая позволяет нам найти дату отправки фруктов в оба направления: как в империю Фарайя, так и в королевство Моро.

Дата
3/5

ВОПРОСЫ И ЗАДАНИЯ

Теперь давайте ответим на некоторые вопросы, чтобы понять, хорошо ли вы поняли реляционные БД. Ответы смотрите на стр. 48.

В1

Как в реляционной базе данных называется ключ, дающий ссылку на столбец в другой таблице?

В2

Следующая таблица отображает информацию о книгах. Какой элемент можно использовать в качестве первичного ключа? ISBN — это международный стандартный книжный номер, уникальный идентификационный номер, присваиваемый каждой публикуемой книге.

ISBN	Название книги	Автор	Дата публикации	Цена

В3

Как называется операция, с помощью которой в данном примере извлекают данные?

Код покупателя	Покупатель
12	Королевство Моро
23	Империя Фарайя
25	Королевство Рамбутан
30	Королевство Кивано

→

Код покупателя	Покупатель
25	Королевство Рамбутан

B4

Как называется операция, с помощью которой в данном примере извлекают данные?

Код покупателя	Покупатель
12	Королевство Моро
23	Империя Фарайя
25	Королевство Рамбутан
32	Королевство Кивано

Код покупателя	Покупатель
15	Королевство Дуриан
22	Королевство Кумкват
31	Королевство Куруба
33	Королевство Черимойя

Код покупателя	Покупатель
12	Королевство Моро
15	Королевство Дуриан
22	Королевство Кумкват
23	Империя Фарайя
25	Королевство Рамбутан
31	Королевство Куруба
32	Королевство Кивано
33	Королевство Черимойя

B5

Как называется операция, с помощью который в данном примере извлекают данные?

Код покупателя	Покупатель
12	Королевство Моро
23	Империя Фарайя
25	Королевство Рамбутан
32	Королевство Кивано

Код покупателя	Дата
12	3/1
23	3/1
12	3/3
32	3/5
12	3/6
25	3/10

Код покупателя	Дата	Покупатель
12	3/1	Королевство Моро
23	3/1	Империя Фарайя
12	3/3	Королевство Моро
32	3/5	Королевство Кивано
12	3/6	Королевство Моро
25	3/10	Королевство Рамбутан

ДА ЗДРАВСТВУЕТ РЕЛЯЦИОННАЯ БАЗА ДАННЫХ!

В реляционной БД для извлечения данных можно использовать восемь различных операций. Извлечённые данные представляют в виде таблицы. Если комбинировать операции, о которых рассказывалось в этой главе, то можно извлекать данные с любыми целями. Например, используя название и цену товара, можно получить сводные данные по валовой выручке от продажи. Реляционные БД пользуются популярностью потому, что в них легко разобраться, и они обеспечивают гибкую обработку данных.

ИТОГИ

- Одна строка данных называется **записью**, а каждый столбец называется **полем**.
- Столбец, с помощью которого можно определить данные, называется **первичным ключом**.
- В реляционной БД для обработки данных используются таблицы.
- В реляционной БД можно обрабатывать данные с помощью математических операций.

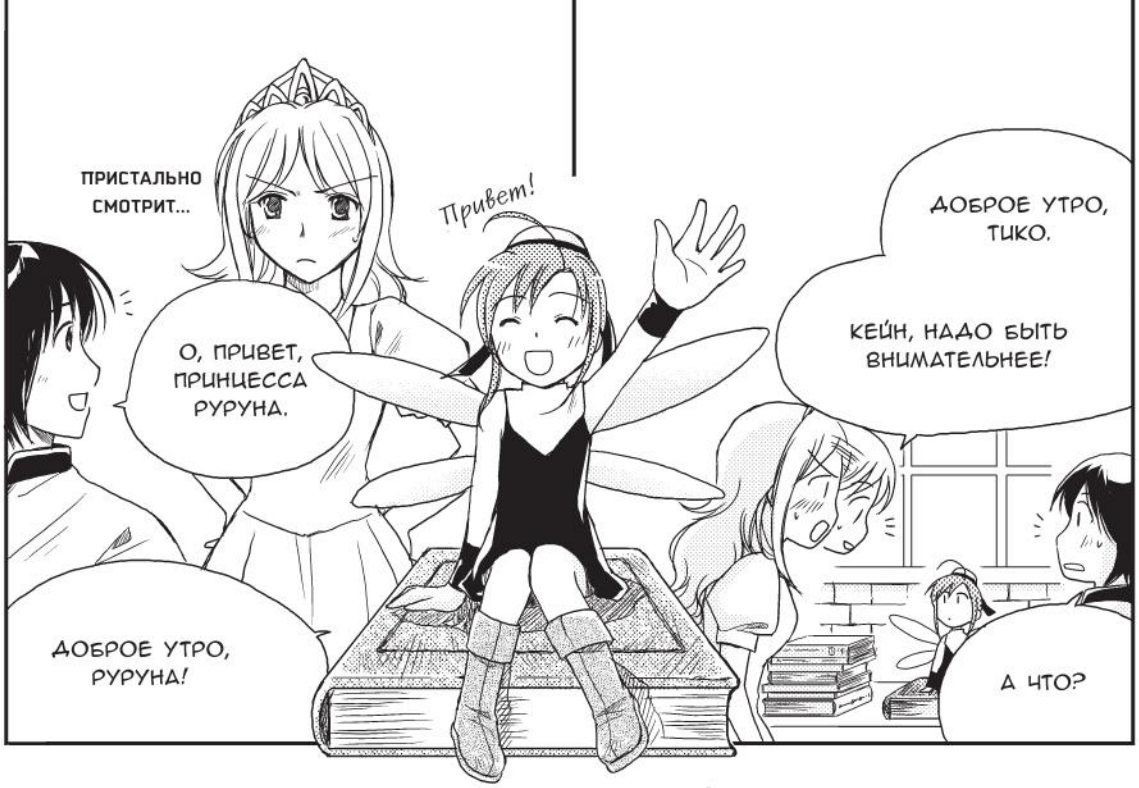
ОТВЕТЫ

01	Внешний ключ
02	ISBN
03	Выборка
04	Объединение
05	Соединение

3.

ДАВАЙТЕ СПРОЕКТИРУЕМ БАЗУ ДАННЫХ!

МОДЕЛЬ СУЩНОСТЬ—СВЯЗЬ (E-R-МОДЕЛЬ)


ПРИСТАЛЬНО СМОТРИТ...
Привет!
ДОБРОЕ УТРО, ТИКО.
О, ПРИВЕТ, ПРИНЦЕССА РУРУНА.
КЕЙН, НАДО БЫТЬ ВНИМАТЕЛЬНЕЕ!
ДОБРОЕ УТРО, РУРУНА!
А ЧТО?

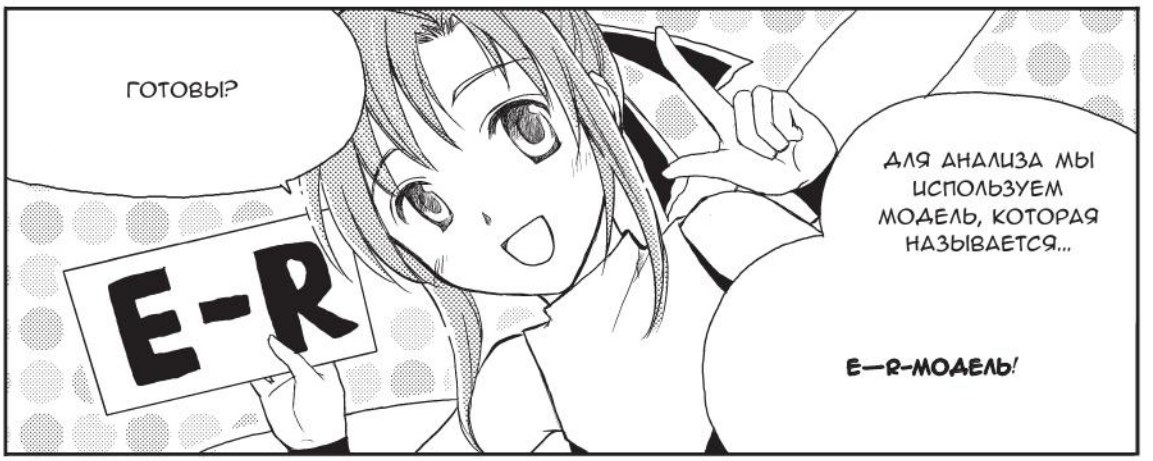
ТЫ ЧТО, ЗАБЫЛ? НИКТО НЕ ВИДИТ ТИКО, КРОМЕ НАС С ТОБОЙ.
А ТО БУДЕШЬ ГЛУПО ВЫГЛЯДЕТЬ, ЕСЛИ СТАНЕШЬ С НЕЙ РАЗГОВАРИВАТЬ ПРИ ПОСТОРОННИХ.
За сумасшедшего сойдёшь
ДА НУ!

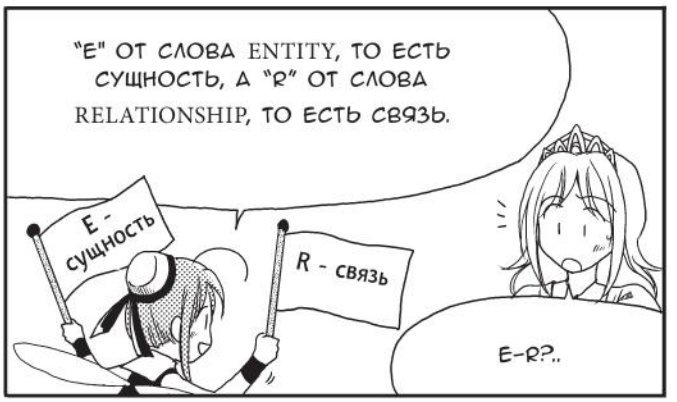
ПРОСТИ, ТЫ ПРАВА.
Да
О ЧЁМ ВЫ ТУТ БЕЗ МЕНЯ РАЗГОВАРИВАЛИ?

Я С ТРУДОМ ПРЕДСТАВЛЯЮ, С ЧЕГО НАДО НАЧИНАТЬ, СОЗДАВАЯ БАЗУ ДАННЫХ,
И Я ОБРАТИЛСЯ К ТИКО ЗА СОВЕТОМ.
Молодец!
ТЫ, НВЕРНОЕ, РАБОТАЛ ВСЁ УТРО.
И?..

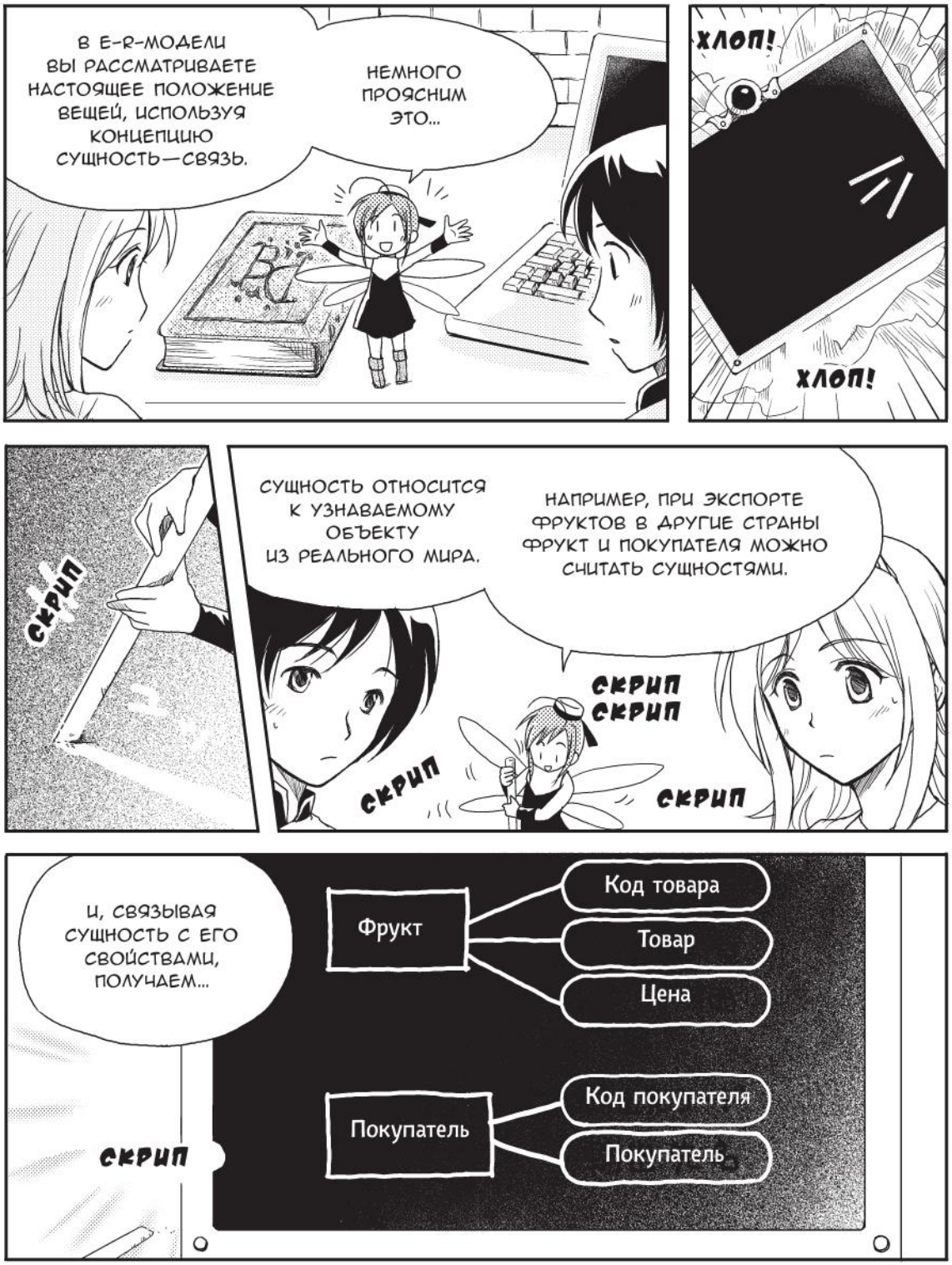
В ПЕРВУЮ ОЧЕРЕДЬ МЫ РЕШИЛИ, ЧТО ЛЕГКО МОГЛИ БЫ ПОСТРОИТЬ БАЗУ ДАННЫХ...
Легко сказать...
Ямс
...ЕСЛИ БЫ ПОНЯЛИ РЕАЛЬНОЕ СОСТОЯНИЕ ДЕЛ В КОРОЛЕВСТВЕ... НУ, ПОКА ЭТО ВСЁ.

ТИКО ГОВОРИТ, ЧТО ХОРОШО БЫ СОЗДАТЬ МОДЕЛЬ И НА ЕЁ ОСНОВЕ ПРОАНАЛИЗИРОВАТЬ, КАК ИДЁТ УПРАВЛЕНИЕ ЭКСПОРТОМ ТОВАРОВ В НАСТОЯЩИЙ МОМЕНТ.
Довольно долгий урок
Спасибо
Присаживайся
НУ, ЧТО ЖЕ, ПРИДЁТСЯ КОЕ-ЧЕМУ НАУЧИТЬСЯ.


ГОТОВЫ?
E-R
ДЛЯ АНАЛИЗА МЫ ИСПОЛЬЗУЕМ МОДЕЛЬ, КОТОРАЯ НАЗЫВАЕТСЯ...
E—R-МОДЕЛЬ!


"E" ОТ СЛОВА ENTITY, ТО ЕСТЬ СУЩНОСТЬ, А "R" ОТ СЛОВА RELATIONSHIP, ТО ЕСТЬ СВЯЗЬ.
E - сущность
R - связь
E-R?..

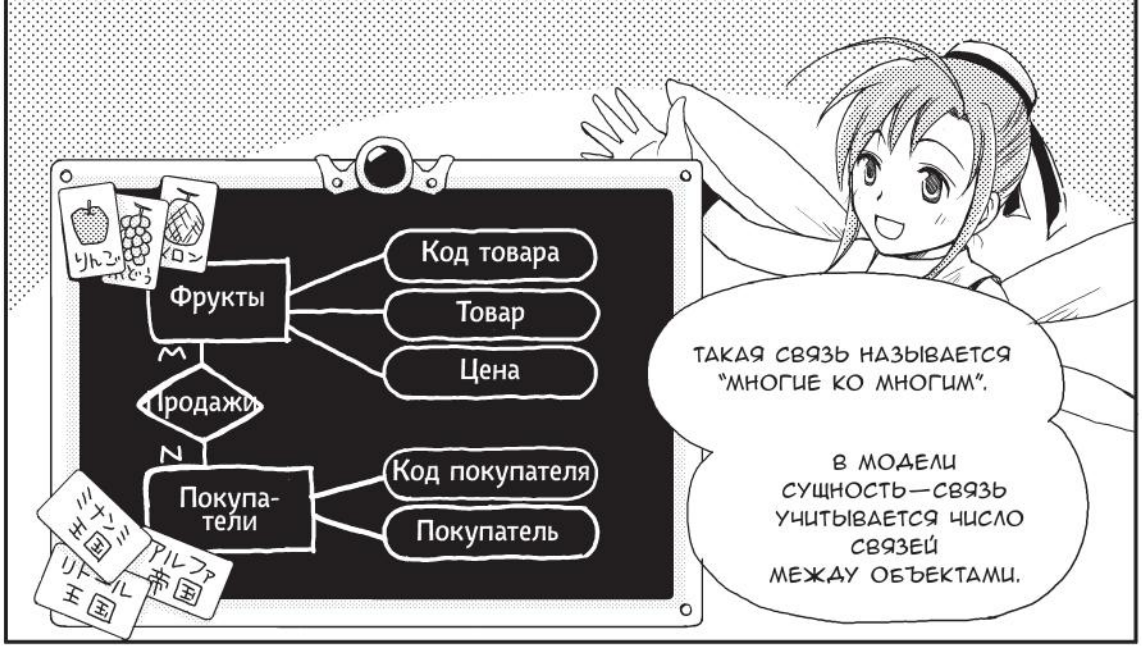
СУЩНОСТЬ И СВЯЗЬ...
?
?
И мне тоже.
МНЕ НЕЗНАКОМЫ ЭТИ ТЕРМИНЫ...

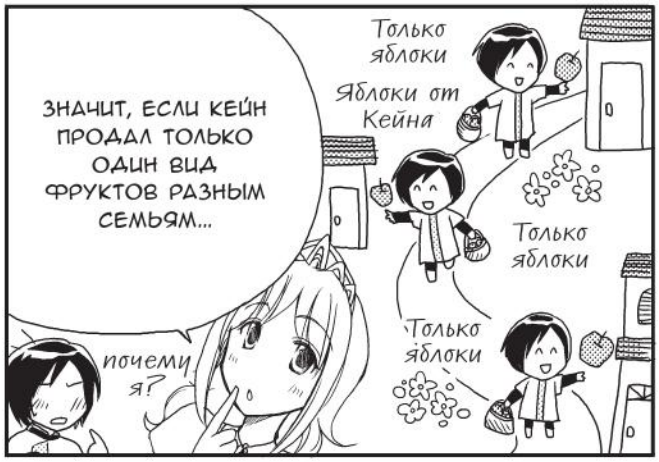

В E-R-МОДЕЛИ ВЫ РАССМАТРИВАЕТЕ НАСТОЯЩЕЕ ПОЛОЖЕНИЕ ВЕЩЕЙ, ИСПОЛЬЗУЯ КОНЦЕПЦИЮ СУЩНОСТЬ—СВЯЗЬ.
НЕМНОГО ПРОЯСНИМ ЭТО...
ХЛОП!
ХЛОП!
СКРИП
СУЩНОСТЬ ОТНОСИТСЯ К УЗНАВАЕМОМУ ОБЪЕКТУ ИЗ РЕАЛЬНОГО МИРА.
НАПРИМЕР, ПРИ ЭКСПОРТЕ ФРУКТОВ В ДРУГИЕ СТРАНЫ ФРУКТ И ПОКУПАТЕЛЯ МОЖНО СЧИТАТЬ СУЩНОСТЯМИ.
СКРИП
СКРИП СКРИП
СКРИП
И, СВЯЗЫВАЯ СУЩНОСТЬ С ЕГО СВОЙСТВАМИ, ПОЛУЧАЕМ...
СКРИП
Фрукт
Код товара
Товар
Цена
Покупатель
Код покупателя
Покупатель

ТЫ ПРОАНАЛИЗИРОВАЛА СВОЙСТВА ФРУКТОВ И ПОКУПАТЕЛЕЙ.
А ТОГДА ЧТО ТАКОЕ СВЯЗЬ?
СВЯЗЬ — ЭТО ТО, КАК СУЩНОСТИ СООТНОСЯТСЯ ДРУГ С ДРУГОМ.
НАПРИМЕР...

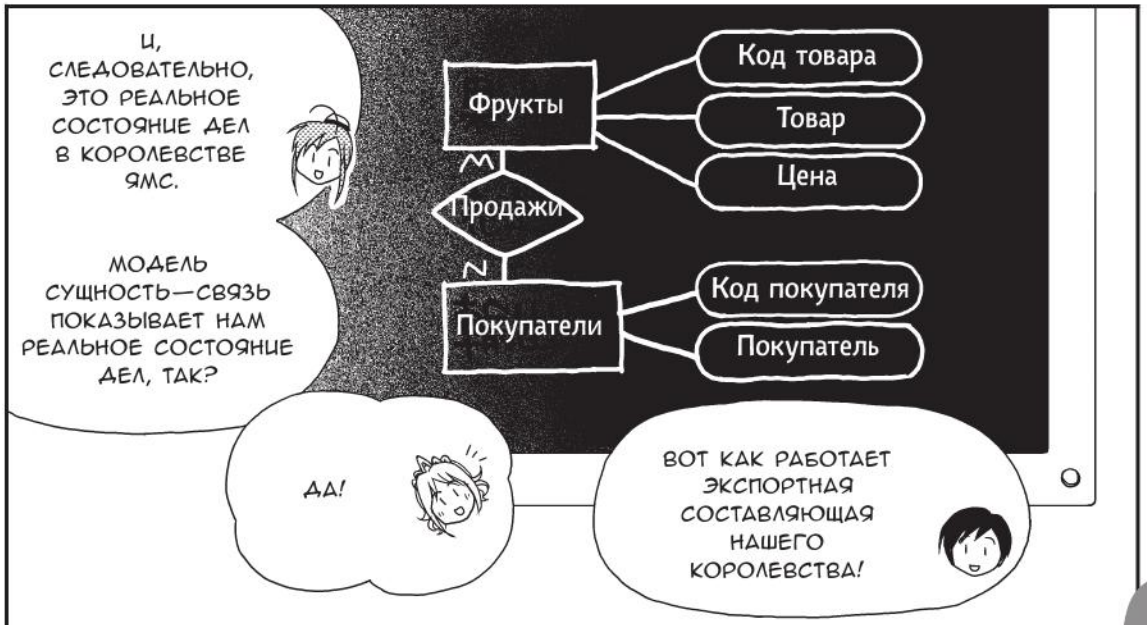
ФРУКТ И ПОКУПАТЕЛЬ СВЯЗАНЫ ДРУГ С ДРУГОМ, ПОТОМУ ЧТО ВЫ ПРОДАЁТЕ ФРУКТЫ СВОИМ ПОКУПАТЕЛЯМ.
Фрукт
Поку-патель
ТО ЕСТЬ МЫ МОЖЕМ РАССМАТРИВАТЬ ПРОДАЖИ КАК СВЯЗЬ.
Фрукты
Продажи
СКРИИИИП!

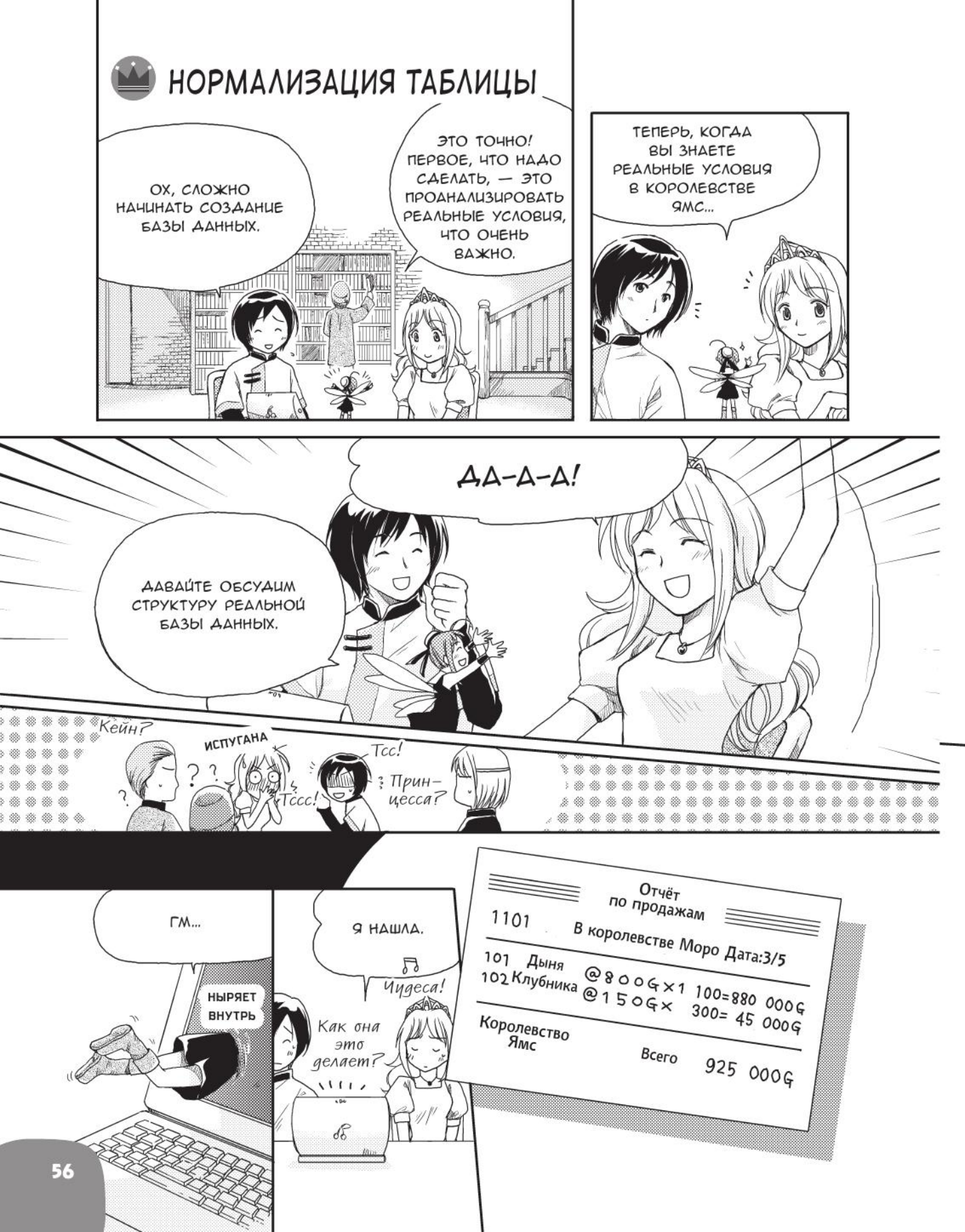
ФРУКТЫ ЭКСПОРТИРУЮТСЯ ВО МНОГИЕ СТРАНЫ, И КАЖДЫЙ ПОКУПАТЕЛЬ ПРИОБРЕТАЕТ МНОГО РАЗНЫХ ФРУКТОВ.
РАЗНЫЕ ПОКУПАТЕЛИ (N) СВЯЗАНЫ С РАЗНЫМИ ФРУКТАМИ (M).
Королевство Ямс
Королевство Рамбутан
Королевство Моро
Империя Фарайя
Фрукты из королевства Ямс
Покупатели
Королевство Моро
Империя Фарайя
Королевство Рамбутан
ЕСЛИ ЗАДУМАТЬСЯ...


Фрукты
Код товара
Товар
Цена
M
Продажи
N
Покупа-тели
Код покупателя
Покупатель
りんご
ぶどう
メロン
アルファ帝国
リトル王国
ジャンボ王国
Такая связь называется "многие ко многим".
В модели сущность—связь учитывается число связей между объектами.


Значит, если Кейн продал только один вид фруктов разным семьям...
Только яблоки
Яблоки от Кейна
Только яблоки
Только яблоки
почему я?

...тогда связь будет "один ко многим"?
Яблоки от Кейна
Фрукты
Продажи
Покупатель
Семья Анны
Семья Риммы
Семья Розы
В точку!
Верно!


И, следовательно, это реальное состояние дел в королевстве ЯМС.
Модель сущность—связь показывает нам реальное состояние дел, так?
Фрукты
Код товара
Товар
Цена
M
Продажи
N
Покупатели
Код покупателя
Покупатель
Да!
Вот как работает экспортная составляющая нашего королевства!


НОРМАЛИЗАЦИЯ ТАБЛИЦЫ
ОХ, СЛОЖНО НАЧИНАТЬ СОЗДАНИЕ БАЗЫ ДАННЫХ.
ЭТО ТОЧНО! ПЕРВОЕ, ЧТО НАДО СДЕЛАТЬ, — ЭТО ПРОАНАЛИЗИРОВАТЬ РЕАЛЬНЫЕ УСЛОВИЯ, ЧТО ОЧЕНЬ ВАЖНО.
ТЕПЕРЬ, КОГДА ВЫ ЗНАЕТЕ РЕАЛЬНЫЕ УСЛОВИЯ В КОРОЛЕВСТВЕ ЯМС...
ДА-А-А!
ДАВАЙТЕ ОБСУДИМ СТРУКТУРУ РЕАЛЬНОЙ БАЗЫ ДАННЫХ.
Кейн?
ИСПУГАНА
Тсс!
Тссс!
Прин-цесса?
ГМ...
НЫРЯЕТ ВНУТРЬ
Я НАШЛА.
Чудеса!
Как она это делает?
Отчёт по продажам
1101 В королевстве Моро Дата:3/5
101 Дыня @800G×1 100=880 000G
102 Клубника @150G× 300= 45 000G
Королевство Ямс Всего 925 000G

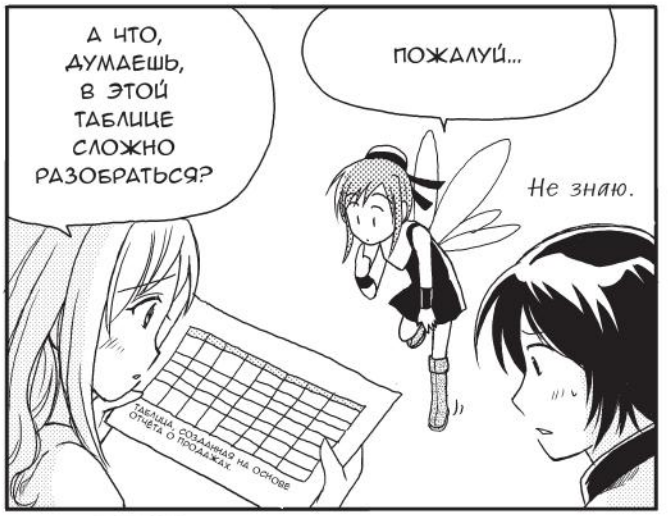
Смотрите!

Код отчёта	Дата	Код покупателя	Покупатель	Код товара	Название товара	Цена	Количество
1101	3/5	12	Королевство Моро	101	Дыни	800 G	1 100
				102	Клубника	150 G	300
1102	3/7	23	Империя Фарайя	103	Яблоки	120 G	1 700
1103	3/8	25	Королевство Рамбутан	104	Лимоны	200 G	500
1104	3/10	12	Королевство Моро	101	Дыни	800 G	2 500
1105	3/12	25	Королевство Рамбутан	103	Яблоки	120 G	2 000
				104	Лимоны	200 G	700

Таблица, созданная на основе отчёта о продажах

ВОТ ЭТО ДА!
КАК ЗДОРОВО! ЭТО МОЖЕТ СТАТЬ НАШЕЙ БАЗОЙ ДАННЫХ.
Фу ты
ИЗВИНИ, НО ТЫ НЕ МОЖЕШЬ ИСПОЛЬЗОВАТЬ ЕЁ В ТАКОМ ВИДЕ.

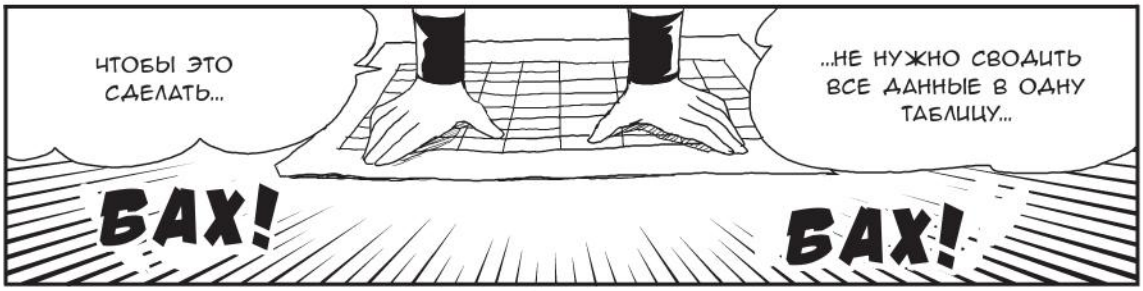
Принцесса!
РАЗДОСАДОВАНА
Ещё нет...
ЧТОБЫ СОЗДАТЬ РЕЛЯЦИОННУЮ БД, НАДО СДЕЛАТЬ ТАБЛИЦУ, МАКСИМАЛЬНО ПРОСТУЮ ДЛЯ ИСПОЛЬЗОВАНИЯ.
Неужели не ясно?

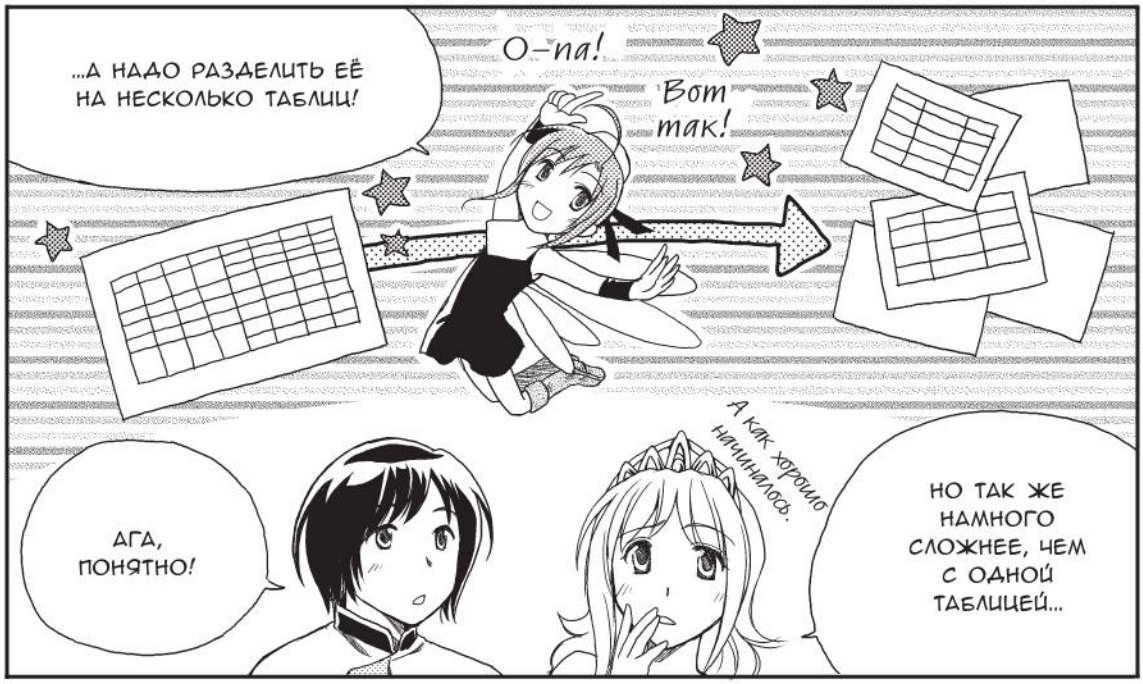

А ЧТО, ДУМАЕШЬ, В ЭТОЙ ТАБЛИЦЕ СЛОЖНО РАЗОБРАТЬСЯ?
ПОЖАЛУЙ...
Не знаю.
ТАБЛИЦА, СОЗДАННАЯ НА ОСНОВЕ ОТЧЁТА О ПРОДАЖАХ.

ВИДИШЬ, ТРИ ТОВАРА ВСТРЕЧАЮТСЯ ПО ДВА РАЗА.
ДА...

ЭТО ИЗ-ЗА ТОГО, ЧТО ИНОГДА МЫ РАБОТАЕМ С ДВУМЯ ИЛИ БОЛЕЕ ВИДАМИ ТОВАРОВ, ИСПОЛЬЗУЯ ОДИН ОТЧЁТ ПО ПРОДАЖАМ.
ТАК И ЕСТЬ.
Отчёт по продажам
1101 В королевство Моро Дата 3/5
101 Дыни @800G×1 100=880 000G
102 Клубника @150G× 300= 45 000G
Королевство Ямс 925 000G

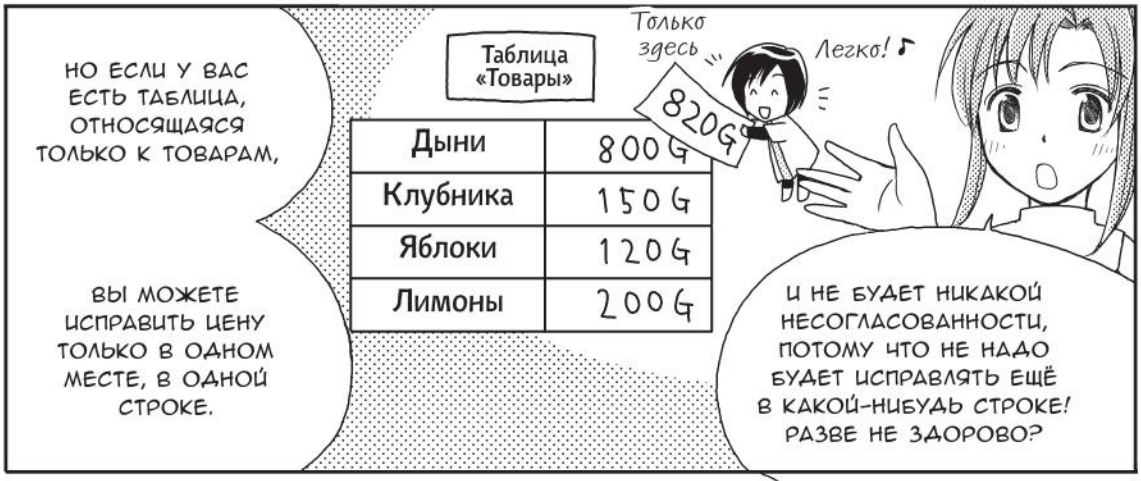
ЛЕГЧЕ БУДЕТ РАЗОБРАТЬСЯ, ЕСЛИ В КАЖДОЙ СТРОКЕ ТАБЛИЦЫ БУДЕТ СТОЯТЬ ПО ОДНОМУ ВИДУ ТОВАРА.
ДЕЙСТВИТЕЛЬНО...
Как же это сделать?
Хм...


ЧТОБЫ ЭТО СДЕЛАТЬ...
...НЕ НУЖНО СВОДИТЬ ВСЕ ДАННЫЕ В ОДНУ ТАБЛИЦУ...
БАХ!
БАХ!


...А НАДО РАЗДЕЛИТЬ ЕЁ НА НЕСКОЛЬКО ТАБЛИЦ!
О-па!
Вот так!
АГА, ПОНЯТНО!
А как хорошо начиналось.
НО ТАК ЖЕ НАМНОГО СЛОЖНЕЕ, ЧЕМ С ОДНОЙ ТАБЛИЦЕЙ...

МОЖЕТ, ЭТО И ВЫГЛЯДИТ СЛОЖНЫМ,
НО ВАЖНО РАБОТАТЬ С ДАННЫМИ КОРРЕКТНО И СОГЛАСОВАННО.
БОРМОЧЕТ БОРМОЧЕТ
Что делать, это наша работа.
Прикинуть реальное состояние,
используя модель сущность-связь
УГУ.

ЧТО? Я ТОЛЬКО ПОДУМАЛ ...
КЕЙН, ЧТО С ТОБОЙ?
Ты часто что-то бормочешь себе под нос.



Дыни	800G
Клубника	150G
Яблоки	120G
Лимоны	200G

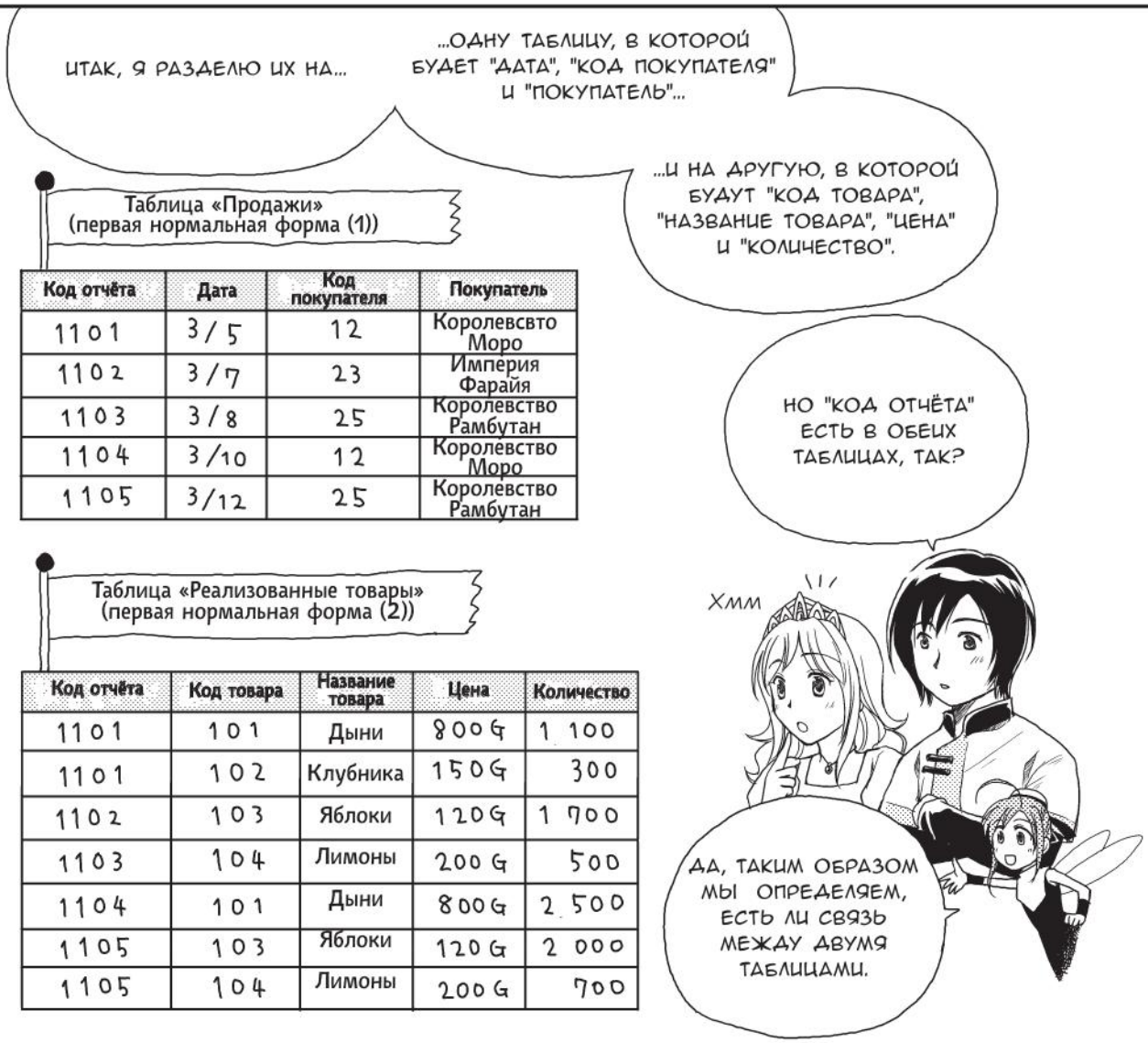


Таблица «Продажи»
(первая нормальная форма (1))

Код отчёта	Дата	Код покупателя	Покупатель
1101	3/5	12	Королевсвто Моро
1102	3/7	23	Империя Фарайя
1103	3/8	25	Королевство Рамбутан
1104	3/10	12	Королевство Моро
1105	3/12	25	Королевство Рамбутан

Таблица «Реализованные товары»
(первая нормальная форма (2))

Код отчёта	Код товара	Название товара	Цена	Количество
1101	101	Дыни	800G	1 100
1101	102	Клубника	150G	300
1102	103	Яблоки	120G	1 700
1103	104	Лимоны	200 G	500
1104	101	Дыни	800G	2 500
1105	103	Яблоки	120 G	2 000
1105	104	Лимоны	200 G	700

ТАБЛИЦА, ПОЛУЧИВШАЯСЯ В РЕЗУЛЬТАТЕ ТАКОГО РАЗДЕЛЕНИЯ, НАЗЫВАЕТСЯ ТАБЛИЦЕЙ ПЕРВОЙ НОРМАЛЬНОЙ ФОРМЫ.
Пусть запомнят мои очки.
1-я нормальная форма,
1-я нормальная форма
Да хватит же бубнить

ТАБЛИЦА, КОТОРАЯ ИМЕЕТ СТРОКИ С ДВУМЯ ИЛИ БОЛЕЕ ОДИНАКОВЫМИ ДАННЫМИ И КОТОРУЮ ПОТОМ ДЕЛЯТ, НАЗЫВАЕТСЯ НЕНОРМАЛИЗОВАННОЙ ФОРМОЙ.
ЭТО ОЗНАЧАЕТ, ЧТО ТАБЛИЦЫ ПЕРВОЙ НОРМАЛЬНОЙ ФОРМЫ СОЗДАЮТСЯ ДЕЛЕНИЕМ НЕНОРМАЛИЗОВАННОЙ ФОРМЫ.
Это необходимо!
Деление
1-я нормальная форма

ДАВАЙТЕ ПОСМОТРИМ...
МИНУТОЧКУ.

МЫ ПОЛУЧИМ ТАБЛИЦЫ "ПЕРВОЙ НОРМАЛЬНОЙ ФОРМЫ". ЗНАЧИТ, ЕСТЬ "ВТОРЫЕ" И "ТРЕТЬИ" НОРМАЛЬНЫЕ ФОРМЫ?
ТОЧНЯК!

В ТАКОМ ВИДЕ ТАБЛИЦА ПЕРВОЙ НОРМАЛЬНОЙ ФОРМЫ ЕЩЁ НЕ МОЖЕТ ИСПОЛЬЗОВАТЬСЯ КАК РЕЛЯЦИОННАЯ ТАБЛИЦА БД.
Держитесь!
ПОНЯТНО...
Вперёд!
Какой долгий путь
Первая нормальная форма
Госпожа реляционная база данных
СПОТЫКАЕТСЯ
СПОТЫКАЕТСЯ

Код отчёта	Код товара	Название товара	Цена	Количество
1101	101	Дыни	800G	1 100
1101	102	Клубника	150G	300

Таблица "Реализованные товары" (первая нормальная форма (2))

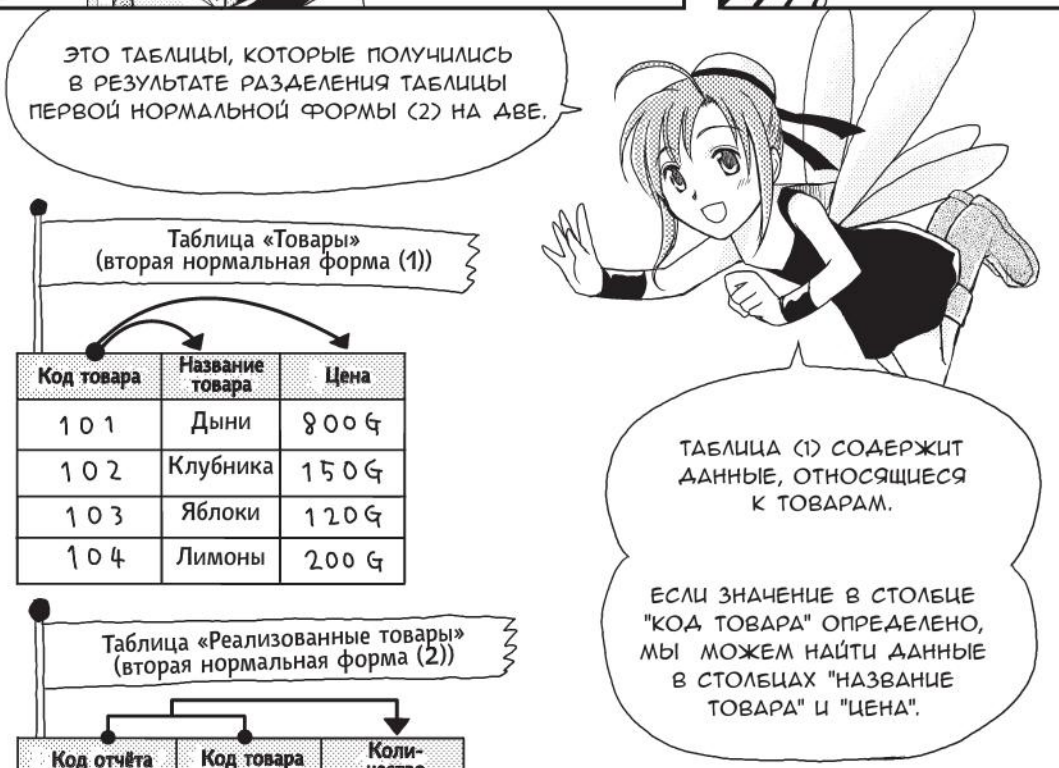


Таблица «Товары» (вторая нормальная форма (1))

Код товара	Название товара	Цена
101	Дыни	800 G
102	Клубника	150 G
103	Яблоки	120 G
104	Лимоны	200 G

Таблица «Реализованные товары» (вторая нормальная форма (2))

Код отчёта	Код товара	Количество
1101	101	1 100
1101	102	300
1102	103	1 700
1103	104	500
1104	101	2 500
1105	103	2 000
1105	104	700

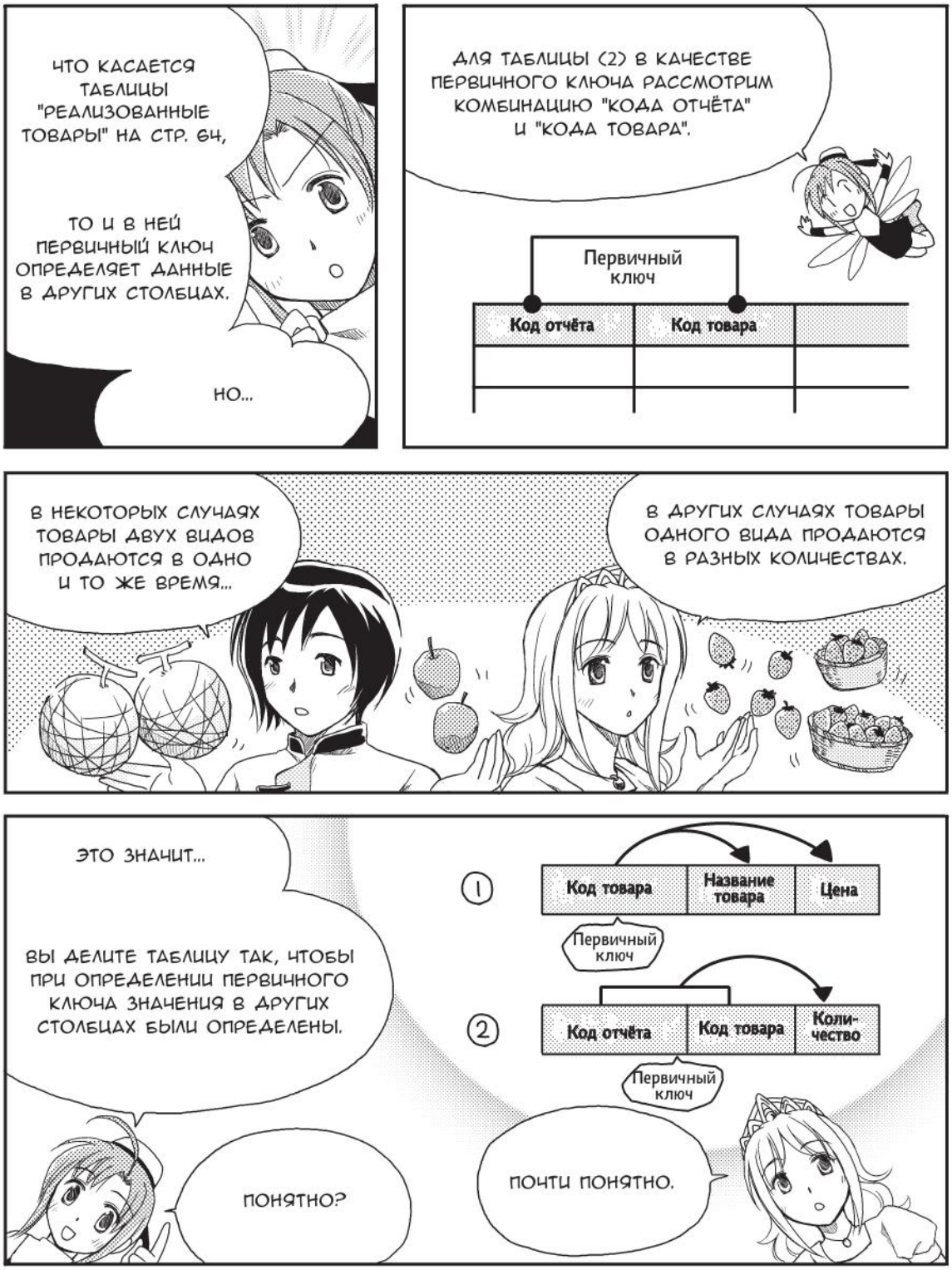

ЧТО КАСАЕТСЯ ТАБЛИЦЫ "РЕАЛИЗОВАННЫЕ ТОВАРЫ" НА СТР. 64,
ТО И В НЕЙ ПЕРВИЧНЫЙ КЛЮЧ ОПРЕДЕЛЯЕТ ДАННЫЕ В ДРУГИХ СТОЛБЦАХ.
НО...
ДЛЯ ТАБЛИЦЫ (2) В КАЧЕСТВЕ ПЕРВИЧНОГО КЛЮЧА РАССМОТРИМ КОМБИНАЦИЮ "КОДА ОТЧЁТА" И "КОДА ТОВАРА".
Первичный ключ
Код отчёта
Код товара
В НЕКОТОРЫХ СЛУЧАЯХ ТОВАРЫ ДВУХ ВИДОВ ПРОДАЮТСЯ В ОДНО И ТО ЖЕ ВРЕМЯ...
В ДРУГИХ СЛУЧАЯХ ТОВАРЫ ОДНОГО ВИДА ПРОДАЮТСЯ В РАЗНЫХ КОЛИЧЕСТВАХ.
ЭТО ЗНАЧИТ...
ВЫ ДЕЛИТЕ ТАБЛИЦУ ТАК, ЧТОБЫ ПРИ ОПРЕДЕЛЕНИИ ПЕРВИЧНОГО КЛЮЧА ЗНАЧЕНИЯ В ДРУГИХ СТОЛБЦАХ БЫЛИ ОПРЕДЕЛЕНЫ.
ПОНЯТНО?
1
Код товара
Название товара
Цена
Первичный ключ
2
Код отчёта
Код товара
Коли-чество
Первичный ключ
ПОЧТИ ПОНЯТНО.

Таблица «Продажи»
(первая нормальная форма (1))

Код отчёта	Дата	Код покупателя	Покупатель
1101	3/5	12	Королевство Моро
1102	3/7	23	Империя Фарайя
1103	3/8	25	Королевство Рамбутан
1104	3/10	12	Королевство Моро
1105	3/12	25	Королевство Рамбутан

Определено — Первичный ключ

Если значение определено — Это значение — Определено

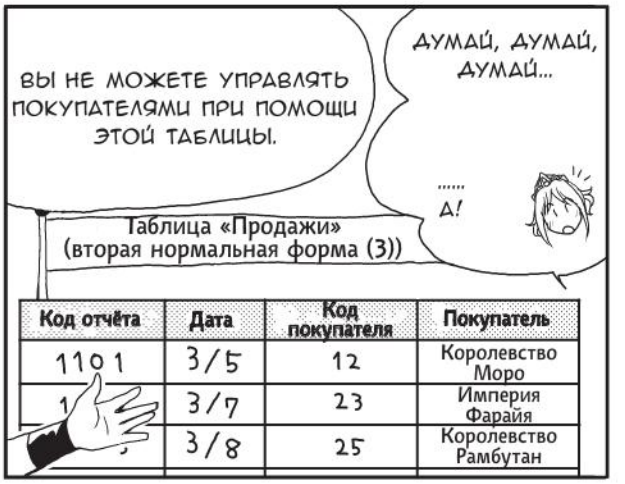
СКАЖИ-КА, КАК НАЗЫВАЕТСЯ ТАБЛИЦА, В КОТОРОЙ ЗНАЧЕНИЯ В ДРУГИХ СТОЛБЦАХ ОПРЕДЕЛЕНЫ, КОГДА ОПРЕДЕЛЁН ПЕРВИЧНЫЙ КЛЮЧ?
ТАБЛИЦА ВТОРОЙ НОРМАЛЬНОЙ ФОРМЫ!
ПОЛУЧАЕТСЯ, ЭТА ТАБЛИЦА ЯВЛЯЕТСЯ ТАБЛИЦЕЙ ВТОРОЙ НОРМАЛЬНОЙ ФОРМЫ, ТАК?

Таблица «Продажи» (первая нормальная форма (1))
Код отчёта | Дата | Код покупателя | Покупатель
12 | Королевство Моро
Империя Фарайя
Королевство Рамбутан
ВЕРНО. ТАБЛИЦУ ПЕРВОЙ НОРМАЛЬНОЙ ФОРМЫ (1) МОЖНО СЧИТАТЬ...
Таблица «Продажи» (вторая нормальная форма (3))
Код отчёта | Дата | Код покупателя | Покупатель
12 | Королевство Моро
Империя Фарайя
Королевство Рамбутан
...ТАБЛИЦЕЙ ВТОРОЙ НОРМАЛЬНОЙ ФОРМЫ (3)!

МЫ ЗАКОНЧИЛИ С ПЕРВОЙ И ВТОРОЙ НОРМАЛЬНЫМИ ФОРМАМИ!
ТЕПЕРЬ НАША РЕЛЯЦИОННАЯ БАЗА ДАННЫХ СОСТАВЛЕНА!
РАДОСТНАЯ
Вторая нормальная форма — это...

ШОК!
СЕКУНДОЧКУ...
Рано радуетесь!

Вперёд!
ЕЩЁ НЕТ...
НО УЖЕ ПОЧТИ!
Вторая нормальная форма
Госпожа реляционная база данных.
Круче, чем раньше.
Ты умеешь летать, Тико. Это не честно!



Вот так **СНОВА РАЗДЕЛИТЬ ИХ!**

Таблица «Продажи» (третья нормальная форма (1))

Код отчёта	Дата	Код покупателя
1101	3/5	12
1102	3/7	23
1103	3/8	25
1104	3/10	12
1105	3/12	25

Таблица «Покупатели» (третья нормальная форма (2))

Код покупателя	Покупатель
12	Королевство Моро
23	Империя Фарайя
25	Королевство Рамбутан

ТА-ДАМ!

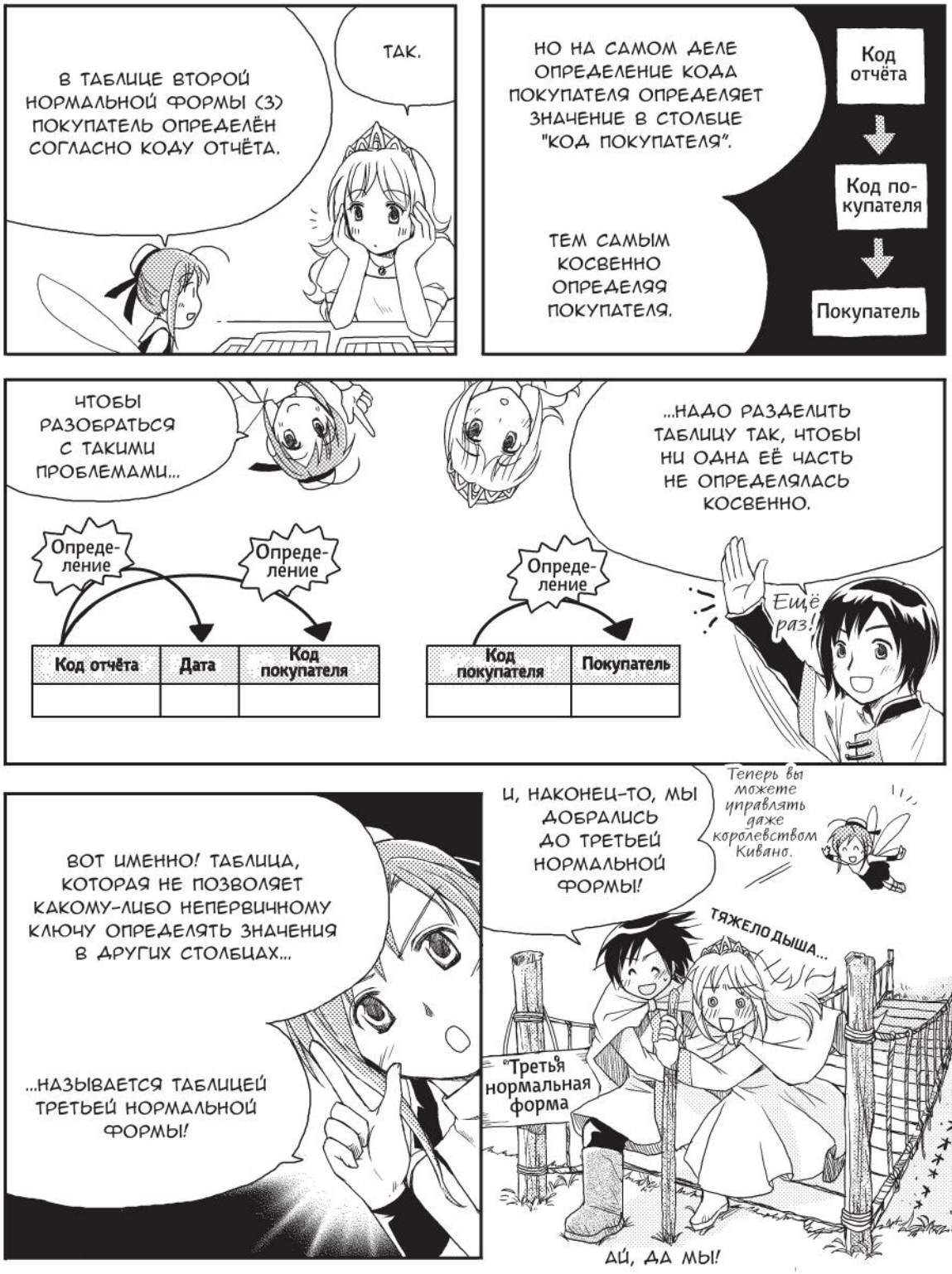

В ТАБЛИЦЕ ВТОРОЙ НОРМАЛЬНОЙ ФОРМЫ (3) ПОКУПАТЕЛЬ ОПРЕДЕЛЁН СОГЛАСНО КОДУ ОТЧЁТА.
ТАК.
НО НА САМОМ ДЕЛЕ ОПРЕДЕЛЕНИЕ КОДА ПОКУПАТЕЛЯ ОПРЕДЕЛЯЕТ ЗНАЧЕНИЕ В СТОЛБЦЕ "КОД ПОКУПАТЕЛЯ".
ТЕМ САМЫМ КОСВЕННО ОПРЕДЕЛЯЯ ПОКУПАТЕЛЯ.
Код отчёта
Код покупателя
Покупатель
ЧТОБЫ РАЗОБРАТЬСЯ С ТАКИМИ ПРОБЛЕМАМИ...
...НАДО РАЗДЕЛИТЬ ТАБЛИЦУ ТАК, ЧТОБЫ НИ ОДНА ЕЁ ЧАСТЬ НЕ ОПРЕДЕЛЯЛАСЬ КОСВЕННО.
Определение
Определение
Определение
Код отчёта | Дата | Код покупателя
Код покупателя | Покупатель
Ещё раз!
ВОТ ИМЕННО! ТАБЛИЦА, КОТОРАЯ НЕ ПОЗВОЛЯЕТ КАКОМУ-ЛИБО НЕПЕРВИЧНОМУ КЛЮЧУ ОПРЕДЕЛЯТЬ ЗНАЧЕНИЯ В ДРУГИХ СТОЛБЦАХ...
...НАЗЫВАЕТСЯ ТАБЛИЦЕЙ ТРЕТЬЕЙ НОРМАЛЬНОЙ ФОРМЫ!
И, НАКОНЕЦ-ТО, МЫ ДОБРАЛИСЬ ДО ТРЕТЬЕЙ НОРМАЛЬНОЙ ФОРМЫ!
Теперь вы можете управлять даже королевством Кивано.
ТЯЖЕЛО ДЫША...
Третья нормальная форма
АЙ, ДА МЫ!

Таблица «Продажи»

Код отчёта	Дата	Код покупателя
1101	3/5	12
1102	3/7	23
1103	3/8	25
1104	3/10	12
1105	3/12	25

Таблица «Покупатели»

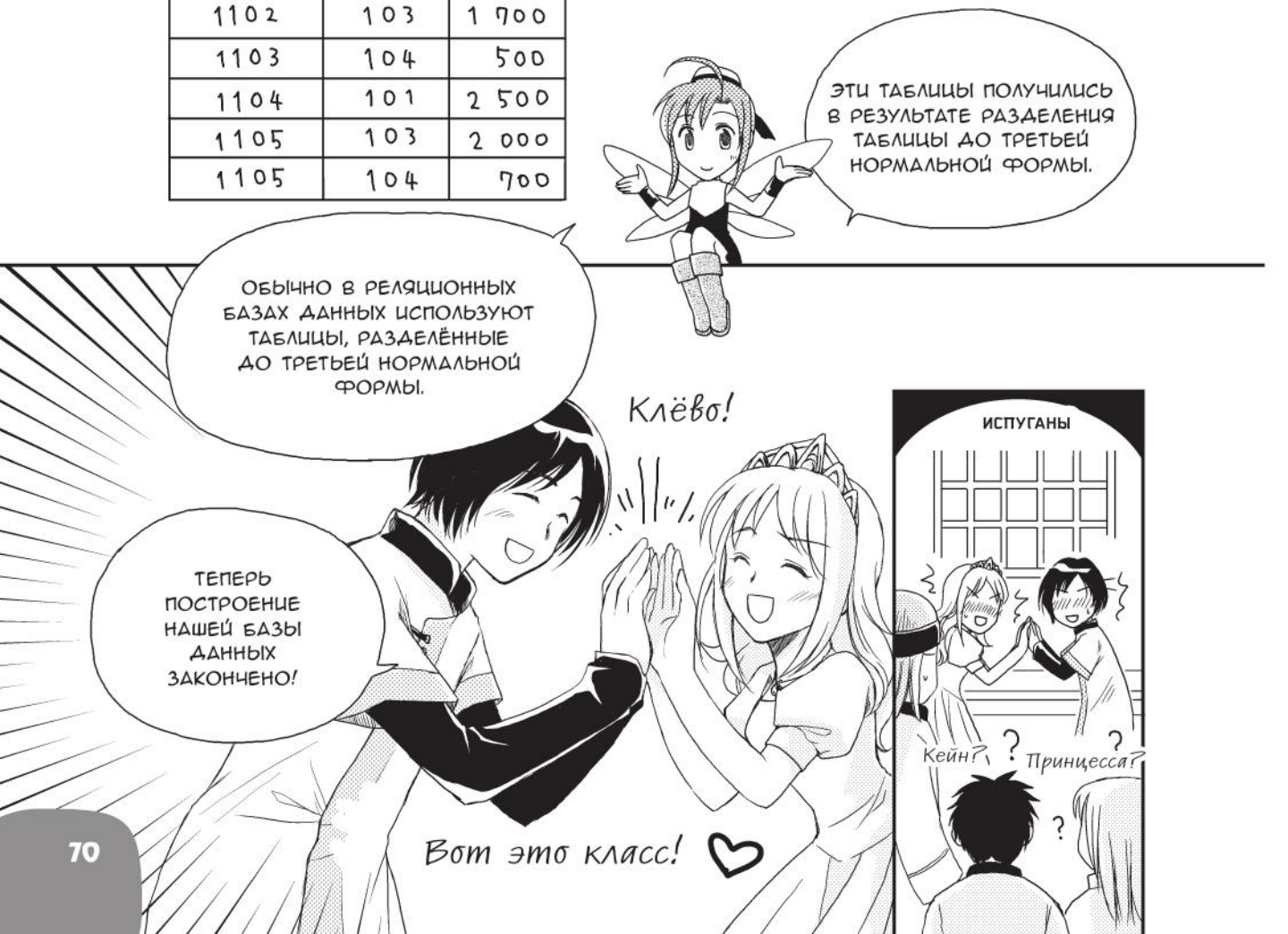
Код покупателя	Покупатель
12	Королевство Моро
23	Империя Фарайя
25	Королевство Рамбутан

Таблица «Реализованные товары»

Код отчёта	Код товара	Количество
1101	101	1 100
1101	102	300
1102	103	1 700
1103	104	500
1104	101	2 500
1105	103	2 000
1105	104	700

Таблица «Товары»

Код товара	Название товара	Цена
101	дыня	800 G
102	клубника	150 G
103	яблоко	120 G
104	лимон	200 G



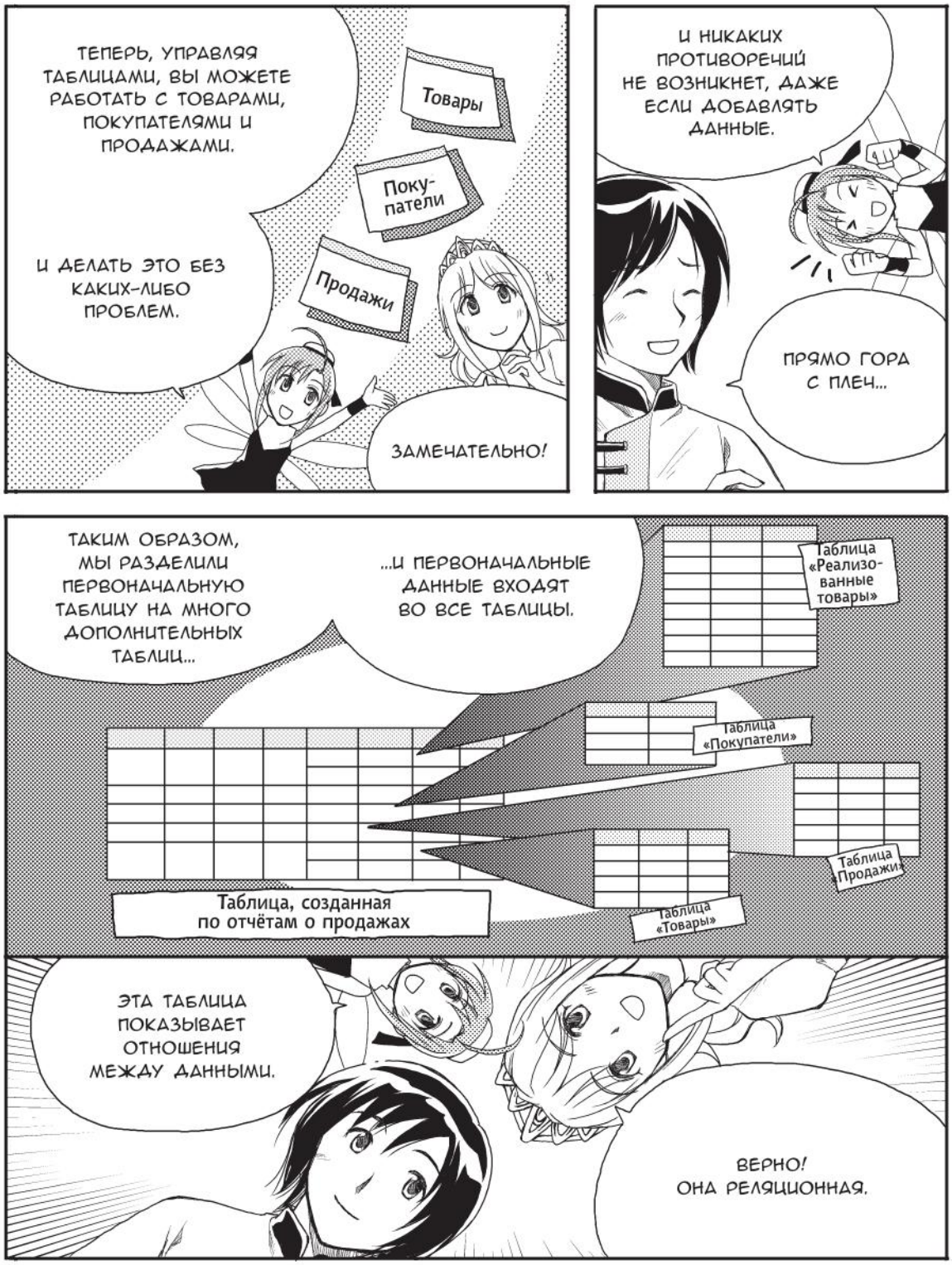

Теперь, управляя таблицами, вы можете работать с товарами, покупателями и продажами.
Товары
Поку-патели
Продажи
И делать это без каких-либо проблем.
Замечательно!
И никаких противоречий не возникнет, даже если добавлять данные.
Прямо гора с плеч...
Таким образом, мы разделили первоначальную таблицу на много дополнительных таблиц...
...и первоначальные данные входят во все таблицы.
Таблица «Реализо-ванные товары»
Таблица «Покупатели»
Таблица «Продажи»
Таблица «Товары»
Таблица, созданная по отчётам о продажах
Эта таблица показывает отношения между данными.
Верно! Она реляционная.

Таблица «Продажи»

Код отчёта	Дата	Код покупателя
1101		

ссылка на...

Таблица «Покупатели»

Код покупателя	Покупатель
12	Королевство
23	Империя Фарайя
25	Королевство

"КОД ПОКУПАТЕЛЯ" В ТАБЛИЦЕ "ПРОДАЖИ" ССЫЛАЕТСЯ НА "КОД ПОКУПАТЕЛЯ" В ТАБЛИЦЕ "ПОКУПАТЕЛЬ".

ТАКИМ ЖЕ ОБРАЗОМ "КОД ТОВАРА" В ТАБЛИЦЕ "РЕАЛИЗОВАННЫЕ ТОВАРЫ" ССЫЛАЕТСЯ НА "КОД ТОВАРА" В ТАБЛИЦЕ "ТОВАРЫ"...

Таблица «Реализованные товары»

ссылка на...

Код отчёта	Код товара	Количество
1101	101	1,100
1101	102	800
1102	103	1,700
1103	104	500
1104	101	2,500
1105	103	2
1105	104	

ссылка на...

Таблица «Товары»

Код товара	Название товара	Цена
101	Дыни	800G
102	Клубника	150G
		120G
		200 G

...И "КОД ОТЧЁТА" В ТАБЛИЦЕ "РЕАЛИЗОВАННЫЕ ТОВАРЫ" ССЫЛАЕТСЯ НА "КОД ОТЧЁТА" В ТАБЛИЦЕ "ПРОДАЖИ".

СТОЛБЕЦ, КОТОРЫЙ ССЫЛАЕТСЯ, НАЗЫВАЕТСЯ ВНЕШНИЙ КЛЮЧ.

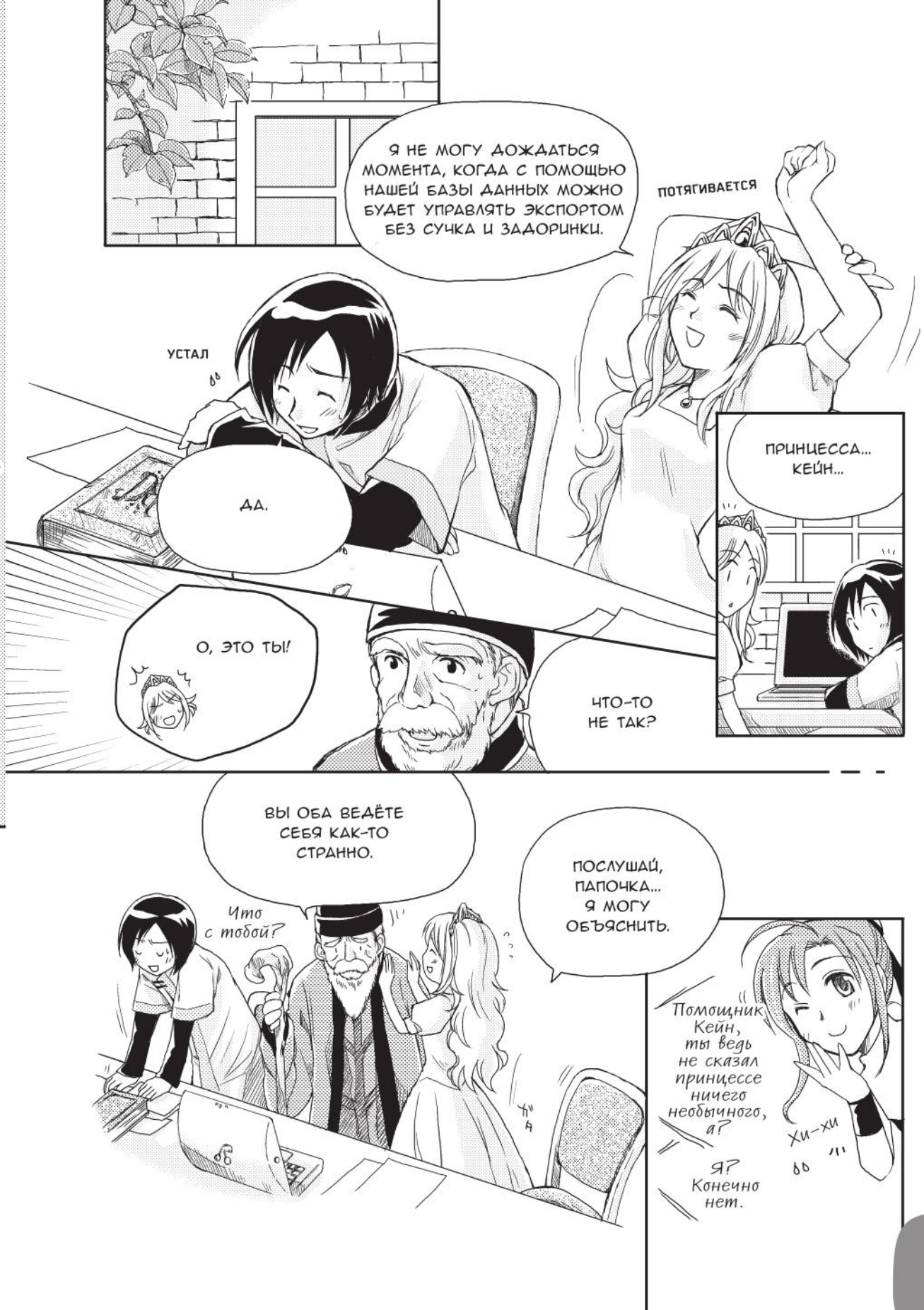
ВНЕШНИЙ КЛЮЧ ССЫЛАЕТСЯ НА ПЕРВИЧНЫЙ КЛЮЧ В ДРУГИХ ТАБЛИЦАХ.

Госпожа реляционная база данных

ЭТИ ТАБЛИЦЫ КАЖУТСЯ ОТДЕЛЬНЫМИ И НЕЗАВИСИМЫМИ ДРУГ ОТ ДРУГА, НО ИХ КРЕПКО ОБЪЕДИНЯЮТ ВНЕШНИЕ КЛЮЧИ.

Мы почти...

на вершине!

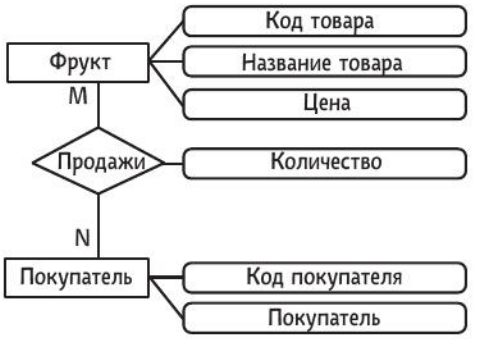

Я НЕ МОГУ ДОЖДАТЬСЯ МОМЕНТА, КОГДА С ПОМОЩЬЮ НАШЕЙ БАЗЫ ДАННЫХ МОЖНО БУДЕТ УПРАВЛЯТЬ ЭКСПОРТОМ БЕЗ СУЧКА И ЗАДОРИНКИ.
ПОТЯГИВАЕТСЯ
УСТАЛ
ДА.
ПРИНЦЕССА... КЕЙН...
О, ЭТО ТЫ!
ЧТО-ТО НЕ ТАК?
ВЫ ОБА ВЕДЁТЕ СЕБЯ КАК-ТО СТРАННО.
Что с тобой?
ПОСЛУШАЙ, ПАПОЧКА... Я МОГУ ОБЪЯСНИТЬ.
Помощник Кейн, ты ведь не сказал принцессе ничего необычного, а?
Хи-хи
Я? Конечно нет.

Что такое модель сущность – связь (E – R-модель)

Принцесса Руруна и Кейн выяснили реальное состояние дел с продажами товаров в королевстве Ямс с помощью модели сущность—связь (E—R-модель). Когда вы пытаетесь создать базу данных самостоятельно, первым делом надо определить состояние данных, которые вы собираетесь использовать.

Используя модель сущность—связь, постарайтесь определить, что является сущностью в ваших данных. Сущность — это реально существующий в жизни объект или «вещь», такая как фрукт или покупатель.

Кроме того, модель сущность—связь показывает связь между сущностями. Принцесса Руруна и Кейн выполнили анализ, предположив, что связью между фруктами и покупателями будут выступать продажи. Фрукты экспортируются многим покупателям, в то время как сами покупатели, в свою очередь, импортируют фрукты многих видов. По этой причине для E—R-модели был проведён анализ, допускающий, что между фруктами и покупателями существуют связи, называемые «многие ко многим». Фрукт M имеет связь с покупателем N. Число связей между сущностями называется **кардинальностью**.

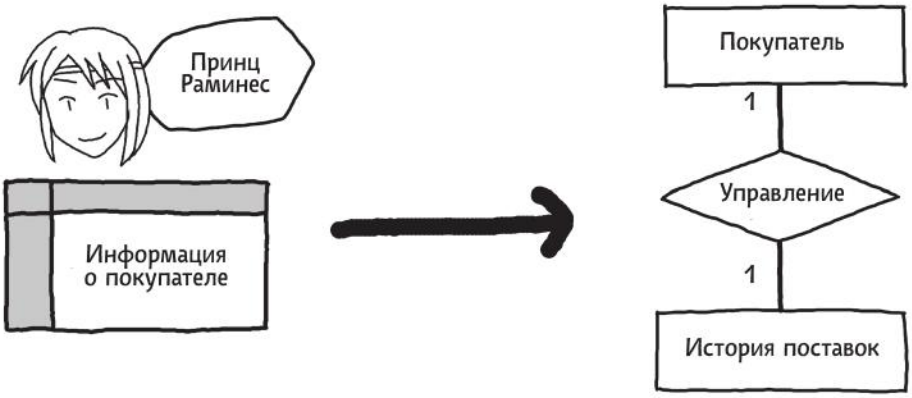


Как анализировать модель сущность – связь

Как бы вы выполнили анализ в случаях, приведённых ниже? Поразмышляйте над этим.

ПРИМЕР 1. СВЯЗЬ «ОДИН К ОДНОМУ»

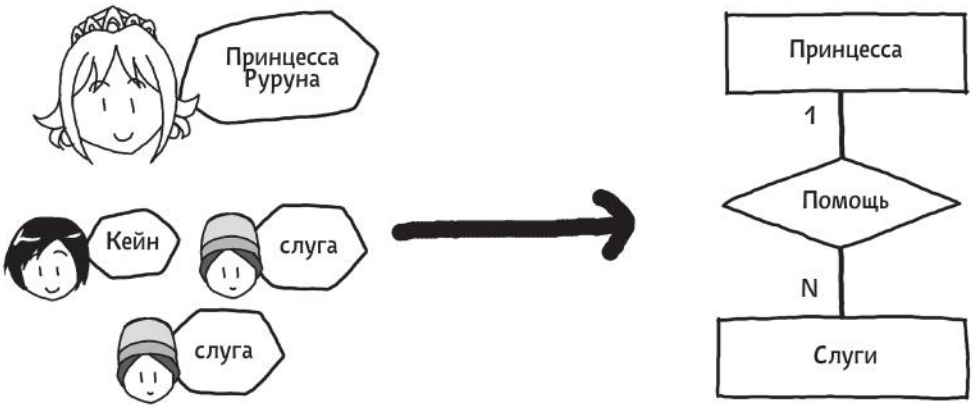
Один покупатель работает с одной частью информации, касающейся истории экспортных поставок.



Этот тип связи называется «один к одному».

ПРИМЕР 2. СВЯЗЬ «ОДИН КО МНОГИМ»

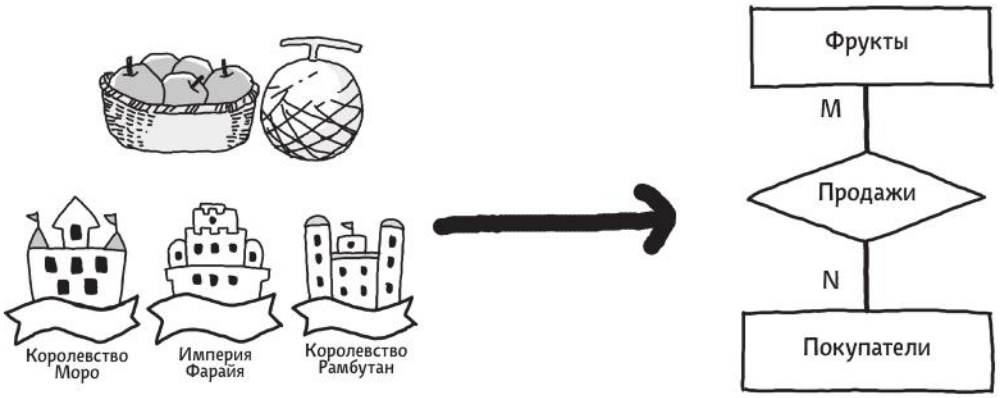
Одну принцессу обслуживают несколько слуг. Эти слуги больше не служат ни другим принцессам, ни даже королю.



Этот тип связи называется «один ко многим».

ПРИМЕР 3. СВЯЗЬ «МНОГИЕ КО МНОГИМ»

Фрукты экспортируются разным покупателям. Покупатели сами импортируют фрукты разных видов.



Этот вид связи называется «многие ко многим».

ВОПРОСЫ И ЗАДАНИЯ

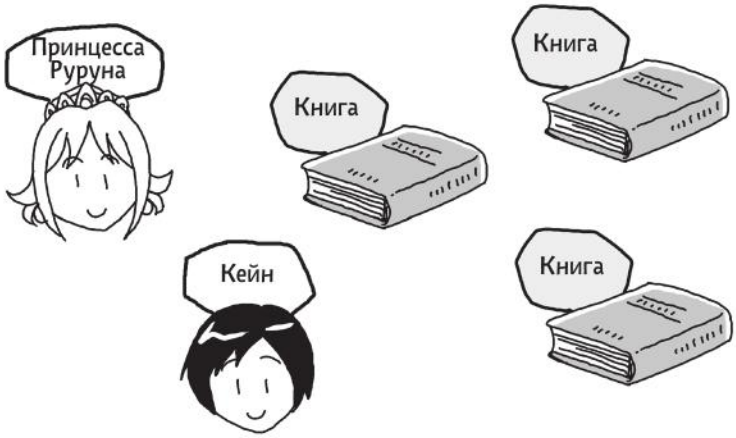
Насколько хорошо вы поняли модель сущность—связь (E—R-модель)? Проанализируйте и нарисуйте модель сущность—связь для каждого из приведенных ниже примеров. Ответы вы найдёте на стр. 82, 83.

B1

Один сотрудник работает сразу со многими клиентами. Один клиент никогда не заключает договор более чем с одним сотрудником.

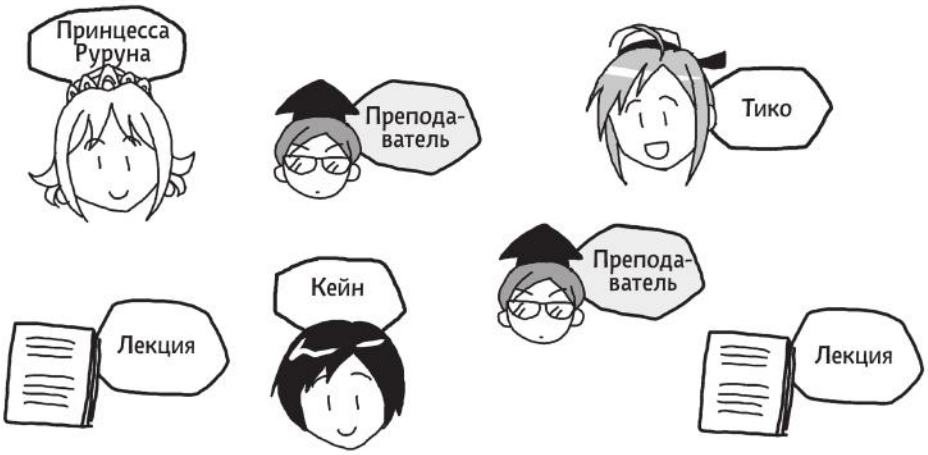
B2

Один человек может сдать много книжек. Книжки могут быть сданы многими студентами в разное время.



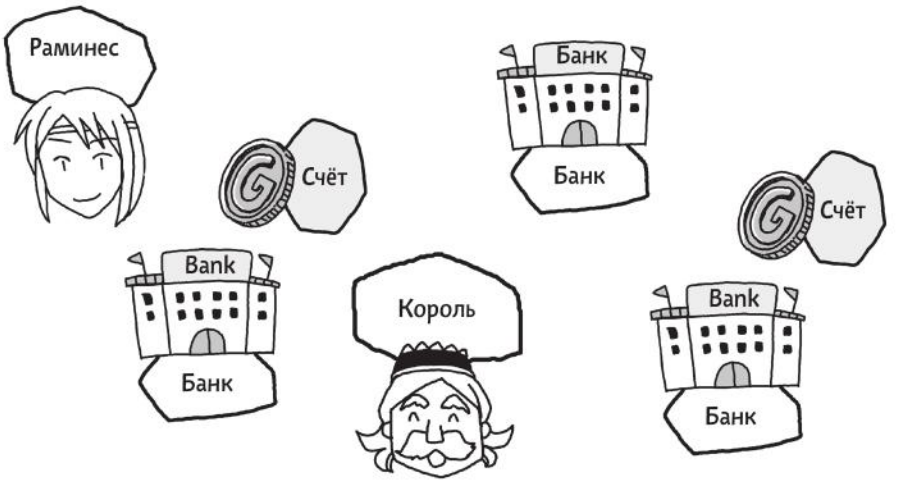
B3

Каждый студент посещает много лекций. На каждую лекцию ходит много студентов. Один преподаватель читает много лекций. Каждую лекцию читает один преподаватель.



ВЧ

Каждый клиент может открыть много депозитных счетов. Каждый депозитный счёт открывается одним клиентом. Каждый банк управляет многими депозитными счетами. Каждый депозитный счёт управляется одним банком.



Не забывайте, что анализ на основе модели сущность—связь необязательно выдает только один «корректный» результат. Для того чтобы отобразить реальные условия с помощью логически выстроенных данных, можно придумать много способов.

НОРМАЛИЗАЦИЯ ТАБЛИЦЫ

Принцесса Руруна и Кейн узнали про нормализацию, процесс сведения данных, взятых из реального мира, в таблицу для создания реляционной БД. Нормализовать данные необходимо для того, чтобы управлять базой должным образом. Процесс нормализации приведён ниже (затемнённые поля — это первичные ключи).

■ НЕНОРМАЛИЗОВАННАЯ ФОРМА

Код отчёта	Дата	Код покупателя	Покупатель	Код товара	Название товара	Цена	Количество

■ ПЕРВАЯ НОРМАЛЬНАЯ ФОРМА (FIRST NORMAL FORM)

Код отчёта	Дата	Код покупателя	Покупатель

Код отчёта	Код товара	Название товара	Цена	Количество

■ ВТОРАЯ НОРМАЛЬНАЯ ФОРМА (SECOND NORMAL FORM)

Код отчёта	Дата	Код покупателя	Покупатель

Код отчёта	Код товара	Количество

Код товара	Название товара	Цена

■ ТРЕТЬЯ НОРМАЛЬНАЯ ФОРМА (THIRD NORMAL FORM)

Код отчёта	Дата	Код покупателя

Код покупателя	Покупатель

Код отчёта	Код товара	Количество

Код товара	Название товара	Цена

Ненормализованная форма — это таблица, в которой элементы, появляющиеся в ней более одного раза, не могут быть удалены. Мы уже убедились, что управлять данными в реляционной БД с помощью такой таблицы нельзя. Следовательно, нам надо разделить таблицу.

Первая нормальная форма (First Normal Form) подразумевает простую двумерную таблицу, которая получается в результате разделения первоначальной ненормализованной таблицы. Можно рассматривать её как таблицу с одним элементом в каждой ячейке. Таблица разделена так, что ни один элемент не появится более одного раза.

Вторая нормальная форма (Second Normal Form) подразумевает таблицу, в которой ключ, обозначающий данные, определяет значения в других столбцах. То есть именно первичный ключ определяет значения в других столбцах.

В реляционной БД значение называется **функционально зависимым** (Functionally Dependent), если оно определяет значения в других столбцах. Во второй нормальной форме таблица делится так, что значения в других столбцах функционально зависят от первичного ключа.

В третьей нормальной форме (Third Normal Form) таблица делится так, чтобы значение не определялось каким-либо непервичным ключом. В реляционной БД значение называется **транзитивно зависимым** (Transitively Dependent), если это значение косвенно определяет значения в других столбцах, что является частью функционально зависимой операции. В третьей нормальной форме таблица делится так, что транзитивно зависимые значения удаляются.

ВОПРОСЫ И ЗАДАНИЯ

Важно суметь спроектировать таблицу реляционной базы данных для различных ситуаций, поэтому давайте рассмотрим некоторые примеры нормализации таблиц. Определите, как была нормализована таблица в каждом из случаев, приведённых ниже. Ответы даны на стр. 83.

B5

Следующая таблица, которая приводилась в примере B2, управляет выдачей книг. До какого уровня она нормализована?

Код выдачи	Дата	Код студента	Имя студента	Адрес студента	Факультет	Год поступления

ISBN	Название книги	Автор	Дата выпуска	Количество страниц

Код выдачи	ISBN	Количество

B6

Следующая таблица также иллюстрирует выдачу книг. До какого уровня она нормализована?

Код выдачи	Дата	Код студента

Код студента	Имя студента	Адрес студента	Факультет	Год поступления

ISBN	Название книги	Автор	Дата выпуска	Количество страниц

Код выдачи	ISBN	Количество

B7

Следующая таблица иллюстрирует ежемесячные продажи, приходящиеся на каждого сотрудника. В каждом отделе числится много сотрудников (клерков). Каждый сотрудник может быть только в одном отделе. Нормализуйте эту таблицу до третьей нормальной формы.

Код сотрудника	Имя сотрудника	Месяц	Продажи сотрудника	Код отдела	Название отдела

B8

Следующая таблица представляет систему получения заказа. Нормализуйте её до третьей нормальной формы, но проводите каждого клиента по коду приёма заказа. На основе одного кода приёма заказа вы можете обрабатывать много товаров. Плюс к этому один код приёма заказа должен соответствовать только одному представителю.

Код приёма заказа	Дата	Код клиента	Имя клиента	Код товара	Название товара	Цена	Код представителя	Имя представителя	Количество

B9

Следующая таблица представляет систему получения заказа. Нормализуйте её до третьей нормальной формы. Пусть товары классифицируются по коду товара.

Код приёма заказа	Дата	Код клиента	Имя клиента	Код товара	Название товара	Цена	Код товарной классификации	Название товарной классификации	Количество

СТАДИИ РАЗРАБОТКИ БАЗЫ ДАННЫХ

Вы узнали, как разработать базу данных! Однако теперь придётся сделать ещё больше, чем вы уже сделали. Вам надо спроектировать детальную файловую структуру внутри базы и придумать методы импорта и экспорта данных. В общем случае вы можете разделить весь процесс разработки БД на три части: концептуальная схема, внутренняя схема и внешняя схема. Концептуальная схема подразумевает метод, который моделирует реальное положения вещей. То есть это способ определения логической структуры базы данных. При разработке концептуальной схемы берётся во внимание, во-первых, понимание реальных условий на основе модели сущность—связь (E—R), во-вторых, нормализация таблицы. Внутренняя схема подразумевает БД, рассматриваемую с точки зрения компьютера. То есть это способ определения физической структуры базы данных. Внутренняя схема проектируется после создания метода быстрого поиска в базе. Внешняя схема подразумевает БД, рассматриваемую с точки зрения пользователя или приложения. Внешняя схема разрабатывается после создания данных, которые требуются для работы программ-приложений.

В этой главе принцесса Руруна и Кейн разработали базу данных, сконцентрировавшись на концептуальной схеме. Они сейчас находятся в середине процесса улучшения БД. И раз вы завершили основное проектирование базы данных, в следующей главе мы перейдём прямо к применению БД.

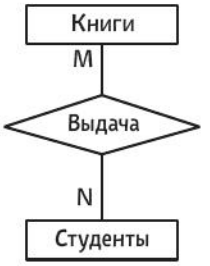
ИТОГИ

- E—R-модель используется для анализа сущностей и связей.
- Связи между сущностями могут иметь вид «один к одному», «один ко многим» и «многие ко многим».
- Данные в таблице должны быть нормализованы до их использования при создании реляционной БД.
- Процесс разработки базы данных можно разделить на три части: концептуальная схема, внутренняя схема и внешняя схема.

ОТВЕТЫ

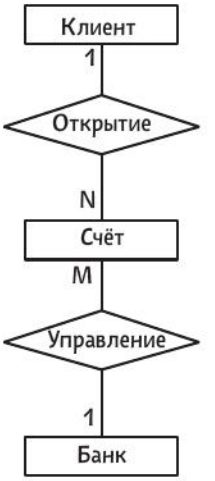
01

02



03

04



05 Вторая нормальная форма

06 Третья нормальная форма

07

Код сотрудника	Месяц	Продажи сотрудника

Код сотрудника	Имя сотрудника	Код отдела

Код отдела	Название отдела

08

Код приёма заказа	Дата	Код клиента	Код представителя

Код клиента	Имя клиента

Код приёма заказа	Код товара	Количество

Код товара	Название товара	Цена

Код представителя	Имя представителя

09

Код приёма заказа	Дата	Код клиента

Код клиента	Имя клиента

Код приёма заказа	Код товара	Количество

Код товара	Код товарной классификации	Название товара	Цена

Код товарной классификации	Название товарной классификации

ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

В этой главе вы узнали, как спроектировать реляционную базу данных. Однако есть и другие методы проектирования БД. Удобство пользования и эффективность БД зависят от анализа и метода проектирования. Следовательно, важно создать соответствующую базу данных на этапе проектирования.

На этапе проектирования, кроме создания таблиц, вам нужно выполнить разнообразные задачи. Например, надо решить, какой тип данных будет использоваться в таблице. Возможно, вам понадобится определить столбцы с числовыми значениями, валютами и символьными строками. Плюс к этому надо будет придумать такой метод поиска, чтобы можно было выполнять быстрый поиск. Иногда при проектировании нужно держать в уме физическую организацию файлов (физическую организацию файловой системы). И, чтобы обеспечить безопасность, вам придётся установить, какие пользователи имеют доступ к БД. Существует множество факторов, о которых вам нужно думать при проектировании базы данных. В следующих главах мы рассмотрим некоторые из них.

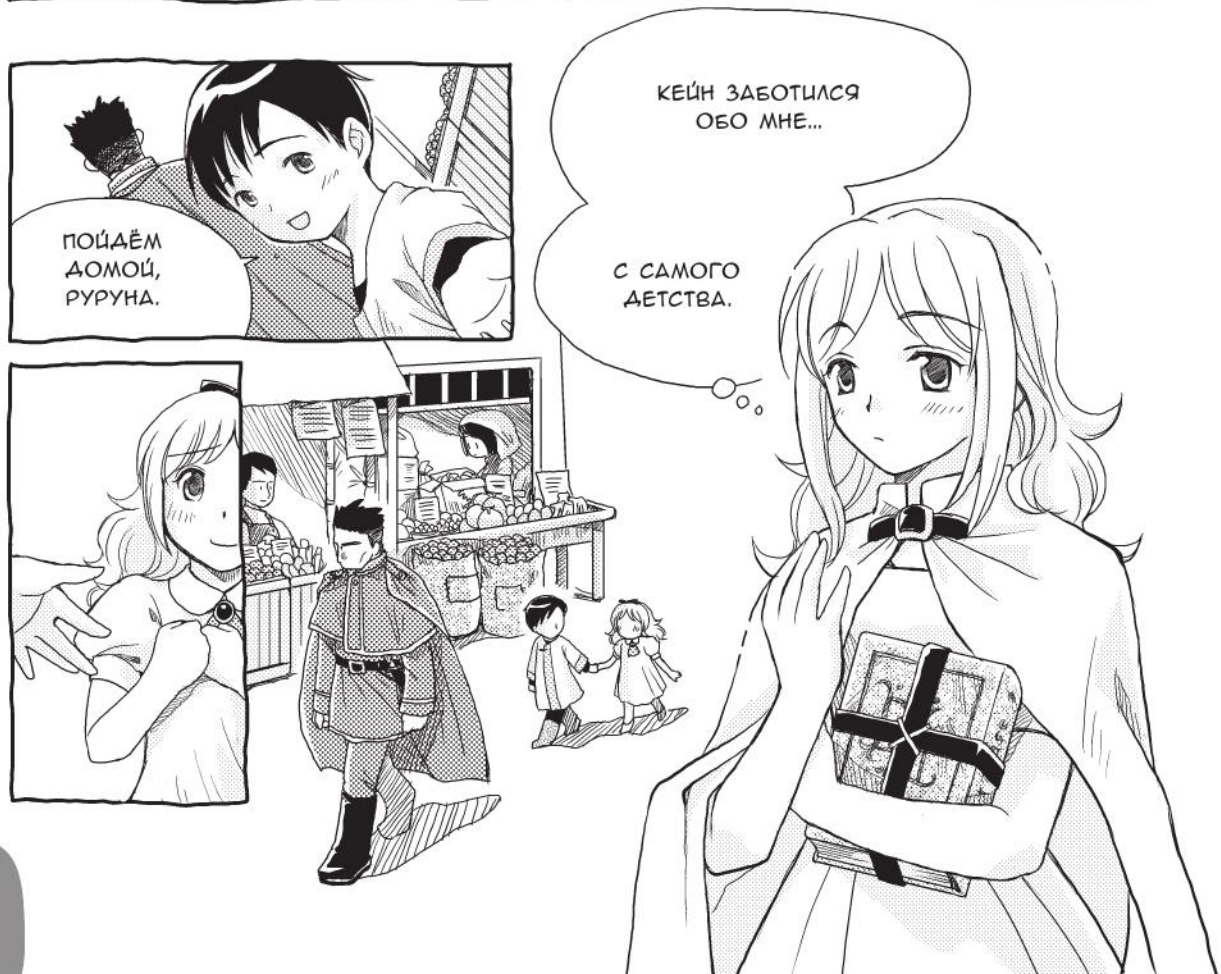
4.

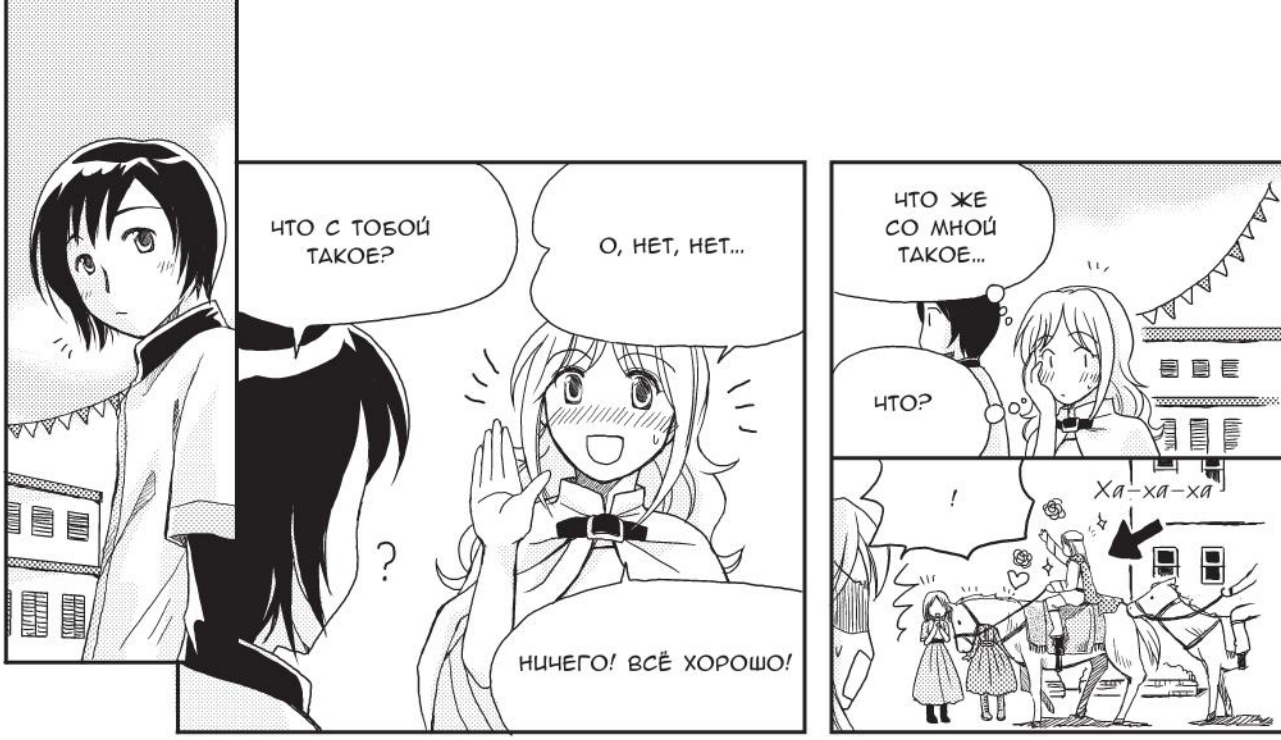
ДАВАЙТЕ ИЗУЧАТЬ SQL!

ПРИМЕНЕНИЕ SQL
ГУЛЯЮ ПО ГОРОДУ И ВСПОМИНАЮ СВОЕ ДЕТСТВО.
ХА-ХА-ХА!
Я ПОМНЮ, КАК ЧАСТО ТЫ ЛИНЯЛА С ЗАНЯТИЙ И ТАЙКОМ ВЫБИРАЛАСЬ ИЗ ЗАМКА.
Много лет назад...
ПРИНЦЕССА! ПРИНЦЕССА РУРУНА!
ТОП-ТОП
入荷
もも
食べごろです
Разве?
НЕЛЬЗЯ ПРОСТО ТАК УБЕГАТЬ ИЗ ЗАМКА, КОГДА ЗАБЛАГОРАССУДИТСЯ.
ФУХ, ФУХ
ПЫХ, ПЫХ

ЗНАЕШЬ ЧТО, КЕЙН? ОСТАВАЛСЯ БЫ ТЫ В ЗАМКЕ, ЕСЛИ ТЕБЕ ТОЛЬКО И НАДО, ЧТО КРИЧАТЬ НА МЕНЯ!
ФУХ
ПЫХ
Н-НО...
СЛЕДОВАТЬ ПОВСЮДУ ЗА ПРИНЦЕССОЙ — ЭТО ОБЯЗАННОСТЬ ПОМОЩНИКА!
Чувство ответственности!
ТЫ ХОДИШЬ ЗА МНОЙ, ПОТОМУ ЧТО ТАК НАДО?
Я-ТО ДУМАЛА, ТЫ ДЕЛАЕШЬ ЭТО, ПОТОМУ ЧТО ТЫ МОЙ ДРУГ.
НУ...
ПОНИМАЕШЬ...
НУ, ТЫ И ПРИДУРОК!
ШУРХ!
НЕТ, Я НЕ ТО ХОТЕЛ СКАЗАТЬ!..
Принцесса...
БАЦ!
УЯ!
ВОТ Я И НАШЁЛ ТЕБЯ, ПРИНЦЕССА РУРУНА!
ОЙ, НАЧАЛЬНИК ОХРАНЫ...
БУМ!

РУРУНА, ТЫ ЗАШЛА СЛИШКОМ ДАЛЕКО.
АЙ!
ВЫ НЕ ПОНЯЛИ!
ШВАХ
ЭТО Я ВИНОВАТ... Я УГОВОРИЛ ПРИНЦЕССУ УБЕЖАТЬ ИЗ ЗАМКА!..
КЕЙН...
???
Да неужели?


ПОЙДЁМ ДОМОЙ, РУРУНА.
КЕЙН ЗАБОТИЛСЯ ОБО МНЕ...
С САМОГО ДЕТСТВА.


ЧТО С ТОБОЙ ТАКОЕ?
О, НЕТ, НЕТ...
?
НИЧЕГО! ВСЁ ХОРОШО!
ЧТО ЖЕ СО МНОЙ ТАКОЕ...
ЧТО?
!
Ха-ха-ха

Принц Раминес...
Ура!
Иго-го
Ха-ха-ха
Какой красавчик!
РАМИНЕС!

КАК ПОЖИВАЕТЕ, МИЛЫЕ ДАМЫ?
ПОДМИГИВАЕТ

О НЕТ, ОПЯТЬ ОН ПРИЕХАЛ.
КЕЙН, СКОРЕЕ СКРОЕМСЯ В ЭТОМ КАФЕ!
Что?
СКРИ-И-ИП
Ха-ха-ха

ДАВАЙ ЗАЙМЁМСЯ БАЗАМИ ДАННЫХ ЗДЕСЬ.

Вот и я!
А вот и вы!
Как хорошо снаружи.
хи-хи
Это что, новое местечко для занятий?!
Фьють
Хлоп
А что там?
Итак, мы остановились на том, что научились проектировать базы данных.
Можно и так сказать.
Дальше нам нужно научиться пользоваться созданной базой данных.
Наконец-то
Тип-гип, Ура!
Рано радуетесь
Как вам уже известно, при использовании базы данных вам надо вводить данные или извлекать их....
...и мы будем делать это с помощью SQL.
SQL
Эс кью эл?
Похоже, непростая штука.
Хи-хи!

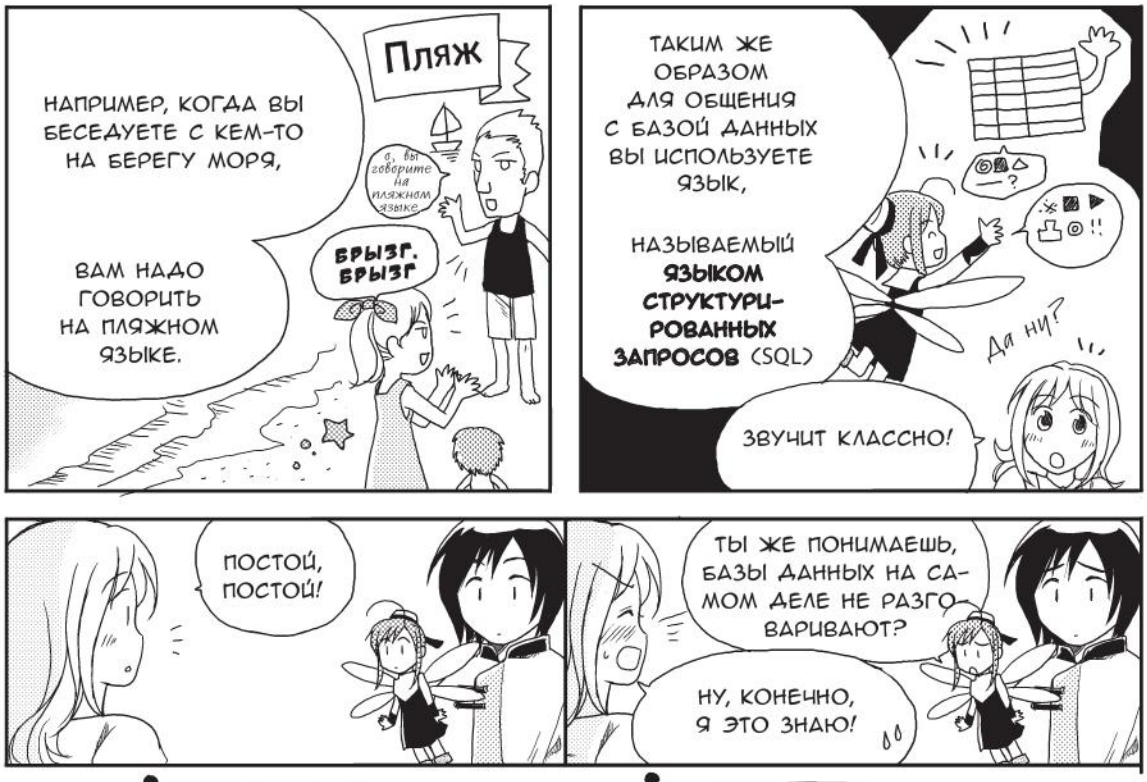


Таблица «Продажи»

Код отчёта	Дата	Код покупателя
1101	3/5	12
1102	3/7	23
1103	3/8	25
1104	3/10	12
1105	3/12	25

Таблица «Покупатели»

Код покупателя	Покупатель
12	Королевство Моро
23	Империя Фарайя
25	Королевство Рамбутан

Таблица «Реализованные товары»

Код отчёта	Код товара	Количество
1101	101	1 100
1101	102	300
1102	103	1 700
1103	104	500
1104	101	2 500
1105	103	2 000
1105	104	700

Таблица «Товары»

Код товара	Название товара	Цена
101	Дыни	800 G
102	Клубника	150 G
103	Яблоки	120 G
104	Лимоны	200 G

А ТЕПЕРЬ ВАМ НАДО С ПОМОЩЬЮ SQL ВНЕСТИ ЭТИ ТАБЛИЦЫ И ДАННЫЕ В БД.
Возможности SQL
• Создание таблиц
• Ввод и извлечение данных
• Управление пользователями
ПРИМЕНЯЯ SQL, ВЫ МОЖЕТЕ "БЕСЕДОВАТЬ" С БАЗОЙ ДАННЫХ, ВЫПОЛНЯЯ ПЕРЕЧИСЛЕННЫЕ ЗАДАЧИ.
Держите! Я же фея, а не канарейка!
Возможности SQL
•Создание таблиц
•Ввод и извлечение данных
•Управление пользователями
ПОХОЖЕ, МЫ МОЖЕМ ДЕЛАТЬ ВСЁ, ЧТО УГОДНО!
Йо-хо-хо
НО... КАЖЕТСЯ, ПРИДЁТСЯ ХОРОШЕНЬКО ПОТРУДИТЬСЯ.
КАКИЕ ПРОБЛЕМЫ!
МЫ УЖЕ ТАК МНОГО ИЗУЧИЛИ!
ТОЧНО.
ЧТО Ж... ВЕРНО... И Я ХОЧУ ВОСПОЛЬЗОВАТЬСЯ БАЗОЙ ДАННЫХ КАК МОЖНО БЫСТРЕЕ.
Вот это сила духа!
Я УЖЕ ВВЕЛА ТАБЛИЦЫ И ДАННЫЕ, КОТОРЫЕ МЫ НЕДАВНО СОСТАВИЛИ.
БИИП
ДАВАЙТЕ ТЕПЕРЬ ИЗВЛЕЧЁМ ИЗ БД КАКИЕ-НИБУДЬ ДАННЫЕ.

ПОИСК ДАННЫХ С ПОМОЩЬЮ КОМАНДЫ SELECT

ЗАПИШЕМ ЭТО ТАК:

```
SELECT название_товара
FROM товары;
```

В ЯЗЫКЕ SQL ОДНА ПРОСЬБА (ЗАПРОС) НАЗЫВАЕТСЯ **КОМАНДОЙ**.

ЭТА SQL-КОМАНДА СОСТОИТ ИЗ ДВУХ ГРУПП: SELECT НАЗВАНИЕ_ТОВАРА И FROM ТОВАРЫ.

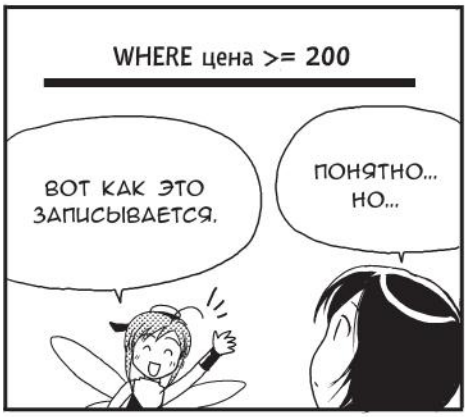
ЭТИ ГРУППЫ НАЗЫВАЮТСЯ **ФРАЗАМИ**.
В SQL ВЫ ЗАДАЁТЕ ИМЯ СТОЛБЦА, КОТОРЫЙ ВЫ ХОТИТЕ ВЫВЕСТИ, С ПОМОЩЬЮ ФРАЗЫ SELECT, И ИМЯ ТАБЛИЦЫ, В КОТОРОЙ НАХОДИТСЯ ЭТОТ СТОЛБЕЦ, С ПОМОЩЬЮ ФРАЗЫ FROM.
FROM
Таблица «товары»
Код товара | Название товара | Цена
101 | Дыни | 800 G
102 | Клубника | 150 G
103 | Яблоки | 120 G
104 | Лимоны | 200 G
SELECT

ВОТ И ПОЛУЧЕННЫЕ ДАННЫЕ.
ТАКИМ ОБРАЗОМ ВЫ МОЖЕТЕ ВЫВЕСТИ ВСЕ НАЗВАНИЯ ТОВАРОВ ИЗ ТАБЛИЦЫ "ТОВАРЫ".
Прошу Вас! ♫
Название товара
Дыни
Клубника
Яблоки
Лимоны

ЗДОРОВО! МЫ БЕСЕДУЕМ С БАЗОЙ ДАННЫХ С ПОМОЩЬЮ SQL.
ДА, ЭТО ТАК. ИСПОЛЬЗУЯ РАЗНЫЕ ВИДЫ ФРАЗ, МОЖНО ВЫВОДИТЬ НЕОБХОДИМЫЕ НАМ ДАННЫЕ.

РАЗНЫЕ ВИДЫ... ХМ.
ХОРОШО, ТОГДА, НАПРИМЕР,

КАК НАСЧЁТ ТОГО, ЧТОБЫ ЗАПРОСИТЬ СПИСОК ТОВАРОВ С ЦЕНОЙ БОЛЬШЕ ИЛИ РАВНОЙ 200 G?
Больше или равно 200 G
?



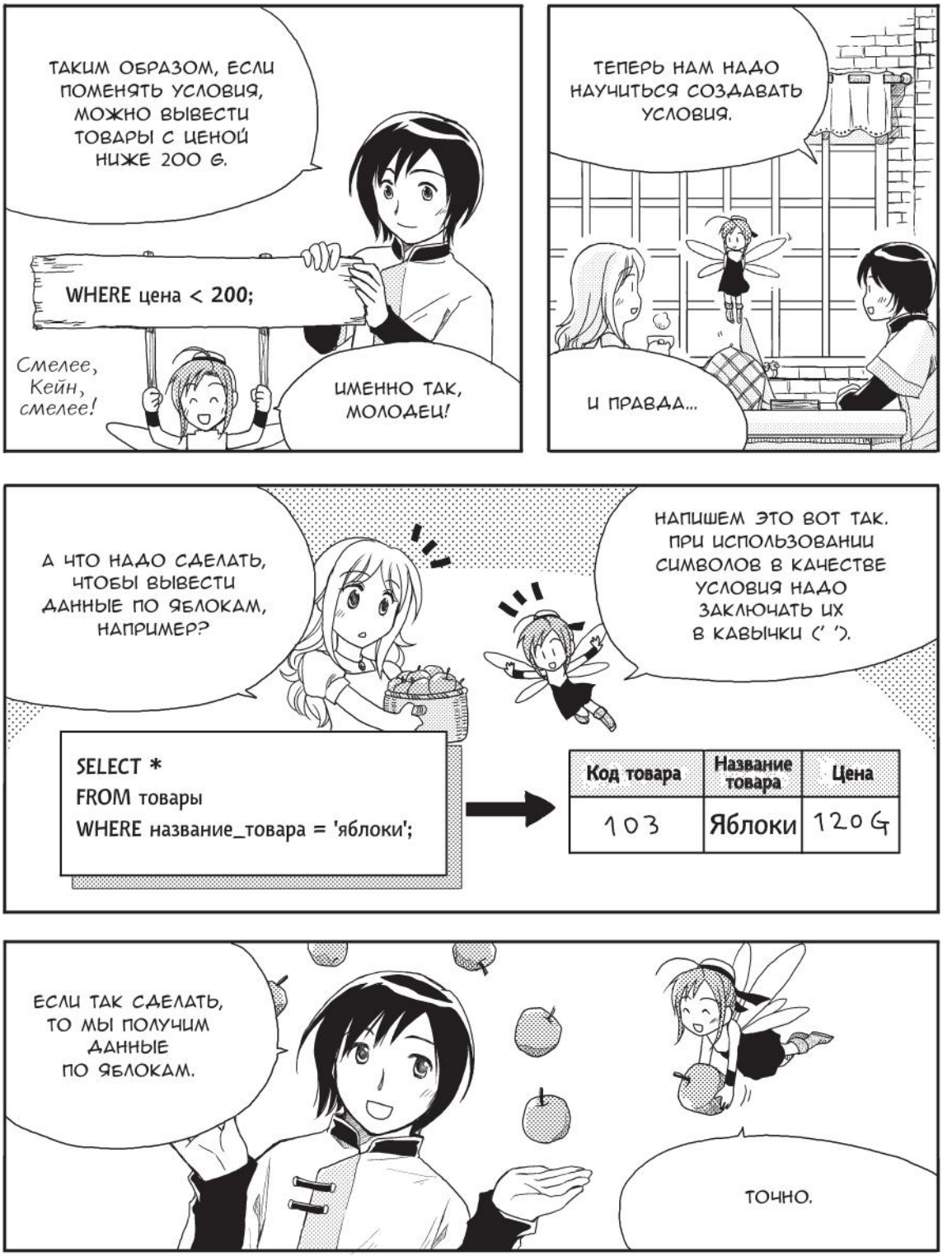
ЭТА КОМАНДА ИЗВЛЕКАЕТ ВСЕ ДАННЫЕ ИЗ ТАБЛИЦЫ "ТОВАРЫ"...

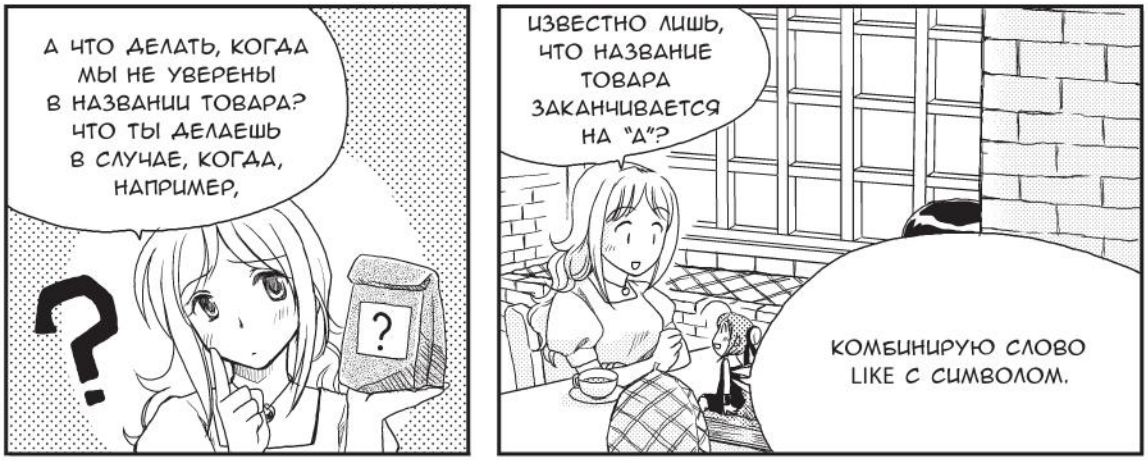
Вот, держите!

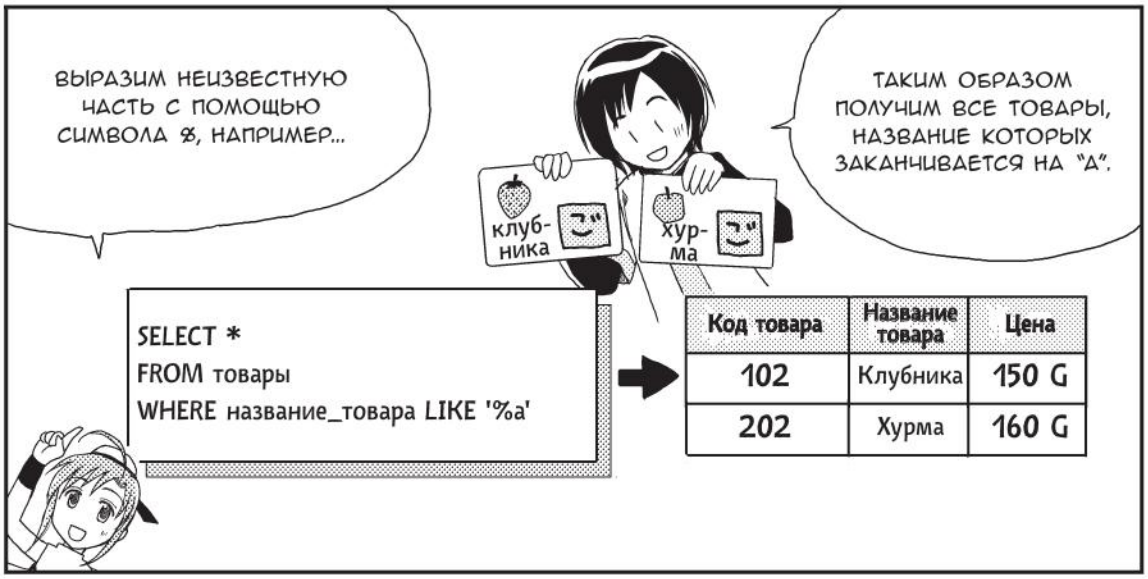
Товары, которые стоят 200 G или более

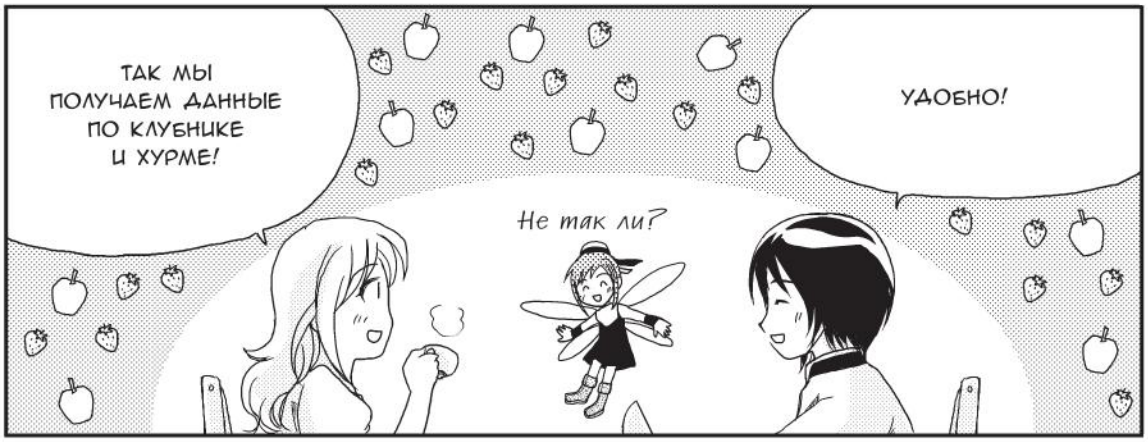
Код товара	Название товара	Цена
101	Дыни	800G
104	Лимоны	200G

...С ЦЕНОЙ БОЛЬШЕ ИЛИ РАВНОЙ 200 G.


ТАКИМ ОБРАЗОМ, ЕСЛИ ПОМЕНЯТЬ УСЛОВИЯ, МОЖНО ВЫВЕСТИ ТОВАРЫ С ЦЕНОЙ НИЖЕ 200 G.
WHERE цена < 200;
Смелее, Кейн, смелее!
ИМЕННО ТАК, МОЛОДЕЦ!
ТЕПЕРЬ НАМ НАДО НАУЧИТЬСЯ СОЗДАВАТЬ УСЛОВИЯ.
И ПРАВДА...
А ЧТО НАДО СДЕЛАТЬ, ЧТОБЫ ВЫВЕСТИ ДАННЫЕ ПО ЯБЛОКАМ, НАПРИМЕР?
НАПИШЕМ ЭТО ВОТ ТАК. ПРИ ИСПОЛЬЗОВАНИИ СИМВОЛОВ В КАЧЕСТВЕ УСЛОВИЯ НАДО ЗАКЛЮЧАТЬ ИХ В КАВЫЧКИ (' ').
SELECT *
FROM товары
WHERE название_товара = 'яблоки';
Код товара
Название товара
Цена
103
Яблоки
120G
ЕСЛИ ТАК СДЕЛАТЬ, ТО МЫ ПОЛУЧИМ ДАННЫЕ ПО ЯБЛОКАМ.
ТОЧНО.

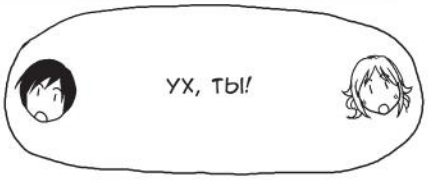

А ЧТО ДЕЛАТЬ, КОГДА МЫ НЕ УВЕРЕНЫ В НАЗВАНИИ ТОВАРА? ЧТО ТЫ ДЕЛАЕШЬ В СЛУЧАЕ, КОГДА, НАПРИМЕР,
ИЗВЕСТНО ЛИШЬ, ЧТО НАЗВАНИЕ ТОВАРА ЗАКАНЧИВАЕТСЯ НА "А"?
КОМБИНИРУЮ СЛОВО LIKE С СИМВОЛОМ.


ВЫРАЗИМ НЕИЗВЕСТНУЮ ЧАСТЬ С ПОМОЩЬЮ СИМВОЛА %, НАПРИМЕР...
ТАКИМ ОБРАЗОМ ПОЛУЧИМ ВСЕ ТОВАРЫ, НАЗВАНИЕ КОТОРЫХ ЗАКАНЧИВАЕТСЯ НА "А".
клуб-ника
хур-ма
SELECT *
FROM товары
WHERE название_товара LIKE '%а'
Код товара | Название товара | Цена
102 | Клубника | 150 G
202 | Хурма | 160 G


ТАК МЫ ПОЛУЧАЕМ ДАННЫЕ ПО КЛУБНИКЕ И ХУРМЕ!
Не так ли?
УДОБНО!

Агрегатные функции в SQL

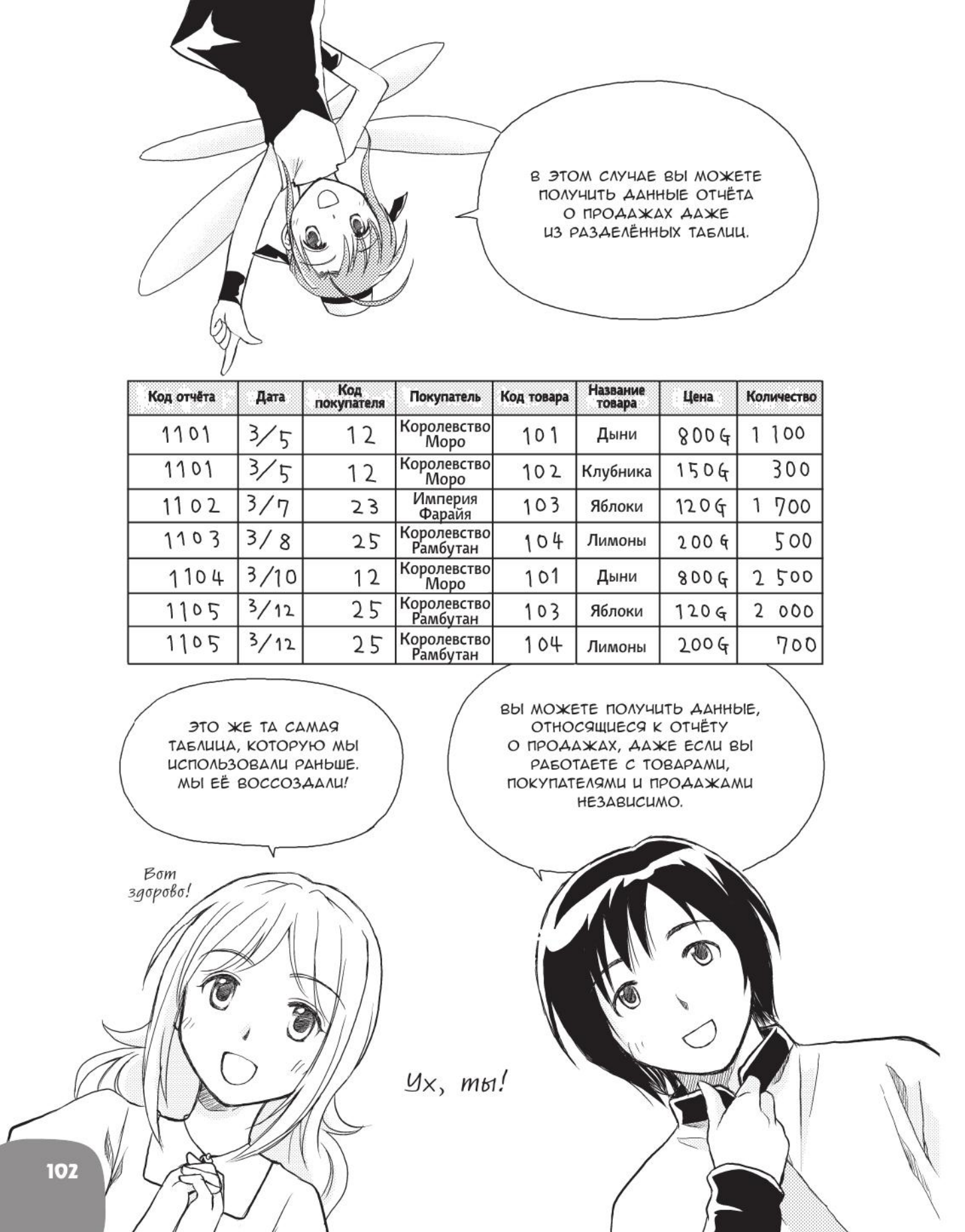
Функция	Описание
COUNT (*)	Получить число строк
COUNT (имя_столбца)	Получить количество непустых ячеек в столбце
COUNT (DISTINCT имя_столбца)	Получить число неодинаковых значений в столбце
SUM (имя_столбца)	Получить сумму значений во всех ячейках столбца
AVG (имя_столбца)	Получить среднее значение от значений в ячейках столбца
MAX (имя_столбца)	Получить максимальное значение в столбце
MIN (имя_столбца)	Получить минимальное значение в столбце



А МОЖНО УЗНАТЬ, СКОЛЬКО ВИДОВ ФРУКТОВ МЫ ПРОДАЁМ?
ВОСПОЛЬЗУЙСЯ КОМАНДОЙ COUNT!
ШУХ, ШУХ
COUNT
А ЧТО СДЕЛАТЬ, ЧТОБЫ НАЙТИ ТОВАР С МАКСИМАЛЬНОЙ ЦЕНОЙ?
ВОСПОЛЬЗУЙСЯ КОМАНДОЙ MAX.
ШУХ, ШУХ
MAX
SQL ПОЗВОЛЯЕТ АГРЕГИРОВАТЬ ДАННЫЕ, ЗНАЧИТ, МЫ МОЖЕМ ПОЛУЧИТЬ КУЧУ ВСЯКОЙ ИНФОРМАЦИИ.
ВЫ ЕЩЁ МОЖЕТЕ ГРУППИРОВАТЬ ДАННЫЕ ДЛЯ ДАЛЬНЕЙШИХ МАНИПУЛЯЦИЙ ИЛИ АНАЛИЗА.
Я ПОНЯЛА. АГРЕГАТНЫЕ ФУНКЦИИ — ЭТО КРУТО.
НО...
КАК НАСЧЁТ НАШЕГО ОТЧЁТА О ПРОДАЖАХ?
Опять эти странные очки!
НУ, ЧТО Ж...
ДАВАЙТЕ СОЗДАДИМ ОТЧЁТ О ПРОДАЖАХ С ПОМОЩЬЮ SQL.
БУМ!

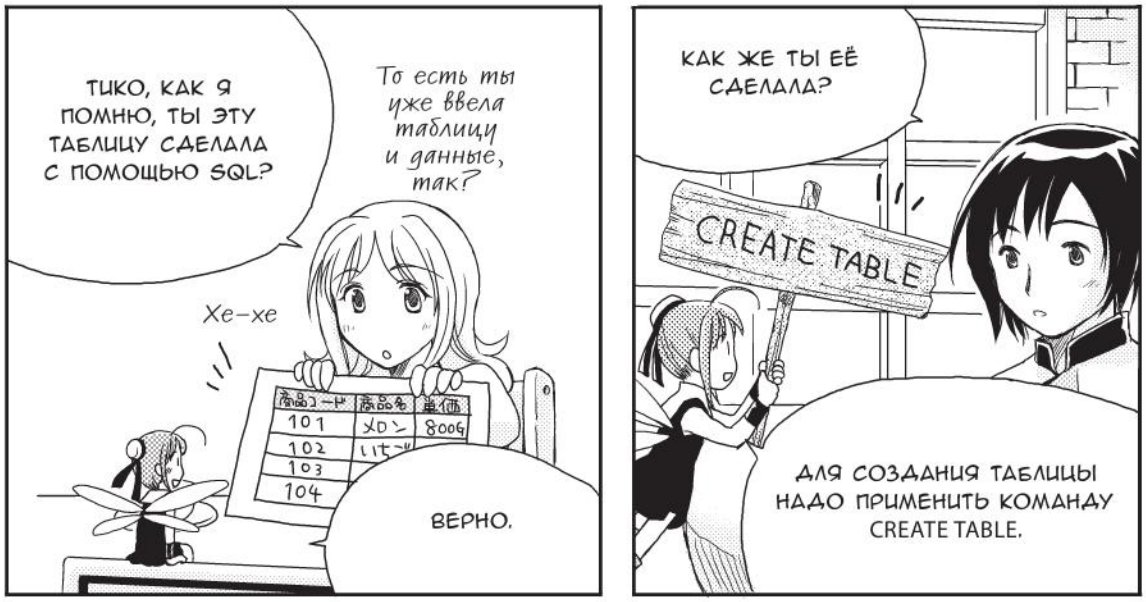
СОЕДИНЕНИЕ ТАБЛИЦ

```
SELECT   продажи.код_отчёта,дата,продажи.код_покупателя,
         покупатель,реализованные_товары.
         код_товара,название_товара,цена,количество
FROM     продажи,реализованные_товары,товары,покупатели
WHERE    продажи.код_отчёта = реализованные_товары.код_отчёта
         AND
         реализованные_товары.код_товара = товары.код_товара
         AND
         покупатели.код_покупателя = продажи.код_покупателя
```



Код отчёта	Дата	Код покупателя	Покупатель	Код товара	Название товара	Цена	Количество
1101	3/5	12	Королевство Моро	101	Дыни	800G	1 100
1101	3/5	12	Королевство Моро	102	Клубника	150G	300
1102	3/7	23	Империя Фарайя	103	Яблоки	120G	1 700
1103	3/8	25	Королевство Рамбутан	104	Лимоны	200G	500
1104	3/10	12	Королевство Моро	101	Дыни	800G	2 500
1105	3/12	25	Королевство Рамбутан	103	Яблоки	120G	2 000
1105	3/12	25	Королевство Рамбутан	104	Лимоны	200G	700

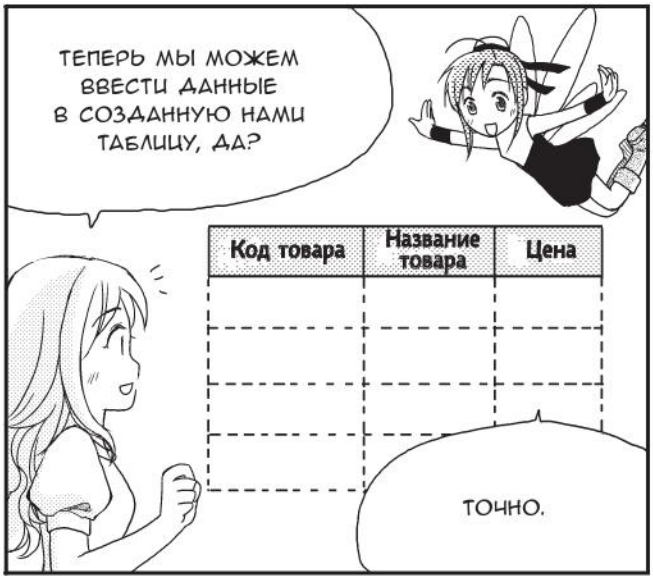
СОЗДАНИЕ ТАБЛИЦ



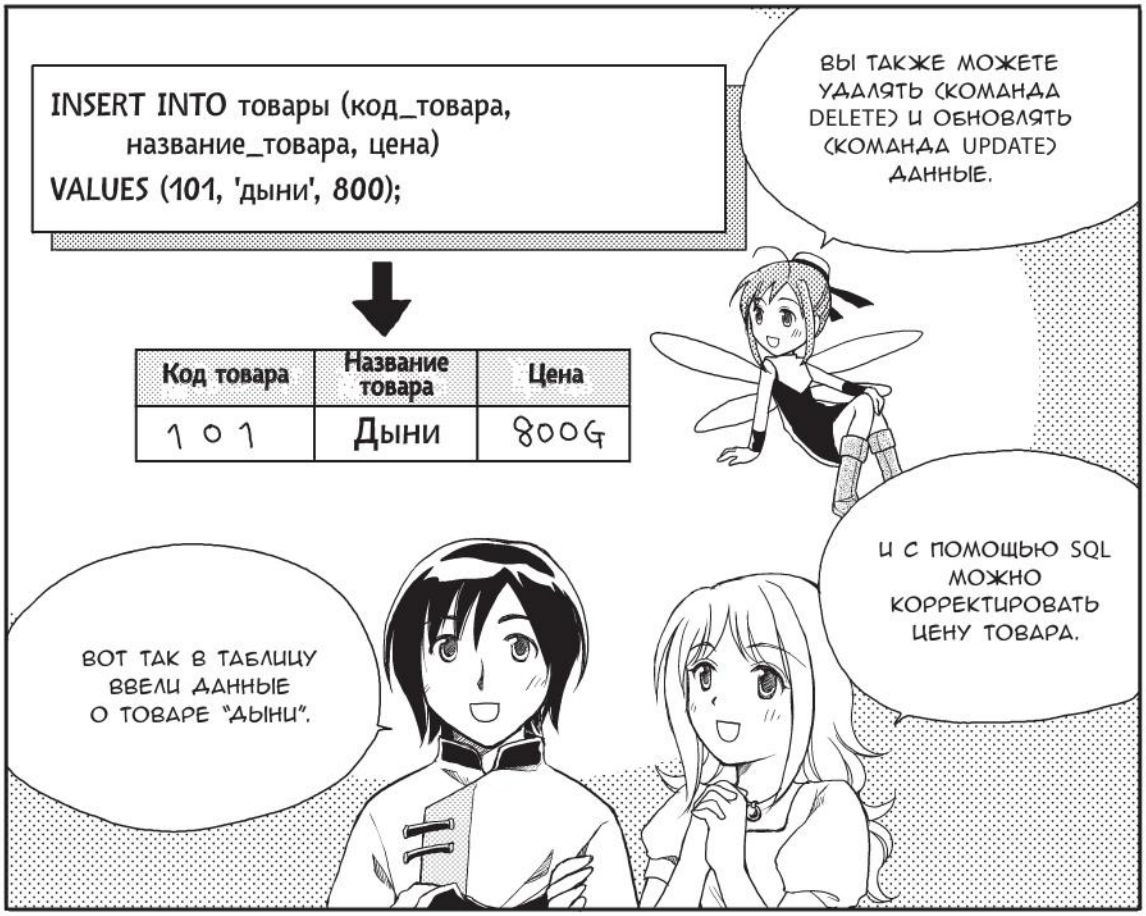

```
CREATE TABLE товары
(
код_товара int NOT NULL,
название_товара varchar(255),
цена int,
PRIMARY KEY (код_товара)
);
```

Код товара	Название товара	Цена

Полное объяснение команды CREATE TABLE смотри на стр. 117.


ТЕПЕРЬ МЫ МОЖЕМ ВВЕСТИ ДАННЫЕ В СОЗДАННУЮ НАМИ ТАБЛИЦУ, ДА?
Код товара
Название товара
Цена
ТОЧНО.

ДЛЯ ДОБАВЛЕНИЯ ДАННЫХ НАДО ИСПОЛЬЗОВАТЬ КОМАНДУ INSERT.
INSERT


INSERT INTO товары (код_товара, название_товара, цена)
VALUES (101, 'дыни', 800);
Код товара
Название товара
Цена
101
Дыни
800G
ВЫ ТАКЖЕ МОЖЕТЕ УДАЛЯТЬ (КОМАНДА DELETE) И ОБНОВЛЯТЬ (КОМАНДА UPDATE) ДАННЫЕ.
ВОТ ТАК В ТАБЛИЦУ ВВЕЛИ ДАННЫЕ О ТОВАРЕ "ДЫНИ".
И С ПОМОЩЬЮ SQL МОЖНО КОРРЕКТИРОВАТЬ ЦЕНУ ТОВАРА.

НАДЕЮСЬ, ВЫ СПРАВИТЕСЬ С УПРАВЛЕНИЕМ ЭКСПОРТОМ ФРУКТОВ С ПОМОЩЬЮ БАЗЫ ДАННЫХ.
КОМАНДА SELECT — ЭТО САМАЯ ВАЖНАЯ ЧАСТЬ SQL. СМОТРИТЕ, УЧИТЕСЬ ХОРОШО.
У нас ещё многое впереди...
Хорошо.
ЕСЛИ ПОЛНОСТЬЮ ЗАДЕЙСТВОВАТЬ SQL, ТО, МОЖЕТ, У НАС И ПОЛУЧИТСЯ УПРАВЛЯТЬ БД...
БОЖЕ, ПОСМОТРИ НА ЧАСЫ!
НАМ ПОРА ВОЗВРАЩАТЬСЯ В ЗАМОК, ПОКА НЕ СТЕМНЕЛО,
ИНАЧЕ НА МЕНЯ СНОВА БУДЕТ КРИЧАТЬ НАЧАЛЬНИК ОХРАНЫ.
Собираемся
Да, да, собираемся
ДА, ВОЗМОЖНО...
СТАРЫЙ ЗАДИРА...
КЕЙН!
ЛЯЗГ ЛЯЗГ...
ДАВАЙ БЫСТРЕЕ, Я УХОЖУ.
ЛАДНО, ПОШЛИ.

ОБЗОР ЯЗЫКА SQL

В этой главе принцесса Руруна и Кейн узнали о языке структурированных запросов SQL (Structured Query Language). Этот язык используется для работы с реляционными базами данных. Команды SQL можно разбить на три отдельных типа:

① Язык описания данных DDL (Data Definition Language) создает таблицы.
② Язык манипулирования данными DML (Data Manipulation Language) вводит и извлекает данные.
③ Язык управления данными DCL (Data Control Language) управляет пользовательским доступом.

① В SQL есть команды, создающие оболочку БД, и команда, которая создает таблицу в самой БД. Также с помощью этого языка можно изменить и удалить таблицу. Язык базы данных, в котором есть эти функции, называется **языком описания данных** DDL (Data Definition Language).

② В SQL есть также команды, которые манипулируют данными в БД, например добавляют, удаляют и обновляют данные. Здесь также есть команда, осуществляющая поиск данных. Язык базы данных, в котором есть эти функции, называется **языком манипулирования данными** DML (Data Manipulation Language).

③ Плюс к этому SQL предоставляет возможность управлять БД так, чтобы не возникала противоречивость данных, даже если базой одновременно пользуются много людей. Язык базы данных, связанный с этими функциями, называется **языком управления данными** DCL (Data Control Language).

ПОИСК ДАННЫХ С ПОМОЩЬЮ КОМАНДЫ SELECT

Принцесса Руруна и Кейн начали изучать SQL с того, что познакомились с основной функцией поиска данных. SQL осуществляет поиск данных тогда, когда вводится одна команда (комбинация фраз). Например, чтобы найти товары с ценой 200 G, надо применить следующую SQL-команду.

```
SELECT *
FROM товары
WHERE цена = 200;
```

Создать SQL-команду, скомбинировав фразы

Команда SELECT — одна из основных в SQL. Она задаёт, какой столбец, из какой таблицы (FROM) и соответствующий каким условиям (WHERE). Вы можете комбинировать эти фразы и интуитивно составлять команды-запросы с помощью SQL. Даже незнакомый с базами данных пользователь может использовать их для поиска данных.

СОЗДАНИЕ УСЛОВИЙ

Кейн недавно сказал: «Теперь нам надо научиться создавать условия». Давайте рассмотрим некоторые способы создания условий в SQL.

ОПЕРАТОРЫ СРАВНЕНИЯ

Один из способов создания условия заключается в использовании операторов сравнения, таких как >= и =. Например, условие «A больше или равно B» выражается с помощью символов >=, а условие «A равно B» — с помощью символа =. Ниже в таблице приведены ещё и другие операторы сравнения.

■ ОПЕРАТОРЫ СРАВНЕНИЯ

Оператор сравнения	Описание	Пример	Описание примера
A = B	A равно B	цена = 200	Цена равна 200 G
A > B	A больше B	цена > 200	Цена больше 200 G
A >= B	A больше или равно B	цена >= 200	Цена больше или равна 200 G
A < B	A меньше B	цена < 200	Цена меньше 200 G
A <= B	A меньше или равно B	цена <= 200	Цена меньше или равна 200 G
A <> B	A не равно B	цена <> 200	Цена не равна 200 G

ЛОГИЧЕСКИЕ ОПЕРАТОРЫ

В некоторых случаях вам надо задать условия, более сложные, чем простое сравнение. Тогда вы можете использовать логические операторы AND, OR и NOT, с их помощью комбинировать операторы сравнения и создавать более сложные условия, как это показано в таблице.

■ ЛОГИЧЕСКИЕ ОПЕРАТОРЫ

Логический оператор	Описание	Пример	Описание примера
AND	A и B	код_товара >= 200 AND цена = 100	Код товара больше или равен 200, и цена равна 100 G
OR	A или B	код_товара >= 200 OR цена = 100	Код товара больше или равен 200, или цена равна 100 G
NOT	не A	NOT цена = 100	Цена не равна 100 G

ШАБЛОНЫ

Когда вы точно не знаете, что искать, можно использовать сравнение по шаблонам с помощью подстановочных знаков (метасимволов). Когда вы сравниваете по шаблонам, в предложении LIKE используйте символы % или _ : тогда будет идти поиск строки символов, которая совпадает с заданным вами шаблоном. Для поиска значения, соответствующего частично заданной строке символов, используйте %, обозначающий строку символов любой длины, и _, обозначающий всего один символ.

Пример запроса с использованием подстановочных символов показан ниже. Здесь идёт поиск строки символов, относящейся к названию товара и имеющей на конце букву «а».

```
SELECT *
FROM товары
WHERE название_товара LIKE '%a';
```

Это предложение сравнивает по шаблону с помощью подстановочного знака.

Код товара	Название товара	Цена
102	Клубника	150 G
202	Хурма	160 G

Ниже приводятся и объясняются подстановочные знаки, которыми можно пользоваться в SQL-командах.

■ ПОДСТАНОВОЧНЫЕ ЗНАКИ

Подстановочный знак	Описание	Пример шаблона	Совпадающая с шаблоном строка символов
%	Соответствует любому числу символов	%a a%	клубника, хурма, арбузы, апельсины
_	Соответствует одному символу	_н н_	он на

ПОИСК

Есть и другие методы поиска. Например, вы можете задать BETWEEN X AND Y для поиска в диапазоне значений. Если задать диапазон, как показано ниже, можно извлечь данные о товарах, у которых цена больше или равна 150 G или ниже 200 G.

```
SELECT *
FROM товары
WHERE цена
BETWEEN 150 AND 200;
```

Задает диапазон поиска

Кроме этого, при поиске строк можно задать IS NULL. Если произвести поиск так, как показано ниже, то можно вывести данные о товарах, у которых цена не определена.

```
SELECT *
FROM товары
WHERE цена IS NULL;
```

Поиск неопределённого значения

ВОПРОСЫ И ЗАДАНИЯ

А теперь создадим SQL-предложение с использованием различных условий. Давайте возьмём таблицу «Покупатели» (за единицу населения примем 10 000 человек). Вам надо ответить на вопросы с помощью SQL-команд. Ответы вы найдёте на стр. 125-128.

■ ТАБЛИЦА «ПОКУПАТЕЛИ»

Код покупателя	Покупатель	Население
12	Королевство Моро	100
23	Империя Фарайя	120
25	Королевство Рамбутан	150
32	Королевство Кивано	80

Для ответов на вопросы 1-6 используйте SQL.

B1

Извлеките из приведённой выше таблицы страны с населением больше либо равным 1 миллиону человек.

Код покупателя	Покупатель	Население
12	Королевство Моро	100
23	Империя Фарайя	120
25	Королевство Рамбутан	150

B2

Извлеките из приведённой выше таблицы страны с населением меньше 1 млн.

Код покупателя	Покупатель	Население
30	Королевство Кивано	80

B3

Найдите страны, у которых код покупателя меньше 20, а население больше или равно 1 млн.

B4

Найдите страны, где код покупателя больше или равен 30, а население больше 1 млн.

B5

Узнайте численность населения королевства Рамбутан.

B6

Найдите страны, у которых в названии есть буква «н».

АГРЕГАТНЫЕ ФУНКЦИИ

Принцесса Руруна и Кейн изучили различные агрегатные функции. Их ещё называют функциями множеств. Вы можете использовать эти функции для агрегирования информации, такой как максимальные и минимальные значения, число элементов и сумма.

Если вы задаёте фразу WHERE вместе с агрегатной функцией, вы можете получить агрегатное значение только для заданных строк. Если вы задаёте фразу, как это показано в примере, вы можете узнать число товаров, у которых цена больше или равна 200 G.

```
SELECT COUNT(*)
FROM  Товары
WHERE цена>=200;
```

COUNT(*)
2

АГРЕГИРОВАНИЕ ДАННЫХ. ГРУППИРОВАНИЕ

Вы можете легко получить агрегатные значения с помощью группирования данных. Например, если вам надо получить число товаров и среднюю цену товара в зависимости от района, вы можете использовать функцию группирования.

Чтобы сгруппировать данные, совместите агрегатную функцию и фразу GROUP BY. Давайте используем для этого таблицу товаров.

■ ТАБЛИЦА «ТОВАРЫ»

Код товара	Название товара	Цена	Район
101	Дыни	800 G	Южный регион
102	Клубника	150 G	Центр
103	Яблоки	120 G	Северный регион
104	Лимоны	200 G	Южный регион
201	Каштаны	100 G	Северный регион
202	Хурма	160 G	Центр
301	Персики	130 G	Южный регион
302	Киви	200 G	Южный регион

Чтобы получить среднее значение цены товара для каждого района, надо задать столбец «Район» и функцию AVG для фразы GROUP BY. Тогда данные будут сгруппированы по району и вы получите среднее значение цены в каждом районе.

```
SELECT район,AVG (цена)
FROM товары
GROUP BY район;
```

Позволяет группировать

Район	AVG (цена)
Южный регион	332,5
Северный регион	110
Центр	155

А что, если мы хотим и дальше накладывать ограничения на результаты выборки, учитывая определённые свойства данных? Предположим, мы хотим найти товары по регионам со средней ценой товара больше или равной 200 G. В этом случае не надо задавать условие во фразе WHERE, а вместо неё воспользуйтесь фразой HAVING. Она позволит вам вывести только те районы, где средняя цена больше или равна 200 G.

```
SELECT район,AVG(цена)
FROM товар
GROUP BY район;
HAVING AVG(цена)>=200;
```

Район	AVG (цена)
Южный регион	332,5

Сначала данные группируются, потом фильтруются

ВОПРОСЫ И ЗАДАНИЯ

Ответьте на приведённые ниже вопросы, используя таблицу покупателей (положим, что за единицу населения мы приняли 10 000 человек). Ответы даны на стр. 125…127.

■ ТАБЛИЦА «ПОКУПАТЕЛИ»

Код покупателя	Покупатель	Население	Район
12	Королевство Моро	100	Южный регион
15	Королевство Дуриан	200	Центр
22	Королевство Кумкват	160	Северный регион
23	Империя Фарайя	120	Северный регион
25	Королевство Рамбутан	150	Южный регион
30	Королевство Кивано	240	Северный регион
31	Королевство Куруба	80	Южный регион
33	Королевство Черимойя	300	Центр

В7

Где самая маленькая численность населения?

В8

Где самая большая численность населения?

В9

Какова суммарная численность населения всех стран, входящих в таблицу покупателей?

B10

Какова суммарная численность населения стран, у которых код страны больше 20?

B11

Сколько в таблице стран с населением больше или равным 1 млн человек?

B12

Сколько стран входят в Северный регион?

B13

В какой стране Северного региона самое большое население?

B14

Какова суммарная численность населения стран, не считая королевства Рамбутан?

B15

Найдите регионы, в которых среднее значение численности населения больше либо равно 2 млн.

B16

Найдите регионы, в которые входят как минимум три страны.

ПОИСК ДАННЫХ

В SQL существуют более сложные способы запроса данных, кроме тех, что мы уже рассмотрели.

ИСПОЛЬЗОВАНИЕ ПОДЗАПРОСА

Например, вы встраиваете один запрос в другой. Это называется подзапрос (subquery). Давайте рассмотрим таблицы товаров и реализованных товаров.

■ ТАБЛИЦА «ТОВАРЫ»

Код товара	Название товара	Цена
101	Дыни	800 G
102	Клубника	150 G
103	Яблоки	120 G
104	Лимоны	200 G

■ ТАБЛИЦА «РЕАЛИЗОВАННЫЕ ТОВАРЫ»

Код отчёта	Код товара	Количество
1101	101	1 100
1101	102	300
1102	103	1 700
1103	104	500
1104	101	2 500
1105	103	2 000
1105	104	700

С помощью этих таблиц вы можете искать названия товаров, объём продаж которых больше или равен 1000. Приведённая ниже SQL-команда проведёт такой поиск.

```
SELECT * FROM товары
WHERE Код_товара IN
(SELECT Код_товара
FROM Реализованные_товары
WHERE Количество>=1000);
```

Эта команда содержит подзапрос

В этой SQL-команде в первую очередь выполянется команда SELECT в круглых скобках. Сначала идёт поиск кода товара в таблице реализованных товаров, и возвращаются коды товара 101 и 103 (так как только в этих отчётах значится объём продаж больше, чем 1000). В предложении SELECT, которое находится за скобками, эти коды товаров используются как часть условия. Для оператора IN условие выполнено, если строка совпадает с каким-либо значением, заключённым в круглые скобки. Следовательно, в результате выведутся товары, соответствующие кодам товара 101 и 103.

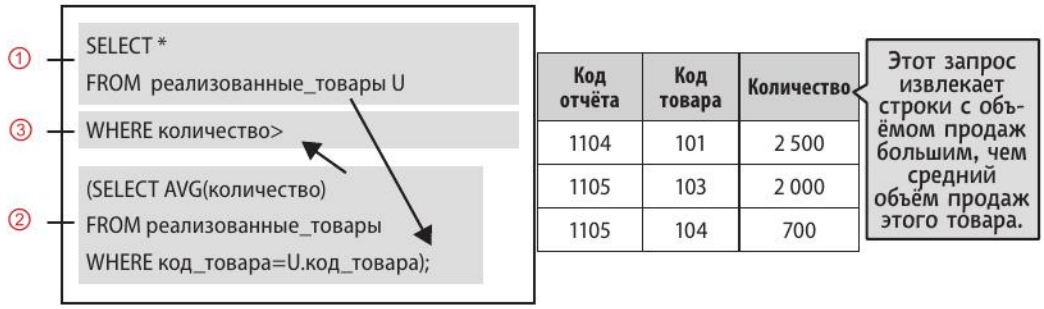
Другими словами, в случае подзапроса результат выполнения команды SELECT в скобках будет передан второй команде SELECT для продолжения поиска. Ниже приведён конечный результат запроса.

Код товара	Товар	Цена
101	Дыни	800 G
103	Яблоки	120 G

КОРРЕЛИРОВАННЫЙ ПОДЗАПРОС

Давайте рассмотрим подзапрос, который содержится внутри другого запроса. Такой подзапрос может обращаться к данным из внешнего запроса и называется коррелированным запросом. В запросе, показанном ниже, таблице «Реализованные товары» во внешнем запросе временно присвоено новое имя U, поэтому подзапрос может обращаться к ней однозначно. Синтаксис U.код_товара указывает, какой столбец «код товара» имеется в виду, так как внутри запроса есть два источника для этого столбца.

Так как подзапрос обращается к данным из внешнего запроса, подзапрос не является независимым от внешнего запроса, как в предыдущих примерах. Такая зависимость называется **корреляцией**.



Давайте посмотрим, как этот коррелированный подзапрос обрабатывается. В данном случае внешний запрос выполняется в первую очередь.

```
① —  SELECT *
     FROM реализованные_товары U
```

Этот результат передаётся внутреннему запросу, чтобы проанализировать его построчно. Давайте рассмотрим анализ первой строки, где код_товара равен 101.

```
② —  (SELECT AVG(количество)
     FROM реализованные_товары
     WHERE код_товара=101)
```

Код товара в первой строке равен 101, что соответствует товару «дыни» — средний объём продаж по дыням равен (2500+1100)/2=1800. Теперь этот результат передаётся в качестве условия для внешнего запроса.

```
③ —  WHERE количество> (1 800)
```

Этот процесс повторяется для каждой строки в таблице реализованных товаров. Шаги 2 и 3 выполняются для всех возможных кодов товара. Другими словами, этот запрос выводит отчёты, в которых объём продаж фрукта (количество) больше, чем средний объём продаж этого же фрукта. Следовательно, извлекаются только пятая, шестая и седьмая строки шага 1.

ВОПРОСЫ И ЗАДАНИЯ

Теперь ответьте на следующие вопросы с использованием таблиц «Товары» и «Реализованные товары». Ответы вы можете найти на стр. 125…127.

B17

Найдите данные о реализованных товарах (фрукты), у которых цена больше или равна 300 G, и получите следующую таблицу.

Код отчёта	Код товара	Количество
1101	101	1 100
1104	101	2 500

B18

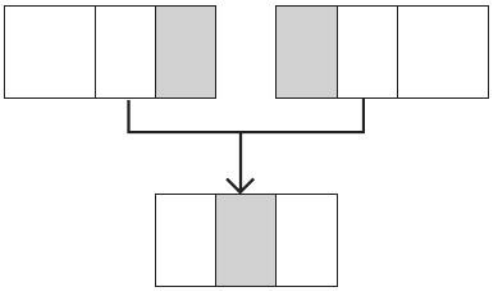
Получите среднее значение объёма продаж по каждому товару и найдите записи, где объём продаж меньше среднего значения.

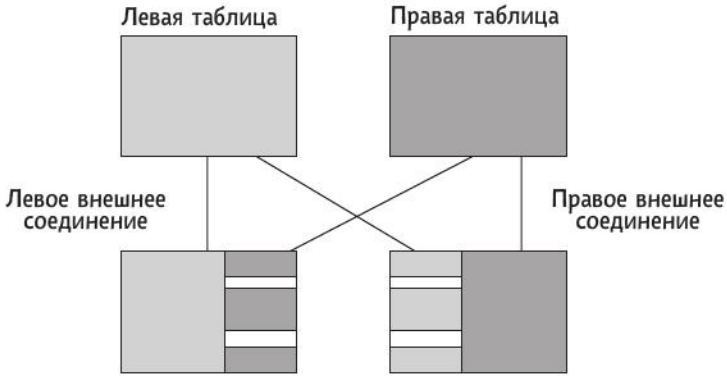
СОЕДИНЕНИЕ ТАБЛИЦ

После того, как принцесса Руруна и Кейн провели поиск с помощью SQL, они создали отчёт о продажах, соединив таблицы. Соединение таблиц путём комбинирования столбцов с одинаковыми именами называется **equi-соединением** (eque join). При таком соединении строки с одинаковыми значениями определены как условия соединения для соединяемых таблиц. Соединение одноимённых столбцов в один называется **естественным соединением** (natura join).

Метод соединения, в котором выбираются только строки с общим значением, как в equi-соединении, называется **внутренним соединением** (inner join).

И наоборот, метод соединения, который сохраняет все строки одной таблицы и задаёт NULL для строк, не входящих в другую таблицу, называется **внешним соединением** (outer join). Если вы поместите таблицу, созданную внешним соединением, справа или слева в SQL-команде, то получится **левое внешнее соединение** (left outer join) или **правое внешнее соединени**е (right outer join), в зависимости от того, какие строки сохраняются.





СОЗДАНИЕ ТАБЛИЦ

Наконец-то принцесса Руруна и Кейн узнали о синтаксисе команды, которая создаёт таблицу, — CREATE TABLE. Синтаксис CREATE TABLE зависит от того, какой тип БД вы используете. Ниже дан пример.

```
CREATE TABLE товары
(
код_товара int NOT NULL,
название_товара varchar(255),
цена int,
PRIMARY KEY (код_товара)
);
```

Эта команда создаёт таблицу

При создании таблицы вы должны задать имена столбцов. Плюс к этому вы можете задавать первичный и внешний ключи для каждого столбца. Например, код товара задан как первичный ключ (PRIMARY KEY), и значение кода товара не может быть нулевым. При создании таблицы вам, возможно, понадобится включить следующие спецификации:

■ ОГРАНИЧЕНИЯ ДЛЯ ТАБЛИЦЫ

Ограничения	Описание
PRIMARY KEY	Устанавливает первичный ключ
UNIQUE	Должен быть уникальным
NOT NULL	Не принимает значение NULL
CHECK	Проверяет диапазон
DEFAULT	Устанавливает значение по умолчанию
FOREIGN KEY/REFERENCES	Устанавливает внешний ключ

Эти спецификации называются **ограничениями** (constraints). Установка ограничений при создании таблицы помогает избежать противоречивых данных в дальнейшем, а также помогает корректной работе с БД.

ВСТАВКА, ОБНОВЛЕНИЕ И УДАЛЕНИЕ СТРОК

Для вставки, обновления или удаления данных из таблицы, созданной с помощью команды CREATE TABLE, вы можете использовать команды INSERT, UPDATE и DELETE. Давайте посмотрим, как с помощью SQL можно вставить, обновить или удалить данные.

```
INSERT INTO товары(код_товара, название_товара, цена)
VALUES(202,'вишня',200);
```

Эта команда добавляет вишню

```
UPDATE товары
SET название_товара='мускатные дыни'
WHERE название_товара='дыни';
```

Эта команда меняет дыни на мускатные дыни

```
DELETE FROM товары
WHERE название_товара='яблоки';
```

Эта команда удаляет яблоки

Код товара	Название товара	Цена
101	Мускатные дыни	800 G
102	Клубника	150 G
103	Яблоки	120 G
104	Лимоны	200 G
200	Вишня	200 G

Обновлено на мускатные дыни

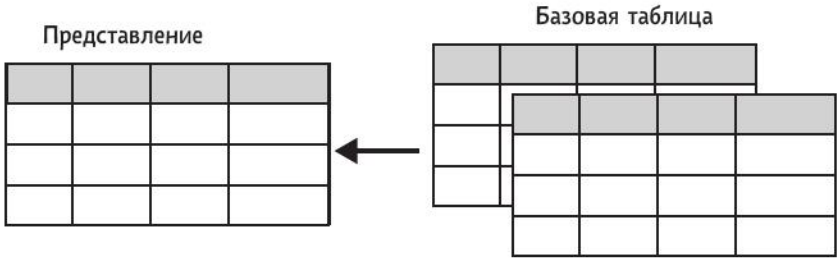
Удалено яблоки

Добавлена вишня

При вставке, обновлении или удалении строки вы не можете нарушать ограничения, установленные оператором CREATE TABLE. Если товар с кодом товара 200 уже существует, вы не сможете добавить вишню, так как нельзя добавлять дублирующие данные в качестве первичного ключа. Когда вы вставляете, обновляете или удаляете данные в БД, вы обязаны учитывать ограничения БД.

ПРЕДСТАВЛЕНИЕ

На основе таблицы, которую вы создали с помощью оператора CREATE TABLE, вы также можете создать виртуальную (производную) таблицу, которая существует, только когда её видит пользователь. Это называется **представлением** (view). Таблица, откуда происходит представление, называется **базовой таблицей** (base table).



Создадим представление с помощью соответствующей команды SQL.

```
CREATE VIEW дорогостоящий_товар(код_товара,
        название_товара, цена)
AS SELECT *
FROM товары
WHERE цена>=200;
```

Эта команда создаёт представление

Таблица дорогостоящих товаров — это представление на основе таблицы товаров, которая является базовой таблицей. Она создалась в результате извлечения из таблицы товаров с ценой за 1 кг больше или равной 200 G.

■ ТАБЛИЦА «ДОРОГОСТОЯЩИЕ ТОВАРЫ»

Код товара	Название товара	Цена
101	Дыни	800 G
104	Лимоны	200 G
202	Хурма	200 G

Когда вы создали представление дорогостоящих товаров, вы можете искать данные точно так же, как вы бы стали это делать в базовой таблице.

```
SELECT *
FROM дорогостоящие_товары
WHERE цена>=500;
```

Позволяет использовать представление аналогично базовой таблице

Представление удобно использовать в случае, если вы хотите сделать общедоступной часть данных в базовой таблице. В SQL также есть операторы, позволяющие удалять базовую таблицу и представление. Ниже показан оператор, удаляющий представление или базовую таблицу.

```
DROP VIEW дорогостоящие_товары;
```

```
DROP TABLE товары;
```

ВОПРОСЫ И ЗАДАНИЯ

Создайте SQL-запросы для следующих вопросов (за единицу населения примем 10 000). Ответы даны на стр. 125-128.

B19

Таблица покупателей создана с помощью команды CREATE TABLE. Добавьте в неё приведённые ниже данные.

■ ТАБЛИЦА «ПОКУПАТЕЛИ»

Код покупателя	Покупатель	Население	Регион
12	Королевство Моро	100	Южный регион
15	Королевство Дуриан	200	Центр
22	Королевство Кумкват	160	Северный регион
23	Империя Фарайя	120	Северный регион

B20

Из таблицы покупателей (B19) создайте представление с названием «Северный регион», которое будет отображать страны, входящие в Северный регион.

■ ТАБЛИЦА «ПОКУПАТЕЛИ»

Код покупателя	Покупатель	Население
22	Королевство Кумкват	160
23	Империя Фарайя	120

B21

Измените население королевства Кумкват в таблице «Покупатели» на значение 1,5 млн.

B22

В таблице «Покупатели» удалите все данные о королевстве Дуриан.

ИСПОЛЬЗОВАНИЕ SQL ИЗ ПРИКЛАДНОГО ПО

Итак, мы познакомились с основными возможностями SQL.

Все операторы SQL, которые мы изучили к настоящему моменту, можно опробовать с помощью интерактивного клиента командной строки или формы веб-клиента базы данных. Результаты выводятся сразу после ввода каждого запроса SQL, как показано на следующем рисунке с изображением окна интерактивного клиента для базы данных MySQL. Аналогично можно опробовать запросы SQL с помощью формы веб-клиента.

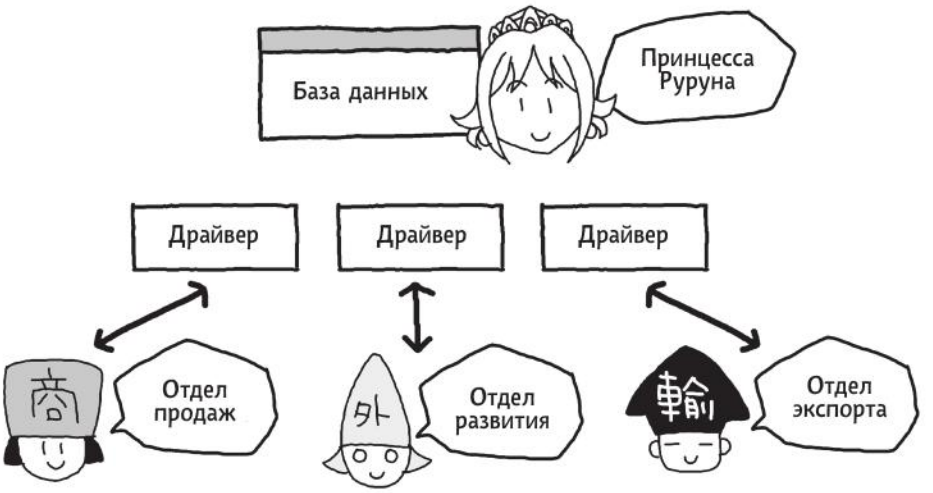
```
MySQL Command Line Client
Query OK, 1 row affected (0.03 sec)

+------+--------+-------+
| id   | name   | price |
+------+--------+-------+
|  101 | メロン | 80G   |
|  102 | いちご | 150G  |
|  103 | りんご | 120G  |
|  104 | レモン | 300G  |
+------+--------+-------+
4 rows in set (0.00 sec)

mysql> SELECT * FROM fruits
    -> ¥g
+------+--------+-------+
| id   | name   | price |
+------+--------+-------+
|  101 | メロン | 80G   |
|  102 | いちご | 150G  |
|  103 | りんご | 120G  |
|  104 | レモン | 300G  |
+------+--------+-------+
4 rows in set (0.00 sec)

mysql>
```

Примеры выполнения запросов в окне интерактивного клиента

Что касается практического применения баз данных, существует множество направлений выгодного использования SQL в программах, например, создание прикладного ПО управления продажами и товарами. [SG&A или SGA: Selling, General and Administrative Expenses].

Существует много приемов использования SQL в языках программирования. Это могут быть статические запросы SQL, реализованные в виде оператора SQL и компилируемые вместе с программой; динамические запросы SQL, генерируемые в процессе работы.

Например, доступ к базе данных из программ на языке Java, с использованием динамического SQL, можно осуществлять следующим образом:



■ ПРИМЕР JAVA-КОДА С ИСПОЛЬЗОВАНИЕМ SQL

```
import java.sql.*;

public class Fruits
{
  public static void main(String args[])
  {
    try{
      String drv = "com.mysql.jdbc.Driver";
      String url = "jdbc:mysql:/// fruitsdb";
      String usr = "";
      String pw = "";

      Class.forName(drv);
      Connection cn = DriverManager.getConnection(url, usr, pw);

      Statement st = cn.createStatement();
      String qry = "SELECT * FROM fruits";

      ResultSet rs = st.executeQuery(qry);

      ResultSetMetaData rm = rs.getMetaData();
      int cnum = rm.getColumnCount();

      while(rs.next()){
        for(int i=1; i<=cnum; i++){
          System.out.print(rm.getColumnName(i) +  "--"+ rs.getObject(i) + "  ");
        }
        System.out.println("");
      }
...
}
```

ПЕРЕМЕЩЕНИЕ ПО ЗАПИСЯМ С ИСПОЛЬЗОВАНИЕМ КУРСОРА

Как мы уже видели ранее, в запросе SQL можно свести результаты в одну таблицу и использовать их в таком виде. Однако в распространенных языках программирования нельзя выбрать и использовать сразу несколько строк в одной команде. Поэтому при использовании SQL в языках программирования необходим метод доступа к каждой из строк в таблице с результатами.

С этой целью используется механизм, который обычно называют курсором, обеспечивающий доступ к текущей строке. Перемещая курсор по строкам, мы можем обращаться к каждой из них. Соответственно, используя оператор цикла языка программирования, можно получить доступ к нескольким строкам. Извлечение данных из одной строки с использованием курсора называется выборкой (fetch). Именно так в программе на Java (предыдущая страница) мы получаем доступ к одной строке данных.

Код товара	Название товара	Цена	Территория
101	Дыни	800 G	Южные моря
102	Клубника	150 G	Центр
103	Яблоки	120 G	Северные воды
104	Лимоны	200 G	Северные воды

Благодаря подобным приемам в прикладном ПО, мы получаем возможность выполнять SQL-запросы к базам данных.

ИТОГИ

- С помощью команд SQL можно задавать данные и управлять ими.
- Для поиска данных используйте оператор SELECT.
- Для постановки условий используйте оператор WHERE.
- Для вставки, обновления и удаления данных, используйте операторы INSERT, UPDATE, DELETE.
- Для создания таблицы используйте оператор CREATE TABLE.

ОТВЕТЫ

01

```
SELECT *
FROM покупатели
WHERE население>=100;
```

02

```
SELECT *
FROM покупатели
WHERE население<100;
```

03

```
SELECT *
FROM покупатели
WHERE код_покупателя<20 AND население>=100;
```

Код покупателя	Покупатель	Население
12	Королевство Моро	100

04

```
SELECT *
FROM покупатели
WHERE код _покупателя>=30 AND население>100;
```

Ни одна страна не отвечает этим критериям, поэтому запрос вернёт пустой набор.

Код покупателя	Покупатель	Население
23	Империя Альфа	120
25	Королевство Ретол	150
32	Королевство Сазана	80

05

```
SELECT население
FROM покупатели
WHERE  покупатель='Королевство Рамбутан';
```

Население
150

06

```
SELECT *
FROM покупатели
WHERE покупатель LIKE '%н%';
```

Код покупателя	Покупатель	Население
25	Королевство Рамбутан	150
32	Королевство Кивано	80

07

```
SELECT MIN(население)
FROM покупатели;
```

MIN (население)
80

08

```
SELECT MAX(население)
FROM покупатели;
```

MAX (население)
300

09

```
SELECT SUM(население)
FROM покупатели;
```

SUM(население)
1 350

010

```
SELECT SUM(население)
FROM покупатели
WHERE код_покупателя>=20;
```

SUM(население)
1 050

011

```
SELECT COUNT(*)
FROM покупатели
WHERE население>=100;
```

COUNT(*)
7

012

```
SELECT COUNT(*)
FROM покупатели
WHERE регион='Северный регион';
```

COUNT(*)
3

013

```
SELECT MAX(население)
FROM покупатели
WHERE регион='Северный регион';
```

MAX (население)
240

014

```
SELECT SUM(население)
FROM покупатели
WHERE NOT(покупатель='Королевство Рамбутан');
```

SUM(население)
1 200

015

```
SELECT регион, AVG(население)
FROM покупатели
GROUP BY регион
HAVING AVG(население)>=200;
```

Регион	AVG(население)
Центр	250

016

```
SELECT регион, COUNT(*)
FROM покупатели
GROUP BY регион
HAVING COUNT(*)>=3;
```

Регион	COUNT(*)
Северный регион	3
Южный регион	3

017

```
SELECT * FROM реализованные_товары WHERE код_товара IN
(SELECT код_товара FROM товары WHERE цена>=300);
```

Код отчёта	Код товара	Количество
1101	101	1 100
1104	101	2 500

018

```
SELECT *
FROM реализованные_товары U
WHERE количество<
(SELECT AVG(количество)
FROM реализованные_товары
WHERE код_товара=U.код_товара);
```

Код отчёта	Код товара	Количество
1101	101	1 100
1102	103	1 700
1103	104	500

019

```
INSERT INTO покупатели(код_покупателя,покупатель,население,регион)
VALUES(12,'Королевство Моро',100,'Южный регион');

INSERT INTO покупатели(код_покупателя,покупатель,население,регион)
VALUES(15,'Королевство Дуриан',200,'Центр');

INSERT INTO покупатели(код_покупателя,покупатель,население,регион)
VALUES(22,'Королевство Кумкват',160,'Северный регион');

INSERT INTO покупатели(код_покупателя,покупатель,население,регион)
VALUES(23,'Империя Фарайя',120,'Северный регион');
```

020

```
CREATE VIEW Северный_регион(код_покупателя,покупатель,население)
AS SELECT код_покупателя,покупатель,население
FROM покупатели
WHERE регион='Северный регион';
```

021

```
UPDATE покупатели
SET население=150
WHERE покупатель='Королевство Кумкват';
```

022

```
DELETE FROM покупатели
WHERE покупатель='Королевство Дуриан';
```

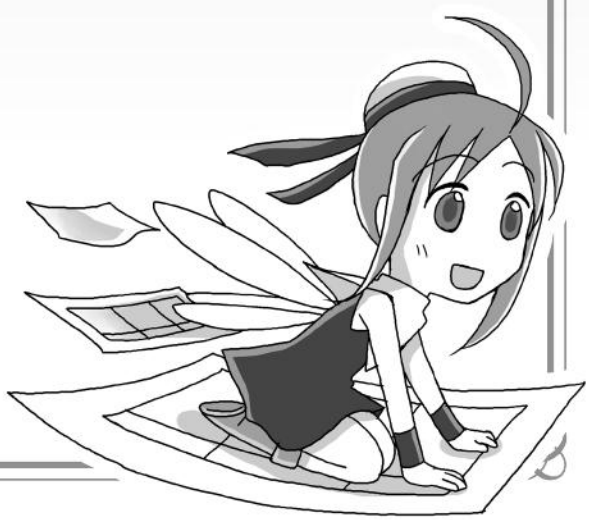
СТАНДАРТИЗАЦИЯ SQL

Язык SQL стандатизизирован Международной организацией по стандартизации (ISO). В Японии он стандартизован комитетом по промышленным стандартам Японии(Japanese Industrial Standards, JIS).

В числе других стандартов SQL можно назвать SQL92, вышедший в 1992 году, и SQL99, вышедший в 1999 году. Системы управления реляционными базами данных проектируются в соответствии с этими стандартами. Некоторые реляционные базы данных имеют свои собственные спецификации. За дополнительной информацией обращайтесь к руководству, составленному для вашей базы данных.

5.

ДАВАЙТЕ УПРАВЛЯТЬ БАЗОЙ ДАННЫХ!



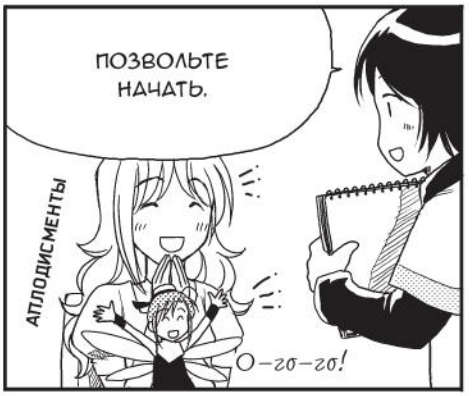
ЧТО ТАКОЕ ТРАНЗАКЦИЯ?
КЛАЦ, КЛАЦ, КЛАЦ...
БИ-ИП
БИ-ИП
КЛАЦ, КЛАЦ, КЛАЦ, КЛАЦ...
КЛАЦ, КЛАЦ, КЛАЦ...
ВИДИШЬ, ВОТ НОВЫЙ ЗАКАЗ, И ВОТ ЕЩЁ.

ДА, КОНЕЧНО, Я ДОБАВЛЮ ИХ В БАЗУ ПРЯМО СЕЙЧАС.

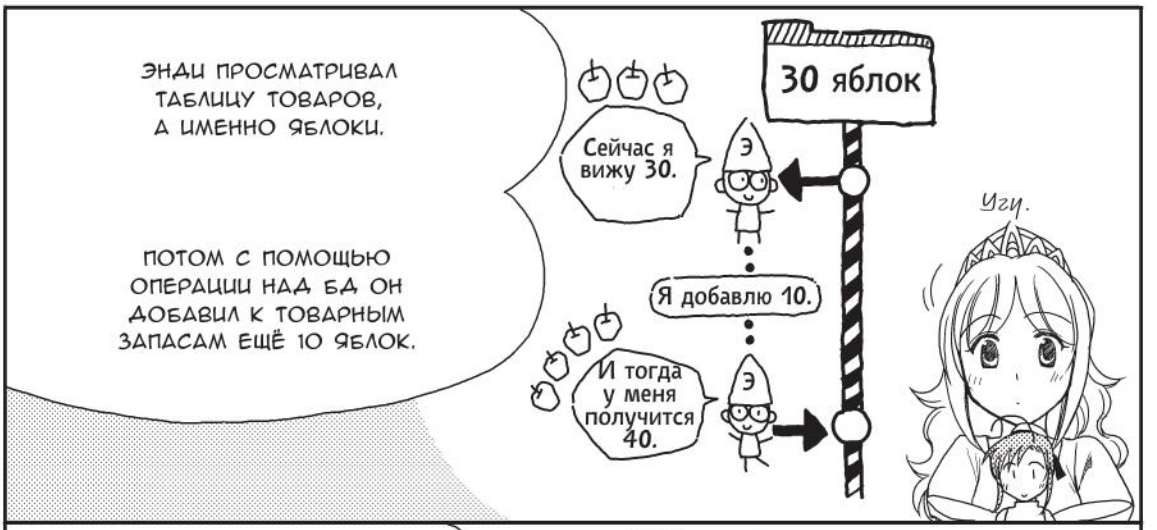
О, МЫ НАКОНЕЦ-ТО УЧИМСЯ ПОЛЬЗОВАТЬСЯ БАЗОЙ ДАННЫХ, ДА?
ФР-Р-Р!
ФУ-ТЫ!

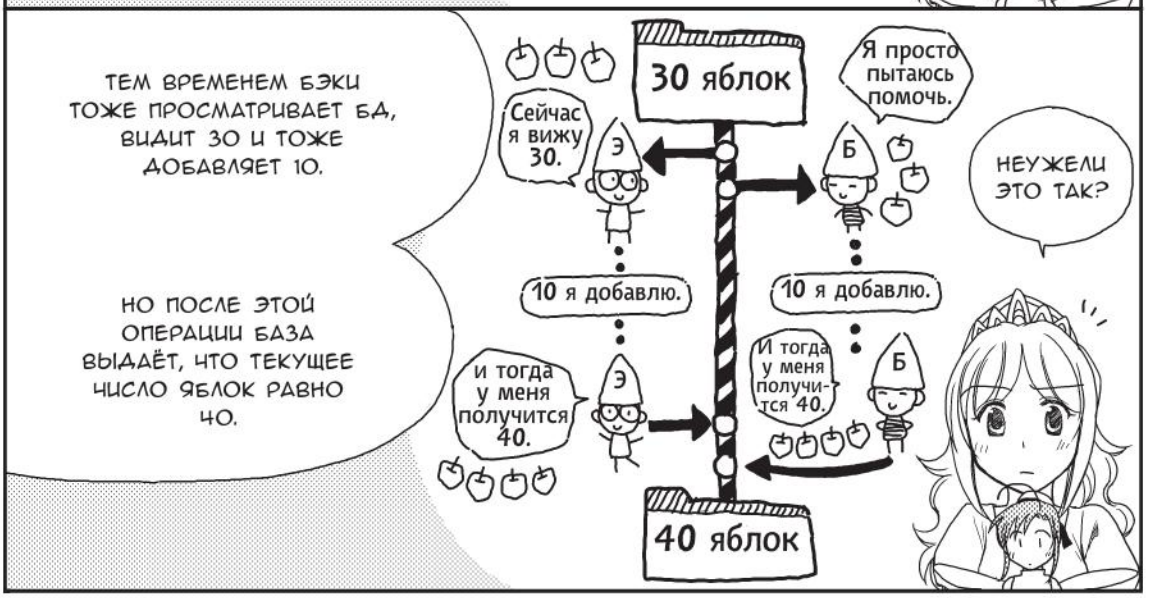
ТЫ ЧТО, НЕ МОЖЕШЬ ПОЯВЛЯТЬСЯ КАК НОРМАЛЬНЫЙ ЧЕЛОВЕК?
Прошу прощения
Тико!
Чёрт!

ВООБЩЕ-ТО, ДОЛЖНА ТЕБЯ ПОБЛАГОДАРИТЬ...
НО НАМ ЕЩЁ ТАК МНОГОМУ НАДО НАУЧИТЬСЯ.
К ПРИМЕРУ, ХОТЕЛОСЬ БЫ ЗНАТЬ, КАК ЕЩЁ МОЖЕТ РАБОТАТЬ БАЗА ДАННЫХ, КОГДА СТОЛЬКО ПОЛЬЗОВАТЕЛЕЙ ОДНОВРЕМЕННО ИМЕЮТ К НЕЙ ДОСТУП.
В ЭТОМ ОТНОШЕНИИ ВОПРОС БЕЗОПАСНОСТИ МЕНЯ ТОЖЕ СЛЕГКА ВОЛНУЕТ.
ВИДИМО, У ТЕБЯ ВОЗНИКАЕТ БЕСПОКОЙСТВО ПО ПОВОДУ СОХРАННОСТИ ДАННЫХ В БД.
ВРОДЕ ТОГО.
ЧТОБЫ ЛУЧШЕ ПОНЯТЬ ЭТИ ВОПРОСЫ,
Кхе!
Я ПРОВЁЛ НЕБОЛЬШОЕ ИССЛЕДОВАНИЕ.
ДА НУ?
ВСПЫШКА!
МОЯ ПРЕЗЕНТАЦИЯ НАЗЫВАЕТСЯ:
КАК БАЗА ДАННЫХ МОЖЕТ ОДНОВРЕМЕННО ДАВАТЬ ДОСТУП НЕСКОЛЬКИМ ПОЛЬЗОВАТЕЛЯМ.
О!
Я ДАЖЕ ПОДГОТОВИЛ ИЛЛЮСТРАЦИИ, ЧТОБЫ ВЫ ЛУЧШЕ ПОНЯЛИ!
Занавес открывается!
Ого, здорово!
Театр БД
Люблю зрелищные шоу!


ПОЗВОЛЬТЕ НАЧАТЬ.
АПЛОДИСМЕНТЫ
О-го-го!

ОДНАЖДЫ ЭНДИ И БЭКИ ОДНОВРЕМЕННО ВОШЛИ В БАЗУ ДАННЫХ.
База данных
Доступ
Э
Б


ЭНДИ ПРОСМАТРИВАЛ ТАБЛИЦУ ТОВАРОВ, А ИМЕННО ЯБЛОКИ.
ПОТОМ С ПОМОЩЬЮ ОПЕРАЦИИ НАД БД ОН ДОБАВИЛ К ТОВАРНЫМ ЗАПАСАМ ЕЩЁ 10 ЯБЛОК.
30 яблок
Сейчас я вижу 30.
Э
Я добавлю 10.
И тогда у меня получится 40.
Э
Угу.


ТЕМ ВРЕМЕНЕМ БЭКИ ТОЖЕ ПРОСМАТРИВАЕТ БД, ВИДИТ 30 И ТОЖЕ ДОБАВЛЯЕТ 10.
НО ПОСЛЕ ЭТОЙ ОПЕРАЦИИ БАЗА ВЫДАЁТ, ЧТО ТЕКУЩЕЕ ЧИСЛО ЯБЛОК РАВНО 40.
30 яблок
Сейчас я вижу 30.
Э
Я просто пытаюсь помочь.
Б
НЕУЖЕЛИ ЭТО ТАК?
10 я добавлю.
10 я добавлю.
И тогда у меня получится 40.
Э
И тогда у меня получи-тся 40.
Б
40 яблок

ТАК И ДОЛЖНО БЫТЬ?
ВЕРНО.
Энди добавил 10. Бэки добавила 10.

И КУДА ДЕЛИСЬ ЕЩЁ 10 ЯБЛОК?
ЗАДУМАЛАСЬ
ДАЙ-КА ПОДУМАТЬ...

ИХ СЪЕЛ КЕЙН!
Я не брал!
Виновен!
НАЧНЁМ С ТОГО, ЧТО ИХ ТАМ И НЕ БЫЛО!
Вот так!

НА САМОМ ДЕЛЕ ПРИ ТАКОЙ ПОСЛЕДОВАТЕЛЬНОСТИ БЭКИ НЕ МОЖЕТ ВЫПОЛНЯТЬ ОПЕРАЦИИ НАД БД, ПОКА ЭНДИ С НЕЙ РАБОТАЕТ.
База данных
Нет
Б
Добавить 10
ПОЭТОМУ 10 ЯБЛОК НИКУДА БЫ НЕ ПРОПАЛИ.

ЧТОБЫ ЭНДИ И БЭКИ МОГЛИ РАБОТАТЬ С БД ОДНОВРЕМЕННО,
ДОЛЖЕН СУЩЕСТВОВАТЬ МЕХАНИЗМ ПРЕДОТВРАЩЕНИЯ ПОДОБНЫХ ПРОТИВОРЕЧИЙ И ДУБЛИРОВАНИЯ.
Э
?
Б
В САМОМ ДЕЛЕ.

ТО ЕСТЬ ВОПРОС СОСТОИТ В ТОМ, КАК БД УПРАВЛЯЕТ ДЕЙСТВИЯМИ ПОЛЬЗОВАТЕЛЕЙ?
И СЕЙЧАС Я ОБ ЭТОМ РАССКАЖУ!
Во дает!
Кейн сегодня бесполезен.

НАПРИМЕР, ОПЕРАЦИЯ ЧТЕНИЯ, ОПЕРАЦИЯ ДОБАВЛЕНИЯ И ОПЕРАЦИЯ ЗАПИСИ ОБРАБАТЫВАЮТСЯ КАК ЕДИНАЯ ОПЕРАЦИЯ.

Транзакция
Э
Чтение данных
Добавить 10
Записать результат
Э
Б
Чтение данных
Добавить 10
Записать результат
Б
Транзакция

ТО ЕСТЬ ОПЕРАЦИИ, КОТОРЫЕ ВЫПОЛНЯЛ ЭНДИ — ЭТО ОДНА ТРАНЗАКЦИЯ,

А ОПЕРАЦИИ БЭКИ — ЭТО УЖЕ ДРУГАЯ ТРАНЗАКЦИЯ.

ЧТО ТАКОЕ БЛОКИРОВКА (LOCK)?

ЭНДИ ЗАБЛОКИРОВАЛ ДАННЫЕ ПЕРЕД ВЫПОЛНЕНИЕМ ЧЕРЕДЫ ОПЕРАЦИЙ.

КОГДА БЭКИ ПЫТАЕТСЯ ВЫПОЛНИТЬ СВОИ ОПЕРАЦИИ, ЕЙ ПРИДЁТСЯ ПОДОЖДАТЬ, КОГДА РАБОТУ ЗАКОНЧИТ ЭНДИ.

30 яблок

lock

30 яблок

Э

10 добавил

40 яблок

Ждёт, когда Энди закончит работу.

Б

Я жду

40 яблок

lock

Б

10 добавила

50 яблок

OK!

50 яблок

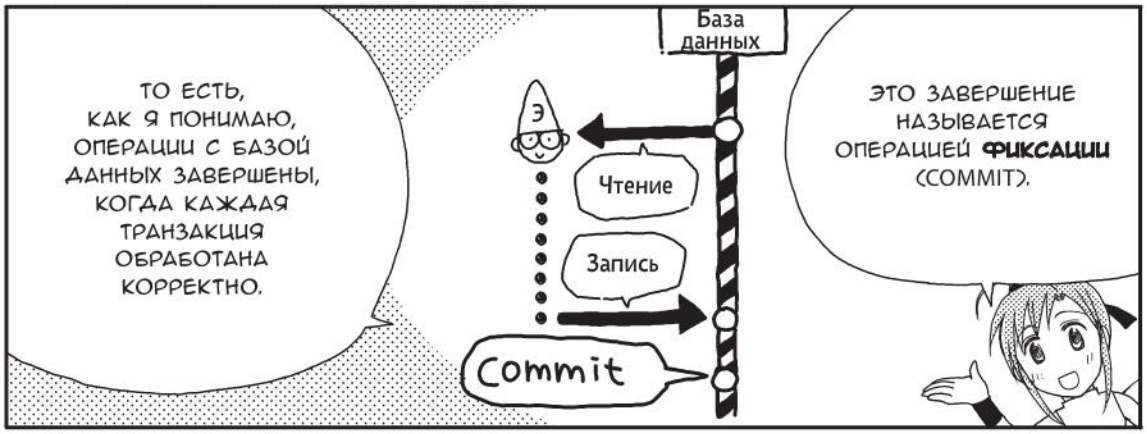
БЭКИ СМОЖЕТ ВЫПОЛНИТЬ СВОИ ОПЕРАЦИИ, ТОЛЬКО КОГДА ЭНДИ ЗАКОНЧИТ СВОИ.

В РЕЗУЛЬТАТЕ БД ВЫДАСТ ЗНАЧЕНИЕ 50, КАК И ДОЛЖНО ПОЛУЧИТЬСЯ.

ОГО, КЕЙН, Я ПОРАЖЕНА.

Да, да

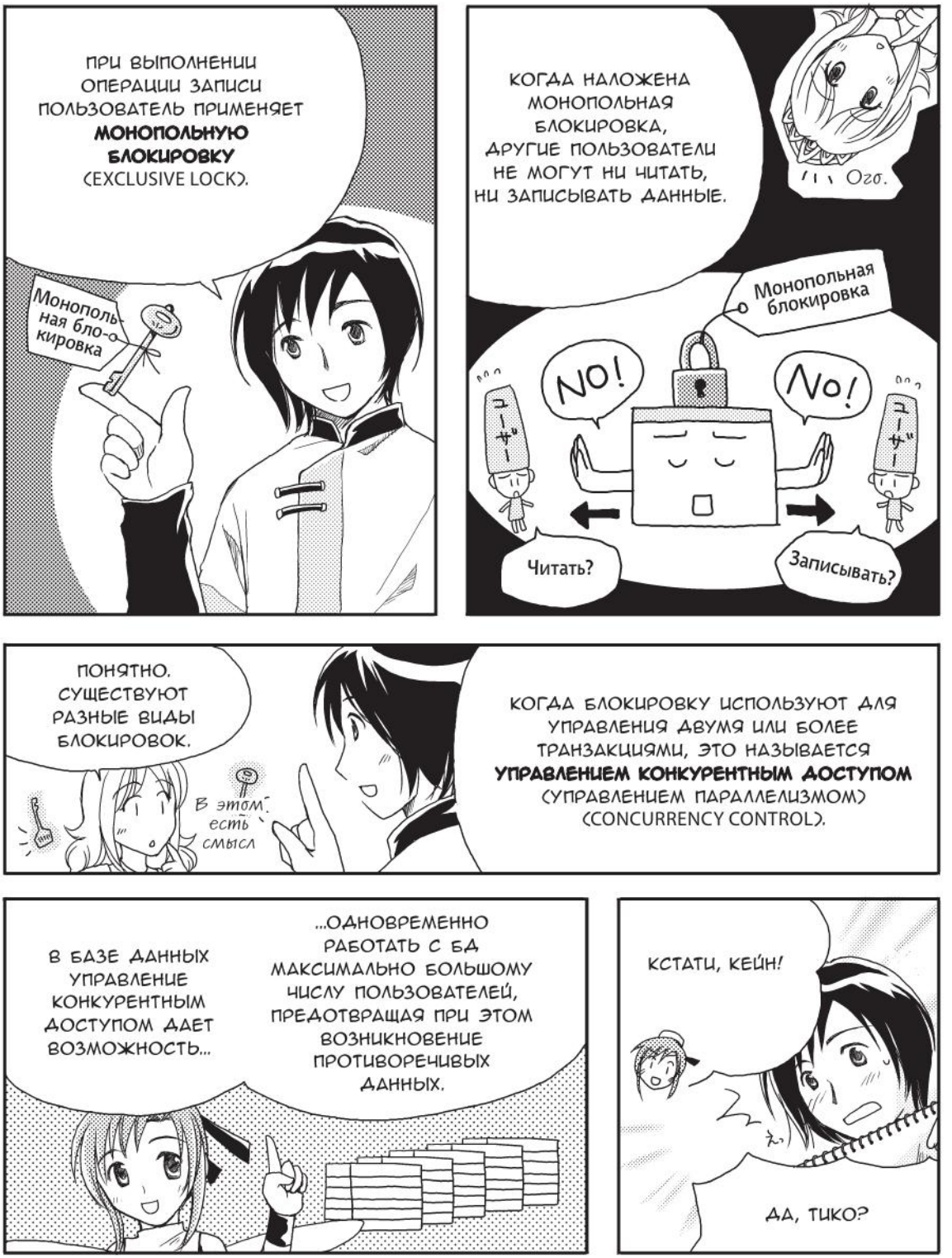
Он нравится мне всё больше.


То есть, как я понимаю, операции с базой данных завершены, когда каждая транзакция обработана корректно.
База данных
Э
Чтение
Запись
Commit
Это завершение называется операцией **фиксации** (commit).

Хотя блокировка играет важную роль в БД, злоупотреблять ею нельзя. Она может воспрепятствовать цели БД: обмену данными между большим количеством людей.
Итак, в зависимости от ситуации применяются разные виды блокировки.
И как это делается?

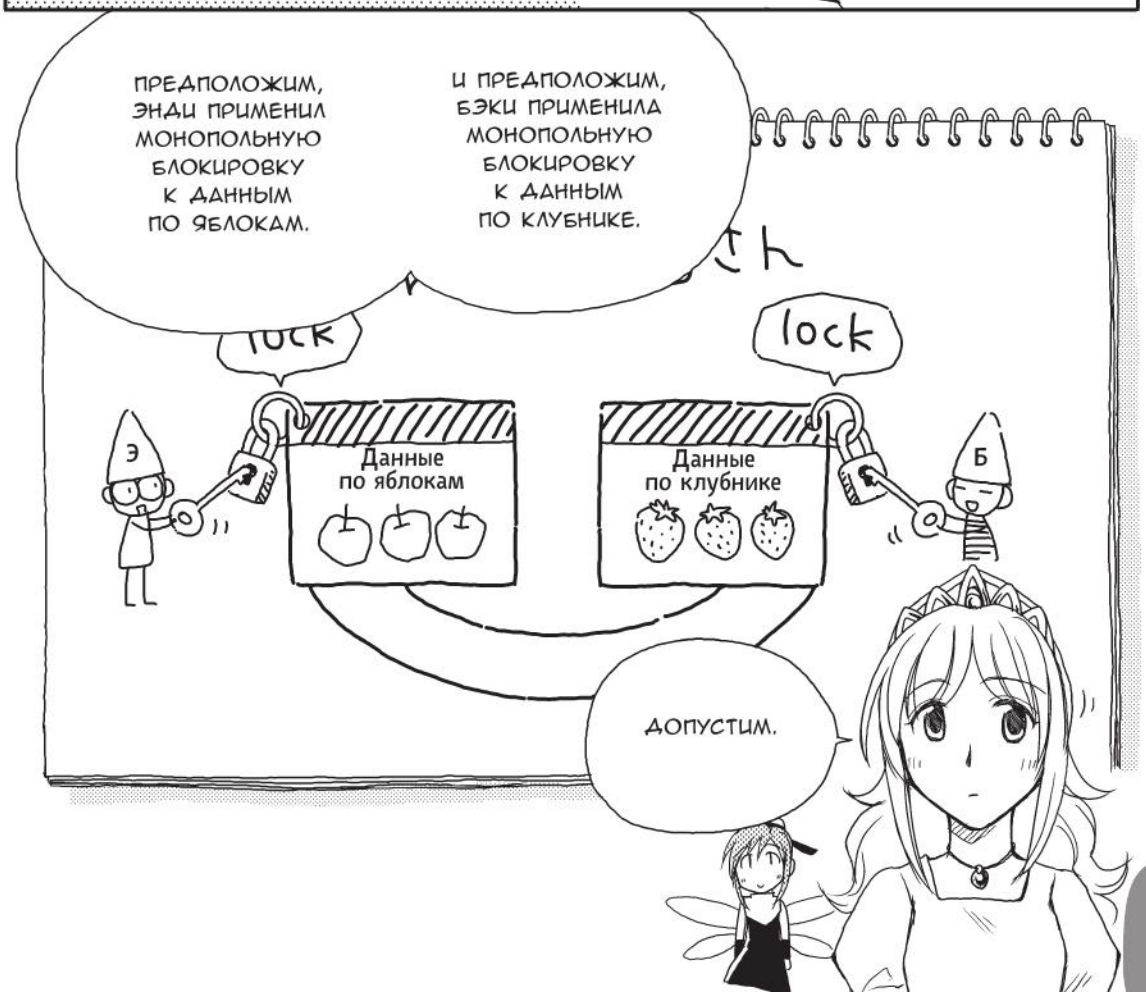
Например, ты можешь использовать **разделяемую** (коллективную) **блокировку** (shared lock) для операции чтения, когда нужна только эта операция.
Разделяемая блокировка

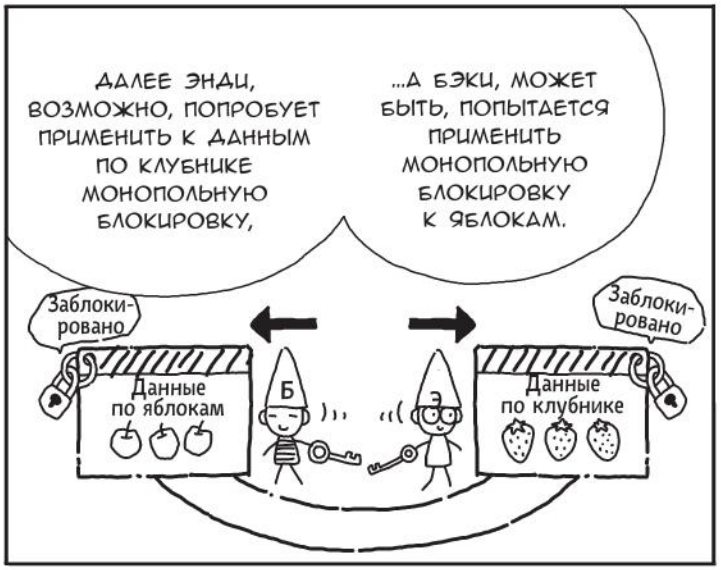
В то время как установлена разделяемая блокировка, другие пользователи могут считывать данные,
но не могут выполнять операции записи.
Разделяемая блокировка
OK
NO
Читать?
Записывать?


При выполнении операции записи пользователь применяет **монопольную блокировку** (EXCLUSIVE LOCK).
Монопольная блокировка
Когда наложена монопольная блокировка, другие пользователи не могут ни читать, ни записывать данные.
Ого.
Монопольная блокировка
NO!
No!
ユーザー
ユーザー
Читать?
Записывать?
Понятно. Существуют разные виды блокировок.
В этом есть смысл
Когда блокировку используют для управления двумя или более транзакциями, это называется **управлением конкурентным доступом** (управлением параллелизмом) (CONCURRENCY CONTROL).
В базе данных управление конкурентным доступом дает возможность...
...одновременно работать с БД максимально большому числу пользователей, предотвращая при этом возникновение противоречивых данных.
Кстати, Кейн!
Да, Тико?

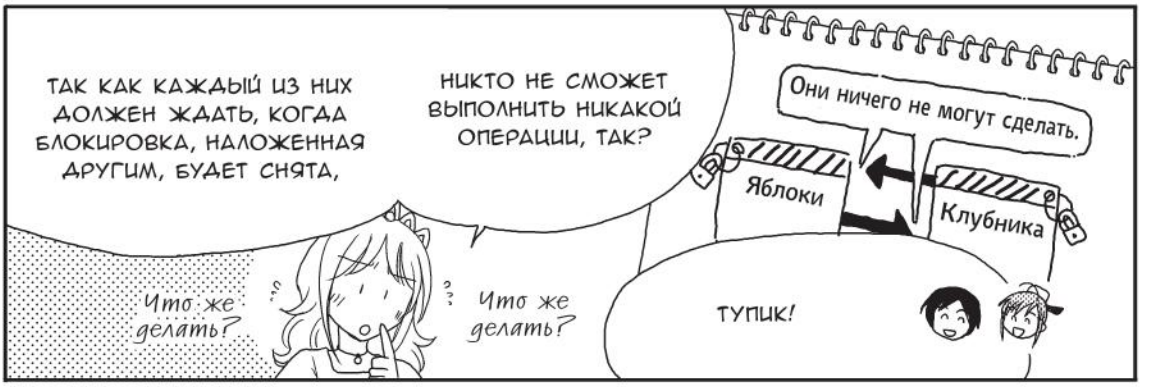
ПОХОЖЕ, ТЫ МНОГО ВСЕГО ИЗУЧИЛ ЗА ПОСЛЕДНЕЕ ВРЕМЯ?
У тебя такое впечатление?
ВООБЩЕ-ТО Я ТАК НЕ ДУМАЮ.
РАЗВЕ ОН НЕ ЗАЛУЖИВАЕТ ПОХВАЛЫ, РУРУНД?
ПОЖАЛУЙ, ДА...
НУ, ТОГДА ВПЕРЕД!
ХОРОШО.

ДАВАЙТЕ ВЕРНЕМСЯ К ПРЕДЫДУЩЕЙ ТЕМЕ.
В НЕКОТОРЫХ СЛУЧАЯХ УПРАВЛЕНИЕ КОНКУРЕНТНЫМ ДОСТУПОМ С БЛОКИРОВКОЙ МОЖЕТ ВЫЗЫВАТЬ ПОБЛЕМЫ.
Энди
Бэки
НАПРИМЕР...


ПРЕДПОЛОЖИМ, ЭНДИ ПРИМЕНИЛ МОНОПОЛЬНУЮ БЛОКИРОВКУ К ДАННЫМ ПО ЯБЛОКАМ.
И ПРЕДПОЛОЖИМ, БЭКИ ПРИМЕНИЛА МОНОПОЛЬНУЮ БЛОКИРОВКУ К ДАННЫМ ПО КЛУБНИКЕ.
lock
lock
Э
Б
Данные по яблокам
Данные по клубнике
ДОПУСТИМ.


Далее Энди, возможно, попробует применить к данным по клубнике монопольную блокировку,
...а Бэки, может быть, попытается применить монопольную блокировку к яблокам.
Заблокировано
Данные по яблокам
Б
Э
Данные по клубнике
Заблокировано

И что тогда произойдёт?
Ну...


Так как каждый из них должен ждать, когда блокировка, наложенная другим, будет снята,
Никто не сможет выполнить никакой операции, так?
Что же делать?
Что же делать?
Они ничего не могут сделать.
Б
Яблоки
Клубника
Э
Тупик!

Эта ситуация, которая называется **взаимная блокировка** (deadlock), не может быть разрешена, пока одна из блокировок не будет снята.
Например, вы можете поискать транзакции, которые встали в очередь за какой-то определённый период времени...
Отмена
И отменить их!

Когда вы отменяете транзакцию, это называется **откат** (rollback).
Откат?

ТЫ ХОЧЕШЬ СКАЗАТЬ, ЧТО МОЖНО ЗА РАЗ ОТМЕНИТЬ ВСЕ ОПЕРАЦИИ В ОДНОЙ ТРАНЗАКЦИИ?
ДА! НАПРИМЕР, ЕСЛИ ТРАНЗАКЦИЯ ПО "СНИЖЕНИЮ ЦЕНЫ НА ФРУКТЫ, ЦЕННИК У КОТОРЫХ НИЖЕ ИЛИ РАВЕН 150 Є" НЕ ПРОШЛА,
ТО ОПЕРАЦИИ ПО ЯБЛОКАМ И КЛУБНИКЕ ДОЛЖНЫ БЫТЬ ОТМЕНЕНЫ ПОЛНОСТЬЮ, ЯСНО?
ПОНЯТНО.

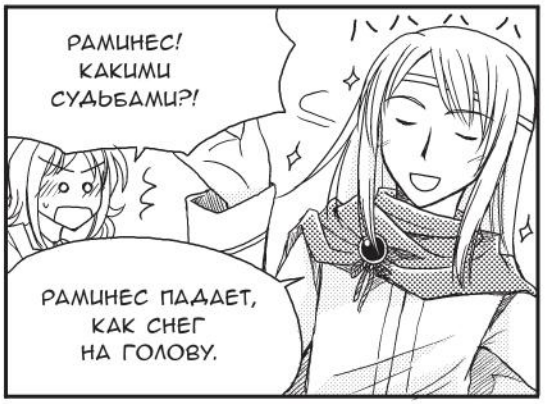
ТО ЕСТЬ БД ВЕДЁТ СЕБЯ ТАК, КАК-БУДТО НИКАКИХ ОПЕРАЦИЙ ВООБЩЕ НЕ ВЫПОЛНЯЛОСЬ?
Б
НУ, ВРОДЕ ТОГО.

ЕСЛИ ВО ВРЕМЯ ТРАНЗАКЦИИ ПРОИСХОДИТ ЧТО-ТО, ЧТО НЕ ДАЁТ ДОВЕСТИ ЕЁ ДО ЗАВЕРШЕНИЯ,
ВМЕСТО ВЫПОЛНЕНИЯ ОПЕРАЦИИ ФИКСАЦИИ (COMMIT) ПРОИЗВОДИТСЯ ОТКАТ (ROLLBACK).
опе-ра-ция
OK
commit
опе-ра-ция
NG
rollback

ВЕРНО. ТРАНЗАКЦИЯ ВСЕГДА ЗАКАНЧИВАЕТСЯ ОПЕРАЦИЕЙ ФИКСАЦИИ ИЛИ ОТКАТА.
ДРУГИМИ СЛОВАМИ, "ТРЕТЬЕГО НЕ ДАНО".

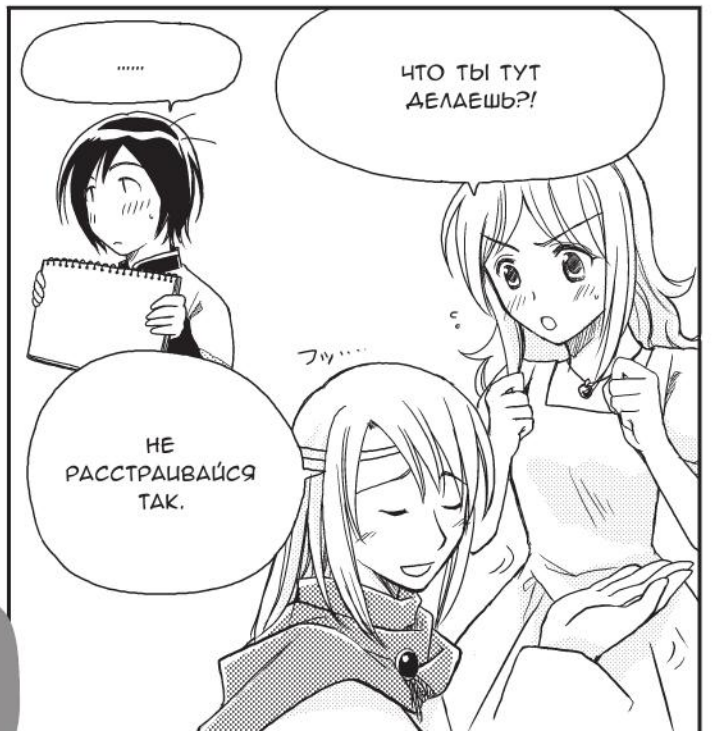
ОТЛИЧНО! ТЕПЕРЬ, МИЛАЯ ТИКО, СЛЕДУЮЩАЯ ТЕМА...

ЯСНО.
ДАЖЕ ЕСЛИ ОДНОВРЕМЕННО СРАЗУ МНОГО НАРОДА ПОЛЬЗУЕТСЯ БД, С НЕЙ НИЧЕГО ПЛОХОГО НЕ СЛУЧИТСЯ, ЕСЛИ, КОНЕЧНО, ОНА ПРАВИЛЬНО РАЗРАБОТАНА.
ОГО! ЭТО ЕЩЁ ЧТО ЗА БЭТМЕН?
ВЖ-Ж-ЖИК


РАМИНЕС! КАКИМИ СУДЬБАМИ?!
РАМИНЕС ПАДАЕТ, КАК СНЕГ НА ГОЛОВУ.

ЖАЛЬ, НО ПРИХОДИТСЯ СКАЗАТЬ...
НА НА НА
.....

Я ВАМ НЕ "МИЛАЯ ТИКО".
Хм

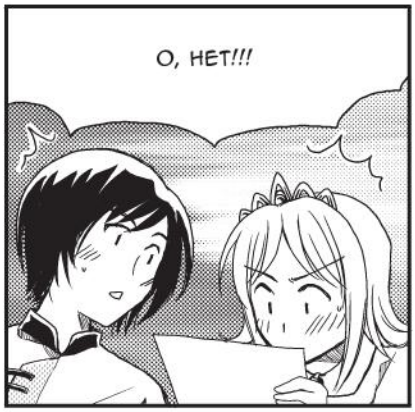

.....
ЧТО ТЫ ТУТ ДЕЛАЕШЬ?!
НЕ РАССТРАИВАЙСЯ ТАК.

ЭТО Я ДОЛЖЕН СЕРДИТЬСЯ.
Тоже мне умники
ВОТ, ПОСМОТРИ НА ЭТО.

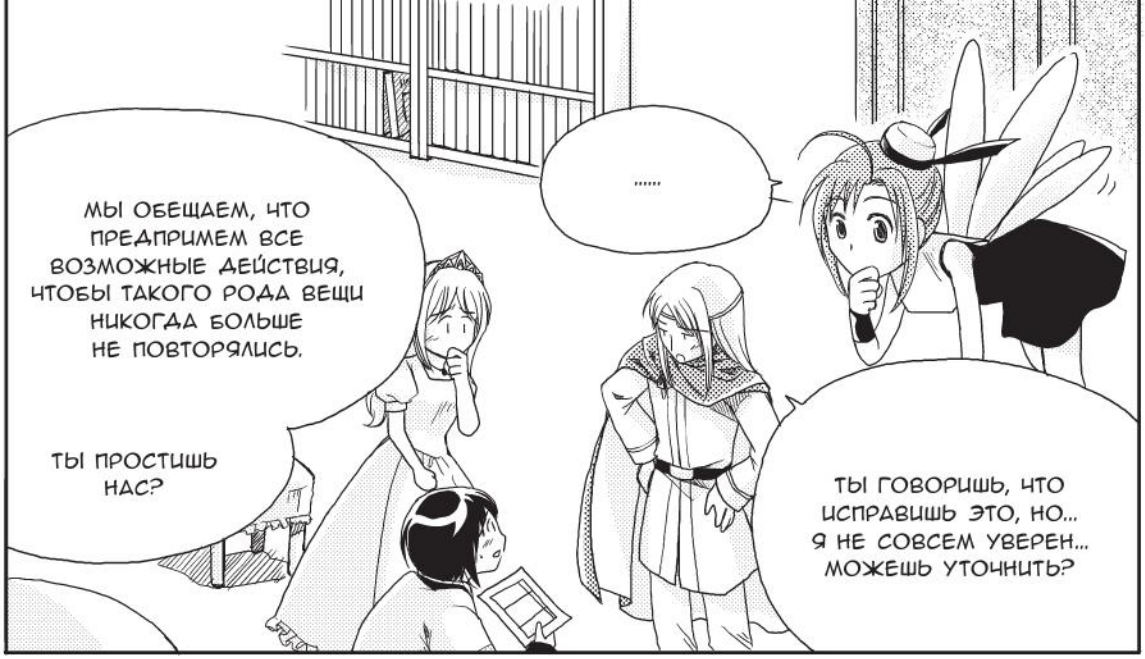
Код товара	Название товара	Цена
1 0 1	Дыни	10 000 G
1 0 2	Клубника	12 500 G
1 0 3	Яблоки	8 000 G
1 0 4	Лимоны	6 000 G
2 0 1	Каштаны	9 000 G
2 0 2	Хурма	12 400 G
3 0 1	Персики	5 000 G
3 0 2	Киви	6 000 G

ЧТО ЖЕ С НЕЙ НЕ ТАК?

ЦЕНЫ. ЦЕНЫ!



Так как ваши счета — это полная неразбериха, в моей стране, то есть у вашего покупателя, царит хаос.
Мерзкая штука, эта ваша база данных.
Вот так!
В качестве компенсации за доставленные проблемы...
Какой аромат!
...Почему бы тебе не принять моё предложение, Руруна? Будь моей невестой и поедем в моё королевство.
ИГНОРИРУЕТ
.....
Возможно, кто-то злонамеренно произвёл несанкционированное наложение данных.
Какой кошмар!
Вы ведёте себя, как будто меня тут нет.
Принц Раминес,
Как же так?
Мы очень сожалеем.

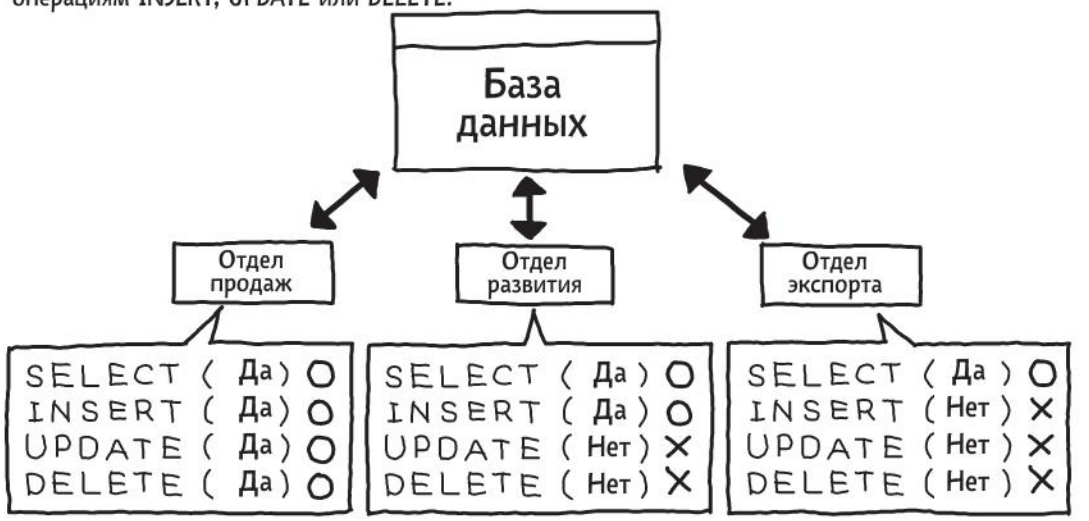

МЫ ОБЕЩАЕМ, ЧТО ПРЕДПРИМЕМ ВСЕ ВОЗМОЖНЫЕ ДЕЙСТВИЯ, ЧТОБЫ ТАКОГО РОДА ВЕЩИ НИКОГДА БОЛЬШЕ НЕ ПОВТОРЯЛИСЬ.
.....
ТЫ ПРОСТИШЬ НАС?
ТЫ ГОВОРИШЬ, ЧТО ИСПРАВИШЬ ЭТО, НО... Я НЕ СОВСЕМ УВЕРЕН... МОЖЕШЬ УТОЧНИТЬ?

ПРИЧИНА ЭТОЙ ПРОБЛЕМЫ В ТОМ, ЧТО КАЖДЫЙ В НАШЕМ КОРОЛЕВСТВЕ ИМЕЕТ ДОСТУП К БАЗЕ ДАННЫХ.
ПРЕЖДЕ ВСЕГО МЫ ОРГАНИЗУЕМ УПРАВЛЕНИЕ ДОСТУПОМ, ЧТОБЫ СОКРАТИТЬ КОЛИЧЕСТВО ПОЛЬЗОВАТЕЛЕЙ.
ТО ЕСТЬ?..

ХОРОШИМ РЕШЕНИЕМ МОЖЕТ СТАТЬ ТРЕБОВАНИЕ ВВЕСТИ ИМЯ ПОЛЬЗОВАТЕЛЯ И ПАРОЛЬ ДЛЯ ВХОДА В БД, ЧТОБЫ УБЕДИТЬСЯ, ЧТО ПОЛЬЗОВАТЕЛЬ ЗАСЛУЖИВАЕТ ДОВЕРИЯ И МОЖЕТ ПОЛУЧИТЬ ДОСТУП К ДАННЫМ.
ХОРОШО ПРИДУМАНО!

ВО-ВТОРЫХ, МЫ СДЕЛАЕМ ТАКИЕ НАСТРОЙКИ, ЧТОБЫ ОПРЕДЕЛЁННЫЕ ОПЕРАЦИИ МОГЛИ СОВЕРШАТЬ ТОЛЬКО ПОЛЬЗОВАТЕЛИ С ОПРЕДЕЛЁННЫМИ ПРАВАМИ.
Вот так!
Разрешение на операции SELECT, INSERT, UPDATE и DELETE над данными о товарах.
разрешение на опереации SELECT и INSERT над данными о товарах, без разрешения UPDATE/DELETE.
разрешение операции SELECT...

- Сотрудники отдела продаж могут производить операции **SELECT, INSERT, UPDATE** и **DELETE** над данными о товарах.
- Сотрудники отдела развития могут производить операции над данными **SELECT** и **INSERT**, но не могут использовать **UPDATE** и **DELETE**
- Сотрудники отдела экспорта могут производить операцию **SELECT**, но не допускаются к операциям **INSERT, UPDATE** или **DELETE**.

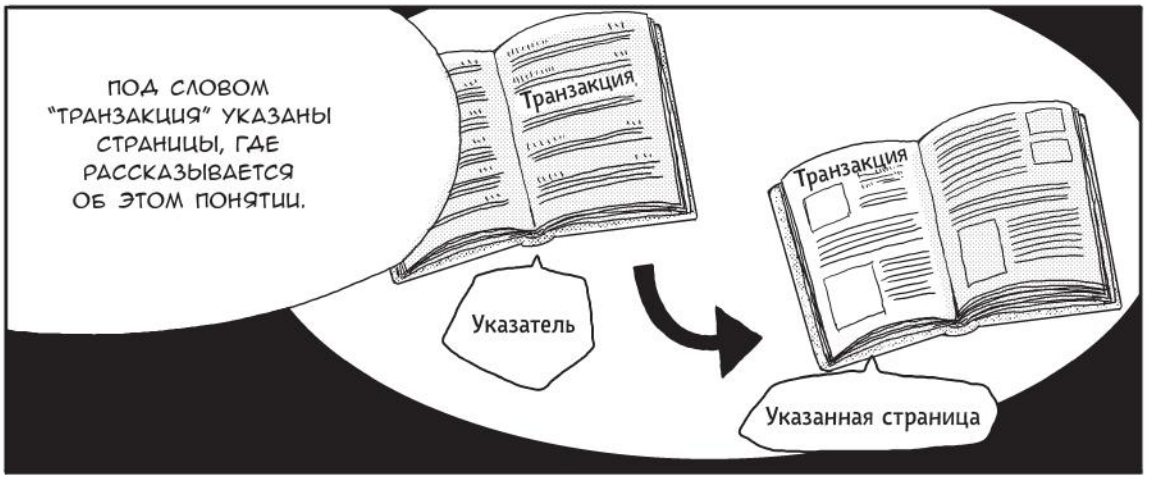


ОДНОЗНАЧНО!

КАК ВСЁ УСКОРИТЬ С ПОМОЩЬЮ ИНДЕКСИРОВАНИЯ

ПРЕДПОЛОЖИМ, НАМ НАДО НАЙТИ ЗНАЧЕНИЕ СЛОВА "ТРАНЗАКЦИЯ" В ЭТОЙ КНИГЕ О БАЗАХ ДАННЫХ.
Предположим

ПОИСК ВСЛЕПУЮ БЫЛ БЫ ОЧЕНЬ УТОМИТЕЛЬНЫМ ЗАНЯТИЕМ. ВМЕСТО ЭТОГО НАЙДЁМ ЭТО СЛОВО В ПРЕДМЕТНОМ УКАЗАТЕЛЕ.
Найдём


ПОД СЛОВОМ "ТРАНЗАКЦИЯ" УКАЗАНЫ СТРАНИЦЫ, ГДЕ РАССКАЗЫВАЕТСЯ ОБ ЭТОМ ПОНЯТИИ.
Транзакция
Указатель
Транзакция
Указанная страница

ИСПОЛЬЗУЯ УКАЗАТЕЛЬ (ИНДЕКС), МЫ МОЖЕМ БЫСТРО НАЙТИ НУЖНУЮ НАМ СТРАНИЦУ!
ТЫ ПРАВ.

ТО ЖЕ САМОЕ С ИНДЕКСОМ В БД, НАПРИМЕР...
?

ЕСЛИ ВЫ СОЗДАЕТЕ ИНДЕКСЫ ДЛЯ КОДОВ ТОВАРА,
ВЫ МОЖЕТЕ СРАЗУ ЖЕ УЗНАТЬ, ГДЕ ХРАНЯТСЯ ДАННЫЕ О ТОВАРЕ, КОТОРОМУ ПРИСВОЕН КОД 101.
101 дыни
Данные
ЭТО ГОВОРИТ НАМ, ГДЕ НА ДИСКЕ НАХОДЯТСЯ ДАННЫЕ ОБ ЭТОМ ТОВАРЕ.

ИНДЕКСИРОВАНИЕ УСКОРЯЕТ ПРОЦЕСС ПОИСКА.
Ну, я так быстро не могу соображать... в любом случае...
УГУ.

ПРОВЕРКА ВСЕХ СТРОК В ПОИСКАХ КОНКРЕТНЫХ ДАННЫХ ЗАНИМАЕТ ОЧЕНЬ МНОГО ВРЕМЕНИ.
Объяснить всё с самого начала?
Нет, нет.
ФР-Р-Р!

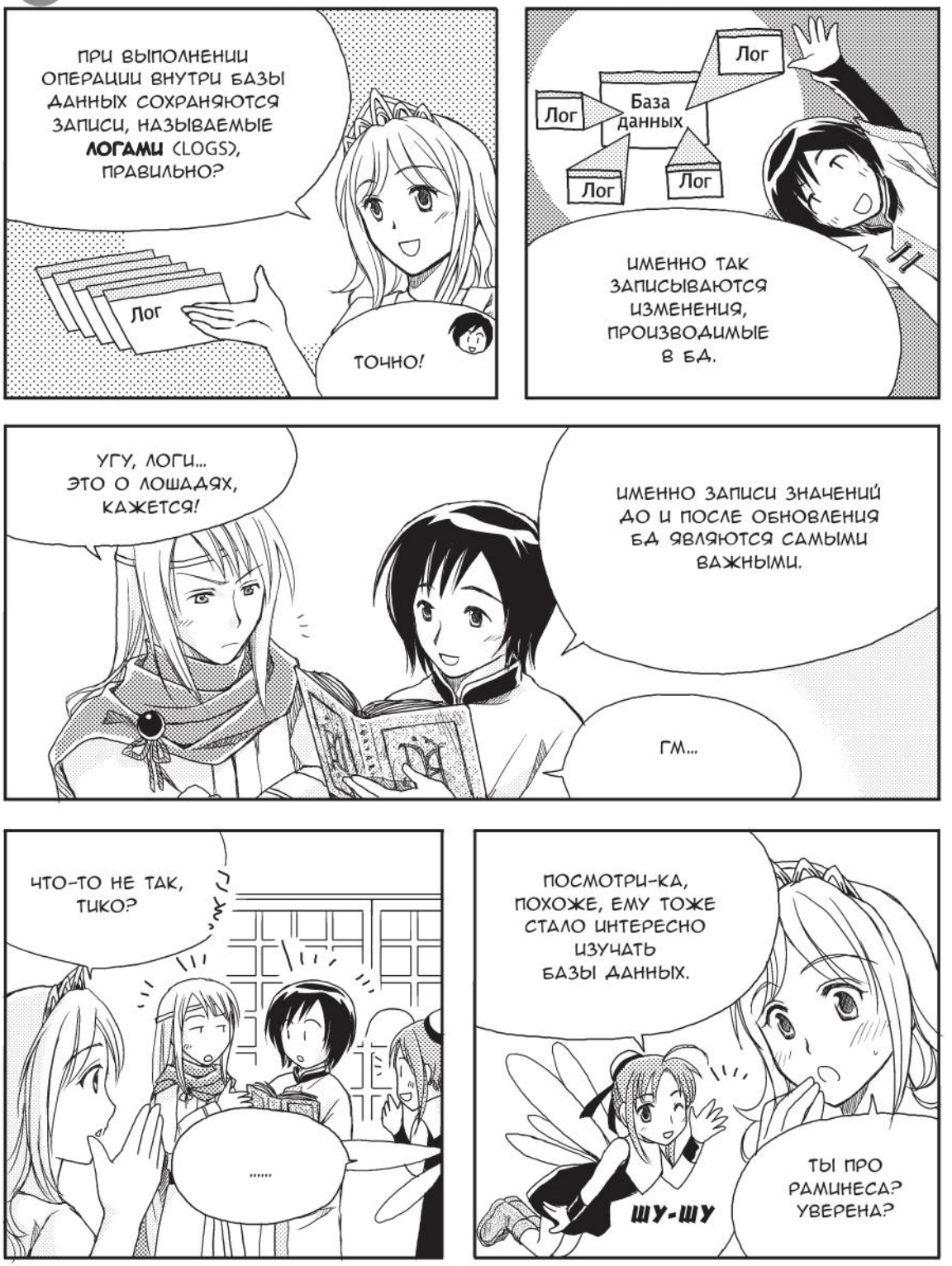
ПРИМЕНЕНИЕ ИНДЕКСОВ УМЕНЬШАЕТ ЧИСЛО ОБРАЩЕНИЙ К ДИСКУ.
ПОНЯТНО! ЕСЛИ УМЕНЬШИТЬ ЧИСЛО ОБРАЩЕНИЙ К ДИСКУ, ТО НАШ ПОИСК СТАНЕТ ГОРАЗДО БЫСТРЕЕ!
Что это?
Ого!
?

ЭЙ, ТЫ С КЕМ СЕЙЧАС РАЗГОВАРИВАЕШЬ?
Тут кто-то ещё?
Принцесса, осторожней!
Ох, я забыла...

ПРОСТО САМА С СОБОЙ БОЛТАЮ!
Ха, ха, ха.
Тико
О-ПА, РАМИНЕС НЕ ВИДИТ ЕЁ.
?
ВСЕМ ВНИМАНИЕ! В СТАНДАРТНОЙ БАЗЕ ДАННЫХ, КАК Я ПОНИМАЮ, ДОБАВЛЯТЬ ИНДЕКСЫ — ЭТО ДЕЛО АДМИНИСТРАТОРА БД.
Надо отвлечь его внимание.
Мозги мне пудрят зачем-то.
ХЛОП
Ещё-что-то...?
С ДРУГОЙ СТОРОНЫ, СОЗДАНИЕ СЛИШКОМ БОЛЬШОГО ЧИСЛА ИНДЕКСОВ МОЖЕТ ПРИВЕСТИ К НЕЭФФЕКТИВНОСТИ.
Так и есть?
ТАК ИНДЕКС — ЭТО ХОРОШО ИЛИ ПЛОХО?
ХЛОП
ШЁПОТ
НУ, ЭТО ПРИМЕРНО ТАК. ПРЕДСТАВЬ, ЧТО В КНИГЕ СЛИШКОМ МНОГО ИНДЕКСОВ В ПРЕДМЕТНОМ УКАЗАТЕЛЕ. ЭТО КАК БУДТО ПОСТАВИТЬ ТЕЛЕГУ ВПЕРЕДИ ЛОШАДИ, ПОНИМАЕШЬ?
БОЛЕЕ ТОГО, КОГДА МЫ ОБНОВЛЯЕМ ДАННЫЕ, МЫ ДОЛЖНЫ ОБНОВЛЯТЬ ИНДЕКСЫ, А ЭТО ВСЁ ЗАЙМЁТ ЕЩЁ БОЛЬШЕ ВРЕМЕНИ.
Зубы заговаривает♡
НЕУЖЕЛИ ВСЁ ЭТО РАБОТАЕТ?
Ха, ха!
ЭТО ТИКО ТАК СКАЗАЛА!

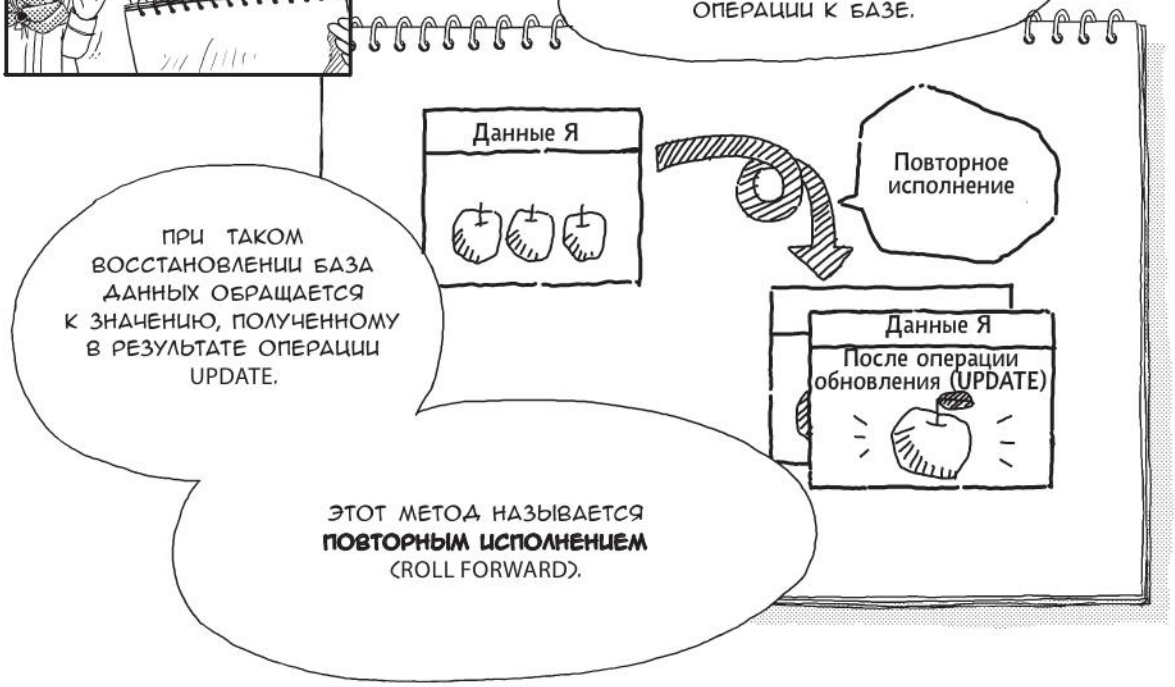
ХМ, Я ПОНЯЛ. БАЗА ДАННЫХ КУДА УМНЕЕ, ЧЕМ Я РАНЬШЕ ДУМАЛ.
Откуда взялась эта роза?
БАЗА ДАННЫХ...
НУ, НЕТ! МЕНЯ ТАК ПРОСТО НЕ ПРОВЕДЁШЬ! ЕСТЬ И ДРУГИЕ ПРОБЛЕМЫ!
ВЗ-З-З
ШВАХ!
НАПРИМЕР...
ЧТО, ЕСЛИ СИСТЕМА, С КОТОРОЙ РАБОТАЕТ БД, ВЫЙДЕТ ИЗ СТРОЯ ИЗ-ЗА ПРОБЛЕМ С ПИТАНИЕМ?
Каюк
Данные
Данные
НЕТ ПРОБЛЕМ.
Вот так!
ЕСТЬ ЖЕ ФУНКЦИЯ АВАРИЙНОГО ВОССТАНОВЛЕНИЯ.
КАКОГО ЕЩЁ АВАРИЙНОГО ВОССТАНОВЛЕНИЯ?
ИМЕЕТСЯ В ВИДУ, ЧТО, ЕСЛИ ПРОИЗОШЁЛ СБОЙ, БД МОЖЕТ ДОЛЖНЫМ ОБРАЗОМ ВОССТАНОВИТЬСЯ ПОСЛЕ НЕГО.
Data
ДА НУ?

АВАРИЙНОЕ ВОССТАНОВЛЕНИЕ

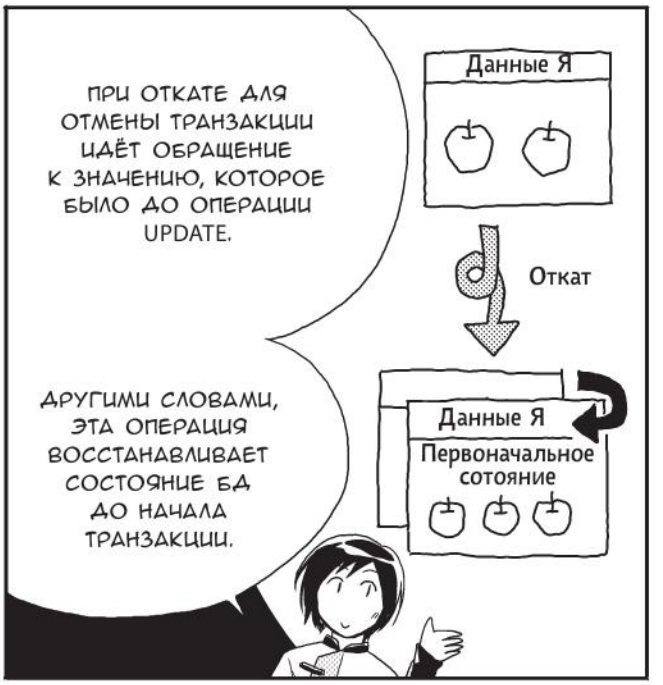


Если проблема возникла после того, как транзакция уже была зафиксирована, это означает, что операции этой транзакции завершены.

Поэтому ты можешь восстановить данные, повторно применив операции к базе.



А ЧТО, ЕСЛИ ТРАНЗАКЦИЯ ЕЩЁ НЕ БЫЛА ЗАФИКСИРОВАНА, А СБОЙ ПРОИЗОШЁЛ?
Опять проблемы.
НЕ ВОЛНУЙСЯ! В ЭТОМ СЛУЧАЕ ПРОИЗВОДИТСЯ ОТКАТ (ROLLBACK).

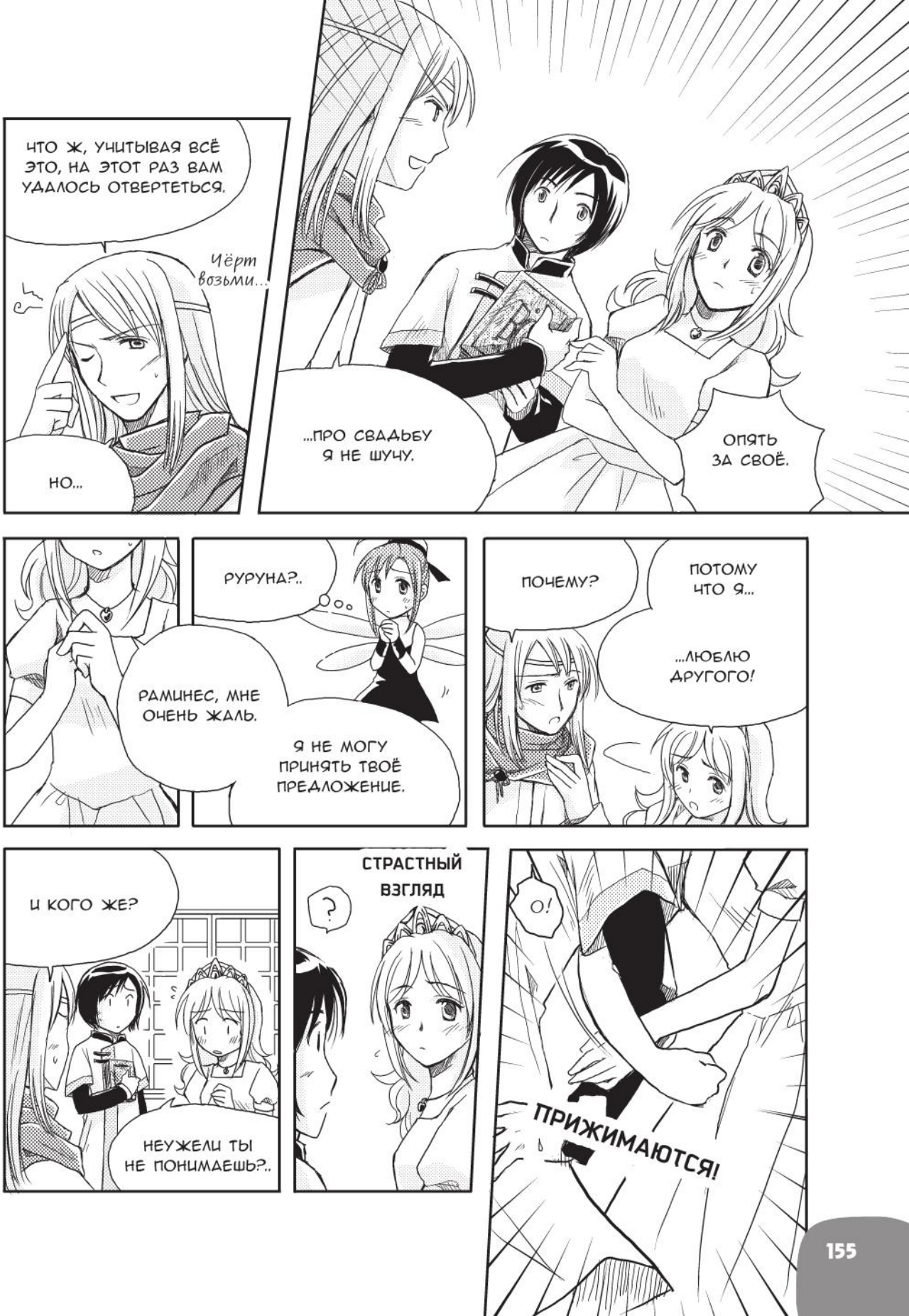

ПРИ ОТКАТЕ ДЛЯ ОТМЕНЫ ТРАНЗАКЦИИ ИДЁТ ОБРАЩЕНИЕ К ЗНАЧЕНИЮ, КОТОРОЕ БЫЛО ДО ОПЕРАЦИИ UPDATE.
ДРУГИМИ СЛОВАМИ, ЭТА ОПЕРАЦИЯ ВОССТАНАВЛИВАЕТ СОСТОЯНИЕ БД ДО НАЧАЛА ТРАНЗАКЦИИ.
Данные Я
Откат
Данные Я
Первоначальное сотояние

СИСТЕМА ВОССТА-НАВЛИВАЕТ ДАННЫЕ, ПРИ ЭТОМ ПРОВЕРЯЯ, ЧТОБЫ НЕ БЫЛО НЕСООТВЕТСТВИЙ.
УГУ.

НУ ЧТО ЖЕ, МНЕ НЕЗНАКОМЫ ТАКИЕ ПОНЯТИЯ, КАК ОПЕРАЦИИ ФИКСАЦИЯ И ТРАНЗАКЦИЯ...
Гмм...

НО ВСЁ ЖЕ НАДЕЮСЬ, ЧТО ВАШИ МЕРЫ БЕЗОПАСНОСТИ СРАБОТАЮТ.
ВЗДОХ
ТЕПЕРЬ ТЫ ПОНИМАЕШЬ?

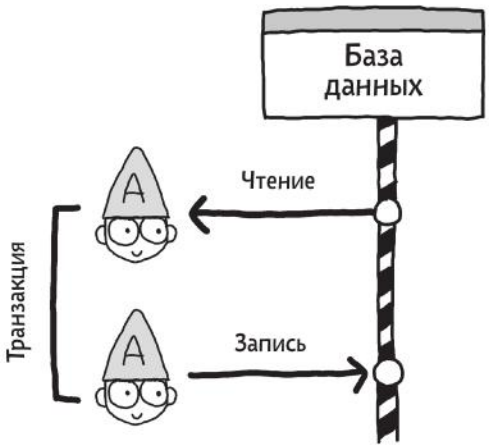
ВИДИШЬ, БАЗА ДАННЫХ — ЭТО СИЛЬНАЯ ВЕЩЬ! ДАЖЕ ЕСЛИ СЛУЧАЕТСЯ СБОЙ.


ЧТО Ж, УЧИТЫВАЯ ВСЁ ЭТО, НА ЭТОТ РАЗ ВАМ УДАЛОСЬ ОТВЕРТЕТЬСЯ.
Чёрт возьми...
НО...
...ПРО СВАДЬБУ Я НЕ ШУЧУ.
ОПЯТЬ ЗА СВОЁ.
РАМИНЕС, МНЕ ОЧЕНЬ ЖАЛЬ.
РУРУНА?..
Я НЕ МОГУ ПРИНЯТЬ ТВОЁ ПРЕДЛОЖЕНИЕ.
ПОЧЕМУ?
ПОТОМУ ЧТО Я...
...ЛЮБЛЮ ДРУГОГО!
И КОГО ЖЕ?
НЕУЖЕЛИ ТЫ НЕ ПОНИМАЕШЬ?..
СТРАСТНЫЙ ВЗГЛЯД
?
О!
ПРИЖИМАЮТСЯ!

Я буду с Кейном навеки, и с помощью нашей базы данных...
Как?.. Ты хочешь сказать...
...мы позаботимся о том, чтобы наше королевство процветало!
Как же так! Нет... не поступай так со мной...
Прости.
Плюх!
Ты выбрала этого скромняги Кейна?
Ну, в общем, да, прости меня.
Кейн, зачем ты извиняешься?
О, да, я не должен. Прошу прощения.
Будь всегда со мной, Кейн.
Да, да, Ваше Высочество!
Почему, ну, почему?
Хорошая из них получится пара. Ведь они оба умеют работать с базами данных!

СВОЙСТВА ТРАНЗАКЦИЙ

Исследования Кейна показали, что пользователи базы даных могут искать (select), вставлять (insert), обновлять (update) и удалять (delete) данные. Набор операций, успешно проведённых одним пользователем, называется транзакцией (transaction).



Когда пользователи работают с одной базой, важно дать гарантию, что многочисленные транзакции можно проводить, не внося противоречий в данные. Также важно защитить данные от несогласованности вследствие возникновения сбоя в период обработки транзакции. В этой связи в приведённой ниже таблице перечислены требования, которым должна отвечать транзакция. В виде аббревиатуры они будут читаться как ACID.

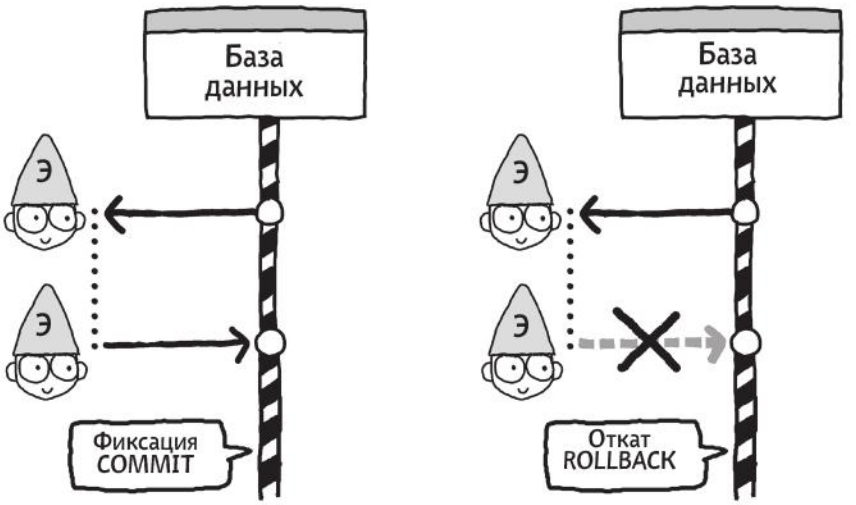
■ ТРЕБОВАНИЯ К ТРАНЗАКЦИИ

Свойства	Значение	Описание
A (Atomicity)	Атомарность	Транзакция должна заканчиваться либо операцией COMMIT, либо операцией ROLLBACK
C (Consistency)	Согласованность	Обработка транзакции никогда не должна приводить к потере согласованности БД
I (Isolation)	Изоляция	Даже если транзакции обрабатываются параллельно, результат должен быть таким же, как при последовательной обработке
D (Durability)	Устойчивость	Сбой не должен оказать влияния на содержание завершённой транзакции

Давайте подробнее рассмотрим каждое из этих свойств.

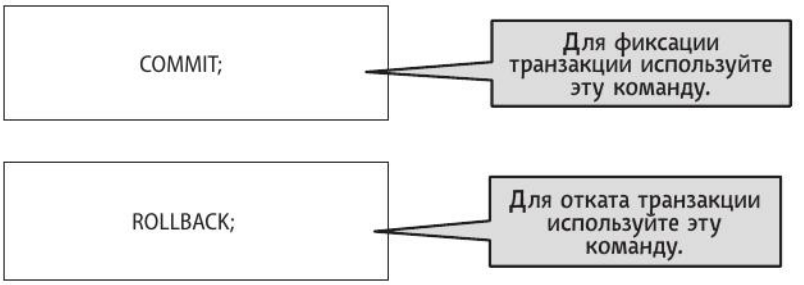
АТОМАРНОСТЬ (ATOMICITY)

Первое свойство, необходимое для транзакции, — это атомарность. Оно означает, что транзакция должна заканчиваться либо фиксацией, либо откатом, для того чтобы в базе данных не появились несогласованности. Проще говоря, либо все действия транзакции завершены, либо отменены. Фиксация (commit) завершает транзакцию. Откат (rollback) отменяет операции в транзакции.



В некоторых случаях фиксация и откат выполняются автоматически. Вы также можете задать, какая из этих операций должна быть выполнена. Например, вы можете задать откат в случае возникновения ошибки.

Для выполнения этих операций вы можете применять команды COMMIT и ROLLBACK.



ВОПРОСЫ И ЗАДАНИЯ

Ответив на эти вопросы, вы сможете проверить своё понимание атомарности. Ответы смотрите на стр. 174.

B1

Напишите SQL-команду, которую можно использовать для завершения транзакции.

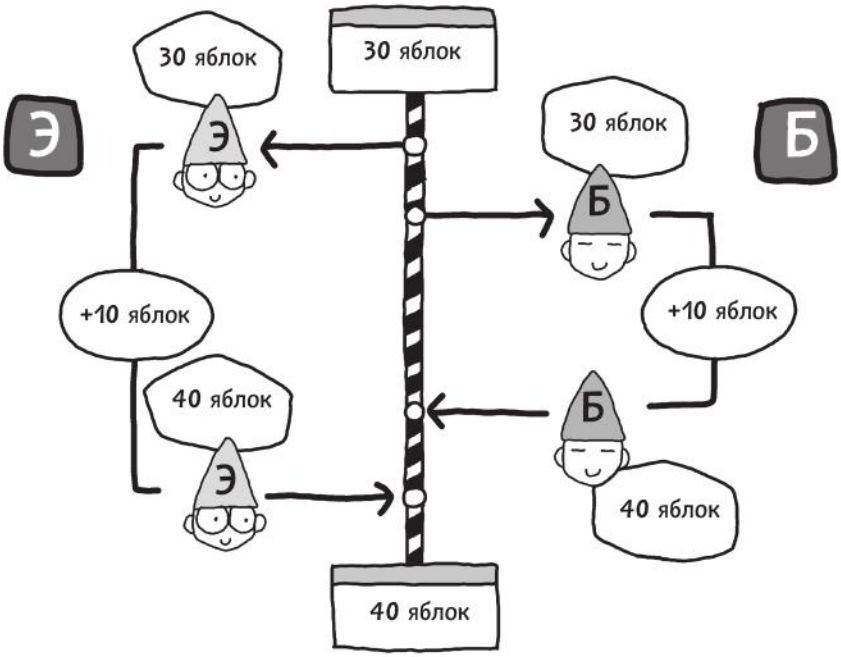
B2

Напишите SQL-команду, которую можно использовать для отмены транзакции.

СОГЛАСОВАННОСТЬ (CONSISTENCY)

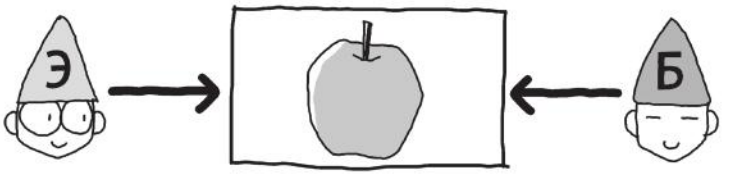
Транзакция не должна создавать ошибок. Если база данных была согласована до выполнения транзакции, то после возникновения этой транзакции она должна остаться согласованной.

Кейн привел пример об Энди и Бэки, каждый из которых пытался добавить 10 яблок к изначально заданному значению 30. Вместо того чтобы выдать верное значение — 50 яблок, база данных показала общий результат, равный 40 яблокам. Этот тип ошибки называется **потерянное обновление** (lost update).



Когда транзакции обрабатываются параллельно, получить доступ к одной и той же таблице или записи в одно и то же время может более чем одна транзакция, поэтому могут возникнуть противоречивые данные.

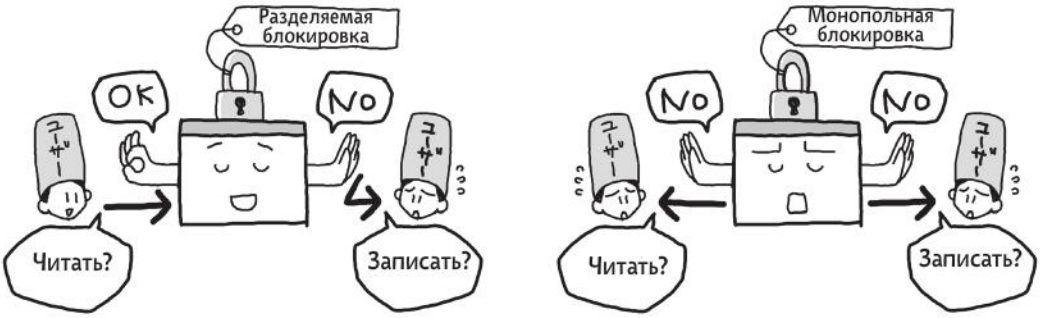
Таблицы и записи, над которыми производятся операции в транзакции, называются **ресурсами** (resources). В БД транзакции должны иметь возможность параллельного доступа к одним и тем же ресурсам, не создавая противоречий.



ИЗОЛЯЦИЯ (ISOLATION)

Когда две или более паралелльных транзакций выдают одинаковый результат, как будто они выполнялись в разное время, такой порядок обработки называется **упорядоченным** (serializable). Изоляция защищает от ошибок, и ей необходима упорядоченная очерёдность.

Для того чтобы сделать порядок обработки упорядоченным, вам нужно иметь возможность управлять одновременно выполняющимися транзакциями. Наиболее часто для этой цели используют метод управления блокировками. Разделяемая блокировка (shared lock) используется при чтении данных, а монопольная блокировка (exclusive lock) — при записи данных.



Когда применяется разделяемая блокировка, другой пользователь может наложить на другие транзакции разделяемую блокировку, но монопольную не может. Когда применяется монопольная блокировка, другой пользователь не может наложить на другие транзакции разделяемую или монопольную блокировку. Следующая таблица описывает отношения между разделяемой и монопольной блокировками.

■ ОТНОШЕНИЯ МЕЖДУ ДВУМЯ ВИДАМИ БЛОКИРОВОК

Вид блокировки	Разделяемая блокировка	Монопольная блокировка
Разделяемая блокировка	Да	Нет
Монопольная блокировка	Нет	Нет

ВОПРОСЫ И ЗАДАНИЯ

Вы поняли, что такое блокировки? Ответьте на следующие вопросы и проверьте правильность ответов на стр. 174.

В3

Если Энди применил разделяемую блокировку, может ли и Бэки её применить?

В4

Если Энди применил монопольную блокировку, может ли Бэки применить разделяемую?

В5

Если Энди применил разделяемую блокировку, может ли Бэки применить монопольную?

В6

Если Энди применил монопольную блокировку, может ли и Бэки её применить?

В7

Бэки не удалось применить монопольную блокировку. Перечислите возможности Энди относительно блокировки.

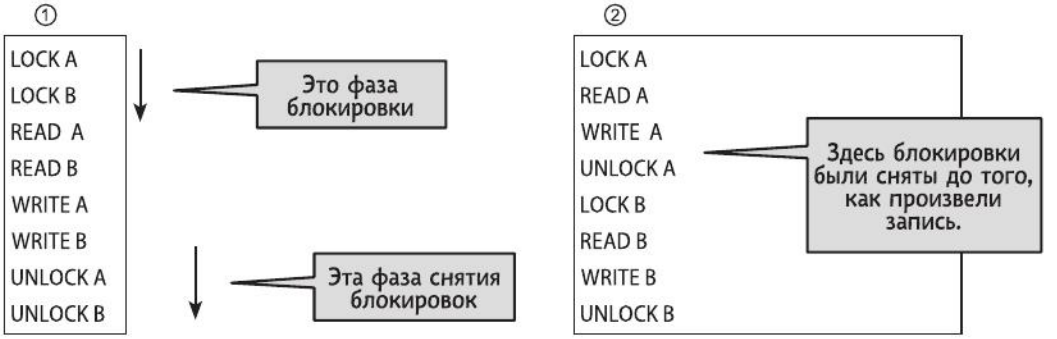
В8

Бэки не удалось применить разделяемую блокировку. Перечислите возможности Энди относительно блокировки.

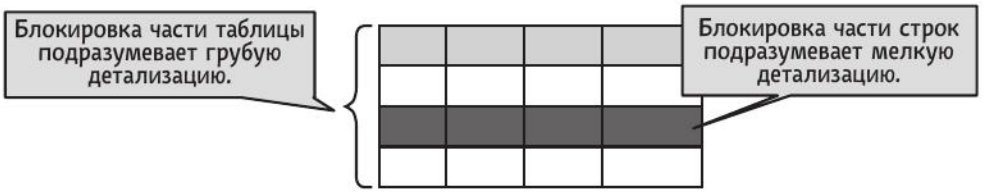
ДВУХФАЗНОЕ БЛОКИРОВАНИЕ (TWO-PHASE LOCKING)

Для того чтобы убедиться, что очередность упорядочена, нам надо применить определённые правила для установки и снятия блокировок. Одно из этих правил — это двухфазное блокирование, то есть для каждой транзакции необходимо применить две фазы: одна — для установки блокировок, другая — для их снятия.

Например, предположим, что ресурсы A и B подвергнуты блокированию. Транзакция ① соблюдает правило двухфазного блокирования, а транзакция ② нет. Упорядочивания можно достичь только, если каждая транзакция подчиняется правилу двухфазного блокирования.



Существуют некоторые ресурсы, которые можно блокировать. Например, вы можете заблокировать данные в части таблиц или части строк. Степень, до которой ресурсы можно заблокировать, называется детализацией (granularity). Грубая детализация (coarse granularity) возникает, когда сразу блокируется много ресурсов, а мелкая детализация (fine granularity) — когда блокируется мало ресурсов.



Когда детализация грубая (или высокая), число блокировок, приходящихся на транзакцию, сокращено, что облегчает управление детализацией. В свою очередь это уменьшает объём обработки данных, необходимый ЦПУ, на котором работает база данных. С другой стороны, когда заблокировано много ресурсов, всё дольше приходится ждать, когда будут сняты блокировки, используемые другими транзакциями. Следовательно при высокой детализации число транзакций, которые вы можете выполнить, стремится к уменьшению.

ВОПРОСЫ И ЗАДАНИЯ

Ответьте на эти вопросы и проверьте правильность ответов на стр. 175.

В9

Ресурс, намеченный для блокировки, был изменён с таблицы на строку. Что произойдёт с числом транзакций, которые вы сможете выполнить параллельно?

В10

Ресурс, намеченный для блокировки, был изменён со строки на таблицу. Что произойдёт с числом транзакций, которые вы сможете выполнить параллельно?

ДРУГИЕ ВИДЫ УПРАВЛЕНИЯ ПАРАЛЛЕЛИЗМОМ (ПАРАЛЛЕЛЬНЫМ ДОСТУПОМ)

Для эффективного выполнения двух или более транзакций одновременно вы можете использовать блокировки. Однако применение блокировок влечёт за собой управление ими, так как могут возникать взаимоблокировки (deadlocks) — места, где действия пользователя противоречат друг другу. Когда у вас небольшое количество транзакций или высоко число операций считывания, можно использовать более простые методы управления параллелизмом, например:

УПРАВЛЕНИЕ МЕТКОЙ ВРЕМЕНИ (TIMESTAMP CONTROL)

Метка, содержащая в себе время доступа, называемая меткой времени (timestamp), присваивается данным, используемым во время транзакции. Если другая транзакция с более поздней меткой времени уже произвела обновление данных, операция не будет разрешена. Когда операции чтения и записи не разрешены, происходит откат транзакции.

ОПТИМИСТИЧЕСКОЕ УПРАВЛЕНИЕ ПАРАЛЛЕЛИЗМОМ (OPTIMISTIC CONTROL)

Этот метод разрешает операцию чтения. Когда происходит попытка записи, проверяется, не выполнилась ли какая-либо другая транзакция. Если другая транзакция уже обновила данные, запись не производится, и транзакция откатывается назад.

В БД можно установить уровень транзакций, которые могут обрабатываться параллельно. Это называется уровнем изоляции (isolation level).

В SQL с помощью команды SET TRANSACTION можно задать уровень изоляции для следующих транзакций:

- READ UNCOMMITTED — чтение незафиксированных данных;
- READ COMMITTED — чтение зафиксированных данных;
- REPEATABLE READ — повторяющееся чтение;
- SERIALIZABLE — упорядочиваемость

```
SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;
```

В зависимости от уровня изоляции, могут возникнуть следующие проблемы:

■

Транзакция	«Грязное» чтение (черновое чтение)	Неповторяющееся чтение	Фантомное чтение
READ UNCOMMITTED	Возможно	Возможно	Возможно
READ COMMITTED	Не возникнет	Возможно	Возможно
REPEATABLE READ	Не возникнет	Не возникнет	Возможно
SERIALIZABLE	Не возникнет	Не возникнет	Не возникнет

«Грязное» чтение (dirty read) возникает, когда транзакция №1 читает данные, измененные, но еще не подтвержденные транзакцией №2, которая впоследствии может откатиться.

Неповторяющееся чтение (non-repeatable read) возникает, когда транзакция считывает одни и те же данные дважды и получает разное значение.

Фантомное чтение (phantom) возникает, когда транзакция ищет строки, совпадающие с определённым условием, но находит не те строки из-за того, что другая транзакция произвела изменения.

УСТОЙЧИВОСТЬ

База данных управляет важными данными, поэтому гарантия безопасности и устойчивости в случае сбоя является ключевым моментом. Обеспечение безопасности также важно для предотвращения записи данных и появления противоречивых данных в результате несанкционированного доступа.

В базе данных вы можете устанавливать права доступа тем, кто может пользоваться базой данных или её таблицами. Кейн избегает опасностей, грозящих базе данных королевства, тем, что совершенствует защиту БД.

В реляционных БД для предоставления прав на чтение и запись используется команда GRANT. С помощью этой команды вы можете давать разрешение другим пользователям БД работать с таблицами, которые вы создали. Установка прав доступа — это важная задача при работе с базой данных.

```
GRANT SELECT,UPDATE ON товары TO отдел_развития;
```

Эта команда даёт разрешение на обработку данных.

С помощью SQL-команд вы можете предоставлять следующие привилегиии (права доступа):

■ ПРИВИЛЕГИИ

Команда	Результат
SELECT	Даёт право пользователю выбирать строки из таблицы
INSERT	Даёт право пользователю вставлять строки в таблицу
UPDATE	Даёт право пользователю обновлять строки в таблице
DELETE	Даёт право пользователю удалять строки из таблицы
ALL	Даёт пользователю все привилегии

Благодаря WITH GRANT OPTION пользователь, указанный в команде GRANT, получает возможность предоставлять привилегии другим пользователям. С помощью приведённой ниже команды отдел развития разрешает другим пользователям производить выборку и обновлять данные в БД.

```
GRANT SELECT,UPDATE ON товары TO отдел_развития
WITH GRANT OPTION;
```

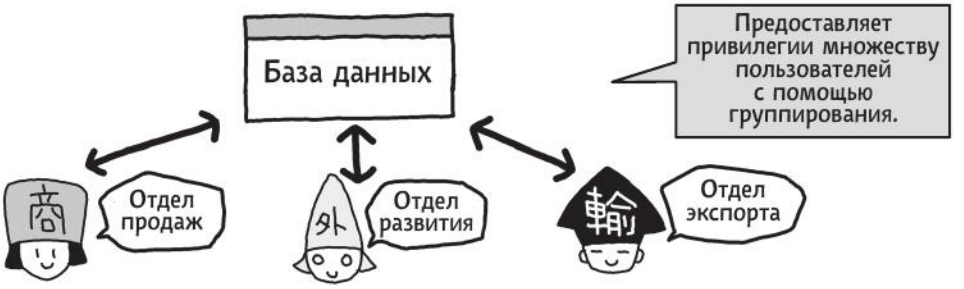
Привилегированный пользователь может предоставить привилегии другим пользователям.

Вы также можете отменить предоставленные пользователю привилегии. Для этого используется команда REVOKE.

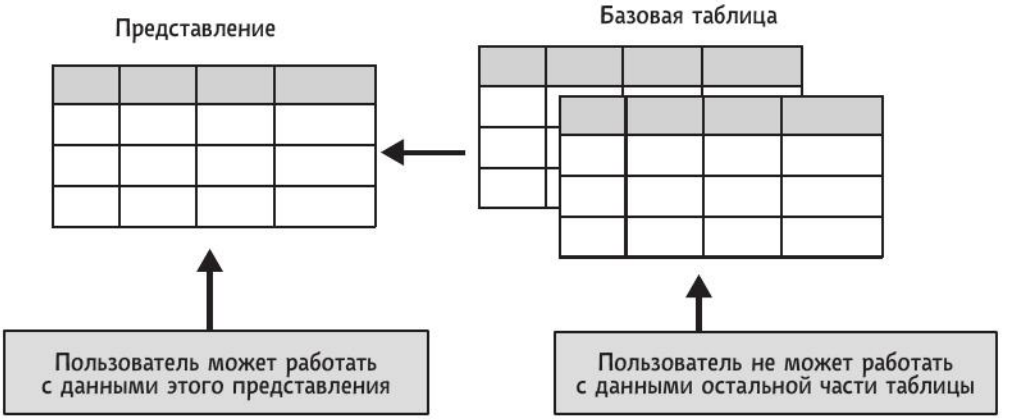
```
REVOKE SELECT,UPDATE ON товары FROM отдел_развития;
```

Эта команда отменяет привилегии пользователя.

Некоторые программные продукты могут группировать несколько привилегий и предоставлять их одновременно множеству пользователей. Группирование облегчает управление привилегиями.



Использование представлений (views), о которых рассказывалось на стр. 119, обеспечивает ещё более эффективное управление и улучшает безопасность. Сначала надо извлечь часть базовой таблицы и создать представление. Установка права доступа (привилегии) для этого представления означает, что привилегия также распространяется на выбранные данные в этом представлении.



ВОПРОСЫ И ЗАДАНИЯ

Попробуйте ответить на вопросы об устойчивости. Ответы на них вы найдёте на стр. 175.

B11

Напишите SQL-команду, которая разрешит отделу экспорта поиск данных в таблице «товары».

B12

Напишите SQL-команду, которая отберёт у отдела развития привелегию удалять данные из таблицы товаров.

B13

Для таблицы товаров, созданной администратором, были установлены права доступа, указанные ниже. Поставьте ДА или НЕТ в каждой ячейке приведённой таблицы, чтобы показать наличие или отсутствие привилегии для каждого отдела соответственно.

```
GRANT ALL ON товар TO отдел_развития;
GRANT SELECT,UPDATE ON товар TO отдел_продаж;
GRANT SELECT,INSERT ON товар TO отдел_экспорта;
```

Отдел	Выборка	Вставка	Обновление	Удаление
Отдел развития				
Отдел продаж				
Отдел экспорта				

ИНДЕКСЫ (INDEX)

База данных управляет огромным количеством данных, поэтому поиск определённых данных может занимать много времени. Но с помощью индексов мы можем его ускорить!

Код товара	Название товара	Цена	Регион
101	Дыни	800 G	Южный регион
102	Клубника	150 G	Центр
103	Яблоки	120 G	Северный регион
104	Лимоны	200 G	Южный регион
201	Каштаны	100 G	Северный регион
202	Хурма	160 G	Центр
301	Персики	130 G	Южный регион
302	Киви	200 G	Южный регион

Последовательный поиск занимает очень много времени

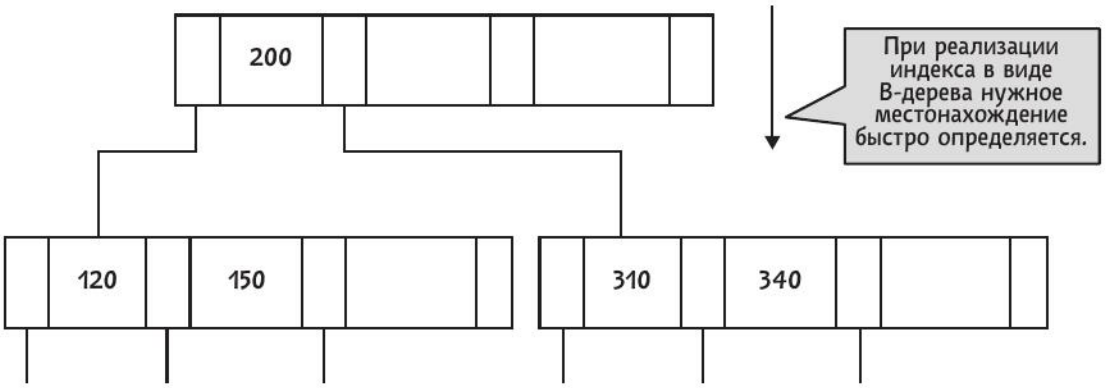
Индекс — это инструмент, который позволяет получить быстрый доступ к расположению искомых данных. Когда нужно найти какие-то данные в большой БД, поиск по индексу сулит быстрый результат.

Код товара	Название товара	Цена	Регион
101	Дыни	800 G	Южный регион
102	Клубника	150 G	Центр
103	Яблоки	120 G	Северный регион
104	Лимоны	200 G	Южный регион
201	Каштаны	100 G	Северный регион
202	Хурма	160 G	Центр
301	Персики	130 G	Южный регион
302	Киви	200 G	Южный регион

Расположение искомых данных можно быстро обнаружить с помощью их индекса.

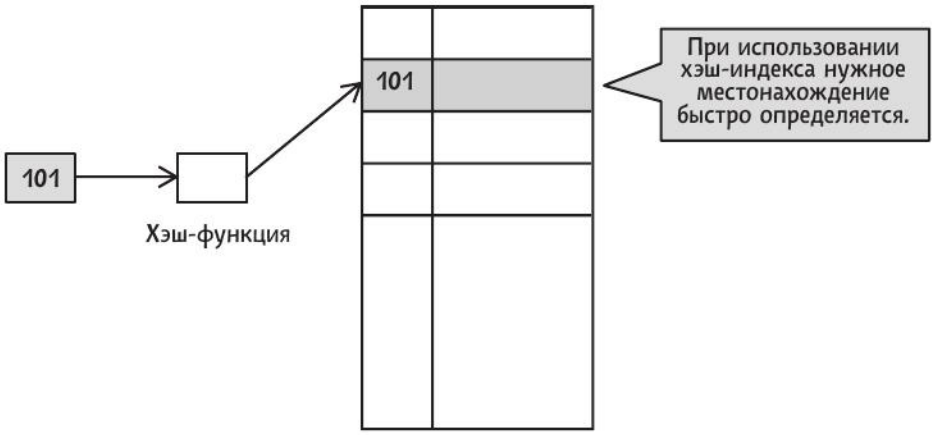
Индекс

Методы индексирования включают в себя B-деревья (B-trees) и хэши (hash). Индекс B-дерева состоит из родительских узлов и потомков, которые, в свою очередь, могут иметь своих потомков. Узлы организованы в определённом порядке. Каждый родитель содержит информацию о минимальных и максимальных значениях, содержащихся во всех его потомках. Это позволяет базе данных быстро ориентироваться в нужном направлении, пропуская целые сегменты дерева, которые, по всей вероятности, не содержат искомого значения.



При реализации индекса в виде B-дерева нужное местонахождение быстро определяется.

Метод хэш-индекса позволяет найти искомые данные, применяя к ключевому значению данных хэш-функцию. Хэш работает как уникальный отпечаток какого-либо значения. С помощью хэш-индексов можно выполнять особый поиск на полное совпадение, например поиск товара с кодом 101. Однако с помощью этого метода нельзя эффективно проводить поиск с условиями для сравнения, например поиск товаров с кодом не меньше, чем 101, или поиск с неявным указанием, например товары, на конце названий которых стоит буква «а».



В некоторых случаях использование индексов не ускоряет процесс поиска — индекс экономит время, только если вы ищете небольшой кусок данных. Бывают случаи, когда индексы заново создаются при каждом обновлении данных, в результате чего замедляется операция обновления данных.

ВОПРОСЫ И ЗАДАНИЯ

Постарайтесь ответить на следующие вопросы об индексах. Ответы вы найдете на стр. 175.

B14

Какой индекс будет более эффективен при поиске на соответствия: B-дерево или хэш?

B15

Какой индекс будет более эффективен при поиске на несоответствия: B-дерево или хэш?

ОПТИМИЗАЦИЯ ЗАПРОСА

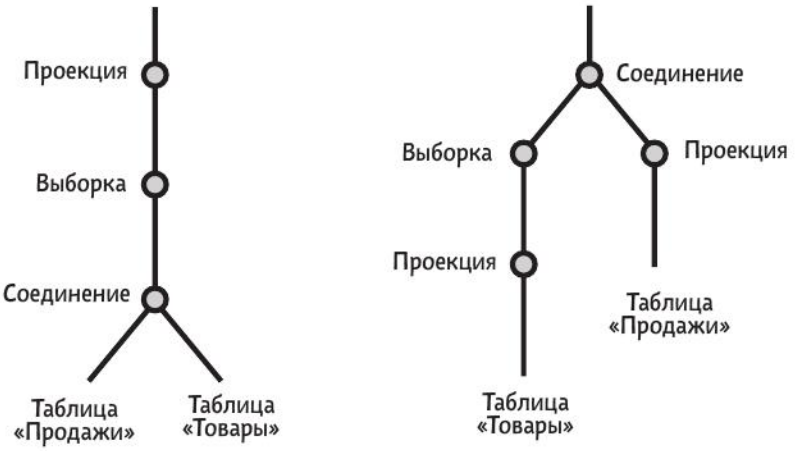
Когда вы создаёте запрос к базе, она анализирует SQL-запрос и решает, как использовать индекс так, чтобы как можно быстрее обработать запрос. Давайте изучим процедуру обработки запроса.

База данных может выбрать оптимальный порядок выполнения запроса. У большинства запросов может быть несколько порядков выполнения, при этом результат будет одинаковым, а вот скорость может оказаться разной. Например, предположим, есть запрос, где требуется получить даты продажи и названия товаров с ценой больше 200 G. Этот запрос состоит из следующих шагов.

```
SELECT дата,товар
FROM товары, продажи
WHERE цена>=200
AND товары.код_товара=продажи.код_товара;
```

① Соединим таблицы товаров и продаж.
② Выберем товары с ценой больше 200 G.
③ Извлечём столбцы «дата» и «товар».

Например, рисунок слева внизу показывает запрос, порядок выполнения которого был от пункта 1 до 3. Рисунок справа показывает запрос, порядок выполнения которого был от пункта 3 до 1. По какому пути не пойди, запросы эквивалентны.



Однако при обработке запроса от 1 до 3 потребуется больше времени, потому что, когда будет выполняться соединение, может создаться промежуточная таблица с большим количеством строк. С другой стороны, на процедуру от 3 до 1 потребуется меньше времени, так как сначала производятся выборка и проекция, отсекающие ненужные данные как можно раньше. Следовательно, на выполнение одного и того же запроса может потребоваться разное время — всё зависит от порядка выполнения операций проекции, выборки и соединения.

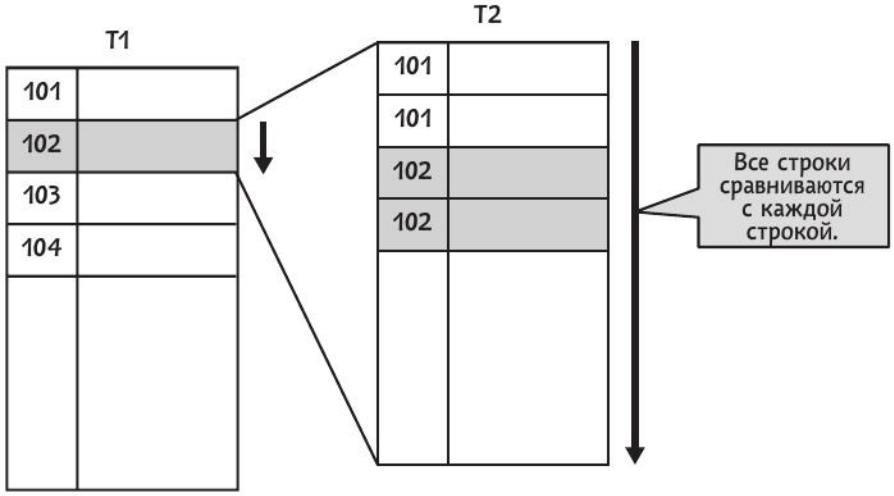
В общем и целом база данных должна использовать следующие правила нахождения оптимального порядка выполнения запроса:

- Сначала выполнить выборку, чтобы уменьшить число строк.
- Выполнить проекцию, чтобы уменьшить число столбцов, не соответствующих результату.
- Выполнить соединение.

Существуют разные способы выполнения проекции, выборки и соединения соответственно. Для выборки вы можете использовать либо поиск на полное совпадение, либо поиск с помощью индекса. Для соединения подходят следующие методы.

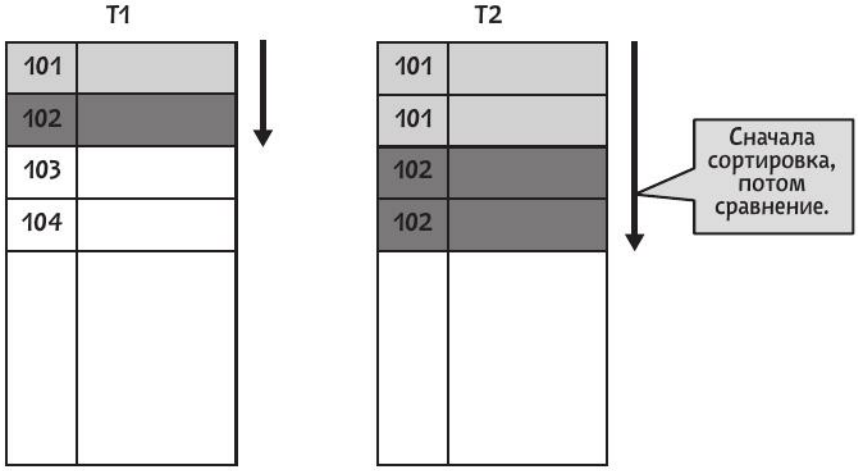
СОЕДИНЕНИЕ ВЛОЖЕННЫХ ЦИКЛОВ (NESTED LOOP JOIN)

Вложенный цикл сравнивает одну строку таблицы с несколькими строками другой таблицы (см. рисунок внизу). Например, одно из значений в строке в таблице T1 используется для нахождения совпадающих строк в таблице T2. Если значения одинаковы, тогда создаётся соединённая строка.



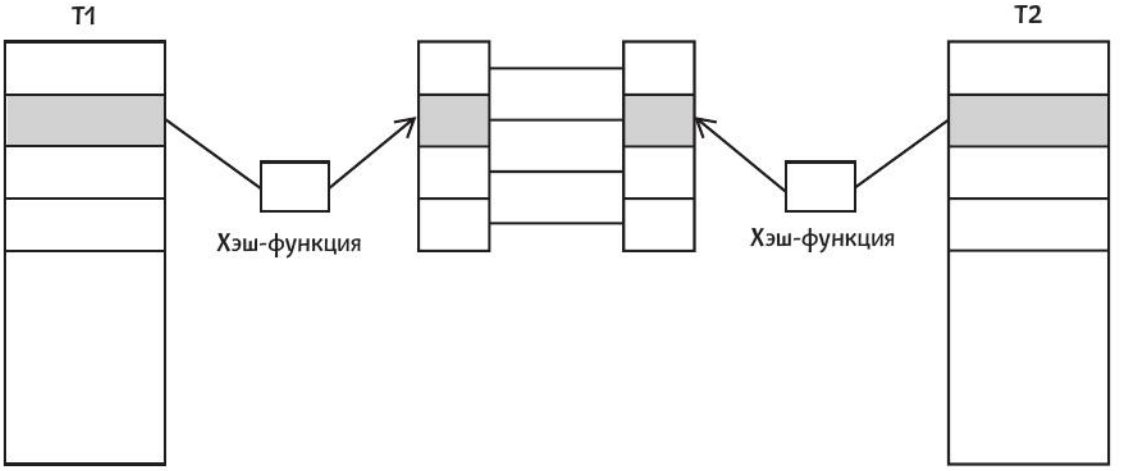
СОЕДИНЕНИЕ СОРТИРОВКА—СЛИЯНИЕ (SORT MERGE JOIN)

Метод «сортировка—слияние» сортирует строки с последующим слиянием в несколько таблиц (см. рисунок внизу). Сначала таблицы T1 и T2 сортируются полностью или частично. Затем они сравниваются, начиная с верхней строки, и каждый раз, когда обнаруживается такое же значение, создаётся соединённая строка. Так как таблицы уже были отсортированы, обработку запроса нужно вести только в одном направлении, поэтому затрачивается меньше времени. Однако вам надо помнить о времени, необходимом для первоначальной сортировки.



ХЭШ-СОЕДИНЕНИЕ (HASH JOIN)

Хэш делит одну из таблиц при помощи хэш-функции с последующим слиянием со строкой другой таблицы, содержащей такое же хэш-значение. Этот метод эффективен для выборки строки для соединения.



ОПТИМИЗАТОР (OPTIMIZER)

При обработке запроса эти различающиеся методы, о которых мы говорили, проверяются на оптимальное выполнение задачи. В базе данных функция, которая отвечает за оптимизацию запросов, называется оптимизатором (optimizer). Существуют два основных типа оптимизатора.

НА ОСНОВЕ ПРАВИЛ (RULE BASED)

Перед выполнением любой операции устанавливаются определённые правила. Например, некоторые операции можно объединить или перестроить, во многом так же, как производят манипуляции с математическим уравнением, и оно всё равно выражает одно и то же. Оптимизатор пытается найти наиболее эффективный способ обработки запроса, при этом результат должен быть одинаков.

ПО СТОИМОСТИ ВЫПОЛНЕНИЯ (COST BASED)

При этом методе происходит подсчёт стоимости обработки запроса, основанный на статистике, которую ведёт БД. Оптимизация по стоимости выполнения иногда более гибкая, чем на основе правил, но она требует периодических обновлений статистики БД. На анализ и управление этой статистикой требуется много времени.

КОГДА НАСТУПАЕТ КАТАСТРОФА

В базе данных должен быть механизм, способный защитить данные в системе в случае возникновения сбоя. Для обеспечения устойчивости транзакций обязательным требованием является недопущение некорректных или искажённых данных, которые могут появиться в ре-

зультате сбоя. Для защиты себя от сбоев база данных выполняет различные операции, к которым относится создание резервных копий и логов транзакций.

ВИДЫ СБОЕВ

Сбои в работе БД могут возникать по разным обстоятельствам. Возможны следующие разновидности сбоев:

- отказ транзакции;
- отказ системы;
- отказ носителя.

Отказ транзакции (transaction failure) возникает по причине того, что транзакция не может быть зафиксирована из-за ошибки в самой транзакции. Когда такое случается, происходит откат транзакции.

Отказ системы (system failure) возникает, когда система перестаёт работать в результате отключения электропитания или подобных сбоев.

Отказ носителя (media failure) возникает в результате повреждения жёсткого диска, на котором находится БД. В таком случае выполнятся аварийное восстановление с использованием резервных копий. Транзакции, зафиксированные после того, как были созданы резервные копии, выполняются повторно.

КОНТРОЛЬНЫЕ ТОЧКИ (CHECKPOINTS)

Чтобы повысить эффективность операции записи данных в БД, их зачастую записывают на короткий интервал времени в буфер (buffer) (сегмент памяти для временного хранения данных). Содержимое буфера и базы данных синхронизируется, а затем записывается контрольная точка (checkpoint). Когда БД записывает контрольную точку, ей в случае сбоя не потребуется выполнять аварийного восстановления транзакций, зафиксированных до контрольной точки. Транзакции, которые не были зафиксированы до контрольной точки, должны быть восстановлены.

Теперь предположим, что показанные ниже транзакции выполняются в момент возникновения отказа системы. Какие из них надо откатить? А какие выполнить повторно?

ВОПРОСЫ И ЗАДАНИЯ

Ответьте на эти вопросы, глядя на приведённую выше диаграмму. Найти ответы можно на стр. 175.

B16

Что надо сделать с T1?

B17

Что надо сделать с T2?

B18

Что надо сделать с T3?

В случае отказа в работе БД описанные выше механизмы восстановления защитят базу от несоответствий. Вот почему при её использовании вы можете быть уверены в её целостности и сохранности.

ИТОГИ

- Вы можете устанавливать привилегии для пользователей БД.
- Блокировка гарантирует согласованность данных, когда базой пользуется много людей.
- Индексы ускоряют поиск.
- В БД есть функции восстановления после сбоев.

ОТВЕТЫ

01 COMMIT;

02 ROLLBACK;

03 Да.

04 Нет.

05 Нет.

06 Нет.

07 Разделяемая блокировка.

08 Монопольная блокировка.

09 Увеличивается.

010 Уменьшается.

011 GRANT SELECT ON товар TO отдел_экспорта;

012 REVOKE DELETE ON товар FROM отдел_развития;

013

Отдел	Выборка	Вставка	Обновление	Удаление
Отдел развития	Да	Да	Да	Да
Отдел продаж	Нет	Да	Нет	Да
Отдел экспорта	Нет	нет	Да	Да

014 Хэш.

015 B-дерево.

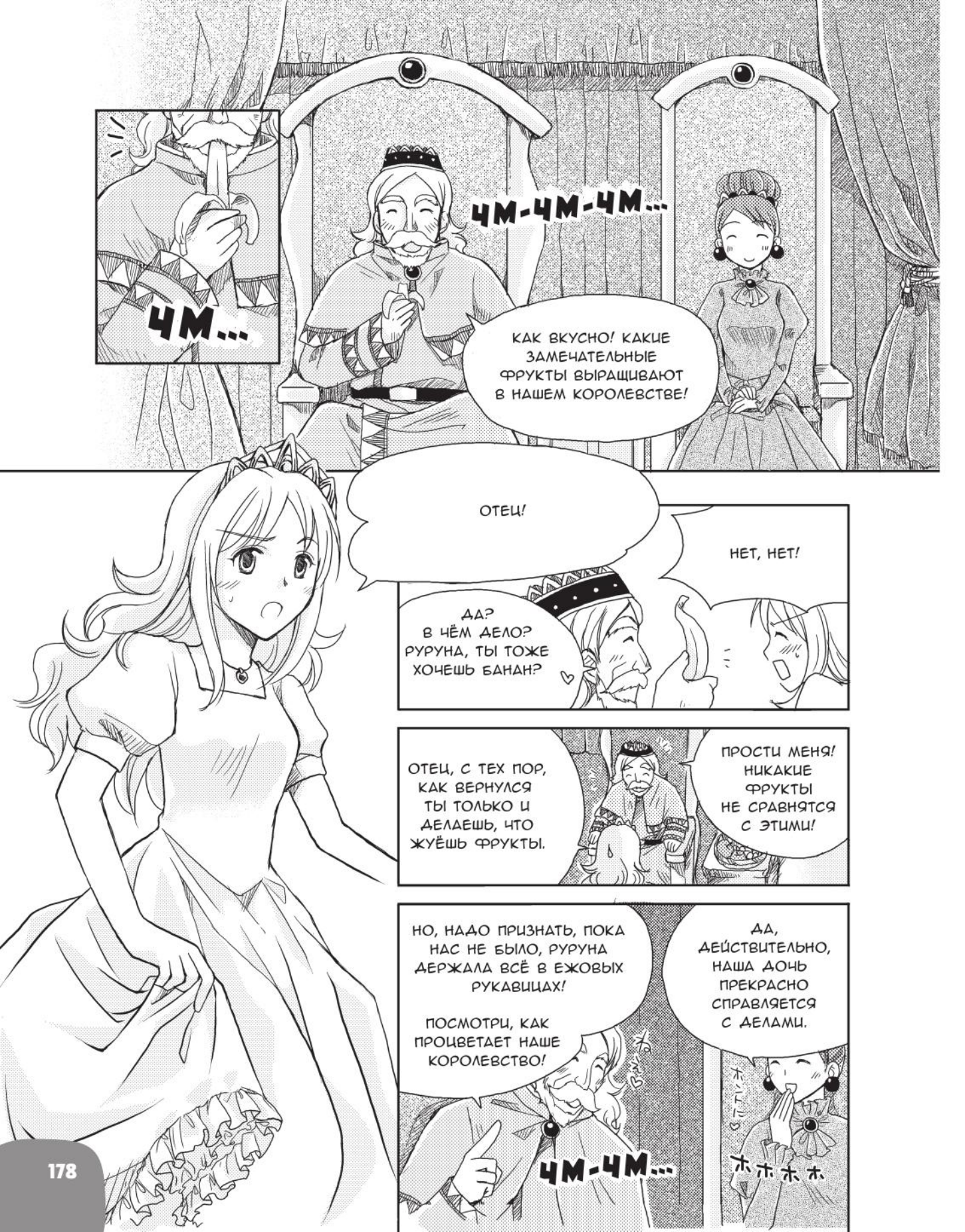
016 Выполняется откат, так как транзакция не зафиксирована в момент возникновения сбоя.

017 Транзакция выполняется повторно, так как она была зафиксирована в момент возникновения сбоя.

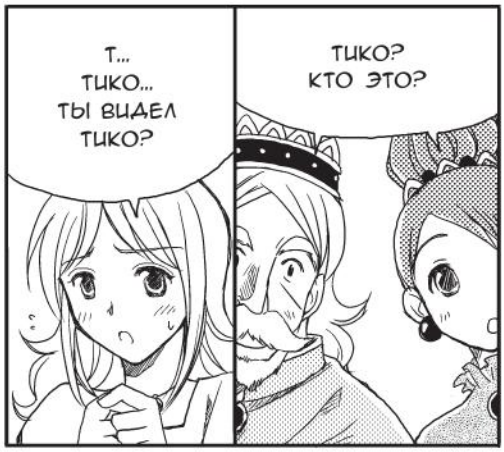
018 Восстановление не требуется, так как транзакция была зафиксирована на момент установки контрольной точки.

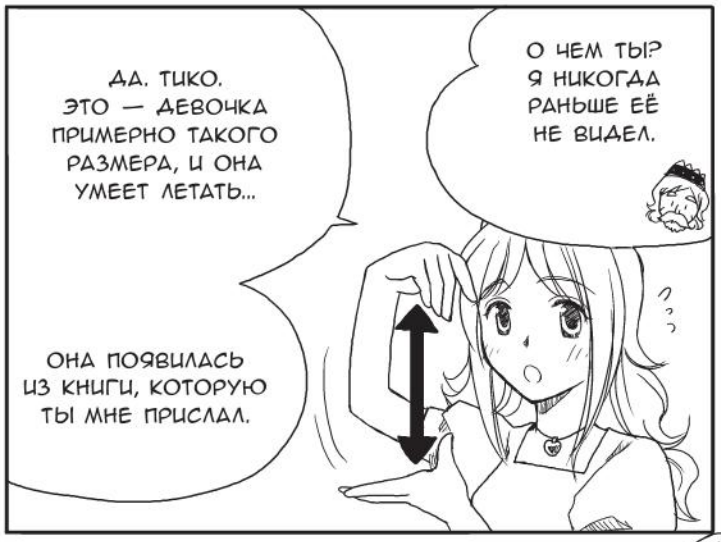
6.

КРУГОМ БАЗЫ ДАННЫХ

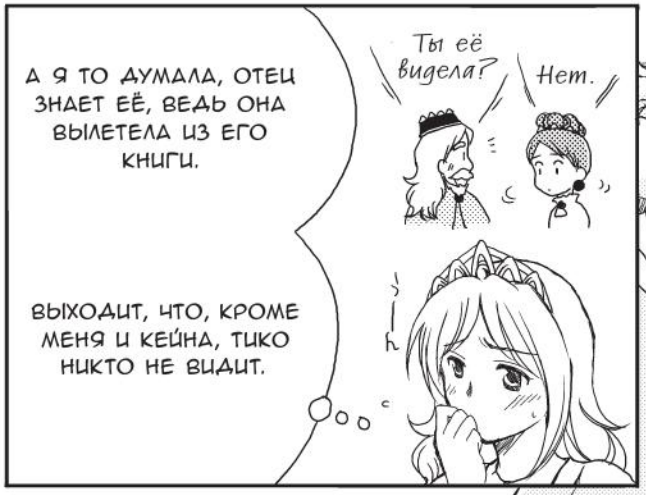

ЧМ...
ЧМ-ЧМ-ЧМ...
КАК ВКУСНО! КАКИЕ ЗАМЕЧАТЕЛЬНЫЕ ФРУКТЫ ВЫРАЩИВАЮТ В НАШЕМ КОРОЛЕВСТВЕ!
ОТЕЦ!
НЕТ, НЕТ!
ДА? В ЧЁМ ДЕЛО? РУРУНА, ТЫ ТОЖЕ ХОЧЕШЬ БАНАН?
ОТЕЦ, С ТЕХ ПОР, КАК ВЕРНУЛСЯ ТЫ ТОЛЬКО И ДЕЛАЕШЬ, ЧТО ЖУЁШЬ ФРУКТЫ.
ПРОСТИ МЕНЯ! НИКАКИЕ ФРУКТЫ НЕ СРАВНЯТСЯ С ЭТИМИ!
НО, НАДО ПРИЗНАТЬ, ПОКА НАС НЕ БЫЛО, РУРУНА ДЕРЖАЛА ВСЁ В ЕЖОВЫХ РУКАВИЦАХ!
ПОСМОТРИ, КАК ПРОЦВЕТАЕТ НАШЕ КОРОЛЕВСТВО!
ДА, ДЕЙСТВИТЕЛЬНО, НАША ДОЧЬ ПРЕКРАСНО СПРАВЛЯЕТСЯ С ДЕЛАМИ.
ЧМ-ЧМ...
ХО ХО ХО

НО У ТЕБЯ, НАВЕРНОЕ, БЫЛИ И ДРУГИЕ ДЕЛА?
ДА, ВЕРНО!


Т... ТИКО... ТЫ ВИДЕЛ ТИКО?
ТИКО? КТО ЭТО?


ДА. ТИКО. ЭТО — ДЕВОЧКА ПРИМЕРНО ТАКОГО РАЗМЕРА, И ОНА УМЕЕТ ЛЕТАТЬ...
ОНА ПОЯВИЛАСЬ ИЗ КНИГИ, КОТОРУЮ ТЫ МНЕ ПРИСЛАЛ.
О ЧЕМ ТЫ? Я НИКОГДА РАНЬШЕ ЕЁ НЕ ВИДЕЛ.

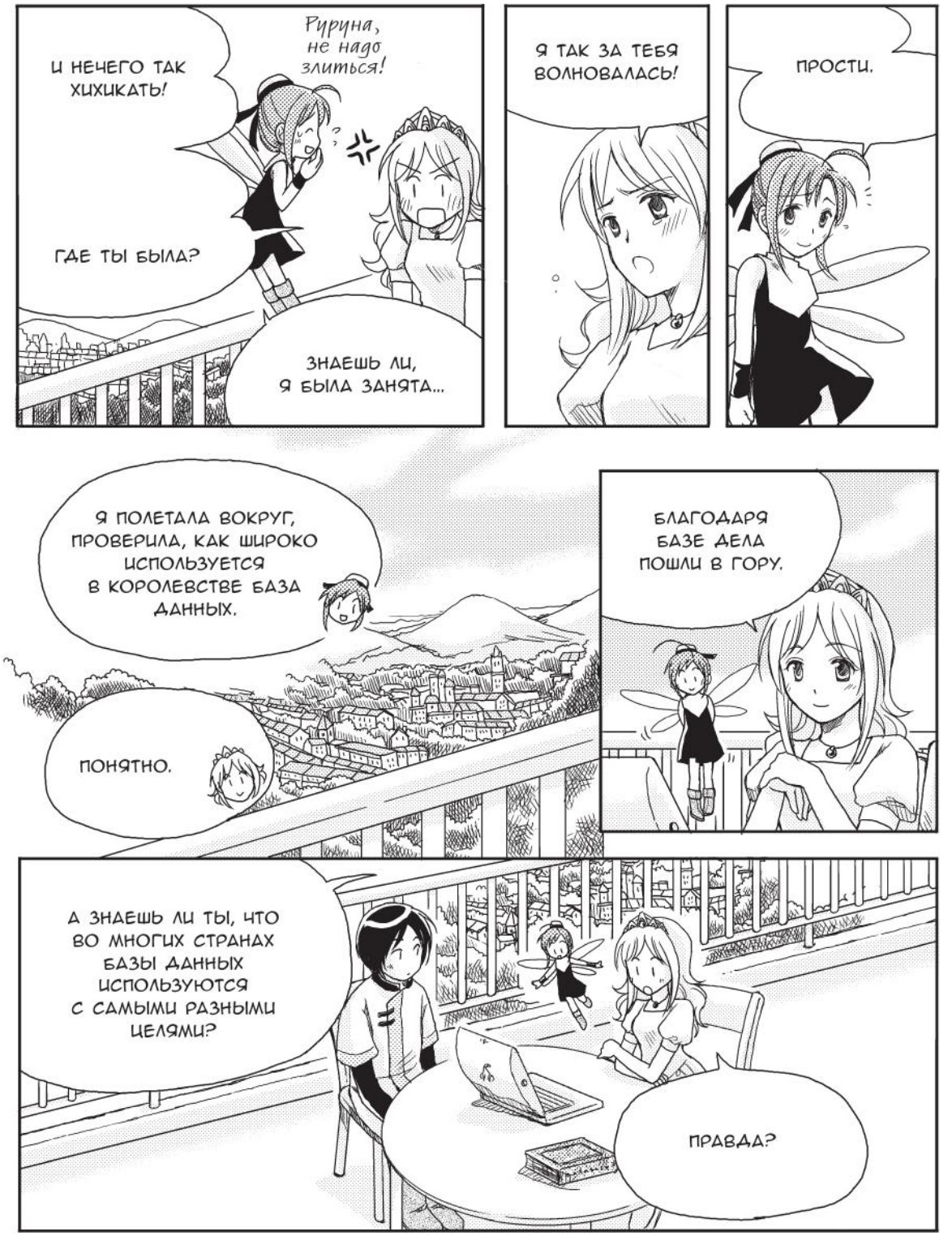
ОНА ОЧЕНЬ ПОМОГЛА НАМ С КЕЙНОМ, ПОКА ТЫ БЫЛ В ОТЪЕЗДЕ.
ЛЯ-ЛЯ-ЛЯ


А Я ТО ДУМАЛА, ОТЕЦ ЗНАЕТ ЕЁ, ВЕДЬ ОНА ВЫЛЕТЕЛА ИЗ ЕГО КНИГИ.
ВЫХОДИТ, ЧТО, КРОМЕ МЕНЯ И КЕЙНА, ТИКО НИКТО НЕ ВИДИТ.
Ты её видела?
Нет.

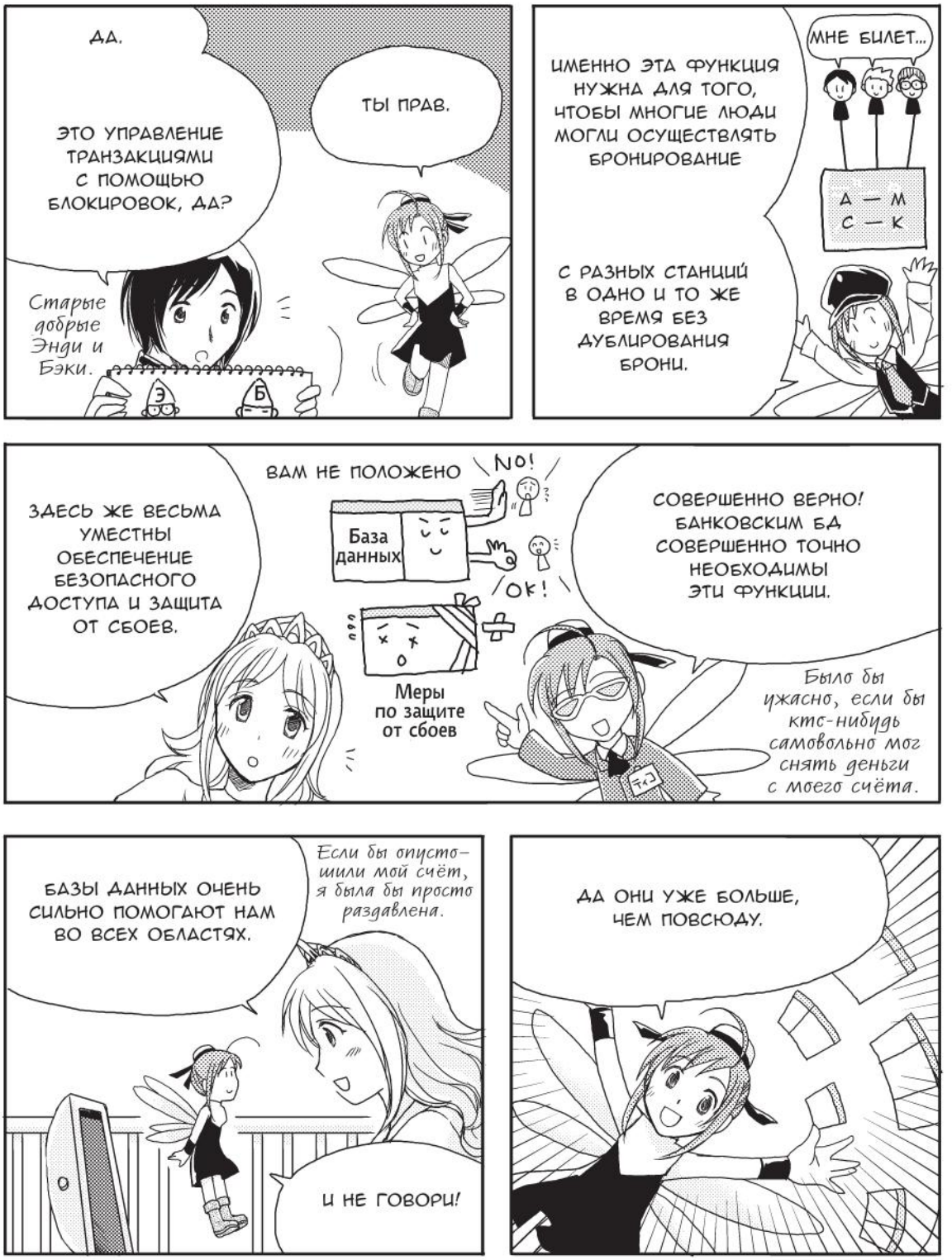
ПРИНЦЕССА РУРУНА!

ТИКО ИСЧЕЗЛА!
И, ПОХОЖЕ, В КНИГЕ ЕЁ НЕТ.
Я ПРОВЕРЮ, НЕТ ЛИ ЕЁ ГДЕ-НИБУДЬ РЯДОМ С ЗАМКОМ!
КУДА ЖЕ ОНА ДЕЛАСЬ?
О, ВАШЕ ВЕЛИЧЕСТВО, ПРОСТИТЕ МЕНЯ!
Извините меня!
ЛАДНО, ЛАДНО
ВАШЕ ВЕЛИЧЕСТВО, ЭТИ ДВОЕ...
ЯВНО ЛАДЯТ ДРУГ С ДРУГОМ, ДА?
Ну и дела!
Никакого этикета!
ОНА ИСЧЕЗЛА!
ФЫРК!
Девочка моя!

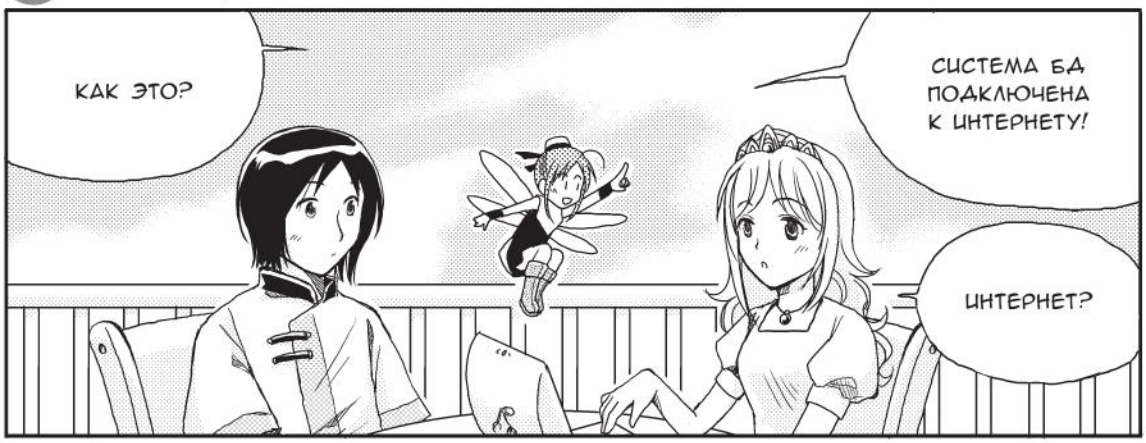
ЕЁ НИГДЕ НЕТ!
ТИКО, КУДА ТЫ ПОДЕВАЛАСЬ НИ С ТОГО НИ С СЕГО?
Я БЫ ХОТЕЛА С НЕЙ...
...ХОТЯ БЫ ПОПРОЩАТЬСЯ.
Я ТОЖЕ.
ИНТЕРЕСНО, КОГО ЭТО ВЫ ПОТЕРЯЛИ?
!!!
ЧТО?
ВОТ И Я!
О!
ТИКО!
БАХ!
ХИ-ХИ-ХИ!


И НЕЧЕГО ТАК ХИХИКАТЬ!
Ририна, не надо злиться!
ГДЕ ТЫ БЫЛА?
ЗНАЕШЬ ЛИ, Я БЫЛА ЗАНЯТА...
Я ТАК ЗА ТЕБЯ ВОЛНОВАЛАСЬ!
ПРОСТИ.
Я ПОЛЕТАЛА ВОКРУГ, ПРОВЕРИЛА, КАК ШИРОКО ИСПОЛЬЗУЕТСЯ В КОРОЛЕВСТВЕ БАЗА ДАННЫХ.
ПОНЯТНО.
БЛАГОДАРЯ БАЗЕ ДЕЛА ПОШЛИ В ГОРУ.
А ЗНАЕШЬ ЛИ ТЫ, ЧТО ВО МНОГИХ СТРАНАХ БАЗЫ ДАННЫХ ИСПОЛЬЗУЮТСЯ С САМЫМИ РАЗНЫМИ ЦЕЛЯМИ?
ПРАВДА?

ПРИМЕНЕНИЕ БАЗ ДАННЫХ


ДА.
ЭТО УПРАВЛЕНИЕ ТРАНЗАКЦИЯМИ С ПОМОЩЬЮ БЛОКИРОВОК, ДА?
ТЫ ПРАВ.
Старые добрые Энди и Бэки.
Э
Б
ИМЕННО ЭТА ФУНКЦИЯ НУЖНА ДЛЯ ТОГО, ЧТОБЫ МНОГИЕ ЛЮДИ МОГЛИ ОСУЩЕСТВЛЯТЬ БРОНИРОВАНИЕ
С РАЗНЫХ СТАНЦИЙ В ОДНО И ТО ЖЕ ВРЕМЯ БЕЗ ДУБЛИРОВАНИЯ БРОНИ.
МНЕ БИЛЕТ...
А — М
С — К
ЗДЕСЬ ЖЕ ВЕСЬМА УМЕСТНЫ ОБЕСПЕЧЕНИЕ БЕЗОПАСНОГО ДОСТУПА И ЗАЩИТА ОТ СБОЕВ.
ВАМ НЕ ПОЛОЖЕНО
NO!
База данных
OK!
Меры по защите от сбоев
СОВЕРШЕННО ВЕРНО! БАНКОВСКИМ БД СОВЕРШЕННО ТОЧНО НЕОБХОДИМЫ ЭТИ ФУНКЦИИ.
Было бы ужасно, если бы кто-нибудь самовольно мог снять деньги с моего счёта.
БАЗЫ ДАННЫХ ОЧЕНЬ СИЛЬНО ПОМОГАЮТ НАМ ВО ВСЕХ ОБЛАСТЯХ.
Если бы опустошили мой счёт, я была бы просто раздавлена.
И НЕ ГОВОРИ!
ДА ОНИ УЖЕ БОЛЬШЕ, ЧЕМ ПОВСЮДУ.

БАЗЫ ДАННЫХ И "ВСЕМИРНАЯ ПАУТИНА"

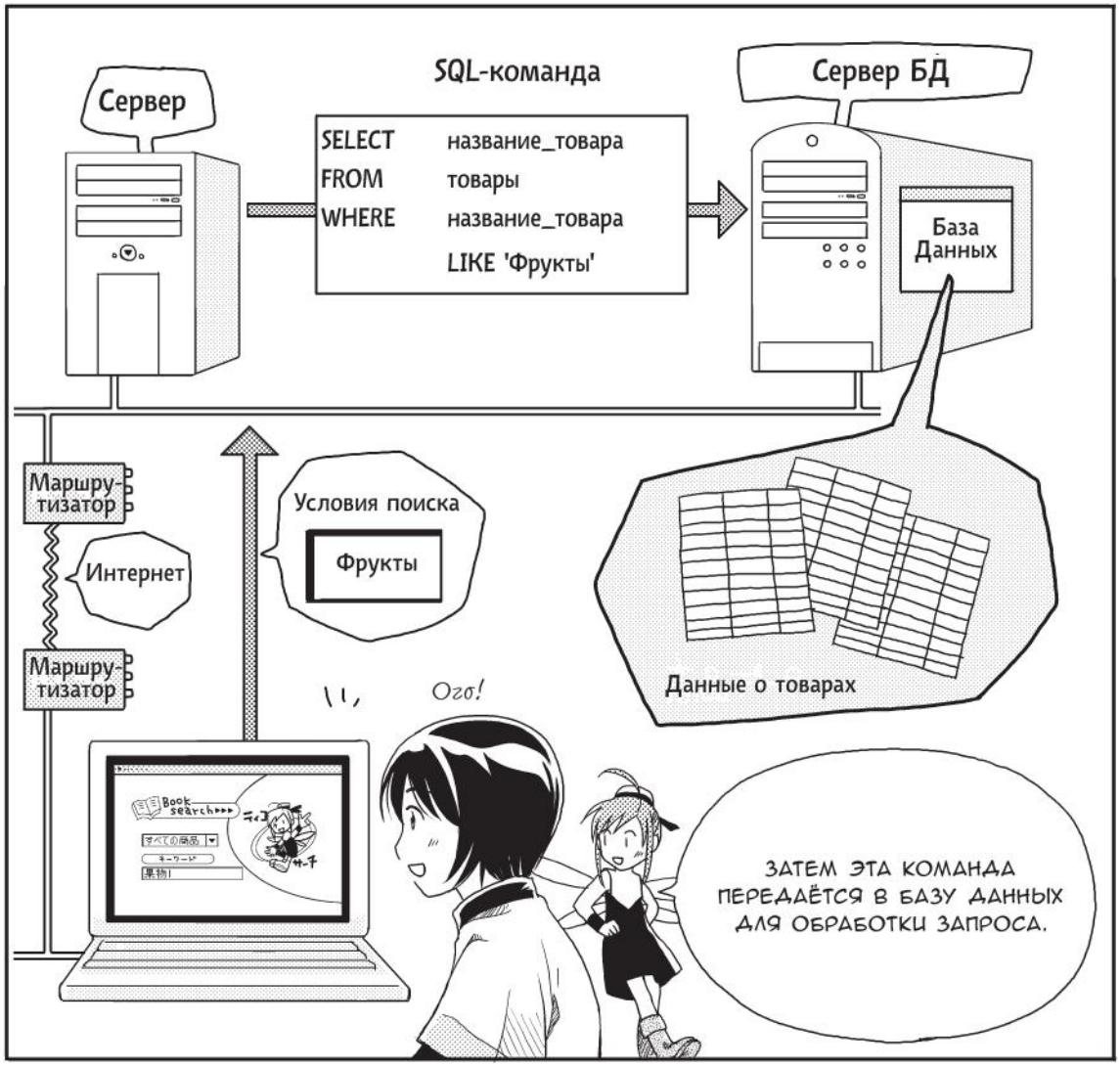


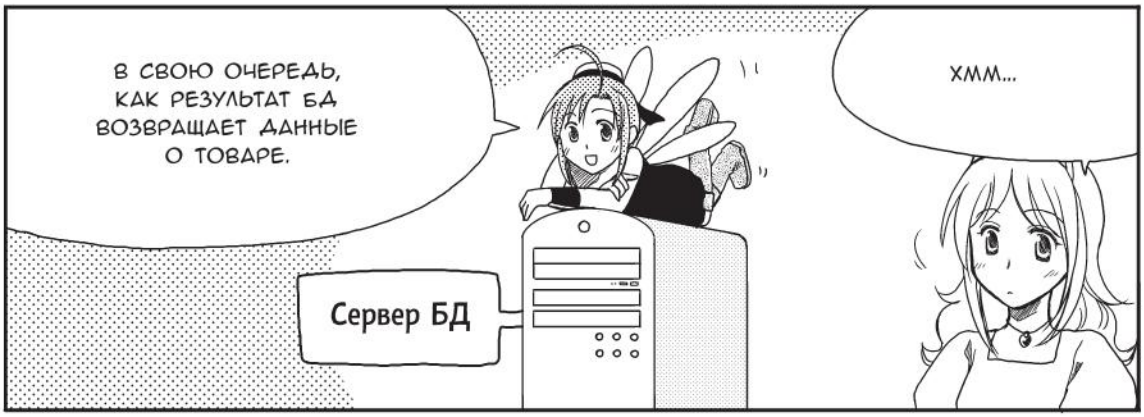
ПРИ ПОИСКЕ КОНКРЕТНОЙ КНИГИ...
ГОТОВО!
...ВВЕДИТЕ КЛЮЧЕВОЕ СЛОВО В СТРОКЕ СИСТЕМЫ ПОИСКА.
Поиск книги
Категории
Ключевое слово
Поиск Тико
Поиск Тико?

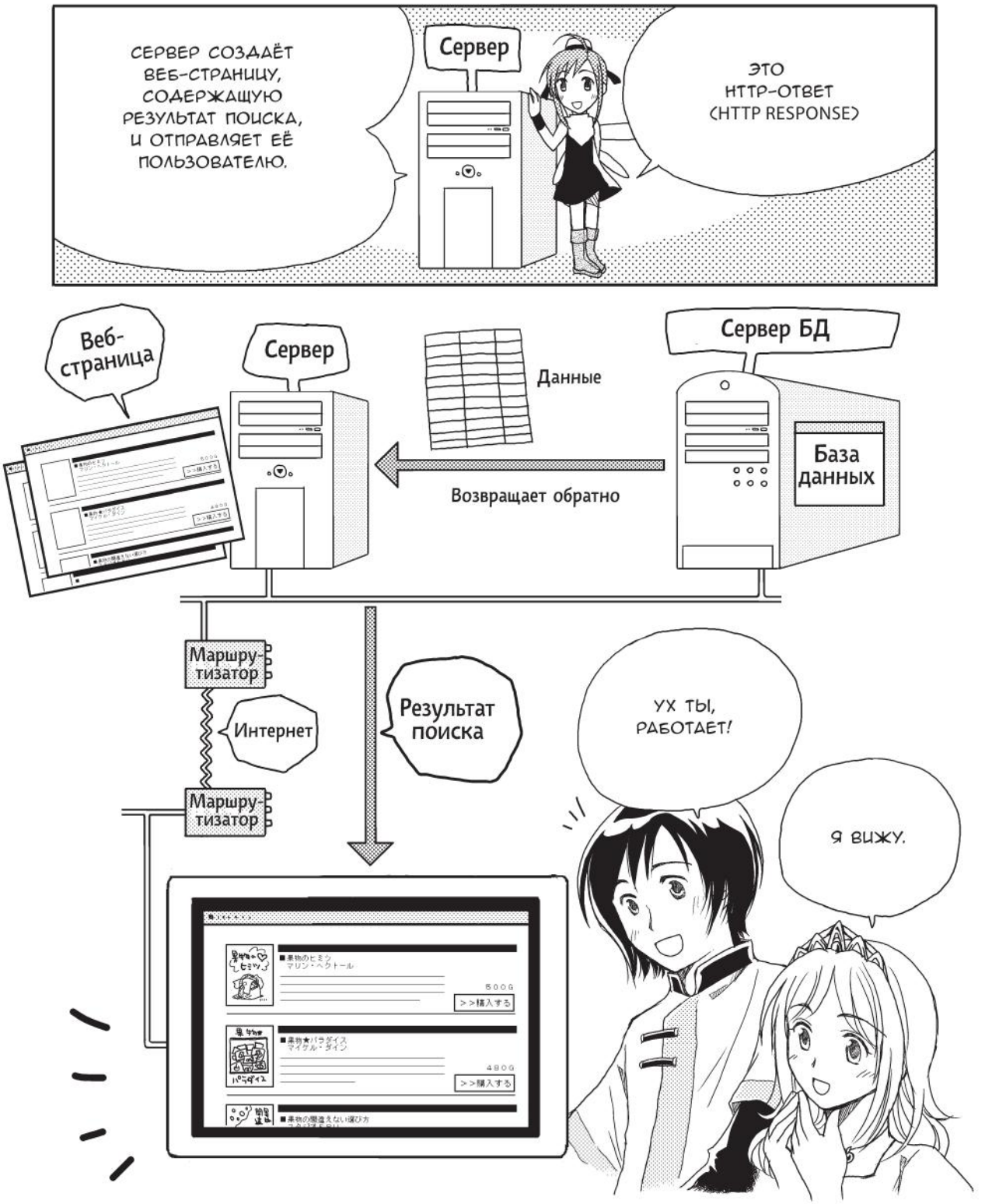
В КАКОЙ КАТЕГОРИИ ТЫ ИЩЕШЬ КНИГУ?
НУ, СКАЖЕМ.... ПУСТЬ БУДУТ "ФРУКТЫ".

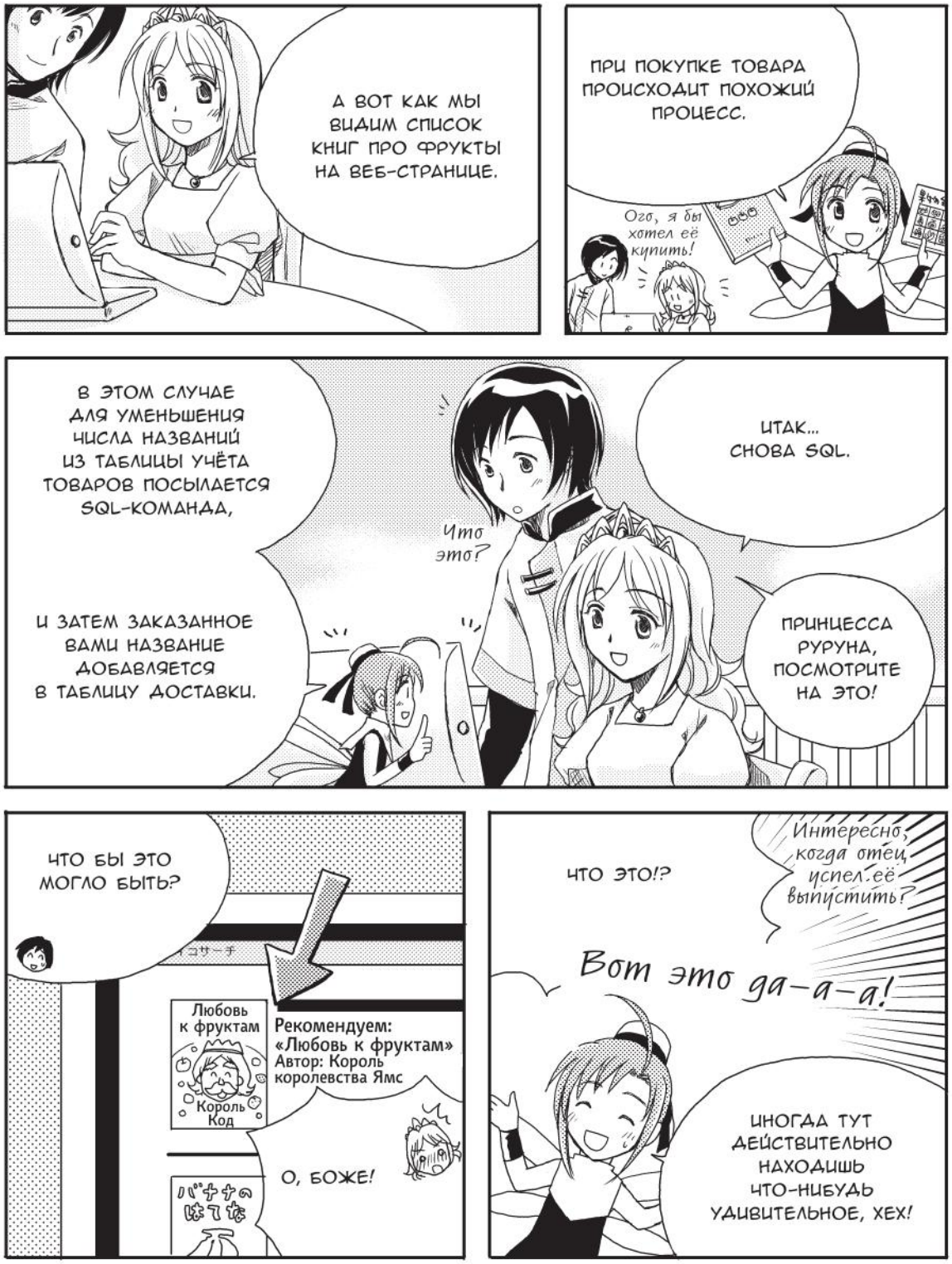
ВВЕДИ СЛОВО "ФРУКТЫ" В ПОЛЕ ДЛЯ КЛЮЧЕВОГО СЛОВА.
Ключевое слово
Фрукты

ЭТО КЛЮЧЕВОЕ СЛОВО БУДЕТ ОТПРАВЛЕНО КАК HTTP-ЗАПРОС (HTTP REQUEST).
КОМПЬЮТЕР, КОТОРЫЙ ПОЛУЧАЕТ ЗАПРОС И ОБРАБАТЫВАЕТ ЕГО, НАЗЫВАЕТСЯ СЕРВЕР (SERVER).
Поиск книги
Категории
Ключевое слово
Фрукты
Я ПОЛАГАЮ, НА СЕРВЕРЕ ФОРМИРУЕТСЯ SQL-КОМАНДА.


Сервер
SQL-команда
SELECT название_товара
FROM товары
WHERE название_товара
LIKE 'Фрукты'
Сервер БД
База Данных
Маршру-тизатор
Интернет
Маршру-тизатор
Условия поиска
Фрукты
Данные о товарах
Ого!
Book search
ЗАТЕМ ЭТА КОМАНДА ПЕРЕДАЁТСЯ В БАЗУ ДАННЫХ ДЛЯ ОБРАБОТКИ ЗАПРОСА.

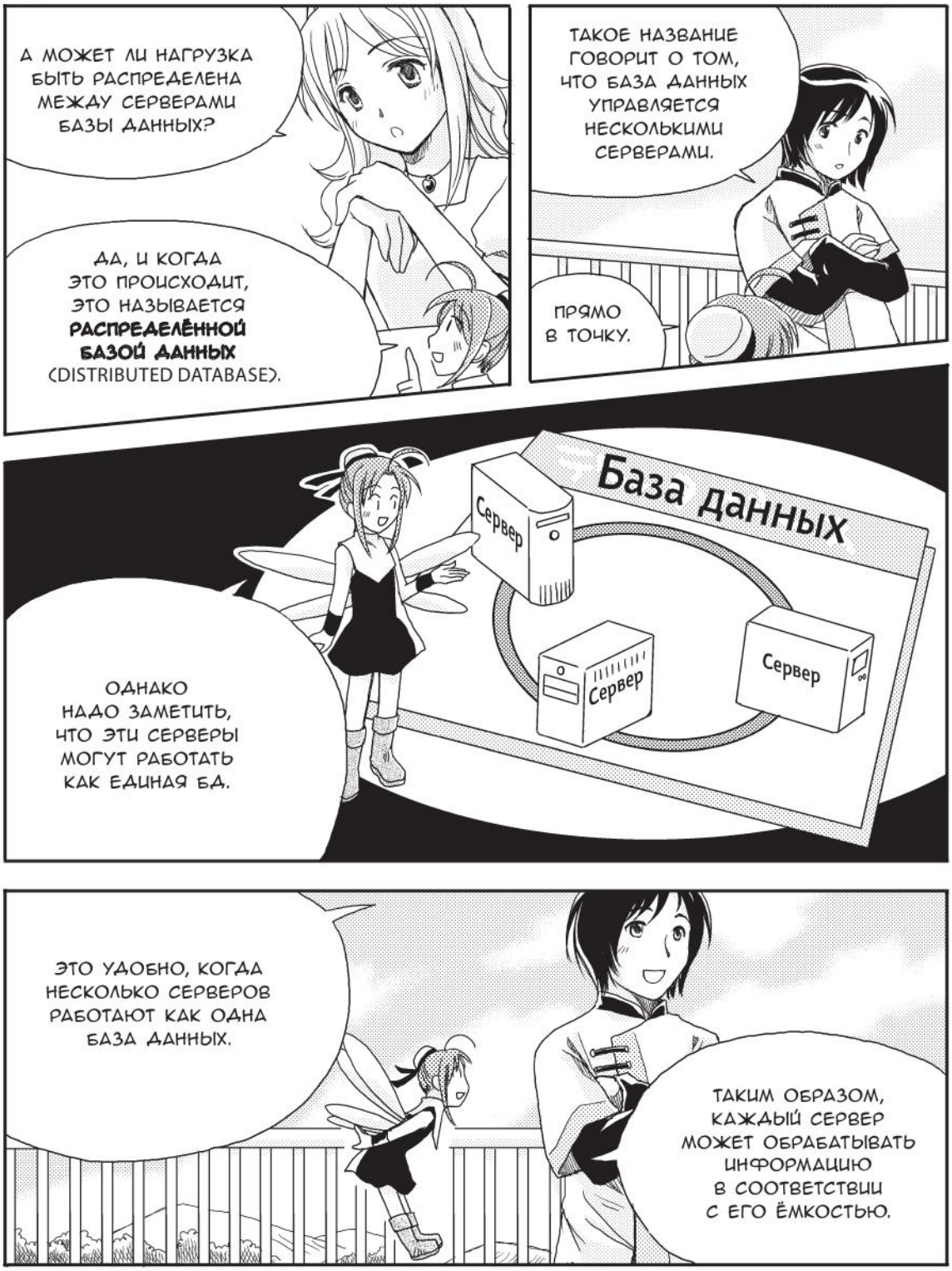

В СВОЮ ОЧЕРЕДЬ, КАК РЕЗУЛЬТАТ БД ВОЗВРАЩАЕТ ДАННЫЕ О ТОВАРЕ.
Сервер БД
ХММ...


Сервер создаёт веб-страницу, содержащую результат поиска, и отправляет её пользователю.
Сервер
Это HTTP-ответ (HTTP RESPONSE)
Веб-страница
Сервер
Данные
Сервер БД
База данных
Возвращает обратно
Маршрутизатор
Интернет
Маршрутизатор
Результат поиска
Ух ты, работает!
Я вижу.


А ВОТ КАК МЫ ВИДИМ СПИСОК КНИГ ПРО ФРУКТЫ НА ВЕБ-СТРАНИЦЕ.
ПРИ ПОКУПКЕ ТОВАРА ПРОИСХОДИТ ПОХОЖИЙ ПРОЦЕСС.
Ого, я бы хотел её купить!
В ЭТОМ СЛУЧАЕ ДЛЯ УМЕНЬШЕНИЯ ЧИСЛА НАЗВАНИЙ ИЗ ТАБЛИЦЫ УЧЁТА ТОВАРОВ ПОСЫЛАЕТСЯ SQL-КОМАНДА,
И ЗАТЕМ ЗАКАЗАННОЕ ВАМИ НАЗВАНИЕ ДОБАВЛЯЕТСЯ В ТАБЛИЦУ ДОСТАВКИ.
Что это?
ИТАК... СНОВА SQL.
ПРИНЦЕССА РУРУНА, ПОСМОТРИТЕ НА ЭТО!
ЧТО БЫ ЭТО МОГЛО БЫТЬ?
Любовь к фруктам
Король Код
Рекомендуем: «Любовь к фруктам» Автор: Король королевства Ямс
О, БОЖЕ!
ЧТО ЭТО!?
Интересно, когда отец успел её выпустить?
Вот это да-а-а!
ИНОГДА ТУТ ДЕЙСТВИТЕЛЬНО НАХОДИШЬ ЧТО-НИБУДЬ УДИВИТЕЛЬНОЕ, ХЕХ!

МОГУ СЕБЕ ПРЕДСТАВИТЬ, КАК МНОГО ПОЛЬЗОВАТЕЛЕЙ ОДНОВРЕМЕННО ИМЕЮТ ДОСТУП К ИНТЕРНЕТ-МАГАЗИНУ.
ДАЖЕ ЕСЛИ БЛОКИРОВКИ И ФУНКЦИИ ОБЕСПЕЧЕНИЯ БЕЗОПАСНОСТИ СОЗДАЮТ УСЛОВИЯ ДЛЯ ПОЛНОЙ ЗАЩИТЫ БД,
Книжный интернет-магазин
ПРИХОДИТСЯ ОБРАБАТЫВАТЬ ОЧЕНЬ МНОГО ЗАПРОСОВ.
В ЭТОМ СЛУЧАЕ ВСЯ ТЯЖЕСТЬ ОБРАБОТКИ РАСПРЕДЕЛЯЕТСЯ МЕЖДУ НЕСКОЛЬКИМИ СЕРВЕРАМИ.
ХОЧЕШЬ СКАЗАТЬ, ЧТО ЗАДЕЙСТВОВАН БОЛЬШЕ, ЧЕМ ОДИН СЕРВЕР?
ДА, НАГРУЗКА РАСПРЕДЕЛЕНА МЕЖДУ РАЗНЫМИ СЕРВЕРАМИ, ТО ЕСТЬ ВЕБ-СЕРВЕРОМ И СЕРВЕРОМ ПРИЛОЖЕНИЙ (APPLICATION SERVER).
Веб-сервер
Сервер приложений
ВЕБ-СЕРВЕР — ЭТО СЕРВЕР, КОТОРЫЙ ОТВЕЧАЕТ ЗА СОЗДАНИЕ ВЕБ-СТРАНИЦЫ, ТАК?
Веб-сервер
Веб-страница
"СЕРВЕР ПРИЛОЖЕНИЙ" ОТВЕЧАЕТ ЗА ОБРАБОТКУ SQL-КОМАНД.
Сервер приложений
SELECT
FROM
WHERE

РАСПРЕДЕЛЁННЫЕ БАЗЫ ДАННЫХ



Многие серверы также обеспечивают дополнительную защиту от сбоев!
Скрип!
Скрип!
Трах!
Ой, больно!
Knock Out!!
Ти-ко! Ти-ко!
Это значит, что даже если на каких-то серверах системы возникнет сбой, вся система не выйдет из строя.
Отсчёт пошёл
Сбой
О, Боже!
Но не надо забывать: чтобы работать с БД таким образом, надо кое о чём позаботиться.
Например?
Когда совершается транзакция, вы должны быть уверены, что ваша распределённая база согласована.
Также, к примеру, все серверы должны должным образом обновляться, на случай если в сети что-то произойдёт.

ХРАНИМЫЕ ПРОЦЕДУРЫ И ТРИГГЕРЫ
В ЛЮБОЙ СРЕДЕ, ГДЕ ИСПОЛЬЗУЕТСЯ НЕСКОЛЬКО СЕРВЕРОВ, СЕТЬ — ЭТО ОБЯЗАТЕЛЬНОЕ УСЛОВИЕ.
ВЕРНО! ВОТ ГДЕ ПРИГОДЯТСЯ ХРАНИМЫЕ ПРОЦЕДУРЫ (STORED PROCEDURES);
ИНОГДА ИХ СОЗДАЮТ, ЧТОБЫ УМЕНЬШИТЬ НАГРУЗКУ НА СЕТЬ.

ХРАНИМЫЕ?..
АГА!

ДРУГИМИ СЛОВАМИ, "ХРАНИТЬ" — ЭТО "ДЕРЖАТЬ В ПАМЯТИ"?
ТОЧНО!
ДЛЯ ТОГО ЧТОБЫ УМЕНЬШИТЬ НАГРУЗКУ НА СЕТЬ, ЧАСТО ИСПОЛЬЗУЕМЫЕ ОПЕРАЦИИ МОЖНО ХРАНИТЬ В БАЗЕ ДАННЫХ.

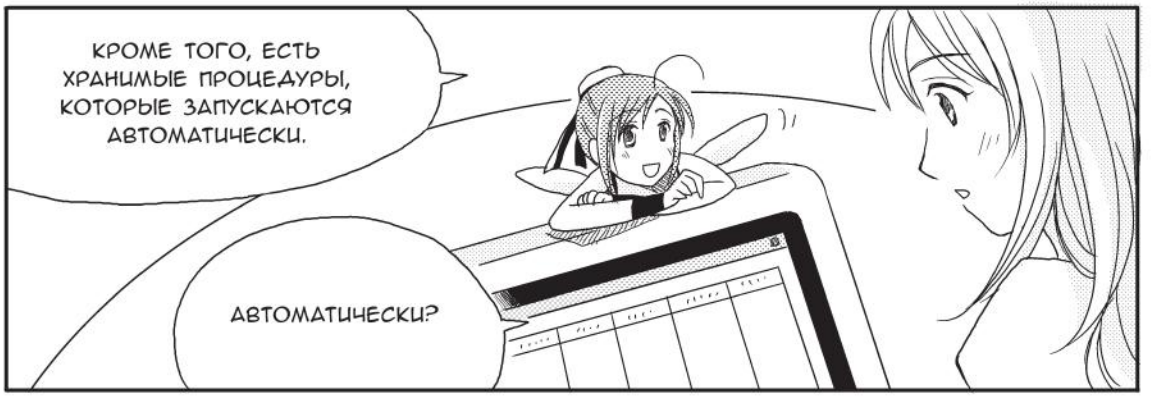
ТЫ ГОВОРИШЬ "ЧАСТО ИСПОЛЬЗУЕМЫЕ ОПЕРАЦИИ"...
ИНТЕРЕСНО, А ЧТО ЭТО ЗА ОПЕРАЦИИ?

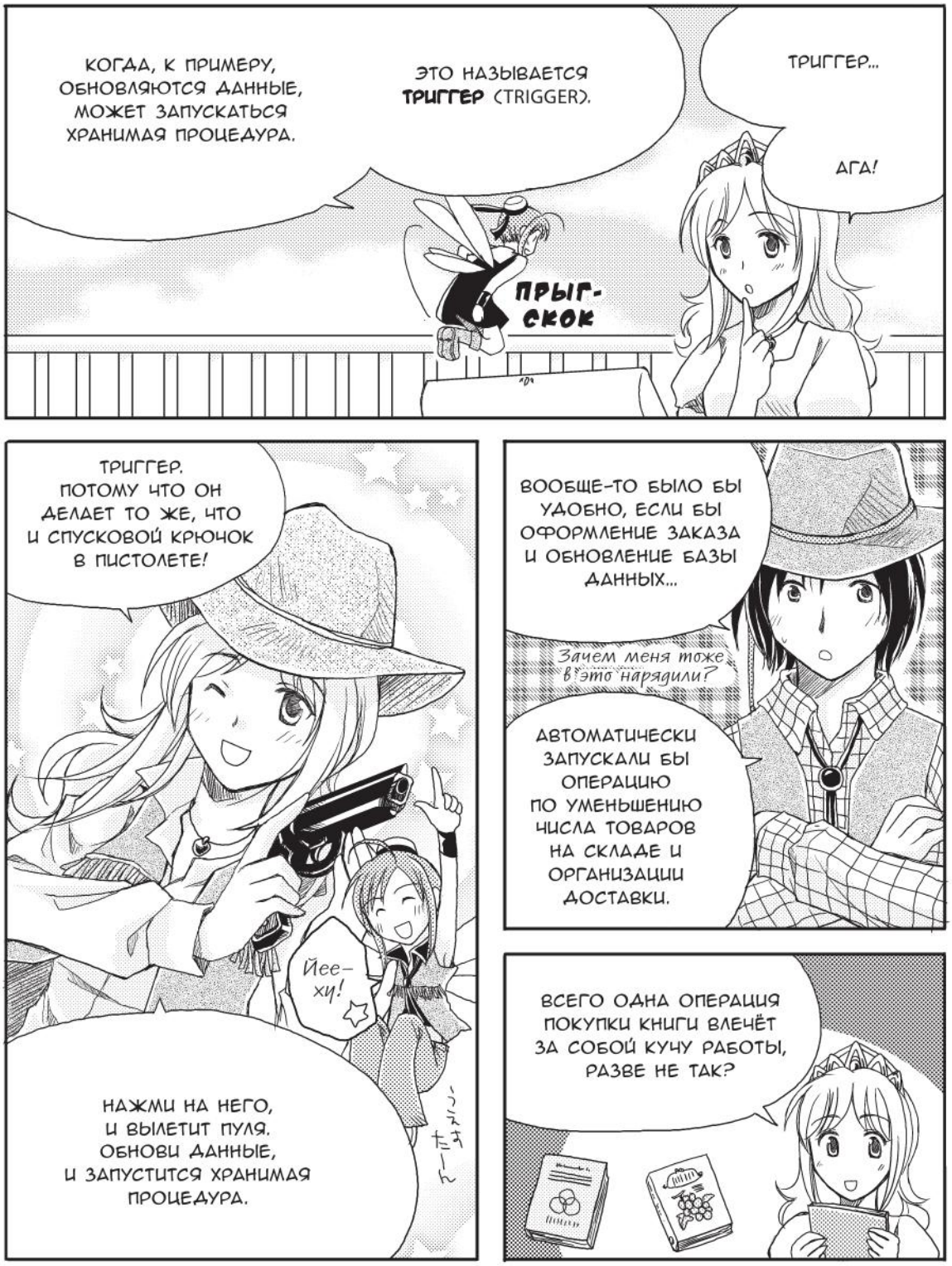
КОГДА МЫ ГОВОРИЛИ ОБ ОПЕРАЦИЯХ ПО ПОКУПКЕ КНИГИ,
ТО РАЗВЕ ОПЕРАЦИИ ИЗВЛЕЧЕНИЯ ИЗ СЧЁТЧИКА "ТОВАРЫ В НАЛИЧИИ" В ТАБЛИЦЕ УЧЁТА ТОВАРА И ДОБАВЛЕНИЯ ДАННЫХ В ТАБЛИЦУ ДОСТАВКИ...
НЕ БУДУТ В ДАННОМ СЛУЧАЕ ТИПИЧНЫМИ?

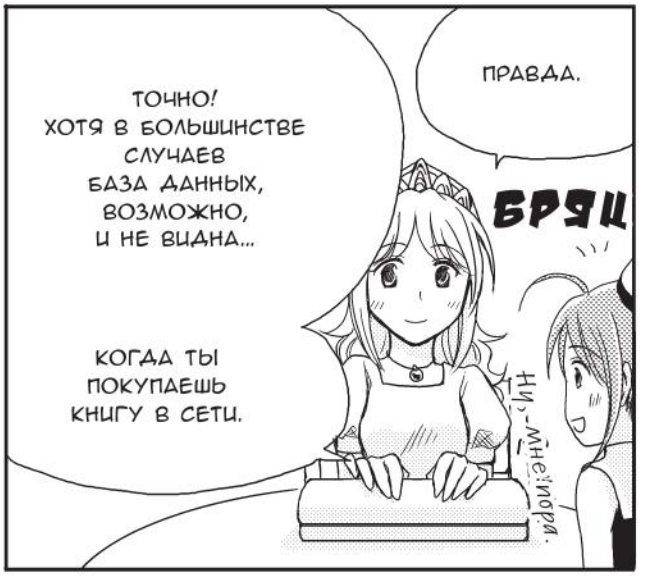
ДА, И ПРАВДА.
ОПЕРАЦИИ, КОТОРЫЕ БУДУТ ИСПОЛЬЗОВАТЬСЯ ЧАСТО, МЫ МОЖЕМ ХРАНИТЬ В ВИДЕ ПРОЦЕДУР.
В БАЗЕ! ЗАРАНЕЕ!

ЕСЛИ МЫ ПРИГОТОВИМ ХРАНИМУЮ ПРОЦЕДУРУ, НАМ НЕ ПРИДЁТСЯ СОЗДАВАТЬ SQL-КОМАНДУ...
КАЖДЫЙ РАЗ, КАК МЫ ЗАХОТИМ УМЕНЬШИТЬ КОЛИЧЕСТВО ТОВАРА НА СКЛАДЕ И ОБРАБОТАТЬ ОПЕРАЦИЮ ДОСТАВКИ.
Это уже лучше
ТАКИМ ОБРАЗОМ, РАБОЧАЯ НАГРУЗКА НА СЕТЬ УМЕНЬШИТСЯ.

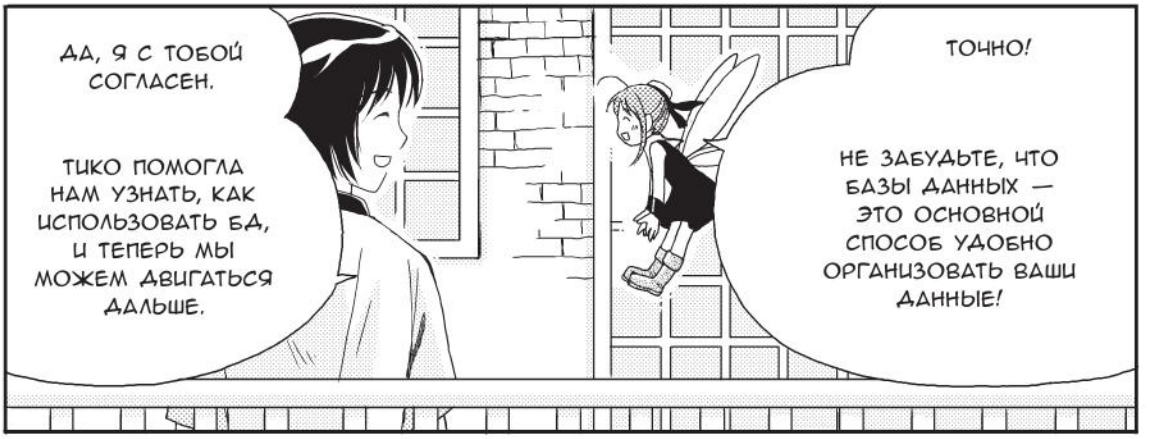
И НАША РАБОТА ТОЖЕ!
О, ДА! ТЫ ПРАВА!


КРОМЕ ТОГО, ЕСТЬ ХРАНИМЫЕ ПРОЦЕДУРЫ, КОТОРЫЕ ЗАПУСКАЮТСЯ АВТОМАТИЧЕСКИ.
АВТОМАТИЧЕСКИ?


КОГДА, К ПРИМЕРУ, ОБНОВЛЯЮТСЯ ДАННЫЕ, МОЖЕТ ЗАПУСКАТЬСЯ ХРАНИМАЯ ПРОЦЕДУРА.
ЭТО НАЗЫВАЕТСЯ ТРИГГЕР (TRIGGER).
ТРИГГЕР...
АГА!
ПРЫГ-СКОК
ТРИГГЕР. ПОТОМУ ЧТО ОН ДЕЛАЕТ ТО ЖЕ, ЧТО И СПУСКОВОЙ КРЮЧОК В ПИСТОЛЕТЕ!
Йее-ху!
НАЖМИ НА НЕГО, И ВЫЛЕТИТ ПУЛЯ. ОБНОВИ ДАННЫЕ, И ЗАПУСТИТСЯ ХРАНИМАЯ ПРОЦЕДУРА.
ВООБЩЕ-ТО БЫЛО БЫ УДОБНО, ЕСЛИ БЫ ОФОРМЛЕНИЕ ЗАКАЗА И ОБНОВЛЕНИЕ БАЗЫ ДАННЫХ...
Зачем меня тоже в это нарядили?
АВТОМАТИЧЕСКИ ЗАПУСКАЛИ БЫ ОПЕРАЦИЮ ПО УМЕНЬШЕНИЮ ЧИСЛА ТОВАРОВ НА СКЛАДЕ И ОРГАНИЗАЦИИ ДОСТАВКИ.
ВСЕГО ОДНА ОПЕРАЦИЯ ПОКУПКИ КНИГИ ВЛЕЧЁТ ЗА СОБОЙ КУЧУ РАБОТЫ, РАЗВЕ НЕ ТАК?


ПРАВДА.
ТОЧНО! ХОТЯ В БОЛЬШИНСТВЕ СЛУЧАЕВ БАЗА ДАННЫХ, ВОЗМОЖНО, И НЕ ВИДНА...
БРЯЦ
КОГДА ТЫ ПОКУПАЕШЬ КНИГУ В СЕТИ.
Ну, мне пора.

НО МЫ ШАГ ЗА ШАГОМ ДОЛЖНЫ БОЛЬШЕ УЗНАТЬ ОБ ЭТОМ.
КЕЙН, ТЫ ТАК НЕ ДУМАЕШЬ?

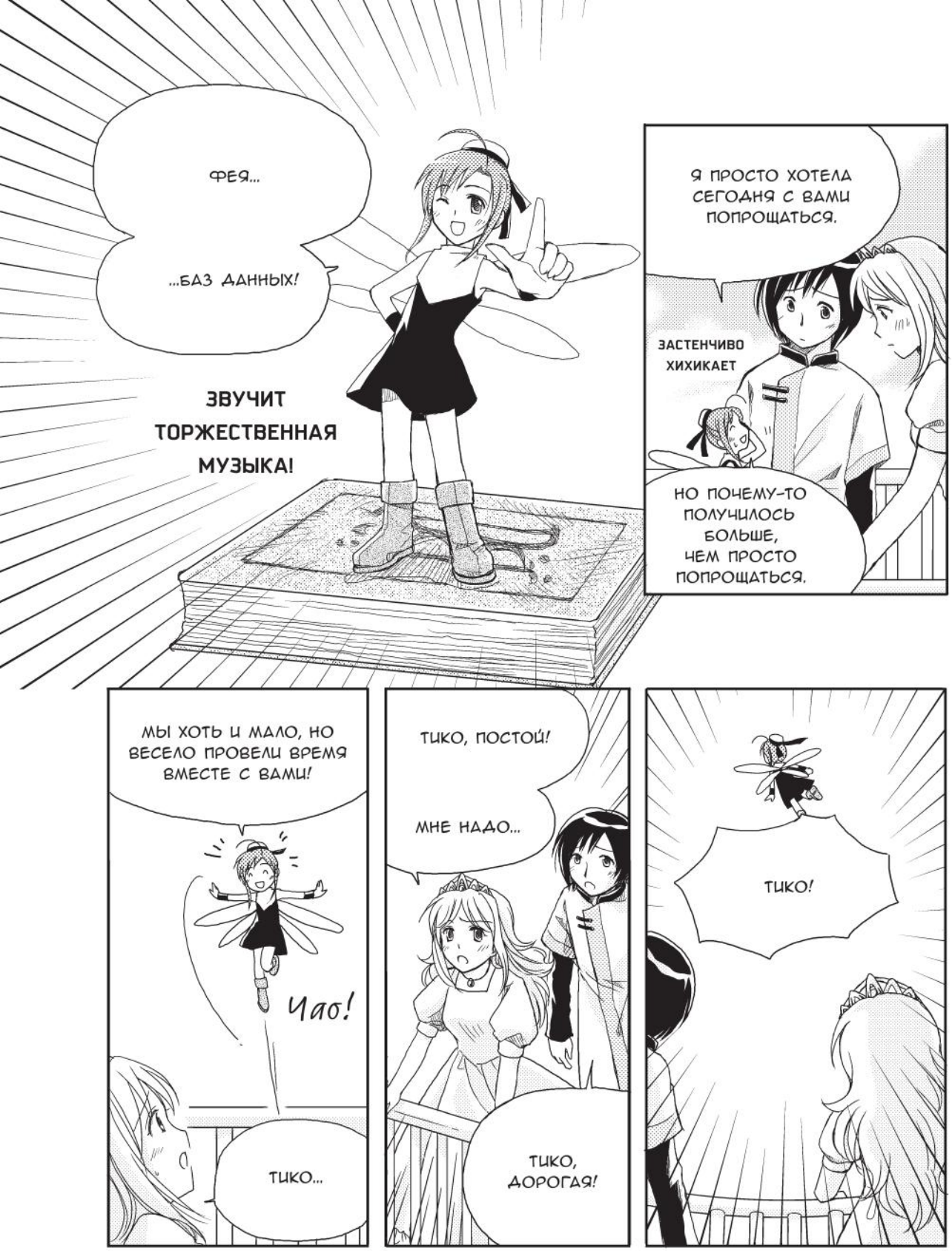

ДА, Я С ТОБОЙ СОГЛАСЕН.
ТИКО ПОМОГЛА НАМ УЗНАТЬ, КАК ИСПОЛЬЗОВАТЬ БД, И ТЕПЕРЬ МЫ МОЖЕМ ДВИГАТЬСЯ ДАЛЬШЕ.
ТОЧНО!
НЕ ЗАБУДЬТЕ, ЧТО БАЗЫ ДАННЫХ — ЭТО ОСНОВНОЙ СПОСОБ УДОБНО ОРГАНИЗОВАТЬ ВАШИ ДАННЫЕ!

ОЧЕНЬ ИНТЕРЕСНО! СМОТРИ, С ПОМОЩЬЮ БАЗЫ ДАННЫХ Я...
...ПОСТРОЮ ПРЕКРАСНУЮ СТРАНУ, ГДЕ КАЖДЫЙ СМОЖЕТ НАСЛАЖДАТЬСЯ УДОБНОЙ И КОМФОРТНОЙ ЖИЗНЬЮ.

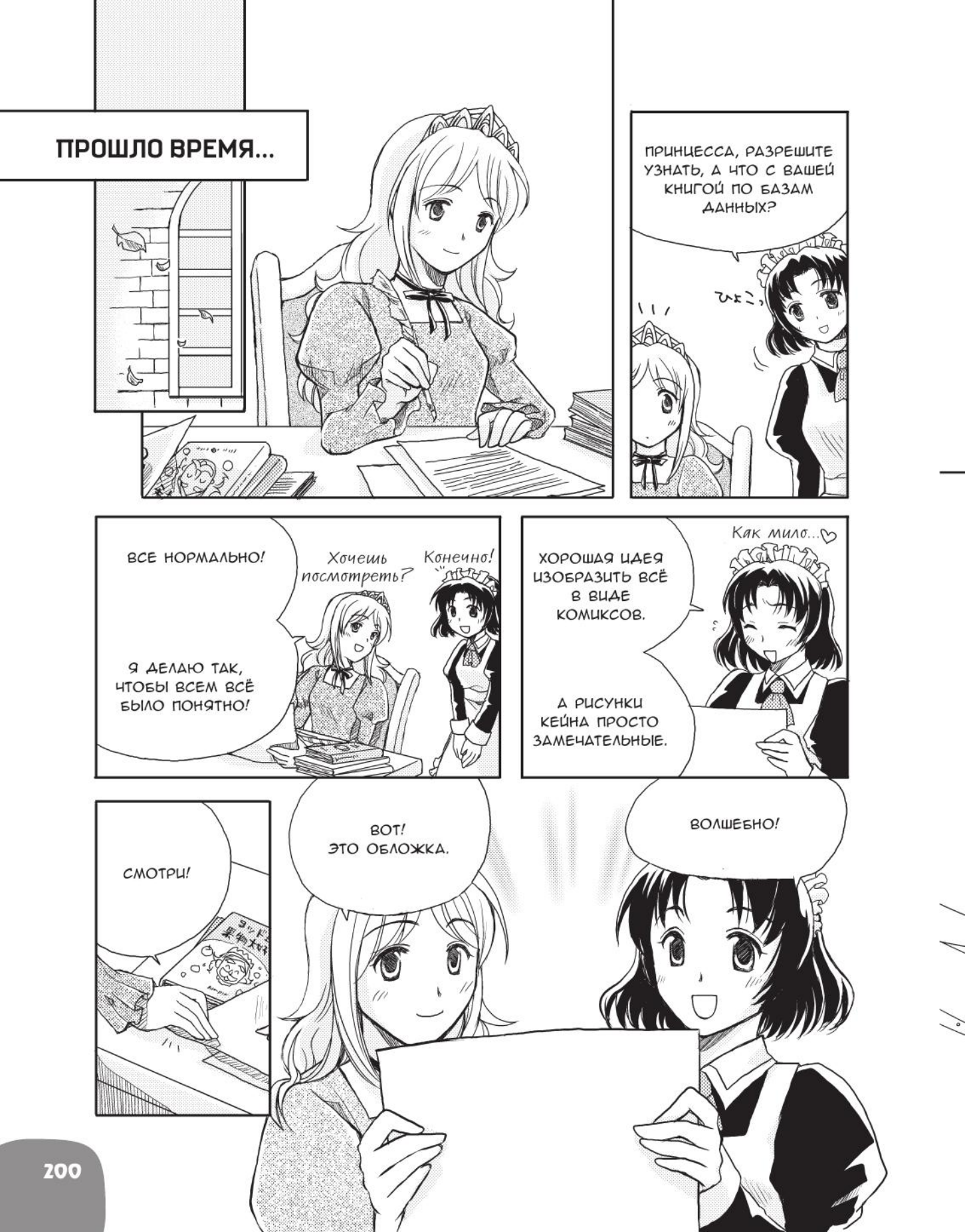
МЫ БУДЕМ СТАРАТЬСЯ ИЗО ВСЕХ СИЛ!
Вау!

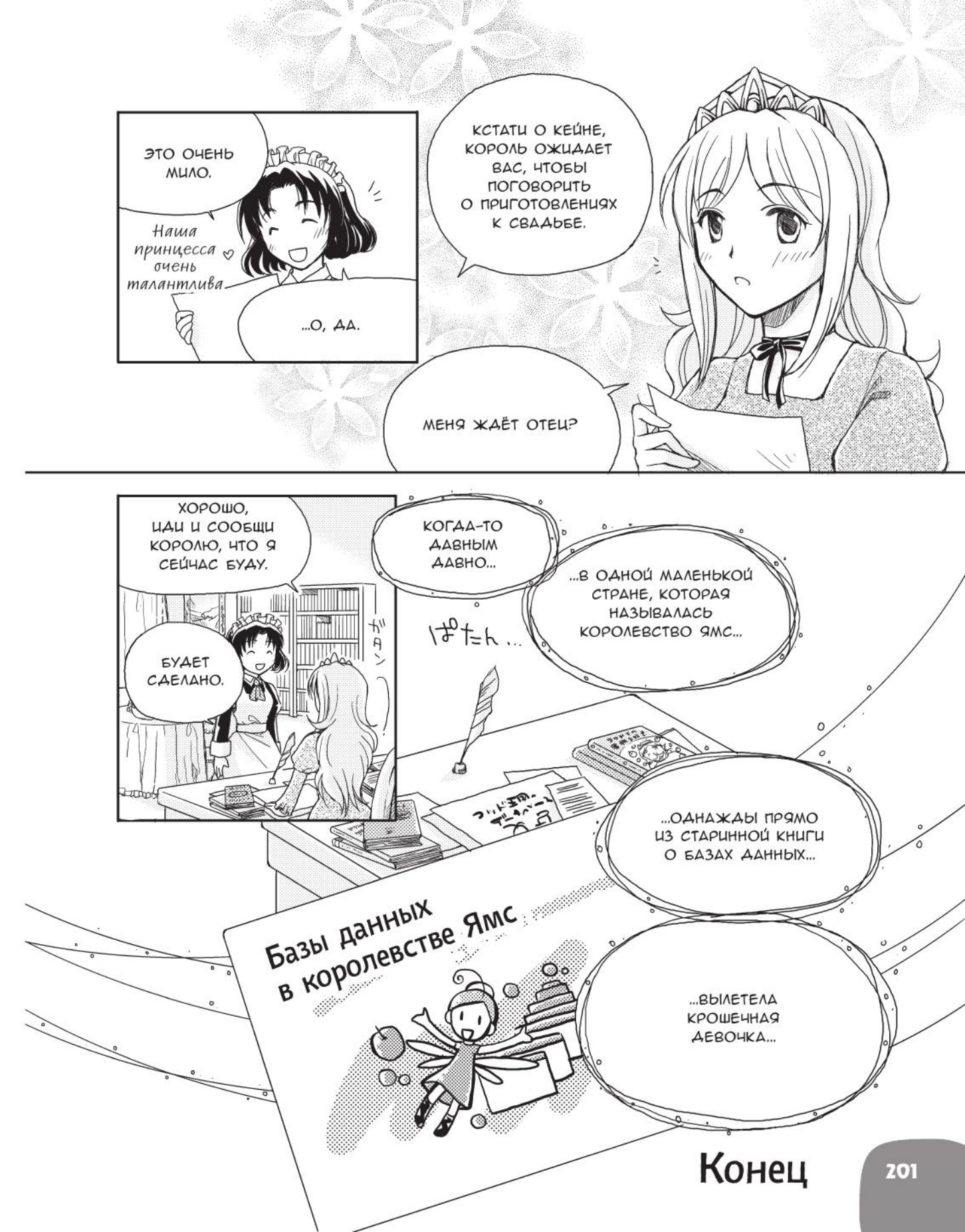
АГА!

ПОНИМАЕТЕ, Я ЖЕ...


ФЕЯ...
...БАЗ ДАННЫХ!
ЗВУЧИТ ТОРЖЕСТВЕННАЯ МУЗЫКА!
Я ПРОСТО ХОТЕЛА СЕГОДНЯ С ВАМИ ПОПРОЩАТЬСЯ.
ЗАСТЕНЧИВО ХИХИКАЕТ
НО ПОЧЕМУ-ТО ПОЛУЧИЛОСЬ БОЛЬШЕ, ЧЕМ ПРОСТО ПОПРОЩАТЬСЯ.
МЫ ХОТЬ И МАЛО, НО ВЕСЕЛО ПРОВЕЛИ ВРЕМЯ ВМЕСТЕ С ВАМИ!
Чао!
ТИКО...
ТИКО, ПОСТОЙ!
МНЕ НАДО...
ТИКО, ДОРОГАЯ!
ТИКО!

СПАСИБО ТЕБЕ!
ОНА ИСЧЕЗЛА.
МНЕ ТАК БОЛЬНО ВИДЕТЬ, КАК ВЫ ОГОРЧЕНЫ, ПРИНЦЕССА.
ТЕПЕРЬ НАША ЗАДАЧА – ВНЕДРИТЬ ЗНАНИЯ, КОТОРЫЕ ДАЛА НАМ ТИКО...
...В РЕАЛЬНУЮ СИСТЕМУ.
О ДА, ТЫ ПРАВ!


ПРОШЛО ВРЕМЯ...
ПРИНЦЕССА, РАЗРЕШИТЕ УЗНАТЬ, А ЧТО С ВАШЕЙ КНИГОЙ ПО БАЗАМ ДАННЫХ?
ひょこっ
ВСЕ НОРМАЛЬНО!
Хочешь посмотреть?
Конечно!
Я ДЕЛАЮ ТАК, ЧТОБЫ ВСЕМ ВСЁ БЫЛО ПОНЯТНО!
ХОРОШАЯ ИДЕЯ ИЗОБРАЗИТЬ ВСЁ В ВИДЕ КОМИКСОВ.
Как мило...
А РИСУНКИ КЕЙНА ПРОСТО ЗАМЕЧАТЕЛЬНЫЕ.
СМОТРИ!
ВОТ! ЭТО ОБЛОЖКА.
ВОЛШЕБНО!

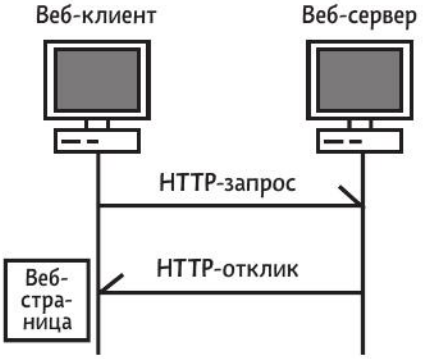

Это очень мило.
Наша принцесса очень талантлива
...О, да.
Кстати о Кейне, король ожидает вас, чтобы поговорить о приготовлениях к свадьбе.
Меня ждёт отец?
Хорошо, иди и сообщи королю, что я сейчас буду.
Будет сделано.
ガタン
Когда-то давным давно...
ぱたん...
...в одной маленькой стране, которая называлась королевство Ямс...
...однажды прямо из старинной книги о базах данных...
Базы данных в королевстве Ямс
...вылетела крошечная девочка...

Конец

БАЗЫ ДАННЫХ В ИНТЕРНЕТЕ

Базы данных используют в различных целях, например для бронирования железнодорожных билетов или в банковских системах. Без них уже невозможно обойтись в нашей повседневной жизни и в среде делового общения. Я уже показала Руруне и Кейну, что системы баз данных для веб-приложений также очень популярны. В веб-приложениях используется протокол передачи данных HTTP (Hyper Text Transfer Protocol — протокол передачи гипертекста). Программное обеспечение, установленное на сервере, ждет запроса от пользователя. Когда делается пользовательский запрос (HTTP request), программа отвечает на него и возвращает соответствующую веб-страницу (HTTP response).

Веб-страница состоит из текстовых файлов в формате HTML. Другие файлы, заданные унифицированным указателем ресурсов (URLs), встроены в веб-страницу для отображения информации, например рисунков.

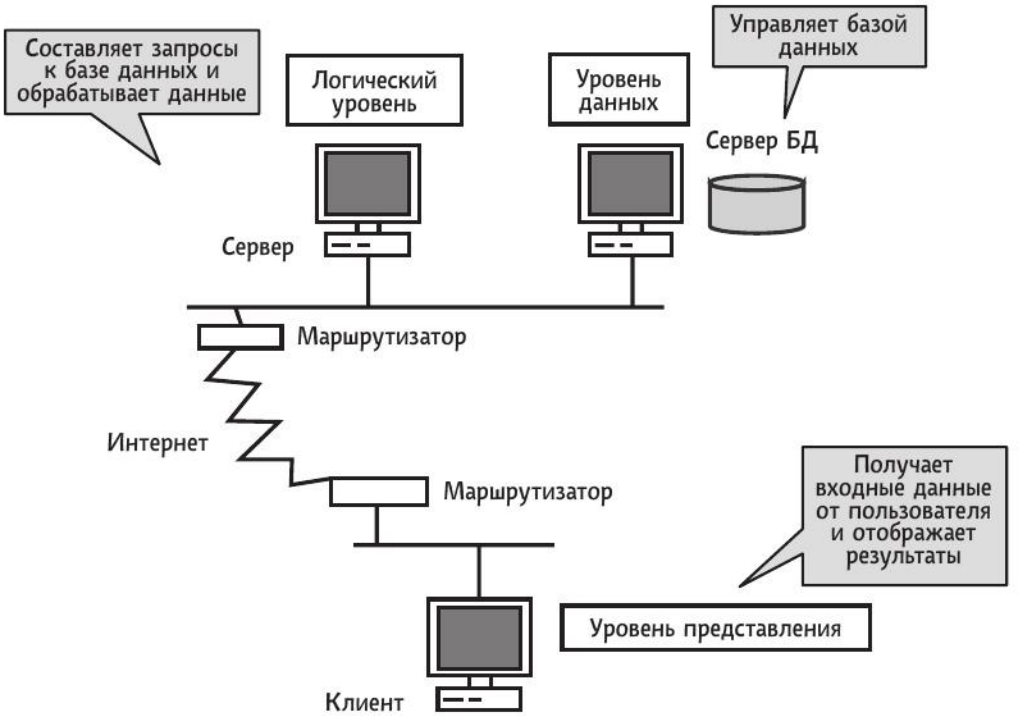


Когда база данных используется с веб-страницей, к системе, которая показана выше, добавляется сервер БД. Эту систему можно сконфигурировать в виде трёх уровней, и тогда она будет называться трёхуровневой системой клиент-сервер (three-tier client/server system). Эта система состоит из уровня представления, логического уровня и уровня данных.

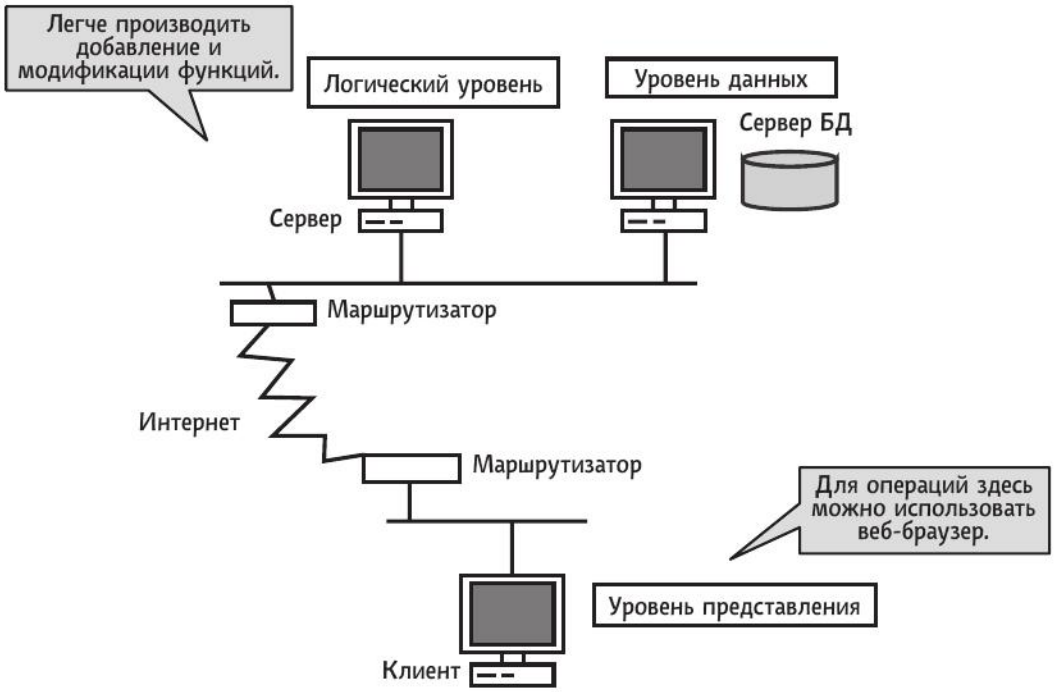
Уровень представления получает входные данные от пользователя, такие как условия поиска, которые нужно передать базе данных. Уровень представления также обрабатывает результаты запроса, полученные от базы, таким образом, чтобы их можно было отобразить. Веб-браузер (например, Internet Explorer или Firefox) выступает в качестве инструмента представления для пользователя.

Логический уровень представляет собой обработку данных. Это уровень, где формируются SQL-команды. Выполняемые здесь процессы написаны на одном или нескольких языках программирования. В зависимости от содержания и нагрузки для успешной обработки могут использоваться несколько серверов, таких как сервер приложений и веб-сервер.

Уровень данных обрабатывает данные на сервере БД. База возвращает результаты поиска в ответ на SQL-запросы.



Трёхуровневая архитектура клиент-сервер — это простая и гибкая модель. Например, когда приложение необходимо дополнить или модифицировать, вы можете отделить необходимую для редактирования часть в качестве логического уровня. На уровне представления можно использовать веб-браузер, исключая тем самым необходимость установки отдельной программы.



ИСПОЛЬЗОВАНИЕ ХРАНИМЫХ ПРОЦЕДУР

В веб-приложении слишком большой трафик в сети может вызвать проблемы. К счастью, вы можете хранить программы в самом сервере БД в виде хранимых процедур.

Хранимые процедуры в сервере БД помогают уменьшить нагрузку на сеть, так как в таком случае исключается необходимость частых передач SQL-запросов. Хранимые процедуры также облегчают написание приложений, поскольку стандартные процессы могут быть сформированы в удобные для использования процедуры. Вообще-то, хранимые процедуры — это просто частный вид более широкой категории, называемой хранимые программы. Оставшиеся две разновидности хранимых программ — это хранимые функции и триггеры.

■ ТИПЫ ХРАНИМЫХ ПРОГРАММ

Хранимая процедура	Программа, не возвращающая значения вызывающей программе
Хранимая функция	Программа, возвращающая значения вызывающей программе
Триггер	Программа, запускаемая автоматически до и после операции изменения базы данных

ВОПРОСЫ И ЗАДАНИЯ

Ответьте на следующие вопросы. Ответы вы найдёте на стр. 219.

В1

На каком уровне в трёхуровневой архитектуре клиент-сервер работает база данных?

В2

На каком уровне в трёхуровневой архитектуре клиент-сервер осуществляется связь с пользователем и отображаются результаты?

ЧТО ТАКОЕ РАСПРЕДЕЛЁННАЯ БАЗА ДАННЫХ (DISTRIBUTED DATABASE)?

В веб-приложении процесс обработки данных распределён между сервером базы данных, веб-сервером и веб-браузером, где каждому предписано выполнение определённых задач. Такой тип распределённой системы предусматривает гибкую систему обработки данных и сокращает объём обработки, необходимый для каждого сервера.

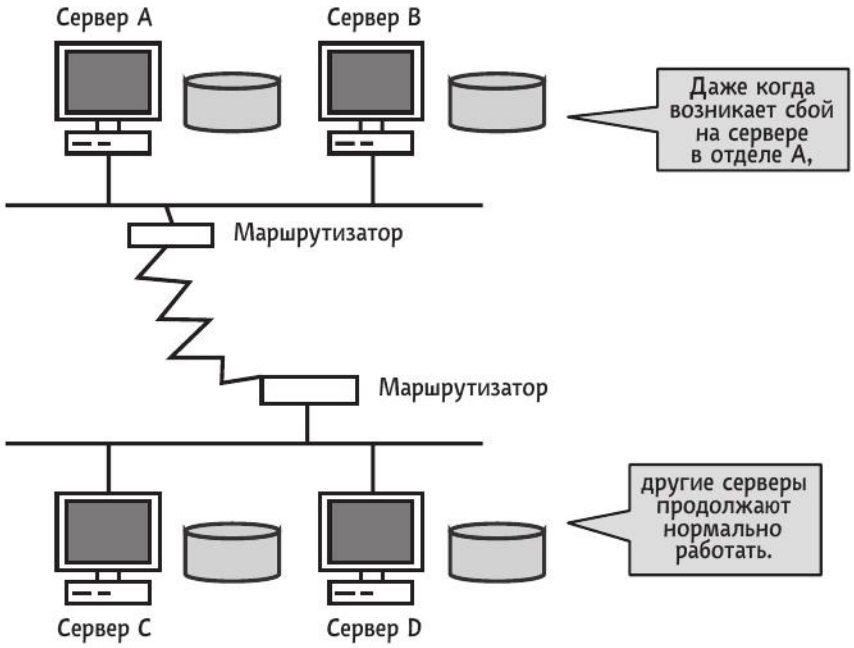
Однако сам сервер БД может быть распределён между несколькими серверами. Распределённые серверы БД могут находиться в разных местах или в одной и той же сети. Заметим, однако, что распределённой БД можно управлять как единой базой. Если оказывается, что распределённая БД — это единый сервер, пользователю не надо думать о расположении или передаче данных.

Как вы скоро увидите, база данных может быть распределена горизонтально и вертикально.

ГОРИЗОНТАЛЬНОЕ РАСПРЕДЕЛЕНИЕ (HORIZONTAL DISTRIBUTION)

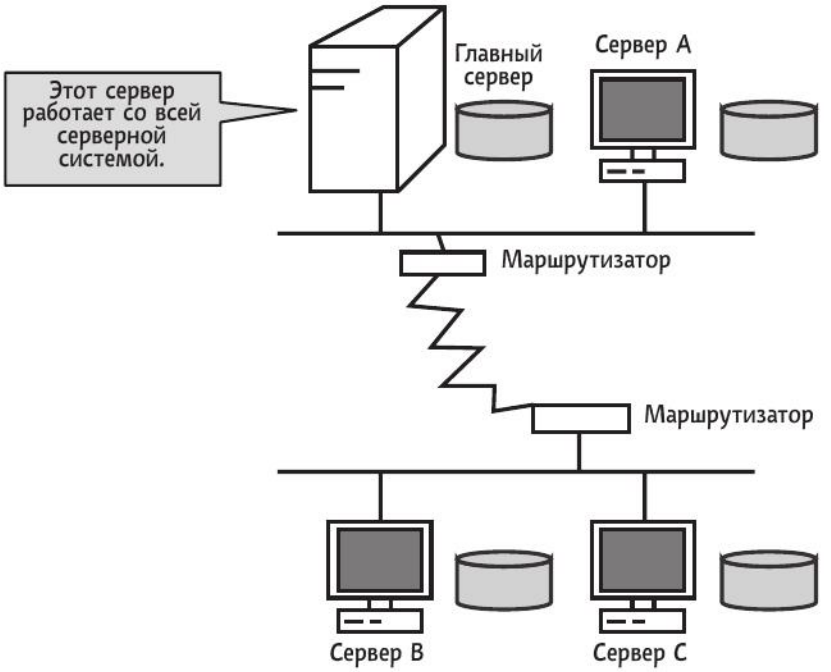
При горизонтальном распределении используются несколько одноуровневых серверов БД. Каждый сервер БД может использовать данные от других серверов БД, и, в свою очередь, каждый из них допускает собственное использование другими серверами БД. Эта структура применяется для системы расширенных БД, которые работают раздельно в каждом отделе.

Горизонтально распределённая база данных — это отказоустойчивая система, так как сбой на одном из серверов не повлияет на работу базы в целом.



ВЕРТИКАЛЬНОЕ РАСПРЕДЕЛЕНИЕ (VERTICAL DISTRIBUTION)

При вертикальном распределении разным серверам БД предписаны разные функции. Один из серверов работает в качестве главного сервера и играет ключевую роль, тогда как другие отвечают за выполнение требуемых задач. Вертикально распределённая БД облегчает управление главным сервером БД, хотя при этом на этот сервер ложится большая нагрузка. В качестве примера вертикального распределения можно привести главный сервер компании и серверы, индивидуальные для каждого отдела.



ДЕКОМПОЗИЦИЯ ДАННЫХ (DATA PARTITIONING)

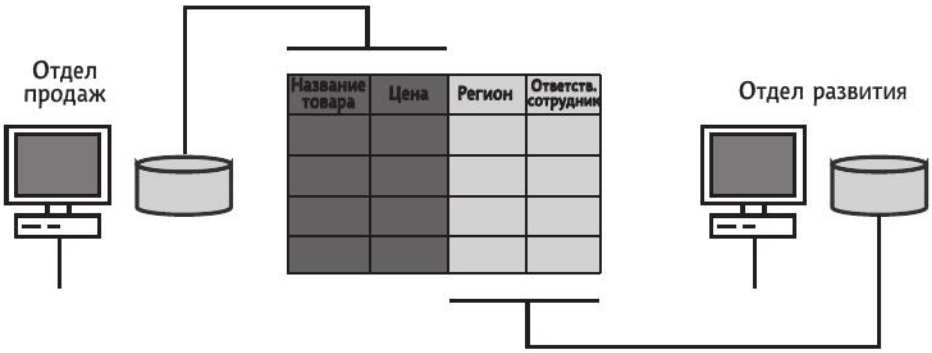
В распределённой БД сохраняемые данные распределяются между серверами. Вы должны тщательно продумать, как разделить данные. Их можно разбить следующими способами.

ГОРИЗОНТАЛЬНАЯ ДЕКОМПОЗИЦИЯ (HORIZONTAL PARTITION)

Горизонтальная декомпозиция делит данные на блоки строк. Строки, полученные в результате такого разделения, распределены между серверами. Эта форма декомпозиции часто применяется в случае, когда данные могут быть организованы в группы таким образом, что связанные данные, которые часто запрашиваются одновременно, хранятся на одном сервере.

ВЕРТИКАЛЬНАЯ ДЕКОМПОЗИЦИЯ (VERTICAL PARTITION)

Вертикальная декомпозиция делит данные на блоки столбцов. Столбцы, полученные в результате такого разделения, распределены между серверами. Например, вертикальная декомпозиция может использоваться для управления и соединения независимых баз данных, принадлежащих, скажем, отделу продаж, отделу развития и отделу экспорта.

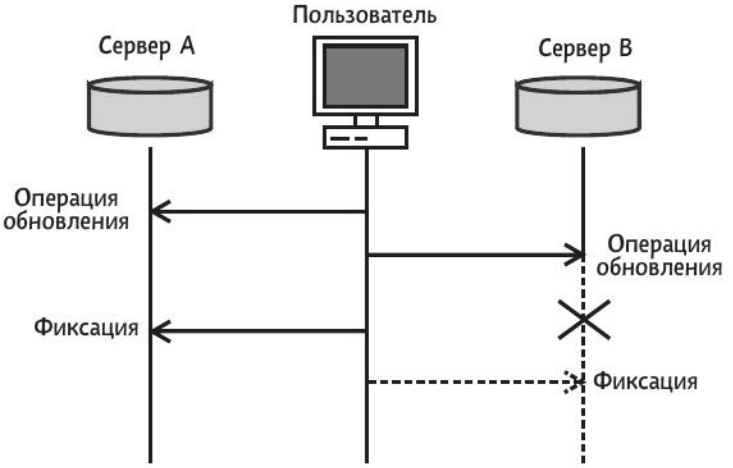


ПРЕДОТВРАЩЕНИЕ НЕСОГЛАСОВАННОСТИ ПРИ ДВУХФАЗНОЙ ФИКСАЦИИ ТРАНЗАКЦИЙ

В распределённой системе базы данных на разных серверах можно сконфигурировать таким образом, что на взгляд пользователя они будут работать как единая БД. Для этого нужно предпринять некоторые шаги, учитывая что данные на самом деле распределены между разными серверами.

Во-первых, когда бы ни происходила фиксация данных, все данные на всех серверах должны обновляться согласовано.

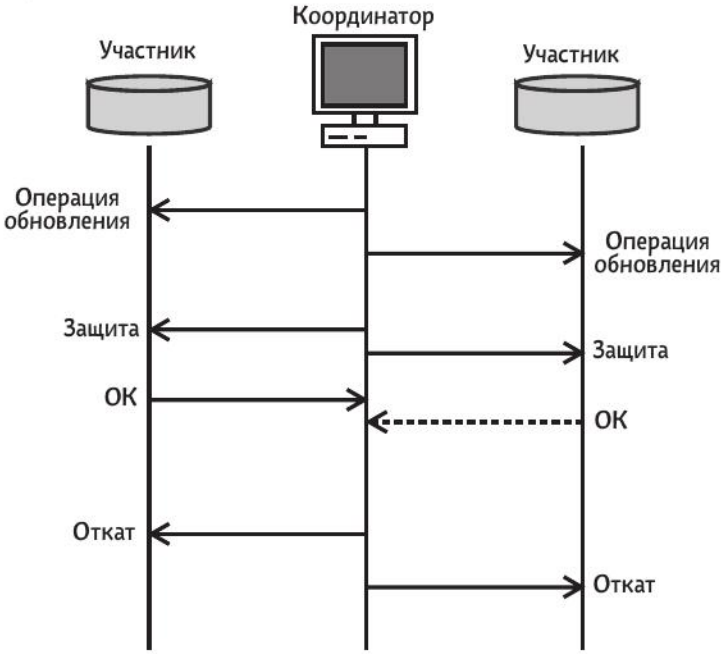
В распределённой системе БД стандартный метод фиксации может привести к тому, что один из серверов будет обновляться, а другой нет, как это показано ниже. Это нарушение свойства атомарности транзакции, так как эта транзакция не закончится ни фиксацией, ни откатом. Это также приведёт в тому, что система баз данных в целом станет несогласованной.



Следовательно, в распределённой системе баз данных принята двухфазная фиксация (two-phase commit). Двухфазная фиксация создаёт одну операцию фиксации для обеих операций фиксации — первой и второй.

Операция двухфазной фиксации подразумевает координатора и участников. В первой фазе такой фиксации координатор спрашивает участников, возможно ли провести операцию фиксации. Если возможно, то участники посылают утвердительный ответ. Этот подготовительный шаг называется преднастройкой (prepare). Во второй фазе координатор даёт инструкции для фиксации, и все участники выполняют фиксацию в соответствии с ними.

Если какой-либо из узлов не обезопасит операцию при двухфазной фиксации, все участники получат команду к откату. Вот каким образом базы данных на всех серверах остаются согласованными друг с другом.



ВОПРОСЫ И ЗАДАНИЯ

Ответьте на следующие вопросы о двухфазной фиксации. Ответы вы найдете на стр. 219.

В3

Какие инструкции даёт координатор во время первой фазы на схеме двухфазной фиксации?

В4

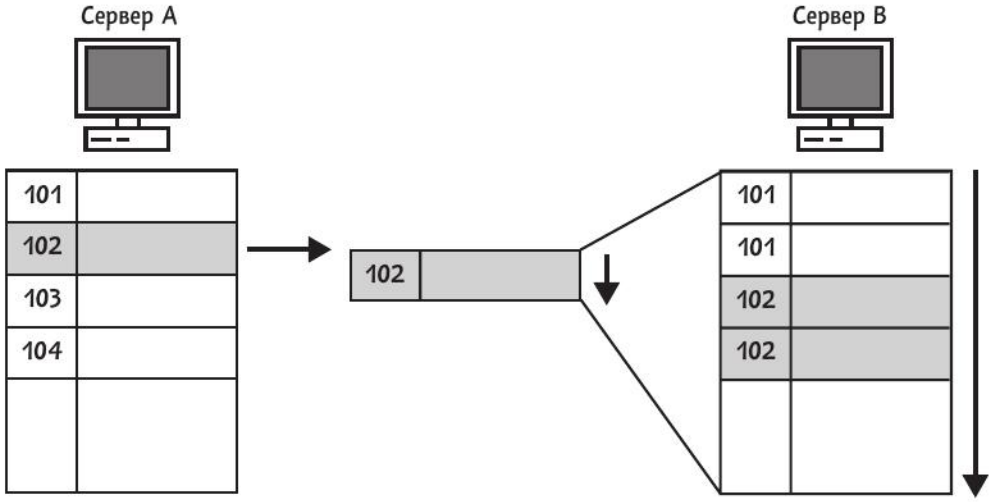
Какие инструкции даёт координатор во время второй фазы на схеме двухфазной фиксации?

СВЯЗАННЫЕ ТАБЛИЦЫ В РАСПРЕДЕЛЁННЫХ БД

Что касается распределённых баз данных, то в пространстве сети величина сообщения растёт и возникают случаи увеличения её загрузки. Особенно внимательными к величине передачи данных нужно быть в случае объединения таблиц в пространстве сервера. Существуют следующие типы объединения таблиц в распределённых базах данных.

ВЛОЖЕННЫЕ ЦИКЛЫ (NESTED LOOP)

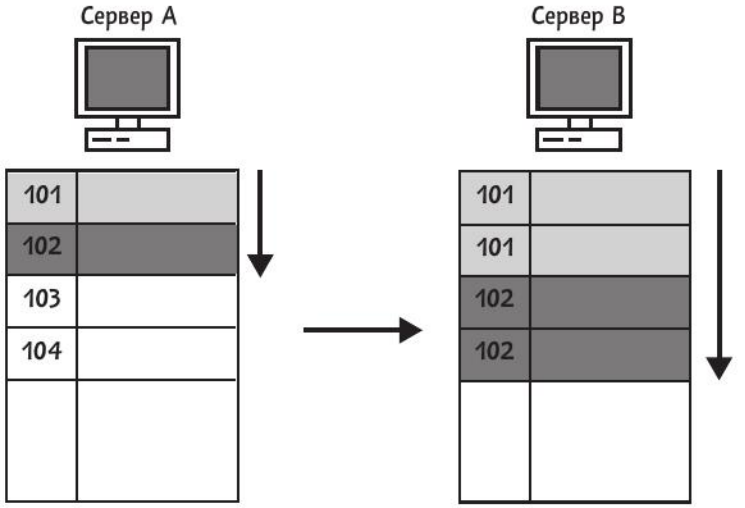
При передаче одной строки таблицы сервера A серверу B осуществляются сравнение и объединение со всеми пунктами таблицы сервера B. Таким образом, происходит повтор всех строк таблицы сервера A.



Вложенные циклы

СОРТИРОВКА СЛИЯНИЕМ (SORT MERGE)

Сортировка слиянием — это метод предварительной сортировки таблиц каждого сервера. Сперва осуществляется сортировка таблиц каждого из серверов A и B, затем они передаются. Для предварительной сортировки можно произвести объединение и считывание таблиц.

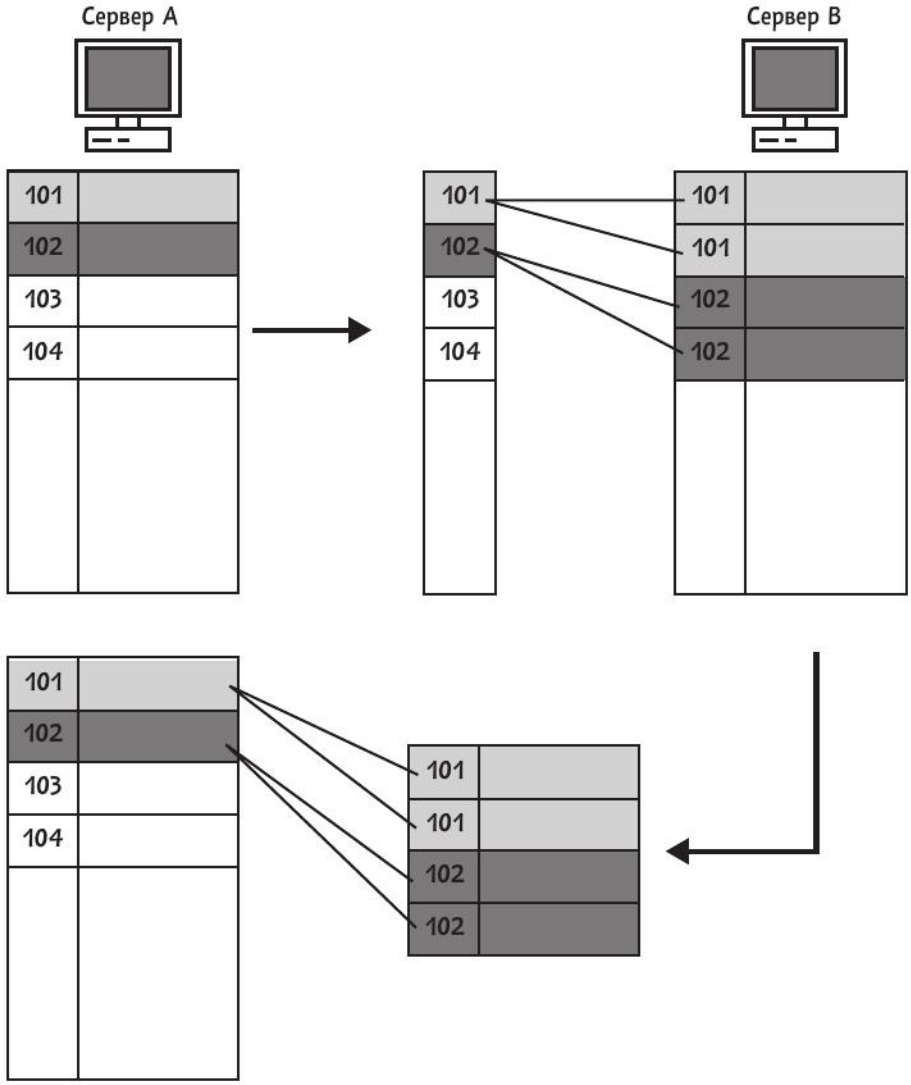


Сортировка слиянием

ПОЛУСЛИЯНИЕ (SEMI JOIN)

Полуслияние — способ осуществления объединения, при котором сливаются только строки, связанные с этим объединением и передаваемые серверу, становящемуся связующим звеном.

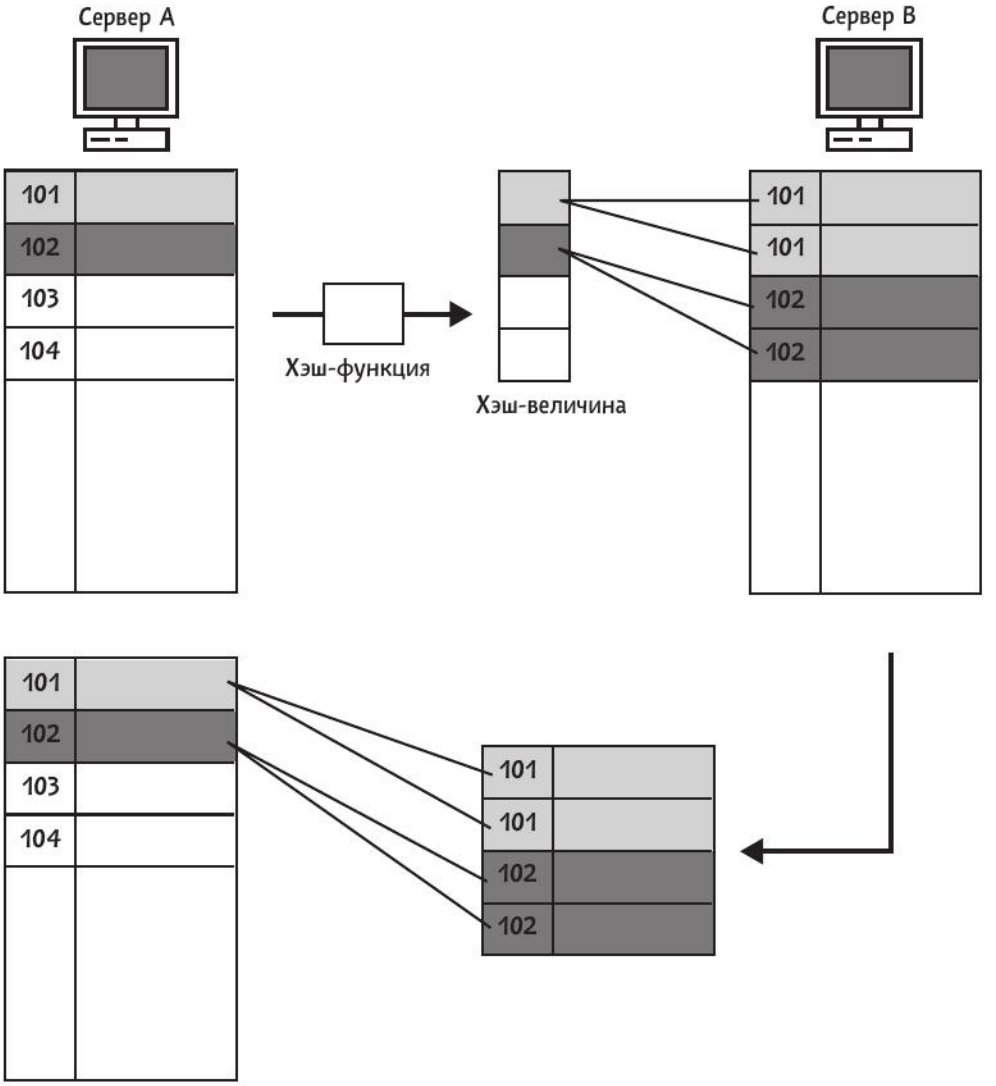
Например, сперва список товарных кодов сервера А передаётся серверу В. Затем извлекаются товарные коды, относящиеся к серверу В. Извлечённые строки возвращаются серверу А. По отношению к ним осуществляется процесс слияния. В случае слияния строк снижается нагрузка на сеть.



Полуслияние

ХЭШ-ПОЛУСЛИЯНИЕ (HASH SEMI JOIN)

Хэш-полуслияние — принятие хэш-величин строк сервера A при передаче серверу B. При этом у сервера B также берётся хэш-величина. Дружественные хэш-величины сравниваются, и в соответствии с этим осуществляется объединение.



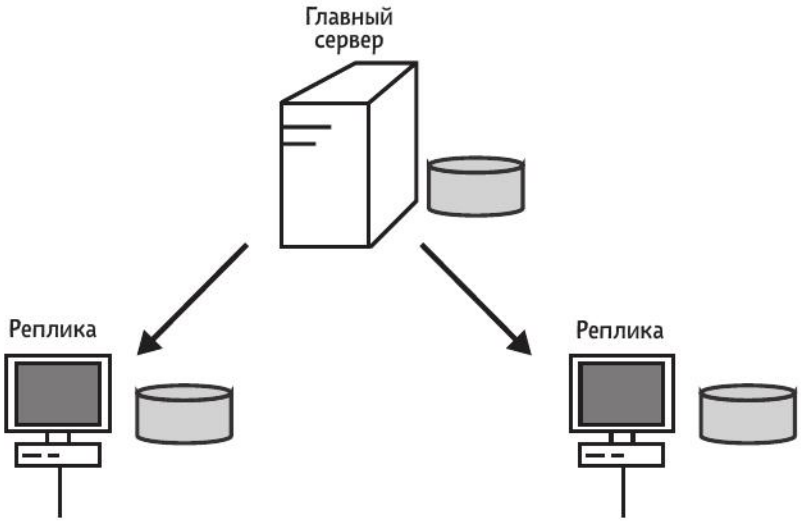
Хэш-полуслияние

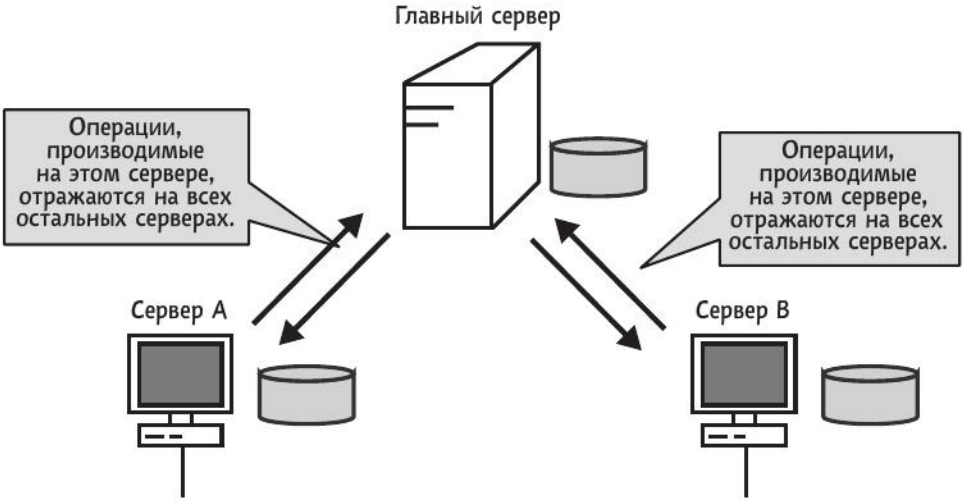
РЕПЛИКАЦИЯ БАЗ ДАННЫХ (DATABASE REPLICATION)

Некоторые распределённые базы данных имеют дублирующую БД, или реплику, которая уменьшает нагрузку на сеть. Эта практика получила название репликация (replication). Первичная база данных называется основной базой данных (master database), а её копия — репликой (replica). Существует несколько видов репликации.

ТОЛЬКО ЧТЕНИЕ (READ-ONLY)

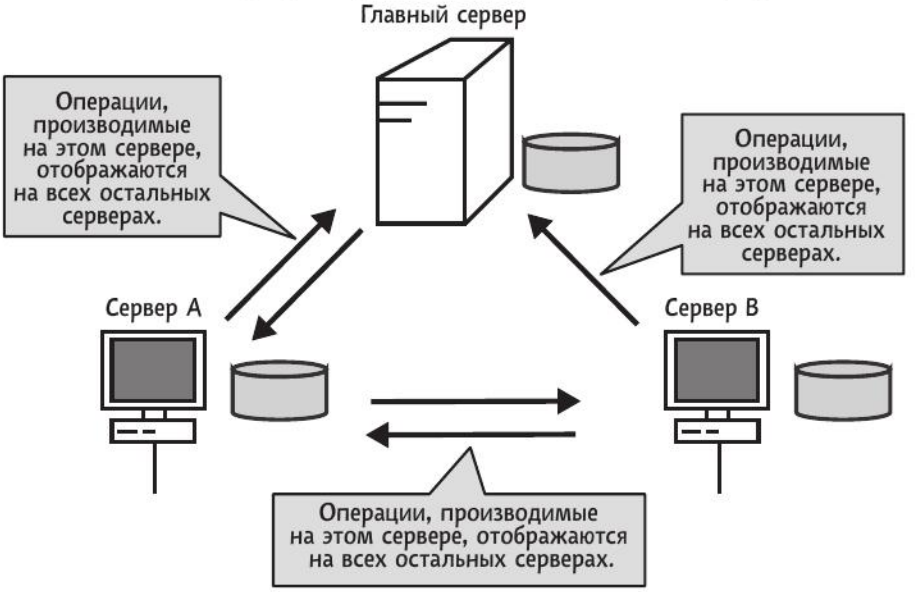
Реплика «только для чтения» создаётся и скачивается с основной базы данных на главном сервере. Чтобы изменить данные, пользователи должны подсоединиться к главному серверу.





РЕПЛИКАЦИЯ, ДОСТУПНАЯ ДЛЯ ВСЕХ СЕРВЕРОВ

При таком методе одну и ту же основную базу данных разделяют на несколько серверов. Обновления на каком-либо из серверов отражаются на всех остальных серверах.



ДАЛЬНЕЙШЕЕ ПРИМЕНЕНИЕ БАЗ ДАННЫХ

Этот заключительный раздел знакомит вас с прикладными технологиями на основе баз данных.

XML

Язык XML (Extensible Markup Language — расширяемый язык разметки) становится все более популярным в качестве способа хранения данных. XML представляет данные, заключая их в теги. Так как эти теги могут давать информацию о данных, которые в них находятся, этот язык применяется для хранения и поиска данных.

XML удобен, потому что его жёстко структурированная грамматика облегчает программируемые процессы. Более того, XML представлен в виде текстовых файлов (которые легко редактировать) и может взаимодействовать с другими системами. По этим причинам иногда вместо базы данных в качестве способа представления данных используется XML.

ОБЪЕКТНО-ОРИЕНТИРОВАННЫЕ БАЗЫ ДАННЫХ (OODB)

Реляционные базы данных хранят данные в виде текста в таблице. Однако реляционные БД не подходят для работы с данными определённого типа. Вот когда прибегают к использованию объектно-ориентированных баз данных (OODB).

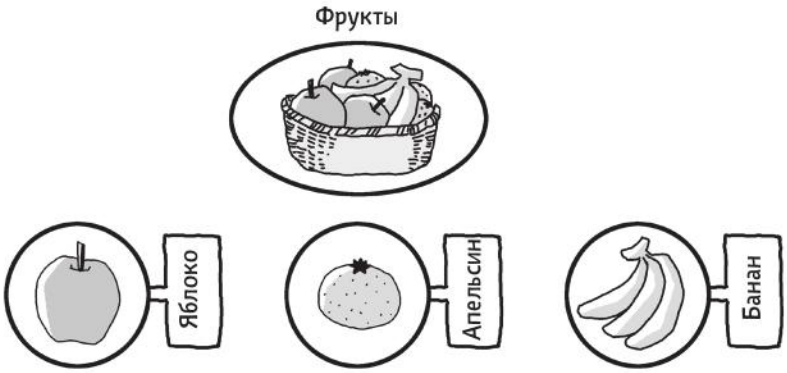
При объектно-ориентированном методе используются объекты (objects) — наборы данных и системы команд, определяющие правила использования этих данных. Иногда объект еще называют экземпляром (instance).

В объектно-ориентированной БД вы также можете управлять составными объектами данных (compound objects) — один объект, вложенный в другой. Это, к примеру, означает, что вы можете хранить данные, состоящие из изображения и текста, как единый объект. Объектно-ориентированная БД обеспечивает гибкое управление сложными типами данных.

В объектно-ориентированной БД существуют разные подходы для упрощения объектно-ориентированной разработки. Шаблон для объектов называется класс (class). Например, предположим, что вы разработали класс Яблоко. Объектами (экземплярами) в этом классе могут быть Яблоко А, Яблоко Б и так далее. Класс Яблоко делает возможным создание этих объектов.

В объектно-ориентированной схеме класс также может иметь иерархические отношения. Вы можете создать класс-потомок, который имеет такие же данные и функции, как и класс-родитель. Этот тип отношений называется наследование (inheritance). Вы также можете дать классу-потомку уникальные функции.

Например, класс Яблоко и класс Апельсин могут наследовать данные и функции класса Фрукты, но каждый из них также имеет свои собственные уникальные данные и функции. При объектно-ориентированной схеме иерархические отношения помогут вам добиться продуктивной разработки.



ИТОГИ

- Трёхуровневая система клиент-сервер — это метод построения веб-приложения.
- База данных действует как уровень данных.
- Система распределённых баз данных управляет данными, которые раздроблены.
- Метод двухфазной фиксации используется в распределённой БД.

ОТВЕТЫ

01 Уровень данных.

02 Уровень представления.

03 Преднастройка.

04 Фиксация или откат.

ПОДВЕДЕНИЕ ИТОГОВ

Вам понравилось изучать базы данных? Прежде чем вы начнёте справляться со всеми аспектами управления базами даных, вам понадобится изучить ещё много материала, но основные принципы работы с базами данных всегда останутся неизменными. Для более глубоко понимания основ вы можете выделить в окружающей нас реальности какие-нибудь значимые данные, спроектировать БД и поработать с ней. Основываясь на полученных фундаментальных знаниях, в дальнейшем вы сможете приобрести глубокие познания и навыки в области построения баз данных. Удачи!

ЧАСТО ИСПОЛЬЗУЕМЫЕ SQL-КОМАНДЫ

ПРОСТОЙ ЗАПРОС

```
SELECT имя_столбца,…
FROM имя_таблицы
```

ЗАПРОС С УСЛОВИЕМ

```
SELECT имя_столбца,…
FROM имя_таблицы
WHERE условие;
```

ПОИСК ПО ШАБЛОНУ

```
SELECT имя_столбца,…
FROM имя_таблицы
WHERE имя_столбца LIKE 'шаблон';
```

УПОРЯДОЧЕННЫЙ ПОИСК

```
SELECT имя_столба,…
FROM имя_таблицы
WHERE условие
ORDER BY имя_столбца;
```

АГРЕГАЦИЯ И ГРУППИРОВАНИЕ

```
SELECT имя_столбца,…
FROM имя_таблицы
WHERE условие
GROUP BY имя_столбца_для_группирования
HAVING условие_для_группируемых_строк;
```

СОЕДИНЕНИЕ ТАБЛИЦ

```
SELECT имя_таблицы1.имя_столбца,…
FROM имя_таблицы1,имя_таблицы2,…
WHERE имя_таблицы1.имя_столбца=имя_
        таблицы2.имя_столбца;
```

СОЗДАНИЕ ТАБЛИЦЫ

```
CREATE TABLE имя_таблицы(
 имя_столбца1 тип_данных,
имя_столбца2 тип_данных
 …
);
```

СОЗДАНИЕ ПРЕДСТАВЛЕНИЯ

```
CREATE VIEW имя_представления
AS SELECT команда;
```

УДАЛЕНИЕ РЕАЛЬНОЙ ТАБЛИЦЫ

```
DROP TABLE имя_таблицы;
```

УДАЛЕНИЕ ПРЕДСТАВЛЕНИЯ

```
DROP VIEW имя_представления;
```

ВСТАВКА СТРОКИ

```
INSERT INTO имя_таблицы(имя_столбца1,…)
VALUES (значение1,…);
```

ОБНОВЛЕНИЕ СТРОКИ

```
UPDATE имя_таблицы
SET имя_столбца=значение1,…
WHERE условие;
```

УДАЛЕНИЕ СТРОКИ

```
DELETE FROM имя_таблицы
WHERE условие;
```

СПРАВОЧНАЯ ЛИТЕРАТУРА

- Исихата Сиёси. Алгоритмы и структура данных. – Иванами Сётэн, 1989.
- Масунага Ёсифуми. Основы релятивных баз данных. – Ohmsha, 1990.
- Ацуси Иизава. Занимательный курс баз данных. – Кёрицу паблишинг, 1993.
- Китагава Хироюки. Системы баз данных. – Сёкодо, 1996.
- C.J. Date, Гид по уровням SQL. 4-е изд., испр. – ASCII, 1999.
- Экзаменационный центр инженеров обработки данных. Способности инженера технического уровня, уровень инженерной обработки данных (базы данных), 2000.
- E. F. Codd, A Relational Model of Data for Large Shared Data Banks, Communications of the ACM, Vol. 13, No. 6, pp. 377-387, 1970
- P. P. Chen, The Entity-Relationship Model: Toward a Unified View of Data, ACM Transactions on Database systems,Vol.1, No.1, pp.9-36, 1976
- ISO/IEC 9075, Information Technology – Database Languages – SQL, 1992
- ISO/IEC 9075, Information Technology – Database Languages – SQL, 1995
- ISO/IEC 9075–1, 2, 3, 4, Information Technology– Database Languages– SQL, 1999.
- Язык баз данных SQL, JIS X3005-1~4, 2002.

ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ

D

DBMS (система управления базой данных) — *21*
DCL (язык управления данными) — *106*
DDL (язык описания данных) — *106*
DML (язык манипулирования данными) — *106*

I

ISBN (Международный стандартный книжный номер) — *46*
ISO (Международная организация по стандартизации) — *128*

O

OODB (объектно-ориентированные базы данных) — *217-219*

R

READ COMMITED, чтение зафиксированных данных — *164*
READ UNCOMMITED, чтение незафиксированных данных — *164*
REPEATABLE READ, повторяющееся чтение — *164*

S

SQL (язык структурных запросов)
- агрегатные функции — *98-100, 111*
- базы данных для веб-приложений — *186-193, 203-204*
- используемые фразы — *94-97, 106, 111*
- логические операторы — *107*
- методы поиска — *93-97, 106, 108, 110, 112-115*
- обзор — *90-92, 106, 118, 128*
- операторы сравнения — *107*
- оптимизация запроса — *172-173*
- подзапросы — *112-114*
- подстановочные знаки — *97, 108*
- представление, создание — *119, 165*
- преобразование данных, манипулирование данными — *90-92, 100, 106, 118*
- соединение таблиц — *44, 101-102, 116*
- сравнение по шаблону — *108*
- стандартизация — *128*
- таблицы, создание — *91-92, 103-106, 117-121*
- условия, создание — *95-96, 101, 107-109*
- фраза GROUP BY — *111*
- фраза HAVING — *112*
- фраза WHERE — *94-97, 106, 111, 125*
- фразы, используемые в — *125*

SQL-команды
- ALL — *165*
- COMMIT — *137, 141-142, 154, 158*
- CREATE TABLE — *103, 117-121*
- CREATE VIEW — *119*
- DELETE — *104, 118, 125, 165*
- DROP TABLE/DROP VIEW — *120*
- GRANT — *165, 176*
- INSERT — *104, 118, 125, 165*
- LIKE — *97, 108*
- ORDER BY — *98*
- REVOKE — *165, 176*
- ROLLBACK — *140-141, 154, 158-159*
- SELECT — *93-98, 105, 106, 115, 125, 165*
- SET TRANSACTION — *164, 165*
- UPDATE — *104, 118, 125, 128, 165*
- агрегация и группирование — *221*
- вставка строки — *221*
- обновление строки — *221*
- поиск по шаблону — *221*
- соединение таблиц — *221*
- создание представления — *221*
- создание таблицы — *221*
- удаление представления — *221*
- удаление реальной таблицы — *221*
- удаление строки — *221*
- упорядоченный поиск — *221*

X

XML (расширяемый язык разметки) — *217*

А

Авторизованные пользователи — *145, 165*
Агрегатные функции — *98-100, 110-111*
Агрегация и группирование — *221*
Атомарность — *158*

Б

Базовые таблицы — *166*
Базы данных
- определение — *6, 10, 15*
- определенный — *195*
- построение на основе существующей системы — *14*
- применение (использование) — *19-21, 183-190*

- типы (разновидности) — *32-39*
- эффективность — *3-4, 15, 19, 150, 182*
Базы данных для Интернета — *217, см. Системы баз данных для веб-приложений*
Безопасность данных — *19, 142-146, 165-166, 173, 184, 190, 192*
Блокировки/управление блокировками — *135-141, 159-161, 166, 173, 183-184, 190*
Буферы — *174*

В

Ввод данных — *21, 90-92, 103-104, 106, 118*
Вводимые данные — *21, 90-92, 103-104, 118*
Вертикальная декомпозиция — *209*
Вертикальное распределение — *209*
Взаимная блокировка — *140*
Внешнее соединение — *116*
Внешние ключи — *44, 48, 72, 101*
Внешняя схема — *81*
Внутреннее соединение — *116*
Внутренняя схема — *81*
Возврат данных — *188*
Возвращаемые данные — *101-102, 217*
Восстановление данных — *20, 151-156, 173-174*
Восстановление после отказа (аварийное восстановление) — *151-154, 173-174*
Вставка строки — *221*
Вторая нормальная форма — *62, 64, 66-69, 78-79, 82*

Г

Горизонтальная декомпозиция — *208*
Горизонтальное распределение — *206*
Грубая детализация — *162*
Группирование — *111, 165*
Грязное чтение — *164*

Д

Двухфазная фиксация — *209-211, 219*
Двухфазное блокирование — *161-162*
Декомпозиция данных — *208-209*
Деление таблиц — *см. Нормализация*
Детализация — *162*
Детализация блокировок — *162*
Дублирование данных — *11, 16, 18, 19, 21, 29*
Дублирующие данные — *118*

З

Замечания — *30-31*
Записи — *27-28, 34, 48, 152-153*
Запрос с условием — *221*
Запросы — *см. SQL (язык структурных запросов)*
Защита данных — *19, 142-146, 165-166, 173, 184, 190, 192*

И

Иерархическая модель данных — *32, 33, 39, 218*
Иерархические отношения — *219*
Извлекаемые данные — *90-92, 95-99*
Изоляция — *158-162*
Имя пользователя — *145*
Индексация хэш-методом — *172*
Индексирование методом B-дерева — *168*
Индексы/индексирование — *147-151, 167-168, 173*
Искажение данных — *20, 21*
Искаженные данные — *158*

К

Кардинальность — *74*
Ключи
- внешний — *44, 48, 72, 101*
- первичный — *35, 44, 48, 65, 67, 72, 78-79, 101, 103, 117*
Команды
- ALL — *165*
- COMMIT — *137, 141-142, 154, 158*
- CREATE TABLE — *103, 117*
- CREATE VIEW — *119*
- DELETE — *104, 118, 125, 165*
- GRANT — *165, 176*
- INSERT — *118, 125*
- LIKE — *97, 108*
- ORDER BY — *98*
- ROLLBACK — *154, 140-142, 158-159*
- SELECT — *93-98, 105, 106, 115, 125*
- UPDATE — *104, 118*
- DROP TABLE/DROP VIEW — *120*
- INSERT — *104, 165*
- REVOKE — *165, 176*
- SELECT — *165*
- SET TRANSACTION — *164, 165*
- UPDATE — *125, 165*
Контроль метки времени — *163*
Контрольные точки — *173*
Концептуальная схема — *81*
Коррелированный подзапрос — *115*

Л

Логи — *152-153*
Логические операторы — *107*
Логический уровень — *202-204*

М

Математические операции — *см. Операции*

Международная организация по стандартизации (ISO) — *128*
Международный стандартный книжный номер (ISBN) — *46*
Мелкая детализация — *162*
Метод вложенный цикл — *172*
Метод сортировка-слияние — *171, 172*
Методы поиска — *93-97, 106, 108, 110, 112-115, см. SQL (язык структурных запросов)*
Механизмы восстановления — *20, 151-154, 173*
Модели данных — *32-39*
Монопольная блокировка — *138-140*
Монопольная группировка — *159-160*

Н

Независимое управление данными — *72*
Ненормализованные формы — *62, 78-79*
Неповторяющееся чтение — *164*
Несанкционированное наложение данных — *144, 148*
Несогласованные данные — *13, 60*
Нормализация — *60-72, 78-81*
Нормализованные таблицы — *72, 91*

О

Обновление строки — *221*
Обрабатываемые данные — *165, 173, 203-206*
Обработка данных — *35-37, 47-48, 134, 165, 173, 190, 203-206*
Обработка запроса по стоимости выполнения — *172, 173*
Общие данные, проблемы — *12, 183*
Объектно-ориентированные базы данных (OODB) — *217-219*
Операторы — *107*
Операторы сравнения — *107*
Операции
- выборка — *37, 39, 43, 47, 48, 172*
- декартово произведение — *37, 39, 42*
- деление — *37, 43, 45*
- извлечение данных — *39-47*
- над множествами — *39-42*
- объединения — *37, 39, 40, 48*
- пересечение — *37, 39, 41*
- проекция — *36, 37, 43, 172*
- разность — *37, 39, 41*
- реляционные — *43-47*
- соединение — *37, 43, 44, 48, 172*
Операция выборки — *48, 172*
Операция вычитания — *41*
Операция Декартово произведение — *42*
Операция деления — *43, 45*
Операция записи — *134, 137-138, 165*
Операция извлечения данных — *36-37, 39-47*
Операция множества — *39-42*
Операция объединения — *39, 40, 48*
Операция пересечения — *39, 41*
Операция проекции — *172*
Операция соединения — *43, 172*
Операция чтения — *134, 137, 138, 165*
Операция выборки — *37, 39, 43, 47*
Операция вычитания — *37, 39*
Операция Декартово произведение — *37, 39*
Операция деления — *37*
Операция объединения — *37*
Операция пересечения — *37*
Операция проекции — *36, 37, 43*
Операция соединения — *37, 44, 48*
Оптимизатор на базе правил — *172, 173*
Оптимизаторы — *173*
Оптимизация запросов — *172, 173*
Оптимистическое управление параллелизмом — *163*
Основная база данных — *216*
Отказоустойчивая система — *192, 206*
Отказы носителя — *172, 173*
Отказы системы — *172, 173*
Отказы, сбои баз данных — *172*
Откат — *140*

П

Память — *см. Команды*
Пароли — *145*
Первая нормальная форма — *66, 62-64, 78-79*
Первичные ключи — *35, 44, 48, 65, 67, 72, 78-79, 101, 103, 117*
Повторное исполнение — *153, 157*
Подзапросы — *112-114*
Подстановочные знаки — *97, 108*
Поиск на полное совпадение — *170*
Поиск по шаблону — *221*
Поля — *27-28, 30, 34, 35, 48*
Потеря данных — *20*
Потерянные данные — *158*
Права доступа — *19, 106, 130-133, 145-146, 165-166, 173*
Правое внешнее соединение — *116*
Представления, создание — *119, 166*
Проблемы управления данными
- дублирование данных — *11, 16, 18, 19, 21, 29*
- искажение/потеря данных — *20*
- искаженные/потерянные данные — *158*
- несогласованные данные — *13*
- несопоставимые данные — *158, 165, 209-211*
- общие данные — *183*

- отказ базы данных — *172*
- противоречивые данные — *13, 17-18, 21, 60, 118, 158, 163*
Проектирование базы данных — *19, 27*
- модель сущность-связь (E-R-модель) — *50-55, 74-77, 81*
- нормализация — *60-72, 78-81*
- определение состояния данных — *74*
- стадии проектирования — *84*
- стадии проектирования — *81*
Простой запрос — *221*
Противоречивые данные — *17-18, 158, 163, 165, 209-211*
Протокол передачи гипертекста (HTTP) — *186, 188, 202, 217*
Пустое значение NULL — *30, 31, 108*

Р

Разделяемая блокировка — *137, 141, 160-161*
Разрешение доступа, полномочия пользователя — *106, 130-132, 145-146, 165-166, 173*
Разрешения — *145-146, 165-166, 173*
Распределенные системы БД
- вертикальное распределение — *207*
- горизонтальное распределение — *206*
- двухфазная фиксация транзакций — *209-211, 219*
- декомпозиция данных — *208*
- обзор — *191-192, 206-208, 219*
- репликация в — *216*
Расширяемый язык разметки (XML) — *217*
Реальное состояние — *52, 55-56, 59, 74*
Резервные копии — *174*
Реляционная модель данных — *33-35, 39, 47, 48*
Реляционные операции — *43-47*
Реплика «только чтение» — *216*
Репликация баз данных — *216*
Реплики — *216*
Ресурсы — *160, 165*

С

Сбои в работе БД (отказы БД) — *172-173*
Связи
- иерархические связи — *32, 33, 39*
- модель E-R (сущность-связь) — *50-55, 74-77, 81*
- принцип — *54, 74*
- связь многие ко многим — *55, 74, 75, 81*
- связь один к одному — *74, 81*
- связь один ко многим — *55, 75, 81*
Связь Многие ко многим — *55, 74, 75, 81*
Связь Один к одному — *74, 81*
Связь Один ко многим — *55, 75, 81*
Серверы — *186-193, 202-205*
Серверы приложений — *190, 203*
Сериализуемость — *164*
Сетевая модель данных — *33, 39*
Символьные строки — *84, 108*
Система на основе документов — *21*
Система управления базой данных (DBMS) — *21*
Системы баз данных для веб-приложений — *185-190, 202-205, 217*
Согласованность — *158-159, 192*
Соединение таблиц — *44, 101-102, 116, 221*
Создание представления — *221*
Создание таблицы — *221*
Сортировка — *См. Агрегатные функции; Индексы/индексирование*
Составные объекты данных — *217*
Сравнение по шаблону — *107, 108*
Столбцы — *34, 84*
Строки — *34, 84, 118*
Сущности — *52-54, 74*
Сущность-связь (E-R), модель — *50-55, 74-77, 81*
Схемы — *81*

Т

Таблицы
- базовая — *166*
- двумерные — *34, 79*
- множественные — *59*
- нормализация — *60-72, 78-81*
- нормализованные — *72, 91*
- ограничения — *117, 118*
- представления — *119, 166*
- принцип — *34, 39, 48*
- соединение — *44, 101-102, 116*
- создание — *57-59, 91-92, 103-106, 117-121*
- формы — *62-70, 81-82*
Табулирование — *57, см. Нормализация*
Теги — *217*
Термины, используемые в базах данных — *26-31*
Транзакции
- восстановление после сбоя — *153-154*
- операции чтения/записи — *134, 137, 138, 165*
- определенные — *130-133*
- отказы — *172*
- свойства — *158-167*
Транзитивно зависимое значение — *79*
Третья нормальная форма — *62, 68, 69-70, 78-79, 81, 82*
Трёхуровневая архитектура клиент/сервер — *202-206, 219*
Триггеры — *195, 204*

У

Удаление представления — *221*
Удаление реальной таблицы — *221*
Удаление строки — *221*
Уникальные поля — *30*
Упорядоченный поиск — *221*
Упорядоченный порядок обработки — *160*
Управление конкурентным доступом — *138*
Управление параллелизмом
- контроль метки времени — *163*
- методы управления блокировками — *135-141, 159-161, 166, 173*
- уровни изоляции — *163*

Управление разрозненными данными — *10, 11, 72*
Управление, доступ пользователей — *19, 130-133, 145-146, 165-166, 173*
Уровень представления — *219*
Уровни данных — *202-204, 219*
Уровни изоляции — *164*
Устойчивость — *158, 165-166*

Ф

Фантомное чтение — *164*
Формы — *62-70, 81-82*
Фраза GROUP BY — *111*
Фраза HAVING — *112*
Фраза WHERE — *94-97, 106, 111, 125*
Функции множества — *98, 111*
Функционально зависимые значения — *79*
Функция AVG (среднее значение) — *98, 99, 111*
Функция COUNT — *99-100, 111*
Функция MAX (максимальное значение) — *99-100*
Функция MIN (минимальное значение) — *99*
Функция SUM — *99, 111*

Х

Хранимые процедуры — *193-196, 205*
Хранимые функции — *205*
Хэш-функция — *173*

Э

Экземпляр — *217*
Эффективная система — *3*

Я

Язык гипертекстовой разметки (HTML) — *202, 217*
Язык манипулирования данными (DML) — *106*
Язык описания данных (DDL) — *106*
Язык структурных запросов — *см. SQL (язык структурных запросов)*
Язык управления данными (DCL) — *106*
Языки программирования — *186, 188, 202, 217*
Японские промышленные стандарты (JIS) — *128*

ОБ АВТОРЕ

Мана Такахаси окончила экономический факультет Токийского университета. В настоящее время специализируется как технический писатель и опубликовала ряд изданий на такие темы, как Java, C, XML, инженерная техника и системное администрирование.

Книги издательства «ДМК Пресс» можно заказать в торгово-издательском холдинге «Планета Альянс» наложенным платежом, выслав открытку или письмо по почтовому адресу:
115487, г. Москва, 2-й Нагатинский пр-д, д. 6А
При оформлении заказа следует указать адрес (полностью), по которому должны быть высланы книги; фамилию, имя и отчество получателя.
Желательно также указать свой телефон и электронный адрес.
Эти книги вы можете заказать и в интернет-магазине: www.alians-kniga.ru.
Оптовые закупки: тел. (499) 782-38-89
Электронный адрес: books@alians-kniga.ru.

Мана Такахаси (автор), Сёко Адзума (художник)

Занимательное программирование. Базы данных. Манга

Издательство выражает благодарность *Панфилову В. О.*

Главный редактор *Д. А. Мовчан*
dmkpress@gmail.com

Перевод *Т. И. Сенниковой*
Научный редактор *И. А. Сенников*
Верстальщик *А. Ю. Анненков*
Корректор *Г. И. Синяева*

Формат 70×90/16. Бумага офсетная.
Печать офсетная. Объём 15 п. л. Усл. п. л. 17,5. Тираж 200 экз.

Веб-сайт издательства ДМК Пресс: www.dmkpress.com