

Оглавление

От автора	13
Часть I. Классификация баз данных	19
Глава 1. Модели данных	21
Глава 2. Другие методы классификации баз данных	61
Часть II. Доступ к данным	73
Глава 3. Структуры хранения данных	75
Глава 4. Обработка данных	149
Часть III. Архитектура СУБД	209
Глава 5. Гарантии корректности данных	211
Глава 6. Устройство СУБД	257
Часть IV. Распределённые базы данных	273
Глава 7. Компромиссы распределённых баз данных	275
Глава 8. Изменение данных в распределённых системах	309
Часть V. Восстановление при сбоях	359
Глава 9. Репликация	361
Глава 10. Резервное копирование	377
Часть VI. Эксплуатация баз данных	389
Глава 11. Управление базой данных	391
Глава 12. Оборудование	403
Глава 13. Коммерческие вопросы эксплуатации	419
Часть VII. Безопасность баз данных	447
Глава 14. Разграничение доступа	449
Глава 15. Защита от внутренних угроз	461
Послесловие	475
Предметно-именной указатель	481

Содержание

От автора	13
Часть I. Классификация баз данных	19
Глава 1. Модели данных	21
1.1. Реляционные БД	21
Исторический экскурс	21
Реляционная алгебра	23
Ограничения целостности	27
Вспомогательные структуры данных	29
Диаграммы «сущность—связь»	30
Нормальные формы	33
1.2. Хранилища «ключ—значение»	38
Java caching API	38
Документоориентированные БД	39
Форматы хранения документов	43
Хранилища семейств колонок	50
БД временных рядов	52
1.3. Другие модели данных	53
Объектные БД	53
Графовые БД	55
1.4. Сравнение моделей данных	57
Литература	59
Глава 2. Другие методы классификации баз данных	61
2.1. Аналитические и транзакционные БД	61
2.2. Монолитные и распределённые БД	63
2.3. БД на диске и в памяти	66
2.4. Карта баз данных	68
Литература	71

Часть II. Доступ к данным	73
Глава 3. Структуры хранения данных	75
3.1. Общая информация о дисковых структурах	75
Изменяемые и неизменяемые структуры	75
Эффективность операций	77
3.2. Изменяемые структуры	79
Неупорядоченная таблица	79
Сжатие данных	83
B-дерево	85
Другие страничные структуры	90
Буферный кеш	93
3.3. Неизменяемые структуры	96
LSM-дерево	96
Фильтр Блума	100
Механизмы хранения на основе LSM-деревьев	102
Прочие неизменяемые структуры	103
3.4. Колоночное хранение	104
Концепция и история	104
Оптимизация доступа	107
Обновление данных	111
3.5. Локализация данных	115
Секционирование	115
Шардирование	122
3.6. СУБД в памяти	132
IMDG и IMDB	132
Обеспечение надёжного хранения	134
Примеры СУБД в памяти	136
Литература	145
Глава 4. Обработка данных	149
4.1. Поиск данных	149
Поиск данных в структурах на основе LSM-дерева	149
Поиск в неупорядоченной таблице	150
Индексирование	153
Поиск в B-дереве	157

Содержание

Самый быстрый способ поиска	161
4.2. Операции реляционной алгебры	162
Соединение вложенными циклами (nested loops)	164
Соединение слиянием (merge join)	165
Соединение хешированием (hash join)	166
Самый быстрый способ соединения	168
4.3. Оптимизация запросов	170
План выполнения запроса	171
Эвристическая оптимизация	174
Оптимизация, основанная на стоимости	175
Кеширование запросов и планов	184
Оптимизация параметризованных запросов	186
Ручное управление планами	190
4.4. Реализация бизнес-логики	192
Клиентские модули	192
ORM	197
Хранимый код	200
Литература	206

Часть III. Архитектура СУБД 209

Глава 5. Гарантии корректности данных 211	
5.1. Транзакции	211
Атомарность	212
Согласованность	213
Изоляция	214
Долговечность (надёжность)	220
5.2. Журналирование	221
Назначение журнала	221
Работа с журналом	222
Устройство журнала	226
Структуры без журналирования	228
Общий алгоритм выполнения транзакции	231
5.3. Блокировки	232
Классификация блокировок	232

Управление блокировками	239
Конкурентные транзакции без блокировок	243
5.4. Версионирование данных	244
Обработка данных без версионирования	244
Версионирование в страничных хранилищах	245
Версионирование в хранилищах на основе LSM-деревьев	250
Версионирование в СУБД в памяти	252
Литература	253
Глава 6. Устройство СУБД	257
6.1. Экземпляр	257
Структура экземпляра	257
Общая память экземпляра	258
Пользовательские процессы	263
Служебные процессы	266
6.2. База данных	267
Связь базы данных и экземпляра	267
Состав базы данных	268
Литература	271
Часть IV. Распределённые базы данных	273
Глава 7. Компромиссы распределённых баз данных	275
7.1. CAP-теорема	275
Формулировка CAP-теоремы	276
Критика CAP-теоремы	278
Системы CP и AP и классификация PACELC	279
7.2. Исторический экскурс: CA-системы	282
Oracle RAC	283
IBM PureData System for Transactions	285
HPE NonStop	287
SAP HANA	289
7.3. Согласованность в распределённых системах	291
Линеаризация изменений	291
Причинная согласованность	296

7.4. Топология кластера	299
Статическая топология	299
Протоколы сплетен	302
Литература	307
Глава 8. Изменение данных в распределённых системах	309
8.1. Основные понятия	309
8.2. Отказоустойчивый кластер с репликацией	311
8.3. Распределённый консенсус	314
Raft	315
Multi-raft	318
Raft	319
Zookeeper Atomic Broadcast	323
Другие алгоритмы распределённого консенсуса	324
8.4. Распределённые транзакции	326
Протокол двухфазной фиксации	326
Детерминированные транзакции (Calvin transactions)	328
Saga	333
8.5. Компенсация несогласованности	336
Нестрогий кворум и направленная передача	336
Восстановление данных и дерево Меркла	337
Версионирование объектов	338
8.6. Безопасные типы данных	342
Счётчик (counter)	343
Множество (set)	344
8.7. Архитектура распределённых платформ	345
Приложение как координатор транзакции	345
Буквальная реализация спецификации X/Open	346
Выделенный координатор транзакций	346
«Настоящее горизонтальное масштабирование»	348
Google Spanner	349
CockroachDB	350
FoundationDB	352
Независимые узлы — участники саги	354
Литература	355

Часть V. Восстановление при сбоях	359
Глава 9. Репликация	361
9.1. Блочная репликация	361
9.2. Физическая репликация	364
9.3. Логическая репликация	367
Репликация триггерами	370
Репликация с помощью журналов СУБД	371
Репликация с помощью CDC	371
Прикладная репликация	372
9.4. Так что же лучше?	373
Литература	375
Глава 10. Резервное копирование	377
10.1. Выгрузка данных	378
10.2. Холодное сохранение файлов БД	379
10.3. Горячее сохранение файлов БД	380
10.4. Восстановление на точку	383
10.5. Инкрементальное резервное копирование	384
Литература	388
Часть VI. Эксплуатация баз данных	389
Глава 11. Управление базой данных	391
11.1. Механизмы управления	391
Мониторинг и журналирование	391
Несколько слов о мониторинге	393
11.2. Мониторинг баз данных	395
Мониторинг доступности экземпляра	395
Мониторинг сервера	395
Мониторинг состояния экземпляра	397
11.3. Настройка производительности	398
Использование динамических представлений	398
Другие методы настройки производительности	399
Литература	402

Глава 12. Оборудование	403
12.1. Серверы	403
Мейнфреймы	403
Мини-ЭВМ	404
Открытые системы	405
Семейство x86	407
Что дальше?	410
12.2. Системы хранения данных	411
Дисковый массив или локальные диски?	411
Протоколы сетевого доступа к дискам	413
Hi-end или mid-range?	414
Отказоустойчивость дисковых массивов	415
Литература	418
Глава 13. Коммерческие вопросы эксплуатации	419
13.1. Надёжность и производительность	419
Тестирование производительности баз данных	419
Влияние оборудования на производительность БД	422
Обеспечение надёжности баз данных	424
13.2. Классификация информационных систем	428
13.3. Базы данных в облаке	434
Монолитные транзакционные платформы	435
Распределённые платформы	437
13.4. Процедура выбора платформы	439
Формирование списка	439
Оценка стоимости	441
Оценка возможностей	442
Результат	444
Литература	445
Часть VII. Безопасность баз данных	447
Глава 14. Разграничение доступа	449
14.1. Ролевая модель доступа	449
Субъекты и объекты	449
Полномочия	452

14.2. Ограничение доступа на уровне строк	454
Фильтры строк	454
Доступ на основе меток (label security)	456
14.3. Модель доступа для транзакционных приложений	458
Литература	460
Глава 15. Защита от внутренних угроз	461
15.1. Принятие решения о защите	461
Классификация данных	461
Принципы защиты данных	463
Модель угроз	464
15.2. Средства защиты данных	465
Шифрование	465
Настройка совмещения ролей	468
Аудит	469
Маскирование данных	472
Литература	474
Послесловие	475
Предметно-именной указатель	481

*Моей дочери Ольге,
благодаря которой всё это имеет смысл*

От автора

Я собирался прочесть массу других книг. В колледже я много читал, а однажды даже написал серию банальных и напыщенных статей для «Йельских новостей». И теперь я решил вернуться к этому занятию и стать самым ограниченным из всех специалистов, «всесторонне развитым человеком».

Фрэнсис Скотт Фицджеральд, «Великий Гэтсби»

О чём эта книга?

Эта книга — о базах данных.

Не самоучитель, который позволит освоить программирование за 21 день, не сборник рецептов, не детальное описание какой-то конкретной платформы. Эта книга — обо всех базах данных сразу. Об алгоритмах, структурах данных и принципах проектирования, лежащих в основе всех современных платформ.

«Но позвольте, — скажет искушённый читатель, — о базах данных написана добрая тысяча книг. Зачем нужна тысяча первая?»

Для того, чтобы дать практическим навыкам прочный фундамент в виде понимания внутреннего устройства инструментов, с которыми вы привыкли работать.

Давным-давно знаменитый мастер боевых искусств после семинара для всех желающих провёл ещё один — закрытый, для «чёрных поясов». Среди приглашённых были опытные бойцы, инструкторы, даже чемпионы Европы.

Сенсей дал задание: «Встали в стойку и пошли в стойке до противоположной стенки, там удар и идём обратно». А сам принялся ходить между участниками и поправлять технику перемещения.

Минут через двадцать кто-то не выдержал и высказал мастеру своё недоумение — мол, это же семинар для «чёрных поясов», не пора ли от базовых упражнений для новичков перейти к чему-то серьёзному? Мастер пожал плечами, снова скомандовал всем встать в стойку и прошёл вдоль строя, толкая каждого в плечо или в грудь.

Чемпионы полетели на землю, как кегли, на ногах остался стоять только один боец.

И тогда мастер объяснил, что именно базовыми вещами спортсмены часто пренебрегают, стремясь поскорее перейти от скучных упражнений к «приёмчикам» и спаррингам. И в итоге добираются до чёрного пояса, не умея как следует стоять. А если люди не умеют твёрдо стоять, они не могут ни нанести сильный удар, ни ловко маневрировать, ни эффективно защищаться. Вся их техника оказывается ущербной. А потом снова поставил учеников в стойку и заставил шагать до стенки и обратно.

Понимание устройства современных систем управления базами данных, алгоритмов и структур, на которых они базируются, подобно базовой технике в боевых искусствах.

Для кого эта книга?

Для всех, кто считает себя специалистом в области информационных технологий, прежде всего — для архитекторов информационных систем. Для тех, кто хочет войти в элиту.

Знаете ли вы, что изобретателем туалета считается сэр Джон Харингтон (Sir John Harington)¹, крестник королевы Елизаветы I? Конечно, в XVII веке туалет был предметом роскоши, доступным лишь королям и высшей знати. Позже он получил широкое распространение, и сегодня пользователями туалета является большинство населения Земли, а мастер, их устанавливающий, никак не ассоциируется у нас с рыцарским достоинством.

Но где-то существуют люди, проектирующие городские канализационные коллекторы или изобретающие новые материалы для труб и стандарты работы с ними. Это и есть настоящая инженерная элита.

Такой же путь проходит любая другая технология, включая и базы данных. Чуть больше полувека назад инженер, работающий с базой данных, казался если не волшебником, то как минимум посвящённым какой-то высокой ступени. Сегодня пользователем баз данных является каждый, у кого есть мобильный телефон, а для создания простых приложений, работающих с базами данных, достаточно прочесть тоненькую брошюрку или даже посмотреть на видео краткий курс лекций. Но для того, чтобы обеспечить эту кажущуюся простоту, нужны люди, на самом глубоком уровне понимающие, как эффективно хранить и обрабатывать данные. И это — тоже элита, системные программисты и архитекторы.

¹ www.thoughtco.com/who-invented-the-toilet-4059858.

С точки зрения программиста, «вошедшего в ИТ» после трёхмесячных курсов, архитектор — это некий мудрец в башне из слоновой кости, изрекающий прописные истины или, наоборот, грезящий о несбыточном. Однако если заглянуть в будущее чуть дальше, чем на год, то окажется, что «прописные истины» прописаны далеко не везде, а несбыточное неожиданно сбывается. В этой книге есть место и прописным истинам, и парадоксальным на первый взгляд утверждениям, которые заставят читателя задуматься и, возможно, пересмотреть отношение к некоторым «истинам». Вероятно, книга также изменит ваши представления о несбыточном и «сбыточном».

Кто же автор,

взавший на себя смелость писать такую книгу?

Такой же специалист, как и вы, — программист, администратор баз данных, архитектор. Специалист, для которого расхожее выражение о том, что знание нескольких закономерностей избавляет от необходимости помнить множество деталей, стало своеобразным профессиональным кредо, и которого это кредо ни разу не подводило.

Сегодня я хочу поделиться найденными закономерностями, ни в коей мере не посягая на ваше право узнать детали и разобраться в них лучше всех, став признанными экспертами.

Приятного чтения!

Благодарности

На обложке книги стоит единственное имя, но эта книга никогда бы не появилась на свет, если бы не помощь многих людей, которым я от души хочу сказать спасибо.

Директору Института системного программирования Российской академии наук **Арутюну Аветисяну** и замечательной команде программистов под его руководством — **Олегу Борисенко** и **Денису Турдакову**. В далёком 2015 году мы проводили большой совместный исследовательский проект, результаты которого оказались несколько неожиданными. Начав глубже разбираться в возникших вопросах, я обнаружил, что ответы на них непосредственно следуют из базовых принципов, на которых строятся все информационные системы. Именно тогда у меня возникла идея написать учебник, в котором эти принципы мог бы прочесть каждый желающий. А когда текст был готов, Олег внимательно его прочитал, с дотошностью настоящего учёного проверив все факты и утверждения.

Бизнес-тренеру, эксперту по партизанскому маркетингу **Александру Левитасу**. В самый разгар так называемой «пандемии» Александр организовал вебинар «Пишем книгу с Левитасом». Пусть я был не самым прилежным учеником, но без концентрированных знаний, полученных на этом вебинаре, книга вышла бы хуже и позже.

Моему коллеге и давнему другу **Егору Рогову**. Егора можно смело назвать соавтором этой книги. Именно он первым читал все написанные куски, и именно он исправил невероятное количество опечаток, оговорок, неточностей и неоднозначностей. Корректность формулировок и правильность языка — его заслуга. Кроме того, описывая устройство PostgreSQL, я всё время сверялся с его книгой «PostgreSQL изнутри».

Ведущему российскому эксперту в области баз данных **Константину Осипову**. Слушая его выступления, я понял, что выпускник факультета вычислительной математики и кибернетики МГУ имени Ломоносова просто обязан написать если не собственную платформу (а Константин создал Tarantool и Picodata, а также внёс значительный вклад в разработку MySQL и ScyllaDB), то хотя бы книгу о том, как устроены чужие платформы. Кроме того, благодаря замечаниям Константина эта книга превратилась, я надеюсь, из сухого перечисления фактов в рассказ не только о том, как устроены базы данных, но и о том, почему они так устроены.

Наталье Пивоваровой, которая руководила проектом внедрения корпоративного хранилища данных в компании «Вымпелком», одним из самых интересных проектов в моей карьере. Именно на этом проекте я почувствовал, что базы данных — это то, чему стоит посвятить свою жизнь. Сейчас Наталья Владимировна читает курс по базам данных в МГТУ имени Баумана, и именно она первой опробовала материал ещё не дописанной книги на студентах.

Компании «Постгрес Профессиональный» и лично **Олегу Бартунову**, её директору, за помощь в издании книги. Мы живём в мире, где информации слишком много, и не читатель ищет автора, а автор должен искать читателя. Многие книги, статьи, рассказы не нашли своего читателя потому, что усилий автора не хватило, чтобы пробиться через глухую стену равнодушия, коммерческой целесообразности и простых случайностей. То, что делает Олег и его команда для популяризации знаний, поистине бесценно.

Кроме того, хочу сказать спасибо моим коллегам, прочитавшим черновик книги и высказавшим множество бесценных замечаний: **Владимиру Харчикову** (Сбербанк-Технологии), **Александру Токареву** (Xsolla), **Игорю Мельникову**

(Постгрес Профессиональный), **Алексею Перегудову** (Инфосистемы Джет), **Андрею Кувалдину** (Сбербанк), **Валерию Марушеву** (ВТБ).

И отдельное спасибо моей дочери **Ольге**, которая не только вдохновляла меня на протяжении всей работы над книгой, но и нарисовала для неё обложку, вместившую символ понимания и мудрости — Ежа — и символ одной из лучших баз данных — Слона.

Часть I

**Классификация
баз данных**

Глава 1

Модели данных

Переход от неформального к формальному существенно неформален.

М. Р. Шура-Бура

Модель данных — это набор абстракций и приёмов, с помощью которых мы пытаемся имитировать понятия реального мира. Модель данных напрямую ничего не говорит ни о производительности базы данных, ни о надёжности, ни о возможности масштабирования. Тем не менее именно модель в большинстве случаев является ключевым фактором выбора платформы.

1.1. Реляционные БД

Ещё в начале XXI века само понятие «база данных» подразумевало именно реляционную базу. Появившись в 70-х годах XX века, реляционные платформы до сих пор остаются наиболее распространёнными и универсальными. В рейтинге популярных баз данных, который ведёт портал db-engines.com, первые четыре места занимают именно реляционные СУБД — Oracle, MySQL, Microsoft SQL Server и PostgreSQL, а среди 50 самых популярных баз данных реляционных — больше половины.

Исторический экскурс

В 50-е и 60-е годы XX века главными покупателями ЭВМ были крупный бизнес и государственные учреждения, которые привыкли работать с каталогами и картотеками. Именно эти инструменты управления большими объёмами информации легли в основу первых навигационных моделей данных — иерархической и более поздней сетевой.

Основным объектом навигационных моделей был тип сегмента (*segment type*) — структура, описывающая некий объект реального мира: пациента, товар, договор. Тип сегмента состоял из полей (*fields*), содержащих данные, — например

фамилию или год рождения. Экземпляр сегмента (segment instance) описывал конкретный объект.

В иерархической модели у каждого типа сегмента был родительский тип — например, «родителем» каждого товара мог быть производитель. Хорошим примером иерархической модели может служить файловая система, которой мы пользуемся каждый день. Важным отличием сетевой модели от иерархической было то, что сегмент мог иметь несколько «родителей».

Для доступа к данным программист получал набор функций, работавших в терминах указателей на конкретные «карточки» — экземпляры сегментов. Типичными функциями были «найти сегмент и указать на первый экземпляр», «перейти к следующему экземпляру», «перейти к родительскому (или дочернему) экземпляру», «найти экземпляр, удовлетворяющий условию».

Навигационные СУБД прекрасно решали задачи, которые ставил перед ними бизнес, особенно с учётом ограниченной мощности компьютеров того времени. Яркими примерами таких платформ могут служить IMS (IBM) и ADABAS (Software AG). Несмотря на то что «золотым веком» навигационных платформ считаются 60-е годы прошлого столетия, они развивались вплоть до 80-х годов — достаточно вспомнить такие продукты, как dBase, FoxBase, Clipper.

Началом эры реляционных баз данных принято считать статью «Реляционная модель данных для больших совместно используемых банков данных» («A Relational Model of Data for Large Shared Data Banks»), опубликованную в 1970 году Эдгаром Коддом (Edgar Frank Codd).

С точки зрения самой структуры данных предложенная Коддом реляционная модель практически не отличалась от навигационной — все отличия касались способов описания структуры и доступа к данным. По мнению Кодда, главный недостаток существующих на тот момент систем — зависимость приложений от физической организации данных. В статье приводятся три зависимости:

- Зависимость от порядка, в котором записаны данные. Если приложение предполагает, что данные записаны в определённом порядке (например, фамилии клиентов отсортированы по алфавиту), то нарушение этого порядка приведёт к ошибкам в работе приложения.
- Зависимость от индексации. Индексом называется вспомогательная структура данных, служащая для ускорения поиска. Если приложение рассчитано на использование какого-то индекса, то уничтожение этого индекса приведёт к неработоспособности приложения, и наоборот — использование нового индекса потребует изменения кода.

Конец ознакомительного фрагмента.

Приобрести книгу можно

в интернет-магазине

«Электронный универс»

e-Univers.ru