Введение

Вдохновение изложить собранный в этой книге материал появилось у меня вследствие приобретенного опыта разработки программного обеспечения Oracle, сотрудничества с другими разработчиками Oracle и оказания им помощи в построении надежных приложений на основе баз данных Oracle. В целом эта книга отражает то, чем мне приходится заниматься ежедневно, и в ней освещены вопросы, с которыми пользователи сталкиваются в своей повседневной работе.

Я раскрыл здесь то, что считаю наиболее важным, а именно — базы данных Oracle и их архитектуру. Я мог бы написать аналогично озаглавленную книгу, посвященную разработке приложений с использованием конкретного языка и архитектуры — например, приложений, которые применяют JavaServer Pages для взаимодействия с компонентами Enterprise JavaBean (EJB), которые, в свою очередь, используют JDBC для обмена данными с Oracle. Однако, по большому счету, для успешного построения таких приложений вы действительно должны хорошо разбираться во всех темах, рассмотренных в настоящей книге. В ней собран материал, который, по моему глубокому убеждению, должен быть всесторонне усвоен для успешной разработки приложений в среде Oracle, будь вы программистом на Visual Basic, применяющим ODBC, программистом на Java, использующим компоненты EJB и JDBC, или программистом на Perl, использующим DBI-интерфейс Perl. В книге не продвигается какая-то специфичная архитектура приложений; в ней не сравниваются трехуровневая и клиент-серверная архитектуры. Вместо этого в книге объясняется, что может делать база данных, и что вы должны понимать в плане особенностей ее функционирования. Поскольку база данных находится в основе архитектуры любого приложения, книга должна иметь широкую аудиторию.

Как следует из названия, эта книга сосредоточена на архитектуре базы данных и работе самой базы данных. Я подробно раскрываю архитектуру базы данных Oracle — файлы, структуры памяти и процессы, которые образуют базу данных и экземпляр. Затем я перехожу к обсуждению таких важных тем, как блокировка, управление параллелизмом, функционирование транзакций, повторение и отмена, и поясняю, почему эти вопросы настолько важны. Наконец, я рассматриваю физические структуры базы данных, такие как таблицы, индексы и типы данных, освещая приемы их оптимального применения.

О чем эта книга

Одна из проблем, обусловленная наличием многочисленных возможностей при разработке, заключается в том, что иногда трудно выбрать вариант, который лучше всего подходит в конкретных обстоятельствах. Каждый желает заполучить максимально возможную гибкость (столько вариантов, сколько вообще можно иметь), но при этом хочет, чтобы все оставалось очень ясным — другими словами, простым. СУБД Oracle предлагает разработчикам практически неограниченные возможности выбора. Никто никогда вам не скажет: “Вы не сможете сделать это в Oracle”. Наоборот, вы услышите: “Сколькими разными способами вы предпочитаете делать это в Oracle?”. Я надеюсь, что данная книга поможет вам производить правильный выбор.

Книга адресована тем, кто высоко ценит наличие выбора, но также предпочитает располагать какими-то руководящими принципами и сведениями о практических реализациях функциональных средств и возможностей Oracle. Например, в Oracle

имеется по-настоящему эффективное средство, которое называется параллельным выполнением. В документации Oracle рассказано, как его использовать и что оно делает. Однако в ней ничего не говорится о том, когда вы должны и — что, пожалуй, еще важнее — когда не должны применять это средство. В документации далеко не всегда сообщаются детали реализации средства, и если вы не осведомлены о них, это может еще не раз дать о себе знать. (Здесь я не имею в виду программные ошибки, а способ, которым средство предположительно функционирует, и для чего оно в действительности было предназначено.)

В этой книге я стремился не только описать, как работает та или иная функция, но также и объяснить, когда и почему вы должны обдумать возможность использования отдельного средства или реализации. Я твердо убежден, что в отношении всех сущностей важно понимать не только "как", но также "когда" и "почему", а заодно "когда нет" и "почему нет".

Кому адресована эта книга

К целевой аудитории этой книги относится любой, кто занимается разработкой приложений на основе баз данных Oracle. Она предназначена для профессиональных разработчиков Oracle, которым необходимо знать способы выполнения действий в базе данных. Практическая природа книги означает, что многие разделы должны быть очень интересными также и администраторам баз данных. В большинстве примеров для демонстрации ключевых средств применяется SQL*Plus, так что не надейтесь обнаружить в них по-настоящему крутой графический пользовательский интерфейс, но вы узнаете, как работает база данных Oracle, что могут делать ее ключевые средства и когда они должны (и не должны) использоваться.

Книга ориентирована на любого, кто желает получить от Oracle максимум, прикладывая минимальные усилия. Она предназначена для любого, кто ищет новые методы применения существующих средств. Она рассчитана на тех, кто хочет увидеть, как эти средства можно задействовать в реальном мире (не просто ознакомиться с примерами использования функции, но в первую очередь понять, почему она оказывается подходящей в той или иной ситуации). Еще одной категорией людей, кого может заинтересовать эта книга, являются технические руководители, ответственные за разработчиков, которые занимаются проектами Oracle. В некоторых отношениях очень важно, чтобы технические руководители понимали, почему знание базы данных имеет решающее значение в достижении успеха. Эта книга может стать серьезным подспорьем руководителям, которые хотят, чтобы их персонал был обучен надлежащим технологиям, или желают убедиться, что персонал уже знает то, что должен знать.

Чтобы извлечь максимальную пользу из этой книги, читатель должен соответствовать следующим характеристикам.

- **Знать язык SQL.** Вовсе не обязательно быть лучшим кодировщиком SQL во все времена, но хорошие практические навыки помогут в освоении.
- **Понимать PL/SQL.** Это не является обязательным условием, но облегчит усвоение примеров. Например, данная книга не научит вас применению цикла `FOR` или объявлению типа записи; вопросы такого рода подробно освещены в документации по Oracle и многочисленных книгах. Тем не менее, нельзя сказать, что вы не изучите много нового о PL/SQL, когда будете читать эту книгу.

Вы освоите разнообразные возможности PL/SQL, узнаете о новых способах решения задач и ознакомитесь с пакетами/функциями, о существовании которых, возможно, даже не подозревали.

- **Иметь представление о каком-нибудь языке третьего поколения, таком как C или Java.** Я уверен, что любой, кто способен читать и писать код на языке третьего поколения, сможет успешно понять примеры, приведенные в этой книге.
- **Быть знакомым с руководством по концепциям базы данных Oracle (Oracle Database Concepts).**

Последний пункт требует дополнительной пары слов: из-за значительного объема набора документации по Oracle многие находят ее устрашающе большой. Если вы только приступили к изучению или пока еще не прочитали ни одного документа из набора, то могу вас заверить, что руководство *Oracle Database Concepts* — это именно то, с чего следует начинать. Оно занимает около 450 страниц (я хорошо это знаю, т.к. некоторые страницы написаны мною, а редактировать мне довелось целиком все руководство) и затрагивает многие из основных концепций Oracle, о которых необходимо знать. Оно может не предоставлять абсолютно все технические детали (им посвящены остальные 10 000 – 20 000 страниц документации), но обучит вас всем важнейшим концепциям. В руководстве *Oracle Database Concepts* раскрыты перечисленные ниже темы (здесь упомянуты далеко не все):

- структуры в базе данных и способ организации и хранения данных;
- распределенная обработка;
- архитектура памяти Oracle;
- архитектура процессов Oracle;
- объекты схемы, которые вы будете использовать (таблицы, индексы, кластеры и т.д.);
- встроенные и определяемые пользователем типы данных;
- хранимые процедуры SQL;
- работа транзакций;
- оптимизатор;
- целостность данных;
- управление параллелизмом.

Время от времени я и сам буду возвращаться к этим темам. Они являются фундаментальными. Без их знания вы будете создавать приложения Oracle, предрасположенные к отказам. Я настоятельно рекомендую прочитать это руководство и усвоить упомянутые вопросы.

Как структурирована эта книга

Чтобы помочь в работе с книгой, большинство глав организовано в виде четырех основных разделов (как описано ниже). Хотя это не жесткое разделение, оно позволяет быстрее переходить к области, в которой требуется дополнительная информация. Книга содержит 15 глав, причем каждая из них подобна “мини-книге” —

практически самостоятельному компоненту. Иногда я ссылаюсь на примеры или средства, описанные в других главах, но любую главу книги вполне можно читать независимо от остальных. Например, вовсе не обязательно сначала читать главу 10, посвященную таблицам базы данных, чтобы понять материал главы 14, в которой рассматривается параллелизм.

Формат и стиль многих глав практически идентичен и представлен ниже.

- Введение в функциональное средство или возможность.
- Причины, по которым может потребоваться применение (или отказ от использования) этого средства или возможности. Здесь даются пояснения относительно того, когда следует подумать о применении этого средства, а когда отказаться от него.
- Как использовать это средство. Приводимые здесь сведения не являются просто копией материала из справочника по SQL. Напротив, они представлены в пошаговом виде: то, что необходимо получить, то, что понадобится сделать, и вопросы, которые нужно выяснить, чтобы начать. В этом разделе рассматриваются следующие темы:
  - способ реализации средства;
  - примеры применения;
  - способ отладки средства;
  - предостережения относительно использования средства;
  - способ обработки ошибок (упреждающим образом);
  - резюме с кратким подведением итогов.

Книга содержит большое количество примеров и кода, которые доступны для загрузки на веб-сайте издательства. В последующих разделах приведены краткие описания содержимого всех глав.

Глава 1. Разработка успешных приложений Oracle

В этой главе я описываю свой подход к программированию для баз данных. Базы данных не создавались одинаковыми. Для того чтобы успешно и своевременно разрабатывать приложения, управляемые базами данных, необходимо четко понимать, что именно конкретная база данных способна делать, и каким образом она это делает. Без знания того, что может делать база данных, возникает риск постоянно заново изобретать колесо, создавая средства, которые база данных уже предоставляет. Без понимания того, каким образом работает база данных, скорее всего, будут получаться приложения, которые функционируют неэффективно и ведут себя непредсказуемо.

В главе предлагается эмпирический взгляд на некоторые приложения, где недостаток понимания базы данных привел к неудаче всего проекта. В соответствии с таким ориентированным на примеры подходом здесь обсуждаются основные средства и функции базы данных, которые должны хорошо пониматься разработчиком. Суть в том, что вы не должны позволять себе трактовать базу данных как черный ящик, который просто выдает ответы и самостоятельно заботится о масштабируемости и производительности.

Глава 2. Обзор архитектуры

В этой главе раскрываются основы архитектуры Oracle. Мы начнем с четких определений двух терминов, которые многими в мире Oracle понимаются совершенно неправильно — *экземпляр* и *база данных*. Затем мы рассмотрим два новых типа баз данных, появившиеся в версии Oracle 12*c*, в частности — *контейнерная база данных* и *подключаемая база данных*. Кроме того, мы кратко обсудим системную глобальную область (System Global Area — SGA) и процессы, лежащие в основе экземпляра Oracle, а посмотрим, каким образом выполняется простое действие "подключения к Oracle".

Глава 3. Файлы

В этой главе подробно описаны восемь типов файлов, которые образуют базу данных и экземпляр Oracle. Начиная с простого файла параметров и заканчивая файлами данных и журнальными файлами повторения, мы посмотрим, что они собой представляют, каково их назначение и каким образом они используются.

Глава 4. Структуры памяти

В этой главе освещены вопросы использования памяти базой данных Oracle, как памяти в отдельных процессах (PGA), так и разделяемой памяти (SGA). Мы исследуем отличия между ручным и автоматическим управлением памятью PGA, автоматическим управлением разделяемой памятью в Oracle 10*g* и автоматическим управлением памятью в Oracle 11*g*, а также выясним, в каких случаях подходит каждый из этих методов. После прочтения этой главы вы получите полное представление о том, каким образом Oracle работает с памятью и управляет ею.

Глава 5. Процессы Oracle

В этой главе предлагается обзор типов процессов Oracle (серверных и фоновых процессов). Кроме того, в ней более подробно описаны отличия между подключением к базе данных с помощью процессов разделяемого и выделенного серверов. Мы также рассмотрим большинство фоновых процессов (таких как `LGWR`, `DBWR`, `PMON`, `SMON` и `LREG`), действующих во время запуска экземпляра Oracle, и обсудим их функции.

Глава 6. Блокировка и защелкивание данных

В разных базах данных задачи решаются по-разному (то, что хорошо работает в SQL Server, может не так хорошо работать в Oracle), и понимание особенностей реализации блокировки и управления параллелизмом абсолютно необходимо для успешного построения приложений. В этой главе рассматривается общий подход, используемый в Oracle для решения этих задач, типы блокировок, которые могут быть применены (DML, DDL и защелки), а также проблемы, которые могут возникать при неаккуратной реализации блокировки (взаимоблокировка, блокирование и эскалация).

Глава 7. Параллелизм и многоверсионность

В этой главе мы будем исследовать мое любимое средство Oracle — многоверсионность, а также его влияние на управление параллелизмом и всю структуру приложения. Здесь будет показано, что все базы данных отличаются друг от друга, а сама их реализация может влиять на структуру приложений. Мы начнем с обзора раз-

нообразных уровней изоляции транзакций, как они определены стандартом ANSI SQL, и посмотрим, каким образом они отображаются на реализацию Oracle (а также соответствие этому стандарту других баз данных). Затем мы проанализируем последствия, которые может иметь для нас многоверсионность — средство, позволяющее Oracle обеспечивать неблокирующее чтение базы данных.

Глава 8. Транзакции

Транзакции являются фундаментальным средством всех баз данных — это часть того, что отличает базу данных от файловой системы. Несмотря на это, их часто понимают неправильно, и многие разработчики даже не подозревают о том, что неумышленно отказываются от их применения. В этой главе показано, как должны использоваться транзакции в Oracle, а также раскрыты плохие привычки, которые могли быть приобретены при разработке приложений для других баз данных. В частности, мы взглянем на последствия атомарности и ее влияние на операторы в Oracle. Мы также обсудим операторы управления транзакциями (`COMMIT`, `SAVEPOINT` и `ROLLBACK`), ограничения целостности, распределенные транзакции (двухфазную фиксацию) и автономные транзакции.

Глава 9. Повтор и отмена

Можно было бы сказать, что разработчикам не обязательно разбираться в деталях реализации повтора (redo) и отмены (undo) в такой же степени, как администраторам баз данных, но разработчики должны знать, какую роль играют повтор и отмена в базе данных. После определения понятия повтора мы посмотрим, что в точности делает оператор `COMMIT`. Мы обсудим, каким образом выяснить объем сгенерированной информации redo и как с помощью конструкции `NOLOGGING` значительно сократить объем информации redo, генерируемой определенными операциями. Мы также исследуем генерацию данных redo в связи с такими проблемами, как очистка блоков и конкуренция за журналы.

В разделе главы, посвященном отмене, мы рассмотрим роль данных undo и операции, которые генерируют наибольший/наименьший объем информации undo. Наконец, мы обсудим печально известную ошибку `ORA-01555: snapshot too old` (ORA-01555: устаревший снимок), возможные причины ее возникновения и способы ее предотвращения.

Глава 10. Таблицы базы данных

В настоящее время Oracle поддерживает множество типов таблиц. В этой главе мы по очереди рассмотрим типы таблиц — традиционная таблица (т.е. стандартная, “нормальная” таблица), индекс-таблица, кластеризованная индекс-таблица, кластеризованная хеш-таблица, вложенная таблица, временная таблица и объектная таблица — и обсудим когда, как и почему они должны использоваться. Бо́льшую часть времени традиционной таблицы вполне достаточно, но данная глава поможет распознать ситуации, в которых лучше подходят другие типы.

Глава 11. Индексы

Индексы являются критически важным аспектом структуры приложения. Корректная реализация требует глубоких знаний данных, их распределения и планируемого использования. Слишком часто при разработке приложений к индексам относятся как к чему-то второстепенному, из-за чего в итоге страдает производительность.

В этой главе подробно рассматриваются различные типы индексов, включая индекс со структурой B-дерева (B*Tree), битовый индекс, индекс на основе функций и индекс предметной области, и обсуждаются обстоятельства, когда они должны, а когда не должны применяться. В разделе, посвященном часто задаваемым вопросам и мифам об индексах, я также отвечу на ряд распространенных вопросов вроде "Работают ли индексы на представлениях?" и "Почему индекс не используется?".

Глава 12. Типы данных

На выбор доступно большое количество типов данных. В этой главе исследуются все 22 встроенных типа данных, приводятся объяснения их реализации и рассматриваются способы и случаи их применения. В начале главы приводится краткий обзор поддержки национальных языков (National Language Support — NLS), знание которой обязательно для полного понимания простых строковых типов в Oracle. Затем мы перейдем к обсуждению вездесущего типа `NUMBER`. Далее мы раскроем типы `LONG` и `LONG RAW` главным образом с исторической точки зрения. Основная цель здесь — показать, как работать в приложениях с унаследованными столбцами `LONG` и переводить их в тип `LOB`. После этого мы углубимся в разнообразные типы данных для хранения значений даты и времени и уделим внимание манипулированию различными типами данных для получения нужного результата. Также будут описаны основы поддержки часовых поясов.

Следующими мы обсудим типы данных `LOB`. Мы рассмотрим способы их хранения и смысл каждого из многочисленных настроек, таких как `IN ROW`, `CHUNK`, `RETENTION`, `CACHE` и т.д. При работе с типами данных `LOB` важно понимать, как они реализованы и каким образом хранятся по умолчанию — особенно, когда дело доходит до настройки их извлечения и хранения. Глава завершается описанием типов `ROWID` и `UROWID`. Это специальные патентованные типы Oracle, которые представляют адрес строки. Мы объясним, когда их использовать в качестве типа данных столбца в таблице (что почти никогда не случается).

Глава 13. Секционирование

Секционирование предназначено для облегчения управления очень большими таблицами и индексами за счет реализации концепции "разделяй и властвуй"; в основном оно сводится к разбиению таблицы или индекса на множество мелких и более управляемых частей. Это область, в которой администратор базы данных и разработчик должны работать вместе с целью обеспечения максимальной доступности и производительности приложения. Здесь также подробно раскрыты функциональные средства, появившиеся в версиях Oracle 11*g* и Oracle 12*c*.

В главе рассматривается секционирование как таблиц, так и индексов. Будет описано секционирование с использованием локальных индексов (распространено в хранилищах данных) и глобальных индексов (распространено в системах OLTP).

Глава 14. Параллельное выполнение

В этой главе представлена концепция и случаи применения параллельного выполнения в Oracle. Мы начнем с рассмотрения ситуаций, когда параллельная обработка полезна и должна использоваться, а также ситуаций, когда она планироваться не должна. После этого мы перейдем к анализу механики параллельного запроса — средства, которое большинство людей ассоциирует с параллельным выполнением.

Затем мы опишем язык PDML (Parallel DML), который позволяет проводить модификации с использованием параллельного выполнения. Мы посмотрим, как язык PDML физически реализован, и почему эта реализация приводит к ряду ограничений, касающихся PDML.

Далее мы займемся исследованием параллельного DDL. По моему мнению, именно здесь параллельное выполнение проявляется во всей своей красе. Обычно администраторы баз данных располагают небольшими окнами обслуживания, в рамках которых должны выполнять крупные операции. Параллельный DDL предоставляет администраторам баз данных возможность в полной мере задействовать все доступные ресурсы компьютера, позволяя им завершать большие и сложные операции за какую-то долю времени, которое они отняли бы при последовательном выполнении.

Глава завершается рассмотрением процедурного параллелизма, т.е. средства, с помощью которого мы можем выполнять код приложения параллельно. Здесь раскрываются две технологии. Первая из них — параллельные конвейерные функции, или способность Oracle параллельно выполнять хранимые функции динамическим образом. Вторая технология — это "самодельный" параллелизм (do-it-yourself — DIY), когда приложение проектируется так, чтобы оно могло выполняться параллельно.

Глава 15. Загрузка и выгрузка данных

Внимание в первой половине этой главы сосредоточено на внешних таблицах — высокоэффективном средстве, с помощью которого осуществляется массовая загрузка и выгрузка данных. Если вы выполняете большой объем загрузки данных, то должны всерьез подумать о применении внешних таблиц. Здесь также подробно обсуждается средство предварительной обработки внешних таблиц, которое позволяет командам операционной системы автоматически выполняться как часть выборки из внешней таблицы.

Вторая половина главы посвящена SQL*Loader (SQLLDR) и разнообразным способам использования этого инструмента для загрузки и модификации данных в базе. Рассматриваются такие вопросы, как загрузка данных с разделителями, обновление существующих строк и вставка новых строк, выгрузка данных и обращение к SQLLDR из хранимой процедуры. SQLLDR — это тщательно продуманный и очень важный инструмент, но с его практическим применением связано немало вопросов.

Исходный код и обновления

Усваивать материал этой книги эффективнее всего путем тщательной проработки и разбора практических примеров. Работая с примерами, вы можете предпочесть вводить весь код вручную. Многие читатели выбирают такой подход потому, что он является хорошим способом ознакомиться с используемыми приемами написания кода.

Решите вы набирать код самостоятельно или нет, исходный код всех примеров, приведенных в настоящей книге, доступен на веб-сайте издательства. Если вы предпочитаете вводить код вручную, то файлы исходного кода можно применять для проверки ожидаемых результатов — это следует делать в первую очередь, когда есть подозрение, что код был набран с ошибкой. Если вам не нравится вводить код вручную, то без загрузки исходного кода с веб-сайта издательства не обойтись. В любом случае файлы кода помогут вам с обновлением и отладкой.