ГЛАВА 11

Возможность подключения и работа в сети

Одна из основных задач администратора БД — установка и поддержание возможности подключения между базой данных, действующей на сервере, и сообществом пользователей. В традиционной модели клиент-сервер пользователи подключаются к базам данных на отдельных серверах посредством клиента. Эта модель все еще применяется во многих организациях для выполнения производственных функций. Однако в настоящее время модели подключения на основе веб-технологий значительно чаще используются в качестве средств подключения к базам данных. Компания Oracle называет свой набор решений по обеспечению возможности подключения (которые охватывают возможность подключения, управляемость и вопросы сетевой безопасности) службами Oracle Net (ранее они назывались службами SQL Net) Services. В этой главе мы рассмотрим использование Oracle Net — компонента Oracle Net Services для создания и поддержания подключений между клиентами и базами данных. Мы рассмотрим также инсталляцию программного обеспечения клиента Oracle Client. Также будет показано, как использовать новый клиент Oracle Instant Client (Мгновенный клиент Oracle), который позволяет подключаться к базе данных Oracle без применения файлов конфигурации сети.

Oracle Database 11*g* предлагает несколько методов подключения серверов баз данных к конечным пользователям. Для небольших наборов пользователей можно использовать файл `tnsnames.ora`, который представляет собой локальный файл, содержащий сведения о подключении. Новый упрощенный метод подключения позволяет клиентам подключаться к базам данных без применения каких-либо файлов конфигурации. Наиболее сложный метод подключения, предоставляемый Oracle — метод именования каталогов, который использует LDAP-совместимую службу Oracle Internet Directory (OID). Помимо поддержания возможностей подключения к базе данных, OID можно использовать для управления параметрами обеспечения безопасности и других целей. Кроме того, существует внешний метод именования, который для конфигурирования клиентских подключений к базе данных Oracle использует внешние службы именования, такие как Network Information Service (Сетевая информационная служба).

В этой главе будет представлен также краткий обзор интерфейса JDBC (Java Database Connectivity — средство организации доступа Java-приложений к базам данных). Вы

узнаете, как подключаться к базе данных Oracle из Java-программ, и ознакомитесь с небольшим примером, который иллюстрирует основные концепции JDBC Oracle.

Сетевые средства Oracle и возможности подключения к базам данных

После создания базы данных и различных ее объектов, а также загрузки необходимых данных следующей серьезной задачей, которую потребуется выполнить, будет установка подключения между сервером БД и пользователями, которые будут ею пользоваться. Oracle Net Services (Сетевые службы Oracle) — набор служб, которые делают это возможным. Компоненты Oracle Net Services должны существовать как на клиенте, так и на сервере, и, как правило, они используют сетевой протокол TCP/IP для установки сетевых подключений между клиентами и сервером базы данных.

Конфигурирование Oracle Net Services выполняется с применением нескольких важных функциональных возможностей, призванных облегчить жизнь администраторам БД.

- *Прозрачность размещения.* Клиентам не обязательно знать сетевое размещение или какую-то иную требующую особых полномочий информацию о службах базы данных, поскольку всю информацию, необходимую для создания подключений к БД, можно поддерживать в централизованном хранилище. Пользователям предоставляется только имя базы данных, и подключение является полностью прозрачным для них.
- *Централизованное конфигурирование.* Для крупных систем централизованные средства установки и обслуживания подключений имеют огромный смысл. LDAP-совместимый сервер каталогов, поддерживаемый Oracle, предоставляет очень эффективное централизованное хранилище для удовлетворения всех сетевых нужд. Сведения о сети, аутентификации и другая информация, связанная с безопасностью, сохраняется в централизованном месте, к которому затем пользователи обращаются за этой информацией. Обслуживание при этом чрезвычайно просто, поскольку независимо от количества клиентов приходится поддерживать только централизованную информацию.
- *Масштабируемость.* Для улучшения масштабируемости Oracle предоставляет специализированную архитектуру под названием *архитектуры разделяемого сервера*. Эта архитектура позволяет нескольким пользователям через процесс диспетчера совместно использовать один и тот же процесс подключения. Небольшое количество подключений к серверу позволяет работать в системе большому количеству конечных пользователей, что увеличивает масштабируемость системы. Кроме того, Oracle предлагает средство Connection Manager (Диспетчер подключений), которое обеспечивает мультиплексирование подключений, когда несколько подключений обслуживаются одновременно.

Сравнение архитектур разделяемого сервера и выделенного сервера

Архитектуру подключения можно определить так, чтобы сервер Oracle запускал отдельный процесс сервера для каждого клиентского подключения, либо чтобы несколько клиентов совместно использовали единственный процесс сервера. Отдельные процессы сервера используют выделенные подключения между каждым клиентом и сервером Oracle, и именно поэтому такая архитектура получила название *архитектуры выделенного сервера*. *Архитектура разделяемого сервера* — термин, обозначающий архитектуру, при которой несколько процессов пользователей используют одно и то же подключение к серверу Oracle для выполнения своей работы.

Архитектура разделяемого сервера

При обработке запросов клиентов на обработку подключения архитектура разделяемого сервера использует службу диспетчера. Один диспетчер может одновременно обслуживать множество клиентских подключений. По существу диспетчеры действуют в качестве посредника между клиентами и разделяемыми серверами. Диспетчеры отвечают за прием запросов от клиентов и помещение их в очередь запросов, из которых их выбирает разделяемый сервер.

При использовании диспетчера (т.е. при использовании подхода с применением разделяемого сервера) прослушивающий процесс будет передавать запрос на подключение не непосредственно серверу базы данных, а диспетчеру. Это процесс называют *прямой передачей* диспетчеру. Прослушивающий процесс может также перенаправить клиентское подключение диспетчеру. В этом случае прослушивающий процесс будет передавать клиентскому подключению сетевой адрес диспетчера, и эта информация позволяет клиенту подключиться к диспетчеру, в то время как подключение к прослушивающему процессу разорвано.

Диспетчер подключений Oracle (Oracle Connection Manager) можно использовать для настройки мультиплексирования сеансов, которое подразумевает объединение в пул нескольких клиентских сеансов по сети в местоположение разделяемого сервера.

Архитектура выделенного сервера

Процессы выделенного сервера не подразумевают никакого совместного использования ресурсов клиентами. Каждому клиенту присваивается подключение к выделенному серверу. Прослушивающий процесс Oracle будет запускать процесс выделенного сервера при каждом запросе на подключение со стороны клиентского процесса. Прослушивающий процесс передает адрес протокола выделенного сервера клиенту; затем клиент использует его для непосредственного подключения к серверу базы данных. Подключение к прослушивающему процессу разрывается, как только он передает адрес выделенного сервера клиенту.

В этой главе рассмотрена только более часто используемая архитектура выделенного сервера. Для ознакомления с установкой конфигурации разделяемого сервера обратитесь к руководству по организации сетей Oracle, *Net Services Administrator's Guide* (Руководство администратора сетевых служб).

Концепции сетевого обмена: работа сетевых средств Oracle

Когда требуется открыть сеанс базы данных с клиента — будь то традиционный клиент или клиент на основе браузера — нужно подключиться к базе данных по сети. Предположим, что по существующей сети устанавливается подключение настольного компьютера к базе данных Oracle, расположенной на сервере UNIX в другой части города. Для этого требуется метод установки соединения между настольным компьютером и базой данных Oracle (предполагающий использование специализированного программного обеспечения), какой-либо интерфейс для проведения сеанса (в данном примере им будет SQL*Plus) и какой-нибудь способ обмена данными со стандартными сетевыми протоколами, такими как TCP/IP.

Для облегчения конфигурирования и управления сетевыми подключениями Oracle предлагает набор служб Oracle Net Services, представляющий собой комплект компонентов, которые предоставляют решения для обеспечения возможности подключения в распределенных гетерогенных компьютерных средах. Набор Oracle Net Services состоит из служб Oracle Net (Сеть Oracle), Oracle Net Listener (Служба прослушивания сети Oracle), Oracle Connection Manager (Диспетчер подключений Oracle), Oracle Net Configuration Assistant (Помощник по конфигурированию сети Oracle) и Oracle Net Manager (Диспетчер

сети Oracle). Программное обеспечение Oracle Net Services устанавливается автоматически в процессе инсталляции программного обеспечения Oracle Database Server (Сервер баз данных Oracle) или Oracle Client (Клиент Oracle).

Oracle Net — программный компонент, который инициализирует, устанавливает и поддерживает подключения между клиентами и серверами. Поэтому компонент Oracle Net должен быть установлен как на клиенте, так и на сервере. Oracle Net состоит из следующих двух основных компонентов.

- *Oracle Network Foundation Layer* (Базовый сетевой уровень Oracle). Отвечает за установку и поддержание подключения между клиентским приложением и сервером, а также за обмен сообщениями между ними
- *Oracle Protocol Support* (Поддержка протокола Oracle). Отвечает за отображение функциональности TNS (Transparent Network Substrate — Прозрачная сетевая среда) на стандартные промышленные протоколы, используемые при подключениях.

Все серверы, содержащие базу данных Oracle, выполняют также службу Oracle Net Listener, слушатель сети Oracle (обычно называемый просто *слушателем*), основная функция которой — прослушивание запросов клиентов на вход в базу данных Oracle. Убедившись, что служба клиента обладает соответствующей информацией базы данных (протоколом, портом и именем экземпляра), слушатель передает запрос клиента базе данных. База данных позволит клиенту выполнить вход, если, конечно, подлинность его имени пользователя и пароля подтверждаются. Как только слушатель передаст запрос пользователя базе данных, клиент и БД окажутся в непосредственном контакте, не требующем никакой помощи со стороны службы прослушивания.

Oracle предлагает ряд утилит, построенных на основе графического интерфейса пользователя, которые призваны облегчить конфигурирование подключений к базам данных. В число этих утилит входит Oracle Connection Manager (Диспетчер подключений Oracle), Oracle Net Manager (Диспетчер сети Oracle) и Oracle Net Configuration Assistant (Помощник по конфигурированию сети Oracle). Названные программные средства могут помочь в удовлетворении всех потребностей, связанных с работой в сети. По завершении прочтения этой главы просто щелкните на пиктограммах этих программ и начните экспериментировать с тестовыми подключениями.

Процесс подключения веб-приложения к Oracle

Чтобы подключиться через Интернет к базе данных Oracle, веб-браузер на клиентском компьютере связывается с веб-сервером и выдает запрос на подключение посредством протокола HTTP. Веб-сервер передает запрос приложению, которое обрабатывает его и связывается с сервером базы данных Oracle посредством службы Oracle Net (сконфигурированной как на сервере базы данных, так и на компьютере клиента). В последующих разделах мы рассмотрим ряд важных терминов, играющих исключительно важную роль в сетевой организации Oracle.

Имена экземпляров базы данных

Как вы уже знаете, экземпляр Oracle состоит из области SGA и набора процессов Oracle. *Имя экземпляра базы данных* указывается в файле инициализации (`init.ora`) в виде параметра `INSTANCE_NAME`. Когда речь идет о системном идентификаторе (SID) Oracle, подразумевается просто экземпляр Oracle.

Обычно каждая база данных может иметь только один связанный с ней экземпляр. Однако в конфигурации Oracle RAC (Real Application Clusters — Кластеры реальных приложений) одна база данных может быть связана с несколькими экземплярами.

Глобальные имена баз данных

Глобальное имя базы данных уникальным образом идентифицирует базу данных Oracle и имеет формат *имя_базы_данных.домен_базы_данных* — например, `sales.us.acme.com`. В этом примере глобального имени базы данных `sales` — имя базы данных, а `us.acme.com` — домен базы данных. Поскольку никакие две базы данных в одном домене не могут иметь одинаковые имена, каждое глобальное имя базы данных уникально.

Имена служб базы данных

С логической точки зрения для клиента база данных выглядит просто службой. Службы и базы данных связаны между собой отношениями типа "многие ко многим", поскольку база данных может быть представлена одной или более службами, каждая из которых предназначена для своего набора клиентов, а служба может охватывать более одного экземпляра базы данных. В системе каждая база данных идентифицируется по ее имени службы, для указания которого служит параметр инициализации `SERVICE_NAMES`. По умолчанию значение имени службы устанавливается соответствующим глобальному имени базы данных.

Обратите внимание, что база данных может адресоваться более чем по одному имени службы. Это может быть реализовано, если нужно, чтобы различные наборы клиентов по-разному адресовались к базе данных для удовлетворения их конкретных потребностей. Например, для одной и той же базы данных можно создать два имени служб наподобие следующих:

```
sales.us.acme.com
finance.us.acme.com
```

Продавцы будут использовать имя службы `sales.us.acme.com`, а сотрудники бухгалтерии и финансового отдела — имя службы `finance.us.acme.com`.

Дескрипторы соединений

Чтобы подключиться к любой службе базы данных в мире со своего настольного компьютера, необходимо предоставить два информационных элемента:

- имя службы базы данных;
- местоположение адреса.

Термин *дескриптор соединения* в Oracle используют для обозначения объединенной спецификации двух обязательных компонентов подключения к базе данных: имени службы базы данных и ее адреса. Часть адреса дескриптора соединения содержит три компонента: коммуникационный протокол, используемый для соединения, имя хоста и номер порта.

Знание коммуникационного протокола облегчает согласование сетевых протоколов, что позволяет установить соединение. Стандартным протоколом является TCP/IP или TCP/IP с SSL (Secure Sockets Layer — уровень защищенных сокетов). На серверах UNIX стандартный номер порта для подключений Oracle — 1521 или 1526. На компьютерах Windows по умолчанию используется порт 1521. Поскольку любой хост не может содержать более одной базы данных с одним и тем же именем службы, имя службы базы данных Oracle и имя хоста будут уникальным образом идентифицировать любую базу данных в мире. Ниже приведен пример типичного дескриптора соединения:

```
(DESCRIPTION
  (ADDRESS=(PROTOCOL=tcp) (HOST=sales-server) (PORT=1521))
  (CONNECT_DATA=
    (SERVICE_NAME=sales.us.acme.com)))
```

В этом дескрипторе соединения строка ADDRESS указывает, что протокол TCP будет использоваться для сетевого обмена данными. HOST определяет сервер UNIX (или Windows), на котором процесс прослушивания Oracle прослушивает запросы на подключение в конкретном порту: 1521. Часть ADDRESS дескриптора соединения называют также *адресом протокола*.

Клиент, желающий подключиться к базе данных, вначале подключается к процессу слушателя Oracle. Слушатель принимает входящие запросы на подключение и передает их серверу базы данных. Как только клиент и сервер базы данных связываются посредством слушателя, они оказываются на прямой связи и слушатель больше не принимает никакого дальнейшего участия в процессе обмена данными для подключения этого клиента.

Идентификаторы соединений

Идентификатор соединения тесно связан с дескриптором соединения. Дескриптор соединения можно использовать в качестве идентификатора соединения или же можно просто отобразить имя службы базы данных на дескриптор соединения. Например, такое имя службы, как sales, можно отобразить на дескриптор соединения, приведенный в предыдущем разделе. Ниже показан пример, демонстрирующий отображение идентификатора соединения sales:

```
sales=
(DESCRIPTION
   (ADDRESS=(PROTOCOL=tcp) (HOST=sales-server) (PORT=1521))
   (CONNECT_DATA=
   (SERVICE_NAME=sales.us.acme.com)))
```

Строки соединений

Подключение к базе данных выполняется путем указания *строки соединения*. Строка соединения содержит сочетание имени пользователя и пароля и идентификатор соединения. Один из наиболее часто используемых идентификаторов соединения — имя сетевой службы, которое представляет собой простое имя службы базы данных.

Следующий пример демонстрирует строку соединения, которая использует полный дескриптор подключения в качестве идентификатора соединения:

```
CONNECT scott/tiger@(DESCRIPTION=
   (ADDRESS=(PROTOCOL=tcp)
   (HOST=sales_server1)
   (PORT=1521))
   (CONNECT_DATA=(SERVICE_NAME=sales.us.acme.com)))
```

К этой же базе данных можно подключиться значительно проще, используя идентификатор соединения sales:

```
CONNECT scott/tiger@sales
```

Оба приведенных примера установят соединение с базой данных sales, но вторая строка соединения (использующая идентификатор соединения sales) явно значительно проще.

Использование инструментальных средств Oracle Net Services

Oracle Net предоставляет несколько графических интерфейсов пользователей и утилит командной строки, позволяющих конфигурировать соединения между клиентами и службами базы данных. Вероятно, наиболее часто придется пользоваться утилитой командной строки lsnrctl, которая помогает управлять службой Oracle Net Listener. Ниже перечислены наиболее важные средства с

графическим интерфейсом пользователя, которые помогают в управлении службами Oracle Net Services.

- *Oracle Net Configuration Assistant* (Помощник по конфигурированию сети Oracle), или NCA. В основном это средство применяют для конфигурирования сетевых компонентов во время установки, и оно позволяет выбрать одну из доступных опций (они рассмотрены позднее в этой главе) для конфигурирования характеристик подключения клиента. Простой в использовании графический интерфейс позволяет быстро конфигурировать подключения клиентов в соответствии с любым выбранным методом именования. В системах UNIX/Linux NCA можно запустить, запуская файл `netca` из каталога `$ORACLE_HOME/bin`. В среде Windows для этого потребуется выбрать команду меню Start⇨Programs⇨Oracle-HOME_NAME⇨Configuration and Migration Tools⇨Net Configuration Assistant (Пуск⇨Программы⇨Имя домашнего каталога Oracle⇨Средства конфигурирования и переноса⇨Помощник по конфигурированию сети).
- *Oracle Net Manager* (Диспетчер сети Oracle). Инструмент Oracle Net Manager может выполняться на клиентах и серверах, и он позволяет конфигурировать различные методы именования и слушателей. С помощью этого средства можно конфигурировать дескрипторы соединений в локальных файлах `tnsnames.ora` или в централизованном OID, а также легко добавлять и изменять методы подключения.

  Чтобы запустить Oracle Net Manager с консоли Oracle Enterprise Manager, выберите команду меню Tools⇨Service Management⇨Oracle Net Manager (Сервис⇨Управление службами⇨Диспетчер сети Oracle). Чтобы запустить его в качестве самостоятельного приложения в среде UNIX, запустите файл `netmgr` из каталога `$ORACLE_HOME/bin`. В среде Windows для этого потребуется выбрать команду меню Start⇨Programs⇨Oracle-HOME_NAME⇨Configuration and Migration Tools⇨Net Manager (Пуск⇨Программы⇨Имя домашнего каталога Oracle⇨Средства конфигурирования и переноса⇨Диспетчер сети).
- *Oracle Enterprise Manager* (Диспетчер предприятия Oracle). В Oracle Database 11*g* OEM может выполнять все, что выполняет Oracle Net Manager, но для нескольких домашних каталогов Oracle, размещенных в нескольких файловых системах. Кроме того, используя OEM, записи именования каталогов можно экспортировать в файл `tnsnames.ora`.
- *Oracle Directory Manager* (Диспетчер каталогов Oracle). Это мощное средство позволяет создавать различные и домены и контексты, необходимые для использования OID. С его помощью можно также управлять политикой использования паролей и решать многие более сложные задачи обеспечения безопасности Oracle. В системах UNIX/Linux OID можно запустить, запуская файл `oidadmin` из каталога `$ORACLE_HOME/bin`. В среде Windows для этого потребуется выбрать команду меню Start⇨Programs⇨Oracle-HOME_NAME⇨Integrated Management Tools⇨Oracle Directory Manager (Пуск⇨Программы⇨Имя домашнего каталога Oracle⇨Встроенные средства управления⇨Диспетчер каталогов Oracle).

Установка соединения Oracle

Чтобы подключиться к базе данных Oracle через сеть, вначале нужно установить сетевое соединение между клиентом и сервером. На компьютере, с которого выполняется подключение, должно быть установлено программное обеспечение клиента Oracle (Oracle Client) либо сервера баз данных Oracle (Oracle Database Server). Инсталляция программного обеспечения Oracle Client рассматривается в следующем разделе.

Чтобы успешно установить соединение, понадобится выполнить следующие действия.

1. Убедитесь, что сервер базы данных установлен, а экземпляр Oracle запущен.
2. Удостоверьтесь, что программное обеспечение клиента Oracle Client установлено на клиентском компьютере.

3. Проверьте, что сервер базы данных и клиент действуют в одной сети. Выполните эту проверку, с помощью команды `ping`:

```
C:\> ping prod1
Pinging prod1.netbsa.org [172.14.152.1] with 32 bytes of data:
Reply from 172.14.152.1: bytes=32 time<1ms TTL=255
Reply from 172.14.152.1: bytes=32 time<1ms TTL=255
Reply from 172.14.152.1: bytes=32 time<1ms TTL=255
Reply from 172.14.152.1: bytes=32 time<1ms TTL=255
Ping statistics for 172.14.152.1:
   Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
   Minimum = 0ms, Maximum = 0ms, Average = 0ms
C:\>
```

   Результат выполнения команды `ping` показывает, что соединение успешно установлено. Если соединение не может быть установлено, отобразится сообщение о просроченном запросе на подключение, а количество отправленных пакетов данных в статистике команды `ping` будет больше количества полученных пакетов.

4. Протокол TCP/IP должен быть установлен и на сервере, и на клиенте. При установке программного обеспечения сервера и клиента Oracle эти протоколы устанавливаются в форме программного компонента Oracle Net.

5. Убедитесь, что служба Oracle Net Listener запущена на сервере и прослушивает соответствующий порт на предмет входящих запросов на подключение.

6. Сконфигурируйте клиента для подключения к базе данных. К базе данных Oracle можно подключиться одним из четырех доступных методов: локальное именование, простое именование подключения, именование каталога и внешнее именование. Эти методы рассматриваются далее в этой главе.

7. Подключитесь к базе данных с помощью интерфейса SQL*Plus или программного средства от независимого разработчика. Например, с помощью SQL*Plus к базе данных можно подключиться, указывая сочетание имени пользователя и пароля вместе с именем базы данных:

```
C:\> sqlplus system/sammyy1@orcl
SQL*Plus: Release 11.1.0.6.0 - Production on Thu Mar 20 09:25:27 2008
Copyright (c) 1982, 2007, Oracle.  All rights reserved.

Connected to:
Oracle Database 11g Enterprise Edition Release 11.1.0.6.0 - Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL>
```

В последующих разделах мы подробнее рассмотрим программное обеспечение Oracle Client, слушателя и методы именования.

Oracle Client

Чтобы получить доступ к базе данных Oracle с ПК, вначале на нем необходимо инсталлировать программное обеспечение Oracle Client. Программное обеспечение Oracle Client поставляется вместе с программным обеспечением Oracle Server. Кроме того, его можно загрузить также из сайта OTN (`http://technet.oracle.com`). Программное обеспечение Oracle Client доступно для загрузки отдельно. Хотя версии Oracle Server и Oracle Client не обязательно должны совпадать, в Oracle рекомендуют применять соответствующие версии ПО, чтобы можно было полностью воспользоваться преимуществами новых функциональных возможностей.

Версию Oracle Client можно выяснить, просмотрев вывод при вызове утилиты SQL*Plus, как показано в следующем примере:

```
$ sqlplus
C:\>sqlplus

SQL*Plus: Release 11.1.0.6.0 - Production on Thu Mar 20 09:27:14 2008

Copyright (c) 1982, 2007, Oracle.  All rights reserved.

Enter user-name:
```

Приведенный вывод команды SQL*Plus показывает, что в системе установлено программное обеспечение Oracle Client версии 11.1.0.6.0.

При установке программного обеспечения Oracle Client можно выбрать одну из четырех опций.

- *Administrator* (Администратор). Позволяет приложениям подключаться к локальным или удаленным базам данных Oracle и управлять ими.
- *Runtime* (Компоненты времени выполнения). Позволяет подключаться к локальным или удаленным базам данным Oracle.
- *Custom* (Пользовательская установка). Позволяет выбирать отдельные компоненты из числа устанавливаемых при вариантах установки Administrator и Runtime.
- *Instant Client* (Мгновенный клиент). Производится инсталляция только библиотек совместного использования, необходимых для OCI (Oracle Call Interface — Интерфейс вызова Oracle), OCCI (Oracle C++ Call Interface — Интерфейс вызова C++ Oracle) и приложениям Java Database Connectivity OCI (Java-OCI подключения к базе данных).

На заметку! Новая опция Instant Client описана в разделе “Мгновенный клиент”.

Инсталляция Oracle Client

Чтобы установить программное обеспечение Oracle Client, выполните следующие действия.

1. Вставьте компакт-диск Oracle Database 11g Client в привод или запустите сценарий `runInstaller` из промежуточного каталога, как описано в главе 9.
2. Выберите пункт меню Install/Deinstall Products (Установка и удаление программных компонентов) и щелкните на кнопке Next (Далее).
3. Откроется Welcome (Приветствие). Щелкните на кнопке Next.
4. На странице Specify File Locations (Укажите расположение файлов) примите заданный по умолчанию каталог файлов или введите имя и путь домашнего каталога Oracle. Щелкните на кнопке Next.
5. На экране Select Installation Type (Выберите тип инсталляции) можно выбрать один из четырех вариантов — Instant Client, Administrator, Runtime или Custom. Выберите из списка опцию Runtime Installation (Установка компонентов времени выполнения) и щелкните на кнопке Next.
6. Просмотрите компоненты, которые будут инсталлированы в процессе этого типа установки и щелкните на кнопке Install (Установить).
7. После завершения установки программного обеспечения Oracle Client откроется помощник Oracle Net Configuration Assistant. Выберите опцию No, I Will Create

Service Names Myself (Нет, я создам имена служб самостоятельно) и щелкните на кнопке Next.

8. В поле Database SID (Системный идентификатор базы данных) введите имя базы данных и щелкните на кнопке Next.
9. В качестве протокола выберите TCP и щелкните на кнопке Next.
10. В поле Host Name (Имя хоста) введите имя сервера своего хоста и выберите стандартный порт. Щелкните на кнопке Next.
11. Щелкните на кнопке Yes (Да), чтобы протестировать возможность подключения, а затем щелкните на кнопке Next.
12. После отображения сообщения Connecting . . . Test Successful (Подключение . . . Тестирование выполнено успешно) щелкните на кнопке Next.
13. Отвечая на вопрос о том, нужно ли конфигурировать другую службу, выберите опцию No (Нет). Щелкните на кнопке Next.
14. Подтвердите завершение настройки имени сетевой службы, щелкнув на кнопке Next.
15. Щелкните на кнопках Finish (Готово) и Exit (Выход).

Совет. При наличии нескольких установок Oracle на ПК система может содержать также несколько файлов `tnsnames.ora`. Пользователь может оказаться не в состоянии подключиться к новой базе данных после добавления информации о конфигурации сети в файл `tnsnames.ora`, если используется не этот файл. Убедитесь, что в пути ПО Oracle Client указан нужный файл `tnsnames.ora`.

Использование переменной среды TWO_TASK

Использование имени Oracle Net можно обойти, устанавливая переменную среды `TWO_TASK` (в системе UNIX/Linux) или переменную среды `LOCAL` (в системе Windows).

Переменная `TWO_TASK` указывает строку подключения для подключения к удаленному компьютеру. Программа SQL*Plus проверит значение переменной среды `TWO_TASK` и автоматически добавит его в строку подключения, как показано в следующем примере:

```
$ export TWO_TASK=mydb
```

Как только переменная среды `TWO_TASK` установлена, к базе данных `mydb` можно подключаться следующим образом:

```
$ sqlplus scott/tiger
```

Обратите внимание, что применять спецификацию `sqlplus scott/tiger@mydb` не пришлось, поскольку используется переменная `TWO_TASK`.

На сервере Windows следующие команды эквивалентны установке переменной среды `TWO_TASK`:

```
$ SET LOCAL=<mydb>
$ sqlplus scott/tiger
```

Мгновенный клиент

Описанный в предыдущем разделе процесс инсталляции ПО Oracle Client требует выполнения подготовительных действий, необходимых для установки регулярного программного обеспечения Oracle Database Server.

К счастью, инсталляция полного программного обеспечения Oracle Client для подключения к базе данных Oracle может требоваться не всегда. Новое программное обеспечение Instant Client Oracle позволяет запускать приложения без установки стандартного ПО Oracle Client или наличия каталога `ORACLE_HOME`. Программное обеспечение Oracle Client не обязательно инсталлировать на каждом компьютере, которому необходим доступ к базе данных Oracle. Все существующие приложения OCI, ODBC и JDBC будут работать с Instant Client (Мгновенный клиент). При желании с ним можно даже использовать интерфейс SQL*Plus.

Instant Client предоставляет следующие преимущества по сравнению с полномасштабным ПО Oracle Client.

- Является бесплатным.
- Занимает меньше дискового пространства.
- Инсталляция выполняется быстрее (приблизительно за пять минут).
- Не требуется компакт-диск.
- Предоставляет все функциональные возможности стандартного программного обеспечения Oracle Client, включая поддержку интерфейса SQL*Plus при необходимости.

Инсталляция Instant Client

Чтобы установить новое программное обеспечение Instant Client и быстро подключиться к базе данных Oracle, выполните следующие действия.

1. Загрузите программное обеспечение Instant Client из веб-сайта OTN. Пакет клиента Basic (Базовый) обязателен для установки. Кроме него можно установить любые дополнительные необязательные пакеты. Пакеты содержат перечисленные ниже элементы.
   - *Basic* (Базовый). Файлы, необходимые для выполнения приложений OCI, OCCI и JDBC-OCI.
   - *SQL*Plus*. Дополнительные библиотеки и исполняемые файлы для запуска SQL*Plus с поддержкой Instant Client.
   - *JDBC Supplemen* (Дополнение JDBC). Дополнительная поддержка XA (eXtended Architecture — расширенная архитектура), Internationalization (Интернационализация) и операций RowSet под управлением JDBC.
   - *ODBC Supplement* (Дополнение ODBC). Дополнительные библиотеки для активизации использования приложений ODBC с поддержкой Instant Client (только в системах Windows).
   - *SDK*. Дополнительные файлы для разработки приложений Oracle с поддержкой Instant Client.
2. Распакуйте выбранные пакеты в один каталог и дайте ему соответствующее имя, например, “`instantclient`”.
3. В системах UNIX и Linux установите переменную среды `LD_LIBRARY_PATH` в `instantclient` (тем самым обеспечив соответствие параметра имени каталога, содержащего пакеты). В системах Windows установите переменную среды `PATH` в `instantclient`.
4. Протестируйте подключение к серверу Oracle.

Слушатель и подключаемость

Слушатель Oracle Net Listener — служба, которая действует только на сервере и прослушивает входящие запросы на подключение. Oracle предоставляет утилиту `lsnrctl`, управляющую процессом слушателя. Место слушателя в сетевой обработке Oracle можно кратко описать следующим образом.

- С помощью слушателя база данных регистрирует информацию о службах, экземплярах и обработчиках служб.
- Клиент устанавливает начальное соединение со слушателем.
- Слушатель принимает и проверяет запрос на подключение клиента и передает его обработчику службы базы данных. Как только слушатель передает запрос клиента, он устраняется из процесса обслуживания данного подключения.

Файл `listener.ora`, который по умолчанию размещается в каталоге `$ORACLE_HOME/network/admin` в системах UNIX и в каталоге `$ORACLE_HOME\network\admin` в системах Windows, содержит информацию о конфигурации слушателя. Поскольку служба слушателя действует только на сервере, клиентские компьютеры не содержат никакого файла `listener.ora`. Типичный файл `listener.ora` приведен в листинге 11.1.

Все параметры конфигурации в этом файле имеют значения по умолчанию. Поэтому службу слушателя не обязательно конфигурировать вручную. После создания первой базы данных на сервере служба слушателя автоматически запускается, и файл конфигурации слушателя, `listener.ora`, помещается в каталог, определенный по умолчанию. При создании новой базы данных ее информация о сетевых подключений и службах автоматически добавляется в файл конфигурации слушателя. При запуске экземпляра база данных автоматически регистрируется в слушателе, и слушатель начинает прослушивать запросы на подключение к этой базе данных.

Листинг 11.1. Типичный файл конфигурации слушателя

```
# Сетевой Файл конфигурация LISTENER.ORA:
/u01/app/oracle/product/11.1.0.6.0 /db_1/network/admin/listener.ora
# Сгенерирован средствами конфигурирования Oracle.
SID_LIST_LISTENER =
   (DESCRIPTION_LIST =
      (DESCRIPTION =
         (ADDRESS_LIST =
            (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC4))
         )
         (ADDRESS_LIST =
            (ADDRESS = (PROTOCOL = TCP)(HOST = NTL-ALAPATISAM)(PORT = 1521))
         )
      )
   )
SID_LIST_LISTENER =
   (SID_LIST =
      (SID_DESC =
         (SID_NAME = PLSExtProc)
         (ORACLE_HOME = /u01/app/oracle/product/11.1.0/db_1)
         (PROGRAM = extproc)
      )
      (SID_DESC =
         (GLOBAL_DBNAME = remorse.world)
         (ORACLE_HOME = /u01/app/oracle/product/11.1.0/db_1)
         (SID_NAME = remorse)
      )
```

```
    (SID_DESC =
      (GLOBAL_DBNAME = finance.world)
      (ORACLE_HOME = /u01/app/oracle/product/11.1.0/db_1)
      (SID_NAME = finance)
    ) )
```

Процесс PMON Oracle отвечает за динамическую регистрацию имен служб баз данных Oracle в слушателе — при создании новые базы данных Oracle будут автоматически регистрироваться в службе слушателя. Процесс PMON будет обновлять файл `listener.ora` после создания каждой базы данных на сервере.

Для обеспечения возможности автоматической регистрации файл `init.ora` или SPFILE должен содержать следующие параметры:

- SERVICE_NAMES (имена служб), например, `sales.us.oracle.com`
- INSTANCE_NAME (имя экземпляра), например, `sales`

Если значение параметра SERVICE_NAMES не указано, по умолчанию ему присваивается значение глобального имени базы данных, являющееся сочетанием параметров DB_NAME и DB_DOMAIN. Значение параметра INSTANCE_NAME, устанавливаемое по умолчанию — идентификатор SID, введенный во время установки Oracle или создания базы данных.

Состояние слушателя на сервере можно проверить с помощью утилиты `lsnrctl`, как показано в листинге 11.2. Вывод показывает длительность работы слушателя и размещение файла конфигурации службы слушателя. Он содержит также имена баз данных, которые слушатель "прослушивает" на предмет запросов на подключение.

Листинг 11.2. Использование утилиты `lsnrctl` для проверки состояния слушателя

```
 $ lsnrctl status
C:\>lsnrctl status

LSNRCTL for 32-bit Windows: Version 11.1.0.6.0 - Production on 20-MAR-2008

Copyright (c) 1991, 2007, Oracle. All rights reserved.

Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=ntl-alapatisam.netbsa.
or
g)(PORT=1522)))
STATUS of the LISTENER
------------------------
Alias                     LISTENER
Version                   TNSLSNR for 32-bit Windows: Version 11.1.0.6.0 - Produ
ction
Start Date                03-MAR-2008 11:15:53
Uptime                    16 days 21 hr. 14 min. 27 sec
Trace Level               off
Security                  ON: Local OS Authentication
SNMP                      OFF
Listener Parameter File   c:\orcl\app\oracle\product\11.1.0\db_1\network\admin\
listener.ora
Listener Log File         c:\orcl11\app\oracle\diag\tnslsnr\ntl-alapatisam\listener\
alert\log.xml
Listening Endpoints Summary...
   (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=ntl-alapatisam.netbsa.org)(PORT=1522
)))
   (DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(PIPENAME=\\.\pipe\EXTPROC1522ipc)))
Services Summary...
```

```
Service "orcl" has 1 instance(s).
   Instance "orcl", status READY, has 1 handler(s) for this service...
Service "orclXDB" has 1 instance(s).
   Instance "orcl", status READY, has 1 handler(s) for this service...
Service "orcl_XPT" has 1 instance(s).
   Instance "orcl", status READY, has 1 handler(s) for this service...
The command completed successfully
C:\>
```

Состояние в разделе `Services Summary` (Сводка по службам) листинга 11.2 может принимать одно из следующих значений:

- `READY` — экземпляр может принимать подключения;
- `BLOCKED` — экземпляр не может принимать подключения;
- `UNKNOWN` — экземпляр зарегистрирован в файле `listener.ora`, а не посредством динамической регистрации служб, поэтому его состояние не известно.

Команды слушателя

После вызова утилиты `lsnrctl` помимо `status` можно выполнять и другие важные команды. Например, команда `services` позволяет выяснить, какие службы слушатель отслеживает на предмет запросов на подключение.

На заметку! Состояние службы слушателя можно проверить из страницы Net Services Administration (Администрирование сетевых служб) в Oracle Enterprise Manager.

Ознакомиться с доступными командами утилиты `lsnrctl` можно с помощью команды `help`, введенной в интерфейсе `lsnrctl`, как показано в листинге 11.3.

Листинг 11.3. Использование команды `help` утилиты `lsnrctl` для вывода списка допустимых команд

```
$ lsnrctl help

LSNRCTL for 32-bit Windows: Version 11.1.0.6.0 - Production on 20-MAR-2008
Copyright (c) 1991, 2007, Oracle. All rights reserved.

The following operations are available
An asterisk (*) denotes a modifier or extended command:

start                stop                status
services             version             reload
save_config          trace               change_password
quit                 exit                set*
show*$
```

После вызова утилиты `lsnrctl` запуск слушателя можно осуществить с помощью команды `start`, а его остановку — с помощью команды `stop`. Если эти команды требуется выдать из командной строки операционной системы, можно использовать команды `lsnrctl start` и `lsnrctl stop`.

При внесении изменений в файл `listener.ora` единственный способ ввода этих изменений в действие — перезапуск слушателя. Другой, более безопасный способ — просто перезагрузка информации слушателя, в результате чего последние выполненные изменения слушателя будут внесены в файл конфигурации. Команда `lsnrctl reload` позволяет перезагрузить слушателя “на лету”, без его перезапуска. Подключенные в

текущий момент клиенты останутся подключенными во время перезагрузки слушателя (или даже при его перезапуске), поскольку слушатель уже "отдал" подключения базе данных и не участвует в обмене данными между клиентом и службой базы данных.

Внимание! Изменять файл `listener.ora` не рекомендуется, если только для этого не существует веской причины. Кроме того, при использовании динамической автоматической регистрации служб необходимость в модификации файла возникает значительно реже. Тем не менее, в некоторых случаях приходится изменять определенную часть файла слушателя, содержащую информацию о конфигурации сети для всех служб, которые слушатель отслеживает на предмет запросов подключения.

Управление слушателем

Хотя установка службы слушателя достаточно проста, после ее выполнения можно предпринять ряд действий для более точной настройки процесса подключения и для обеспечения безопасности службы слушателя. Некоторые из этих параметров описаны в последующих разделах.

Несколько слушателей

На одном сервере могут действовать более одной службы слушателя, но обычно такую конфигурацию придется применять в Oracle RAC. При использовании нескольких служб слушателя можно конфигурировать параметр `CONNECT_TIME_FAILOVER`, который определяет длительность ожидания клиентом подключения через одного слушателя, прежде чем будет предпринята попытка подключения с помощью другого слушателя.

Установка размера очереди

Иногда большое количество одновременных запросов на подключение со стороны клиентов могут перегрузить службу слушателя. Чтобы уберечь слушателя от сбоя, можно использовать параметр `QUEUESIZE` в файле конфигурации `listener.ora` для указания допустимого количества параллельно выполняющихся запросов на подключение.

Для большинства операционных систем значение параметра `QUEUESIZE` — достаточно небольшое число, наподобие 5. Ниже приведен пример установки параметра `QUEUESIZE`:

```
LISTENER=
 (DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521)(QUEUESIZE=10)))
```

Установка пароля для слушателя

При первоначальной установке слушателя утилита не имеет никакой защиты паролем. Любой пользователь, имеющий доступ к операционной системе, без труда может остановить слушателя и воспрепятствовать клиентам подключаться, просто введя команду **`lsnrctl stop`** в командной строке.

На заметку! Установленный по умолчанию пароль службы слушателя — `listener`, и при использовании слушателя этот пароль указывать не нужно.

Собственный пароль для утилиты слушателя можно установить, как показано в листинге 11.4.

Листинг 11.4. Установка пароля для слушателя

```
LSNRCTL> set password
Password:
The command completed successfully
LSNRCTL> change_password
Old password:
New password:
Reenter new password:
Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=ntl-alapatisam.netbsa.org
)(PORT=1521)))
Password changed for LISTENER
The command completed successfully
LSNRCTL> save_config
Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=ntl-alapatisam.netbsa.org
)(PORT=1521)))
Saved LISTENER configuration parameters.
Listener Parameter File
/u01/app/oracle/product/11.1.0/db_1/network/admin/listener.ora
Old Parameter File /u01/app/oracle/product/11.1.0/db_1/network/admin/listener.bak
```

После того как пароль успешно изменен, службу слушателя нельзя будет останавливать или запускать как раньше — для этого придется ввести пароль пользователя. Для указания слушателю нового пароля необходимо использовать выражение `set password` в приглашении утилиты `lsnrctl`, после чего можно еще раз запустить или остановить службу слушателя. Обратите внимание, что выражение `set password` не устанавливает новый пароль, а просто вынуждает слушателя запрашивать пароль для выполнения административных задач.

Результат неудачной попытки остановки слушателя вследствие отсутствия предоставленного пароля показан в листинге 11.5. Затем слушатель был корректно остановлен посредством применения команды `set password`.

Листинг 11.5. Останов слушателя с защитой паролем

```
$ lsnrctl stop
LSNRCTL for 32-bit Windows: Version 11.1.0.6.0 - Production on 20-MAR-2008

Copyright (c) 1991, 2001, Oracle Corporation. All rights reserved.

Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=ntl-alaptisam.netbsa.org
)(PORT=1521)))
TNS-01169: The listener has not recognized the password
$ lsnrctl set password
Password:
The command completed successfully
LSNRCTL> stop
Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=IPC)(KEY=EXTPROC0)))
The command completed successfully
```

Именование и подключение

В ранее приведенных примерах дескрипторов и идентификаторов подключений идентификатор подключения `sales` служил для подключения к службе `sales`. Идентификатор подключения может быть самим дескриптором подключения или более простым именем (наподобие `sales`), вместо которого затем подставляется дескриптор

подключения. Обычно используемый простой идентификатор подключения называют *именем сетевой службы*. Таким образом, идентификатор подключения `sales` в ранее приведенных примерах является именем сетевой службы.

Поскольку указание полного дескриптора подключения при каждой установке подключения является довольно утомительной задачей, применение имен сетевых служб более целесообразно. Однако для этого потребуется поддерживать центральное хранилище всех соответствий между именами сетевых служб и информацией дескрипторов подключений, чтобы Oracle мог проверять актуальность имен сетевых служб. Таким образом, когда пользователь запускает процесс подключения, указывая имя сетевой службы `sales`, центральное хранилище ищет дескриптор подключения, соответствующий имени `sales`. Как только дескриптор подключения найден, Oracle Net инициирует подключение к базе данных на указанном сервере.

Oracle допускает использование нескольких типов хранилищ именования, и доступ к информации преобразования имен, хранящейся в этих хранилищах, можно получить одним из следующих методов именования.

- *Метод локального именования*. Этот метод для подключения к серверу базы данных использует файл `tnsnames.ora`, хранящийся на каждом клиенте.
- *Метод именования простым подключением*. Этот метод допускает подключения без какой-либо конфигурации имени службы.
- *Метод внешнего именования*. Этот метод использует независимую службу именования для преобразования имен служб.
- *Метод каталожного именования*. Этот метод использует LDAP-совместимый сервер для преобразования имен служб.

Независимо от применяемого метода именования процесс преобразования имен остается неизменным. Следующие действия понадобится выполнить при использовании любого из методов именования для разрешения дескриптора подключения именем сетевой службы.

1. Выберите метод именования — локальный, простое подключение, внешнее именование или именование с помощью службы каталогов.
2. Отобразите дескрипторы подключений на имена служб.
3. Сконфигурируйте клиенты, чтобы они использовали метод именования, выбранный на первом шаге.

Метод локального именования

Локальное именование — простейший и наиболее легкий способ установки подключения Oracle. При использовании этого метода имена служб и их дескрипторы подключения сохраняются в локальном файле конфигурации `tnsnames.ora`. По умолчанию этот файл всегда находится в каталоге `$ORACLE_HOME/network/admin`. Oracle предоставляет для использования образец файла `tnsnames.ora`, который можно найти в каталоге по умолчанию. (Файл `tnsnames.ora` можно считать аналогичным файлу `/etc/hosts`, который содержит информацию по организации сети в системах UNIX/Linux.) Файл `tnsnames.ora` всегда присутствует на компьютере клиента. Если сервер базы данных применяется также для установки подключений клиентского типа, он будет содержать файл `tnsnames.ora` для других баз данных, к которым необходимо подключаться с этого сервера.

При инициализации подключения посредством интерфейса SQL*Plus либо других средств необходимо указать имя пользователя и пароль для той базы данных, к которой осуществляется подключение. Как только это будет сделано, службе Oracle Net

придется выяснить, на каком сервере действует база данных, поэтому она обращается к файлу tnsnames.ora для определения сетевого адреса, протокола и порта сервера базы данных. Успешно справившись с этой задачей, она инициализирует соединение со слушателем на том компьютере, где расположен сервер базы данных. После того как слушатель передает подключение серверу базы данных, база данных выполняет аутентификацию имени пользователя и пароля.

После того как подключения сконфигурированы для применения метода локального именования, все подключения к базе данных, выполняются они непосредственно через интерфейс SQL*Plus или через страницу регистрации приложения, будут применять файл tnsnames.ora для преобразования имен служб.

При использовании метода локального именования клиентские компьютеры кроме файла tnsnames.ora используют еще один файл — sqlnet.ora. Этот файл хранится на каждом клиенте и содержит важные параметры сетевой конфигурации. (Естественно, если сервер служит и в качестве клиента, он также будет содержать файл sqlnet.ora.) Использование параметра SQLNET.AUTHENTICATION_SERVICES для конфигурирования аутентификации операционной системы описано в главе 11.

Типичный файл sqlnet.ora имеет следующий вид:

```
# SQLNET.ORA Network Configuration File:
/u01/app/oracle/product/10.1.0/db_1/network/admin/sqlnet.ora
# Generated by Oracle configuration tools.
NAMES.DEFAULT_DOMAIN = wowcompany.com
SQLNET.AUTHENTICATION_SERVICES= (NTS)
NAMES.DIRECTORY_PATH= (TNSNAMES)
```

Обычно файлы конфигурации tnsnames.ora и sqlnet.ora располагаются в каталоге $ORACLE_HOME/network/admin в системах UNIX/Linux и в каталоге $ORACLE_HOME\network\admin в системах Windows. Однако эти файлы можно размещать в любом удобном месте. Если они помещены в каталоге, отличном от заданного по умолчанию, потребуется в переменной среды TNS_ADMIN указать их местоположение. Oracle будет искать эти два файла в следующих местах и воспользуется первым из найденных экземпляров.

1. В каталоге, указанном в переменной среды TNS_ADMIN.
2. Поиск файла tnsnames.ora будет выполняться в глобальном каталоге конфигурации. Для систем UNIX/Linux это обычно каталог /var/opt/oracle.
3. В стандартных сетевых каталогах: $ORACLE_HOME\network\admin в системах UNIX/Linux и в каталоге $ORACLE_HOME\network\admin в системах Windows.

Изменение файла tnsnames.ora вручную

Чтобы сконфигурировать метод локального именования, нужно внести изменения в файл tnsnames.ora, предоставленный Oracle при создании базы данных. Для этого достаточно перейти в заданный по умолчанию каталог хранения файла tnsnames.ora, $ORACLE_HOME/network/admin, и отредактировать этот файл, чтобы отразить информацию об именах данной сети и базы данных. При добавлении новой базы данных в систему необходимо также физически добавить отображение имени службы новой базы данных в файл tnsnames.ora каждого пользователя или отправить всем пользователям новый обновленный файл tnsnames.ora, чтобы они заменили им старый файл. В листинге 11.6 показан типичный файл tnsnames.ora.

Листинг 11.6. Типичный файл tnsnames.ora

```
# TNSNAMES.ORA Network Configuration File:
/u01/app/oracle/product/10.1.0/db_1/network/admin/tnsnames.ora
# Generated by Oracle configuration tools.
```

```
finance =
   (DESCRIPTION =
      (ADDRESS_LIST =
         (ADDRESS = (PROTOCOL = TCP)(HOST = localhost)(PORT = 1521))
      )
      (CONNECT_DATA =
         (SERVICE_NAME = finance.world)
      )
   )
salesprod =
   (DESCRIPTION =
      (ADDRESS_LIST =
         (ADDRESS = (PROTOCOL = TCP)(HOST = 172.11.150.1)(PORT = 1521))
      )
      (CONNECT_DATA =
         (SERVICE_NAME = salesprod.world)
      )
   )
custprod =
   (DESCRIPTION =
      (ADDRESS_LIST =
         (ADDRESS = (PROTOCOL = TCP)(HOST = custprod)(PORT = 1521))
      )
      (CONNECT_DATA =
         (SERVICE_NAME = custprod.world)
      )
   )
```

В файле `tnsnames.ora`, приведенном в листинге 11.6, перечислены три базы данных, и каждая из них отличается собственными характеристиками. Первая запись — база данных `finance`, предназначенная для настольного компьютера `NTL-ALAPATISAM`. База данных `salesprod` расположена на сервере UNIX с IP-адресом 172.11.150.1, к которому служба Oracle Net может подключаться, используя порт 1521 и протокол TCP. Последняя база данных для обозначения сервера хоста вместо IP-адреса использует символьное имя `custprod`.

Если бы в этот файл `tnsnames.ora` нужно было добавить четвертую базу данных `orderprod`, расположенную на хосте с IP-адресом 172.16.11.151, в файл `tnsnames.ora` потребовалось бы добавить соответствующий идентификатор подключения, как показано в следующем примере:

```
orderprod =
   (DESCRIPTION =
      (ADDRESS_LIST =
         (ADDRESS = (PROTOCOL = TCP)(HOST = 172.16.11.151)(PORT = 1521))
      )
      (CONNECT_DATA =
         (SERVICE_NAME =orderprod.world)
      )
```

После того как имя сетевой службы сконфигурировано, а файл `tnsnames.ora` должным образом модифицирован, подключение к базе данных должно выполняться так, как описано ниже.

1. Распространите новую конфигурацию имени службы своим клиентам. Это можно выполнить, копируя файлы `tnsnames.ora` и `sqlnet.ora` на компьютеры клиентов, на которых должен присутствовать установленный Oracle Client. В качестве альтернативы, для конфигурирования имен сетевых служб на самом клиенте

можно воспользоваться помощником Oracle Net8 Assistant или Net8 Configuration Assistant.

2. Удостоверьтесь, что слушатель запущен на том сервере, где действует база данных. Проверьте, что слушатель использует тот же протокол и адрес, которые были сконфигурированы для имени сетевой службы в файле `tnsnames.ora`. Убедитесь также, что слушатель использует протокол TCP/IP и прослушивает заданный по умолчанию порт 1521.
3. Удостоверьтесь, что целевая база данных, к которой вы пытаетесь подключиться, действует.
4. Протестируйте новое подключение с помощью следующей команды:

```
CONNECT имя_пользователя/пароль@имя_сетевой_службы
```

Хотя локальное именование достаточно просто в реализации, применение этого метода оказывается обременительным при наличии большого количестве клиентских компьютеров, которым требуется непосредственный доступ к серверу баз данных, поскольку в этом случае приходится поддерживать локальные копии файла `tnsnames.ora` на всех локальных клиентах. Более того, при изменении хостов или добавлении баз данных в систему приходится обеспечивать внесение изменений во все клиентские файлы `tnsnames.ora`. Естественно, если база клиентов невелика, поддержание файлов `tnsnames.ora` не должно составлять проблему.

Изменение `tnsnames.ora` с помощью Net Configuration Assistant

Для добавления новой службы в свой файл `tnsnames.ora` можно отдать предпочтение помощнику Net Configuration Assistant (NCA) Oracle, а не делать все вручную. Подобно файлу `listener.ora`, файл `tnsnames.ora` является несколько сложным, учитывая все используемые в нем скобки, и при редактировании его вручную легко допустить ошибку. Создавать новые службы с помощью графического интерфейса пользователя очень просто. Помощник NCA запрашивает имя сервера, имя базы данных, сетевой адрес и тип протокола. По завершении конфигурирования подключения в заданном по умолчанию каталоге появится новый или обновленный файл `tnsnames.ora`, включающий в себя добавленные службы баз данных.

Чтобы можно было применять NCA, вначале на клиентском компьютере потребуется установить программное обеспечение Oracle Client с компакт-диска Oracle Client CD. NCA поставляется как с серверной, так и с клиентской версией ПО. Создание и тестирование подключения займет пару минут.

Чтобы использовать NCA для конфигурирования нового имени службы в файле `tnsnames.ora`, нужно выполнить следующие действия.

1. Запустите на сервере UNIX/Linux помощник Net Configuration Assistant с помощью команды `netca`, как показано в следующем примере:

```
$ export DISPLAY=172.16.14.15:0.0
$ netca
```

На заметку! В системе Windows помощник NCA запускается выбором команды меню Start⇨Programs⇨Oracle-HOME_NAME⇨Configuration and Migration Tools (Пуск⇨Программы⇨Имя домашнего каталога Oracle⇨Средства конфигурирования и переноса).

---

2. Откроется страница Welcome (Приветствие). Выберите опцию Local Net Service Name Configuration (Локальная конфигурация имен сетевых служб) и щелкните на кнопке Next (Далее).

3. На странице Net Service Name Configuration (Конфигурация имен сетевых служб) выберите опцию Add (Добавить) и щелкните на кнопке Next.
4. На странице Service Name Configuration (Конфигурация имен служб) введите имя службы, которую необходимо сконфигурировать. В данном примере это база данных `emrep.netbsa.org`. Обратите внимание, что в общем случае имя службы совпадает с глобальным именем базы данных. Щелкните на кнопке Next.
5. На странице Select Protocol (Выберите протокол) выберите протокол TCP и щелкните на кнопке Next.
6. На странице TCP/IP Protocol (Протокол TCP/IP) введите имя хоста, на котором действует база данных. Выберите стандартный номер порта, 1521, и щелкните на кнопке Next.
7. На странице Test (Тестирование) щелкните на кнопке Yes, Perform a Test (Да, выполнить тестирование), а затем щелкните на кнопке Next.
8. NCA предпримет попытку подключения к базе данных с применением новой конфигурации и отобразит результаты. В случае неудачи подключения удостоверьтесь, что слушатель целевой базы данных запущен, а сочетание имени пользователя и пароля, по умолчанию используемое процессом тестирования (`system/manager`), изменено на действующее сочетание. Убедитесь также, что имена базы данных и домена указаны правильно.
9. Затем на странице Net Service Name (Имя сетевой службы) NCA попросит подтвердить имя сетевой службы. Щелкните на кнопке Next.
10. На странице Another Service Name (Имя другой службы) можно сконфигурировать дополнительные имена служб.
11. На странице Net Service Name Configuration Done (Конфигурация имени сетевой службы завершена) щелкните на кнопке Next. Когда снова откроется страница Welcome, щелкните на кнопке Finish (Готово).

На заметку! Имена сетевых служб можно конфигурировать и на странице Net Services Administration (Администрирование сетевых служб) в Oracle Enterprise Manager или в графическом интерфейсе пользователя Oracle Net Manager.

Метод именования простым подключением

Администраторы БД Oracle могут упростить конфигурирование клиентов, используя метод именования простым подключением. С помощью этого метода клиенты базы данных могут подключаться к ней без применения файла `tnsnames.ora` в средах TCP/IP. Все что им потребуется — это имя хоста, необязательный номер порта и имя службы базы данных. Таким образом, мы располагаем не требующей конфигурирования и доступной изначально возможностью подключения к любой базе данных в системе по протоколу TCP/IP.

Единственным условием для применения метода именования простого подключения является наличие поддержки протокола TCP/IP как на стороне клиента, так и на стороне сервера. Однако при этом отпадает необходимость в настройке файла `tnsnames.ora`. Этот новый метод подключения можно рассматривать в качестве расширения метода именования хоста, предложенного в Oracle9*i*.

Синтаксис применения этого нового метода подключения выглядит следующим образом:

```
$ CONNECT имя_пользователя/пароль@[//]хост[:порт][/имя_службы]
```

В этой синтаксической конструкции следует обратить внимание на четыре описанных ниже элемента.

- //. Этот элемент не обязателен.
- *хост*. Этот параметр обязателен. Можно указать либо символическое имя хоста, либо IP-адрес сервера, на котором расположена целевая база данных.
- *порт*. Этот параметр не обязателен. Если порт не указан, по умолчанию используется порт 1521.
- *имя_службы*. Этот параметр указывает имя службы базы данных (по умолчанию оно совпадает с именем хоста) и является необязательным. Если имя хоста и имя сервера базы данных идентичны, этот параметр можно опустить. Если же они не совпадают, понадобится указать допустимое имя службы для идентификации базы данных.

Следующий пример демонстрирует результат подключения к базе данных dev1, расположенной на сервере hp50. Подключение осуществляется непосредственно из приглашения операционной системы, поэтому вместо ключевого слова CONNECT используется SQLPLUS:

```
$ sqlplus system/system_passwd@ntl-alapatisam.netbsa.org:1521/emrep.netbsa.org
-
SQL*Plus: Release 11.1.0.6.0 - Production on Thu Mar 20 09:38:15 2008

Copyright (c) 1982, 2007, Oracle. All rights reserved.


Connected to:
Oracle Database 11g Enterprise Edition Release 11.1.0.6.0 - Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL>
```

Обратите внимание, что подключение также можно выполнить, не используя необязательный номер порта, как показано в следующем примере:

```
$ sqlplus system/system_passwd@ntl-alaptisam.netbsa.org/emrep.netbsa.org
```

Обратите внимание, что основные параметры метода простого подключения совпадают с информацией подключения метода локального именования, которая требуется в файле tnsnames.ora. Приведенная в предыдущем примере информация в файле tnsnames.ora была бы сконфигурирована следующим образом:

```
(DESCRIPTION=
   (ADDRESS=(PROTOCOL=tcp)(HOST=ntl_alapatisam.netbsa.org)(PORT=1521))
      (CONNECT_DATA=
        (SERVICE_NAME=emrep.netbsa.org)))
```

При подключении из интерфейса SQL*Plus можно применять следующий синтаксис:

```
$ sqlplus /nolog

SQL*Plus: Release 11.1.0.6.0 - Production on Thu Mar 20 09:38:15 2008

Copyright (c) 1982, 2007, Oracle. All rights reserved.
SQL> connect system/system_passwd@ntl-alaptisam.netbsa.org:1521/emrep.netbsa.
org

Connected.
SQL>
```

На заметку! Из четырех элементов, которые должны быть указаны при использовании в качестве метода именования простого подключения, обязательным является только имя хоста.

Конфигурирование именования простым подключением

Как видно из его названия, метод именования простым подключением требует очень мало в плане конфигурирования. Для указания метода простого подключения в качестве значения переменной `NAMES.DIRECTORY_PATH` в файле `sqlnet.ora` применяется ключевое слово `EZCONNECT`. Рассмотрим следующий файл `sqlnet.ora`:

```
# sqlnet.ora Network Configuration File:
/u01/app/oracle/10.1.0/db_1/network/admin/sqlnet.ora
# Generated by Oracle configuration tools.
NAMES.DEFAULT_DOMAIN = netbsa.org
SQLNET.AUTHENTICATION_SERVICES = (NTS)
NAMES.DIRECTORY_PATH = (TNSNAMES,EZCONNECT)
```

Последняя строка показывает методы подключения, которые служба Oracle Net будет использовать для преобразования идентификаторов подключений в дескрипторы подключений. Параметр `NAMES.DIRECTORY_PATH` задает порядок, в котором служба Oracle Net будет применять методы именования для преобразования идентификаторов подключений в дескрипторы подключений. В этом примере `TNSNAMES` является первой настройкой, поэтому Oracle Net будет использовать файл `tnsnames.ora` по умолчанию. Если ей не удастся подключиться с применением файла `tnsnames.ora`, она попытается подключиться методом `EZCONNECT`.

Если требуется, чтобы метод `EZCONNECT` применялся по умолчанию, можно вручную отредактировать файл `sqlnet.ora`, переместив значение `EZCONNECT` на первое место в параметре `NAMES.DIRECTORY_PATH`:

```
NAMES.DIRECTORY_PATH = (EZCONNECT, TNSNAMES)
```

Ограничения метода именования простым подключением

Ниже перечислены ограничения в применении метода именования простым подключением.

- На стороне клиента должно быть установлено программное обеспечение Oracle Database 11*g* Net Services.
- Поддержка протокола TCP/IP должна быть обеспечена как на стороне клиента, так и на стороне сервера баз данных.
- Нельзя использовать какие-либо расширенные сетевые функциональные возможности Oracle, вроде поддержки пула соединений, вызовов внешних процедур или гетерогенных служб (Heterogeneous Services).

Резидентный пул соединений базы данных

Вплоть до появления версии Oracle Database 11*g* для подключения сеансов пользователя к базе данных можно было применять два способа: процесс выделенного сервера, который в каждый момент времени обрабатывает один процесс пользователя, и процесс разделяемого сервера, который обслуживает несколько процессов пользователей. В Oracle Database 11*g* появился третий способ подключения сеансов к базе данных, который является разновидностью подхода с использованием выделенного сервера и действует на основе концепции использования пула серверов при обслуживании запросов на подключение.

Как правило, веб-приложения устанавливают соединение, быстро его используют и быстро освобождают. Для этих приложений типичным является совместное или повторное использование сеансов. Обычно веб-приложения не поддерживают постоянного активного соединения с базой данных, а используют ее время от времени и, как правило, не сохраняют состояние при повторном подключении к БД. Применение пула соединений помогает обслуживать тысячи конечных пользователей с помощью небольшого количества сеансов, тем самым повышая масштабируемость базы данных. Технологии вроде PHP сами по себе не могут воспользоваться преимуществами пула соединений на сервере приложений, поскольку каждый процесс веб-сервера требует выделенного соединения с базой данных.

Совершенно новый метод подключения с помощью *резидентного пула соединений базы данных* (database resident connection pooling — DRCP) использует пулы серверов для обслуживания большого количества сеансов пользователей. Применение DRCP снижает требования к памяти по сравнению с конфигурациями, использующими выделенные и разделяемые серверы. DRCP разработан специально для поддержки таких архитектур, как PHP с сервером Apache, которые не в состоянии получить преимущества от пула соединений промежуточного слоя, поскольку они применяют мультипроцессные однопоточные серверы приложений. DRCP позволяет таким приложениям легко масштабировать серверных подключений вплоть до десятков тысяч.

DRCP во многом аналогичен выделенному серверу в плане того, что работает подобно конфигурации с выделенным сервером, но при этом каждое подключение пользователя не должно сохранять связь с единственным выделенным сервером в течение времени своего существования. Каждое соединение с базой данных занимает сервер на краткий промежуток времени. Когда сеанс пользователя завершает свою работу, он освобождает соединение с сервером, делая его снова доступным для пула серверов.

Принцип работы DRCP

DRCP использует брокер соединений, который присваивает каждый новый сеанс клиента доступному серверу из пула. Как только запрос на подключение клиента обслужен базой данных, соединение освобождает сервер, возвращая его в пул серверов. Таким образом, сеансы используют память и другие ресурсы только в то время, когда база данных действительно выполняет задачи для обслуживания сеансов, и освобождают эти ресурсы, освобождая сервер и возвращая его в пул серверов.

До тех пор пока количество серверов в пуле базы данных не достигло максимального значения, брокер соединений будет создавать сервер в пуле для назначения его новому клиентскому соединению, если не сможет найти свободный сервер в пуле. По достижении максимального количества серверов пула брокер не сможет создавать новые серверы. В этом случае он будет отправлять клиентские подключения в очередь ожидания на освобождение серверов пула. В отличие от подхода с применением выделенного сервера, где используемый объем памяти (PGA) пропорционален количеству сеансов пользователей, при использовании DRCP объем памяти пропорционален числу активных серверов из пула. Следующий пример демонстрирует, как значительно сэкономить память, перейдя от конфигурации с применением выделенного сервера к конфигурации с DRCP.

В этом примере предполагается, что существует 5000 клиентских соединений, и что каждый клиентский сеанс требует 200 Кбайт, а каждый процесс сервера — 5 Мбайт памяти. Кроме того, будем считать, то максимальное количество необходимых серверных соединений равно 200. Общий объем памяти, требуемый для конфигураций с применением выделенного сервера и с применением DRCP можно вычислить следующим образом:

```
Выделенный сервер
Общий необходимый объем памяти = 5000 × (200 Кбайт + 5 Мбайт) = 260 Гбайт

Резидентный пул соединений базы данных

Общий необходимый объем = 200 (200 Кбайт + 5 Мбайт) = 502 Мбайт

Разделяемый сервер
500 × 200 Кбайт + 200 × 5 Мбайт = 11 Гбайт
```

Как видите, в то время как конфигурация с выделенным сервером требует 260 Гбайт, для конфигурации с применением DRCP необходимо лишь не многим больше 0,5 Гбайт.

Активизация и отключение DRCP

По умолчанию база данных изначально конфигурируется с используемым по умолчанию пулом соединений SYS_DEFAULT_CONNECTION_POOL. Однако чтобы воспользоваться преимуществами, предоставляемыми DRCP, потребуется запустить заданный по умолчанию пул соединений. Его запускают с помощью процедуры START_POOL из пакета DBMS_CONNECTION_POOL, как показано в следующем примере:

```
SQL> connect sys/sammyy1 as sysdba
SQL> exec dbms_connection_pool.start_pool();

PL/SQL procedure successfully completed.
SQL>
```

Состояние пула соединений можно проверить с помощью следующего запроса:

```
SQL> select connection_pool, status, maxsize from dba_cpool_info;

CONNECTION_POOL                STATUS    MAXSIZE
-----------------------------  ------    --------
SYS_DEFAULT_CONNECTION_POOL    ACTIVE    80
SQL>
```

После запуска пула соединений он останется открытым даже при остановке базы данных и ее повторном запуске. Пул соединений можно остановить, выполняя процедуру STOP_POOL:

```
SQL> exec dbms_connection_pool.stop_pool();

PL/SQL procedure successfully completed.
SQL>
```

Пулом соединений управляет фоновый процесс CMON (Connection Monitor — Монитор соединений). Приложения передают процессы выделенного сервера процессу CMON, который возвращает процесс пулу соединений.

DRCP-подключение можно указать следующим образом.

При использовании строки EZ Connect укажите в ней ключевое слово POOLED:

```
myhost.comany.com:1521/mydb.company.com:POOLED
```

При использовании файла tnsnames.ora укажите параметр SERVER=POOLED в строке подключения TNS:

```
mydb = (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp) (HOST=myhost.company.com)
       (SERVER=POOLED)))
```

Конфигурирование DRCP

Для конфигурирования DRCP в базе данных используют следующие параметры. Пул соединений можно конфигурировать в зависимости от требований к применению БД.

Основные параметры конфигурации DRCP перечислены ниже.

- INACTIVITY TIMEOUT. Максимальное время бездействия, разрешенного помещенного в пул серверу, прежде чем он будет остановлен.
- MAX_LIFETIME_PER_SESSION. Длительность времени жизни (time to live — TTL) для сеанса, помещенного в пул.
- MAX_USES_PER_SESSION. Максимальное число раз предоставления помещенного в пул сервера пулу соединений.
- MAX_SIZE и MIN_SIZE. Максимальное и минимальное количество помещенных в пул серверов в пуле соединений.
- MAX_THINK_TIME. Максимальное время, в течение которого клиент может оставаться неактивным после получения сервера из пула соединений.

Для одновременной настройки нескольких параметров конфигурации пула соединений можно выполнить процедуру CONFIGURE_POOL из пакета DBMS_CONNECTION_POOL. Чтобы определить значение одного параметра конфигурации пула соединений, можно запустить процедуру ALTER_PARAM из пакета DBMS_CONNECTION_POOL, как показано в следующем примере:

```
SQL> exec dbms_connection_pool.alter_param('   ','INACTIVITY_TIMEOUT','2400')
```

Приведенный пример демонстрирует, как можно определить значение параметра INACTIVITY_TIMEOUT. В соответствии со значением параметра INACTIVITY_TIMEOUT, равным 2400, база данных разрешает помещенному в пул серверу оставаться бездействующим на протяжении периода длительностью до одного часа, прежде чем соединение будет разорвано.

Значения параметров конфигурации пула соединений, заданные по умолчанию, можно восстановить, выполняя процедуру RESTORE_DEFAULTS, как показано в следующем примере:

```
SQL> exec dbms_connection_pool.restore_defaults()
```

Процедура RESTORE_DEFAULTS помогает восстановить заданные по умолчанию значения для всех параметров конфигурации пула соединений.

Мониторинг DRCP

Для отслеживания работы DRCP используются следующие представления.

- DB_CPOOL_INFO. Отображает имя пула соединений, его состояние, максимальное и минимальное количества соединений и тайм-аут для бездействующих сеансов
- V$CPOOL_STAT. Отображает статистические сведения, такие как количество запросов сеансов и время ожидания запросов сеансов.
- V$CPOOL_CC_STATS. Отображает подробные статистические сведения о соединении на уровне класса.

Метод внешнего именования

Для преобразования имен сетевых служб внешний метод именования использует внешние службы именования наподобие Network Information Service (Сетевая информационная служба) (NIS), которая первоначально была разработана компанией Sun Microsystems. Системы NIS содержат центральную базу данных имен хостов и используют двумерное пространство имен, основанное на главном сервере.

Чтобы использовать метод внешнего именования для преобразования имен, потребуется выполнить следующие действия.

1. Попросите своего системного администратора сконфигурировать NIS, если это еще не было сделано.
2. Создайте файл `tnsnames.ora`, как это было бы сделано при использовании метода локального именования.
3. Преобразуйте файл `tnsnames.ora` в карту `tnsnames`, которая впоследствии понадобится для сервера NIS. Для извлечения карты `tnsnames` из файла `tnsnames.ora` системный администратор должен использовать команду `tns2nis`, как показано в этом примере:

   ```
   #  tns2nis tnsnames.ora
   ```

4. Скопируйте файл карты связей `tnsnames` на сервер, на котором действует служба NIS.
5. Установите файл карты `tnsnames` на сервер NIS, используя NIS-программу `makedbm`:

   ```
   #  makedbm tnsnames /var/yp/'domainname'/tnsnames
   ```

6. Протестируйте установку карты связей `tnsnames` в службе NIS с помощью следующей команды:

   ```
   # ypmatch net_service_name tnsnames
   ```

   Вы должны получить подтверждение в следующей форме:

   ```
   description=(address=(protocol=tcp)
    (host=имя_хоста)(port=номер_порта)))
    (connect_data=(service_name=имя_службы)))
   ```

7. Отредактируйте файл `sqlnet.ora`, как показано ниже:

   ```
   NAMES_DIRECTORY_PATH=(nis, hostname, tnsnames)
   ```

Метод `nis` должен быть указан первым в скобках, т.к. служба Oracle Net будет пытаться преобразовать имя службы, вначале используя службу NIS. Порядок следования остальных элементов в скобках значения не имеет.

Именование с помощью службы каталогов

По установившейся традиции сетевая информация хранилась на нескольких серверах, часто в различных форматах, но современные Интернет-приложения подвергают многие организации огромному риску в плане безопасности. Децентрализованные системы служат постоянным источником беспокойства для большинства специалистов в области безопасности. Централизованные службы каталогов, предназначенные для аутентификации пользователей и реализации политик безопасности, повышают возможности организаций по защите своих сетевых ресурсов.

Службы каталогов — это огромные централизованные хранилища, которые содержат все метаданные, связанные с базами данных, сетями, пользователями, политиками безопасности и т.п. Каталог, поддерживающий работу этих служб, может замещать множество локализованных файлов, таких как `tnsnames.ora`, и предоставлять единственное место выполнения преобразования имен и аутентификации. Эти каталоги представляют собой сравнительно редко обновляемые базы данных, по отношению к которым выполняется значительное количество операций чтения. Скорость получения информации — основной фактор успешности работы службы каталогов.

Вот несколько примеров видов данных, которыми такие каталоги могут эффективно управлять:

- имена пользователей и пароли;
- профили пользователей;
- политики предоставления полномочий;
- конфигурация сети и информация сетевых служб.

Существует много видов доступных коммерческих служб каталогов, в том числе Active Directory от Microsoft и Oracle Internet Directory (OID), и их можно арендовать для выполнения функций хоста для организации.

Метод именования с помощью службы каталогов хранит информацию о соединениях базы данных на сервере каталогов, совместимом с протоколом LDAP (Lightweight Directory Access Protocol — Облегченный протокол доступа к каталогам). Идентификаторы соединений сохраняются в контексте Oracle, который содержит записи, предназначенные для использования с OID.

Хотя вначале централизованная настройка может показаться пугающе сложной, в действительности она достаточно проста. Первоначальные затраты могут оказаться выше, но впоследствии затраты на управление информацией минимальны. Кроме облегчения клиентам подключения к центральным сетям и базам данных, такие каталоги, как OID, чрезвычайно ценны с точки зрения обеспечения безопасности в пределах всей организации.

Oracle Internet Directory

OID — это LDAP-совместимая служба каталогов, которая, помимо прочего, хранит также идентификаторы соединений. LDAP — популярный протокол доступа к сетевым службам, и он является Интернет-стандартом хранения данных и доступа к каталогам. Служба OID чрезвычайно масштабируема, поскольку она реализована на основе в высшей степени масштабируемой базы данных Oracle. Это позволяет хранить огромный объем информации о каталогах и легко получать к ней доступ. Данные защищены, поскольку они хранятся в базе, а OID является в высшей степени доступной службой, как и база данных Oracle. Спецификация LDAP привлекательна также минимально необходимым объемом клиентского программного обеспечения.

OID можно использовать для многих приложений, таких как адресные книги, хранилища сертификатов безопасности и корпоративные службы каталогов. Компания Oracle настоятельно рекомендует переходить к применению OID в качестве способа конфигурирования соединений баз данных. Путем отхода от Oracle Names (Имена Oracle) — метода подключения, предоставляемого в прошлом — компания Oracle позиционирует OID в качестве основной альтернативы традиционного метода локального именования, который требует использования файла конфигурации сети `tnsnames.ora`. База данных Oracle может применять OID для хранения имен пользователей и паролей и для хранения верификатора пароля вместе с записью каждого пользователя. Другие компоненты Oracle используют OID в различных целях.

- *Oracle Application Server Single Sign-On* (Единый интерфейс входа в сервер приложений Oracle). Применяет OID для хранения записей пользователей.
- *Oracle Collaboration Suite* (Пакет для обеспечения сотрудничества Oracle). Использует OID для осуществления централизованного управления информацией о пользователях и группах.
- *Oracle Net Services* (Сетевые службы Oracle). Пользуется OID для хранения и преобразования имен служб базы данных и сетевых служб.

- *Oracle Advanced Security* (Расширенная служба безопасности Oracle). Использует OID для централизованного управления аутентификационными мандатами пользователей, авторизацией, преобразованиями в схему совместного использования, аутентификацией посредством единственного пароля, безопасностью пользователей предприятия и центральным хранилищем мандатов инфраструктуры PKI (Public Key Infrastructure — инфраструктура открытых ключей).

OID включает в себя следующие элементы.

- *Oracle directory server* (Сервер каталогов Oracle). Предоставляет имена служб и другую информацию, используя при этом многоуровневую архитектуру поверх стека протоколов TCP/IP.
- *Oracle directory replication server* (Сервер репликации каталогов Oracle). Реплицирует данные LDAP между серверами каталогов Oracle.
- *Средства администрирования каталогов, в том числе*:
  - Oracle Directory Manager (Диспетчер каталогов Oracle), графический интерфейс пользователя, который облегчает администрирование OID, и другие утилиты администрирования на основе командной строки;
  - средства управления сервером каталогов, входящие в состав консоли Application Server Control (Управление сервером приложений) Oracle Enterprise Manager 10*g*, которые позволяют отслеживать события реального времени из веб-браузера.

Основная идея применения OID проста. Пользователи подключаются к OID, который является приложением, действующим в базе данных Oracle. Пользователи предоставляют OID идентификатор службы Oracle (имя базы данных). Каталог возвращает полную информацию о соединении — имя хоста, протокол подключения, номер порта и имя экземпляра базы данных — клиенту, который затем подключается к серверу базы данных. Идентификаторы соединений хранятся в контексте Oracle, который содержит такие записи, как имена баз данных и служб, предназначенные для использования с программным обеспечением Oracle, подобным OID.

Функция Advanced Security Oracle использует OID для централизованного управления информацией, связанной с пользователями. При использовании технологии реплицированной базы данных каталог Oracle OID будет весьма полезным в плане преодоления сложностей, связанных с использованием нескольких серверов и сетевых протоколов.

Хотя компания Oracle предпочла бы, чтобы все сетевые конфигурации были преобразованы в OID, это совершенно не означает, что предоставляемые этой службой преимущества окупают дополнительные накладные расходы для большинства небольших или средних предприятий. Помните, что OID не является продуктом, предназначенным исключительно для конфигурирования сети. Сетевые соединения базы данных — лишь небольшая часть возможностей OID. Подход локального именования (или новый метод именования простым подключением) благодаря своей простоте по-прежнему остается удобным для большинства организаций.

Как OID осуществляет подключения к базам данных

При использовании OID для преобразования имен следует помнить, что клиент не располагает файлом, таким как `tnsnames.ora`, содержащим информацию о преобразовании. При использовании именования с помощью службы каталогов клиенты Net Services подключаются к базе данных следующим образом.

1. Лицо, которое желает выполнить подключение, вводит на клиентском компьютере свое обычное сочетание имени пользователя/пароля и идентификатор соединения.

2. Файл `sqlnet.ora` на компьютере клиента указывает, что он использует OID для преобразования имен, поэтому клиент Net Services передает свой запрос процессу слушателя/диспетчера OID.
3. Слушатель/диспетчер OID передает запрос LDAP серверу каталогов Oracle.
4. Сервер каталогов подключается к базе данных OID и преобразует идентификатор соединения в соответствующий дескриптор соединения, который содержит информацию о сети, сервере и протоколе. Затем он отправляет эту подробную информацию дескриптора соединения клиенту Net Services.
5. Клиент отправляет полученный дескриптор соединения службе слушателя Oracle Net Listener (или диспетчеру, если используются разделяемые серверы).
6. Служба слушателя принимает запрос на подключение и, после его проверки, пересылает базе данных.

Организация OID

Каталог содержит набор сведений о различных объектах, таких как имена и адреса сотрудников или информацию об именах служб баз данных (как описано в этой главе). Информация в каталоге организована в виде иерархической структуры, получившей название DIT (Directory Information Tree — Дерево информации каталога). Каждая запись каталога образуется из различных объектных классов и атрибутов следующим образом:

- каталоги образуются из объектных классов;
- объектные классы — это группы атрибутов;
- атрибуты представляют собой контейнеры, содержащие данные.

Каталог состоит из *записей*, которые представляют собой коллекции информации об объекте. Для однозначной идентификации записи требуется какое-либо средство для указания ее местоположения в структуре каталога. Этим однозначным локатором адреса является *отличительное имя* (distinguished name — DN) записи. DN подобно адресу, который точно указывает местоположение записи в каталоге — оно предоставляет полный путь от вершины иерархии до местоположения записи.

Ниже приведен пример DN:

```
cn=nina
ou=finance
c=us
o=wowcompany
```

Это DN записи `nina` содержит следующие узлы:

- `cn` — общее имя (common name);
- `ou` — организационная единица (organizational unit);
- `c` — страна (country);
- `o` — организация (organization).

Таким образом, DN-имя `nina.finance.us.wowcompany` *уникальным образом определяет лицо с именем Nina*, которое работает в финансовом отделе (`finance`) филиала в США (`US`) компании `Wowcompany`. Обратите внимание, что каждый из отдельных узлов называют *относительными отличительными именами* (relative distinguished names — RDN), поэтому, по сути, DN — это всего лишь строка имен RDN.

Контекст именования — это смежное поддерево на одном сервере каталогов. *Контекст Oracle* содержит соответствующие записи, предназначенные для использования с такими функциями Oracle, как именование с помощью службы каталогов

Oracle Net Service и безопасность пользователя предприятия. Один сервер каталогов может содержать несколько контекстов Oracle. OID создаст заданный по умолчанию контекст Oracle в корне дерева информации каталога. В дереве DIT контекст Oracle RDN (`cn=OracleContext`) является местоположением по умолчанию, которое клиенты применяют для поиска в каталоге соответствующих дескрипторов соединений для указанных идентификаторов соединений.

Контекст Oracle в дереве каталогов должен содержать все имена служб, включая полную информацию о сетевых подключениях и подключениях к серверам. В дополнительных подзаписях, поддерживающих именование с помощью службы каталогов, контекст Oracle содержит записи, поддерживающие функции безопасности предприятия. Поэтому при попытке подключения к базе данных на сервере серверу OID не нужно просматривать все дерево каталогов от корневой записи до последнего узла. Нужно просто снабдить его частичным именем DN с указанием пути от корневого узла до контекста Oracle. Контекст Oracle будет содержать имена сетевых служб, а те, в свою очередь, будут содержать подробную информацию о подключении.

Административный контекст, называемый также *контекстом именования с помощью каталогов*, — это запись каталога, которая содержит контекст Oracle. Следующий простой пример демонстрирует эти несколько запутанные понятия.

Полное имя DN для базы данных `orcl` выглядит так:

```
dc=com,dc=wowcompany
cn=orcl,
cn=description,
cn=address,
cn=port,
cn=service_name
```

В записи DN `dc` означает *domain component* (компонент домена) и обычно используется для описания элементов доменов в каталоге.

Важно отметить, что поскольку вся информация дескриптора соединения хранится в контексте Oracle RDN, полное имя DN не обязательно указывать при каждом поиске информации о подключении к базе данных. Приведенное достаточно длинное имя DN можно заменить следующим имеющим обобщенный вид именем DN:

```
dc=com,dc=wowcompany,cn=OracleContext
```

Обратите внимание, что `dc` определяет компонент домена, а `cn` означает общее имя (common name). В этом примере и `com`, и `wowcompany` — компоненты домена, поэтому они расположены в верхней части дерева каталога.

Инсталляция OID

OID можно установить с применением программного обеспечения сервера приложений Oracle Application Server 10*g* Release 2 (10.1.2.0.0). При использовании универсального инсталлятора Oracle в окне Select a Product to Install (Выберите продукт для установки) потребуется выбрать опцию Oracle AS Infrastructure 10g (Инфраструктура сервера приложений Oracle 10*g*). Эта опция позволяет установить новую службу OID на сервер. На следующей странице универсального инсталлятора Oracle — Select Installation Type (Выберите тип установки) — выберите опцию Identify Management and Metadata Repository (Управление идентификацией и хранилище метаданных). Она создает хранилище метаданных Metadata Repository, существование которого обязательно для инсталляции OID.

Рассмотрение установки и управления OID выходит за рамки настоящей книги. За более подробной информацией обращайтесь к документации Oracle Application Server Release 2 на веб-сайте `http://technet.oracle.com`.

Как только служба OID будет сконфигурирована, можно приступать к вводу в этот каталог имен сетевых служб. Для этого применяется несколько методов. Проще всего добавлять имена служб в Oracle Net Manager, если записи добавляются индивидуально, или же импортировать в OID весь файл `tnsnames.ora`, используя Oracle Enterprise Manager.

Oracle и интерфейс подключения Java к базам данных

Часто для выполнения задач по манипулированию данными программам Java требуется подключиться к базе данных. JDBC — это интерфейс, который позволяет программе Java подключаться к базе данных и выдавать операторы DML и DDL. Интерфейс JDBC позволяет использовать динамические операторы SQL в тех ситуациях, когда количество и тип столбцов неизвестно до времени выполнения. (При написании статического SQL-кода можно применять интерфейс SQLJ, который позволяет вставлять операторы SQL в код Java.) JDBC предоставляет обширную библиотеку подпрограмм, которые помогают открывать и закрывать соединения с базами данных и обрабатывать данные.

В последующих разделах показано использование JDBC для подключения и работы с базами данных Oracle из программ Java.

Установка соединения с базой данных

Прежде чем можно будет подключиться к базе данных, необходимо выбрать соответствующие драйверы. Oracle предоставляет четыре основных вида драйверов JDBC.

- *Тонкий драйвер JDBC*. Этот простейший клиентский драйвер Java предоставляет прямое соединение с базой данных посредством протокола TCP/IP. Этот драйвер требует наличия слушателя и использует сокеты для установления соединений базами данных.
- *OCI-драйвер JDBC*. Этот драйвер требует наличия клиентской инсталляции Oracle, поэтому он является специфичным для Oracle. Этот драйвер в высшей степени масштабируем, и он может использовать пул соединений для обслуживания большого количества пользователей.
- *Тонкий драйвер серверной стороны JDBC*. Действуя на сервере, этот драйвер подключается к удаленным базам данных и предоставляет те же функциональные возможности, что и тонкий драйвер клиентской стороны.
- *Внутренний драйвер серверной стороны JDBC*. Как видно из его названия, этот драйвер располагается на сервере и используется виртуальной машиной Java Virtual Machine (Virtual Machine Java — JVM) для обмена данными с сервером базы данных Oracle.

После выбора конкретного типа драйвера JDBC необходимо указать драйвер JDBC одним из двух способов: используя статический метод `registerDriver()` класса `DriverManager` JDBC или применяя метод `forName()` класса `java.lang`. Эти два метода указания драйвера JDBC имеют следующий вид:

```
DriverManager.registerDriver ("new oracle.jdbc.OracleDriver()");
```

и

```
Class.forName("oracle.jdbc.driver.OracleDriver")
```

Как только драйвер JDBC загружен, с помощью статического метода `getConnection()` класса `DriverManager` можно установить соединение с базой данных. Этот метод создаст экземпляр класса `connection` JDBC. Соответствующий код приведен в листинге 11.7.

Листинг 11.7. Создание соединения с базой данных

```
connection conn=DriverManager.getConnection(
"jdbc:oracle:thin:@prod1:1521:finprod", username, passwd);
/* Различные части объекта соединения означают следующее: */
jdbc=protocol
oracle=vendor
thin=driver
prod1=server
1521=port number
finprod=Oracle database
username=database username
password=database password
```

Если вся предоставленная информация верна, соединение с базой данных будет установлено из приложения Java.

Работа с базой данных

Теперь, когда вы научились подключаться к базе данных, используя интерфейс JDBC, пора выяснить, как посредством JDBC-соединения можно обрабатывать операторы SQL в базе данных.

SQL-код нельзя выполнять непосредственно из программы Java. Вначале потребуется создать операторы JDBC, а затем нужно будет выполнить SQL-операторы. Давайте подробно рассмотрим эти два этапа.

Создание объекта `Statement`

Чтобы передать SQL-операторы базе данных, необходимо создать JDBC-объект `Statement`. Этот объект свяжет себя с открытым соединением, и в дальнейшем будет действовать в качестве канала, по которому SQL-операторы будут передаваться из программы Java в базу данных для выполнения. Объект `Statement` JDBC создается следующим образом:

```
statement stmt = conn.createStatement();
```

С объектом `stmt` никакие SQL-операторы не связаны. Однако класс `Statement` содержит еще один объект, названный `PreparedStatement`, который, кроме того, что служит каналом для выполнения операторов, всегда содержит SQL-оператор. Этот SQL-оператор компилируется немедленно, причем он может компилироваться только один раз, а затем использоваться многократно, что является огромным преимуществом.

Однако для простоты в этом разделе описано только использование объекта `Statement`. Теперь рассмотрим выполнение SQL-операторов.

Выполнение SQL-операторов

Чтобы понять работу SQL-операторов JDBC, следует отделить операторы `SELECT`, выполняющие запросы к базе данных, от всех остальных операторов. В отличие от других операторов, операторы `SELECT` не изменяют состояние базы данных.

Вначале рассмотрим операторы обработки запросов.

Обработка запросов

Для получения результатов запроса операторы SELECT используют метод executeQuery(). Этот метод возвращает результаты в объекте ResultSet. Пример приведен в листинге 11.8.

Листинг 11.8. Получение результатов запроса

```
string first_name,last_name,manager;
number salary;
resultSet rs = stmt.executeQuery("SELECT * FROM Employees");
 while (rs.next()) {
  first_name = rs.getString("first_name");
  last_name = rs.getString("last_name");
  manager = rs.getString("manager");
  salary = rs.getNumber("salary");
  system.out.println(first_name + last_name "works for"
  Manager "salary is:" salary.");
```

Обратите внимание, что rs — это экземпляр объекта resultSet, и именно он содержит результаты запроса. Объект resultSet предоставляет также курсор, что позволяет обращаться к результатам одним за другим. При каждом вызове метода resultSet курсор перемещается к следующей строке в результирующем наборе.

Обработка операторов DDL и не связанных с запросами операторов DML

Любой оператор, который изменяет состояние базы данных — будь то оператор DDL или оператор DML, такой как INSERT, UPDATE либо DELETE — выполняется с помощью метода executeUpdate(). Обратите внимание, что слово “update” (“обновление”) в имени метода указывает, что SQL-оператор будет изменять что-либо в базе данных.

Вот несколько примеров операторов executeUpdate()

```
statement stmt = conn.createStatement();
   stmt.executeUpdate("CREATE TABLE Employees" +
      "(last_name VARCHAR2(30), first_name VARCHAR2(20),
         manager VARCHAR2(30), salary(number");
   stmt.executeUpdate("INSERT INTO Employees " +
      "VALUES ('Alapati', 'Valerie', 'Shannon', salary)");
```

Приведенные операторы создают объект Statement, а затем создают таблицу и вставляют в нее определенные данные.

При использовании интерфейса для выполнения SQL-операторов все обычные свойства SQL-транзакций, такие как целостность и продолжительность, сохраняются. По умолчанию каждый оператор утверждается после его выполнения, поскольку, как видно из следующего примера, значение conn.setAutoCommit() установлено равным true. Утверждение после выполнения каждого оператора можно гарантировать любым из следующих способов (при желании можно применить метод conn.rollback(), чтобы выполнить откат оператора):

```
conn.setAutoCommit(false);
```

или

```
conn.commit();
```

Ниже приведен простой пример, который демонстрирует использование операторов commit() и rollback():

```
conn.setAutoCommit(false);
     Statement stmt = conn.createStatement();
stmt.executeUpdate("INSERT INTO employees
VALUES('Alapati','Valerie','Nicholas',50000 )");
       conn.rollback();
stmt.executeUpdate("INSERT INTO employees
VALUES('Alapati','Nina','Nicholas',50000)");
       conn.commit();
```

Обработка ошибок

Все программы должны содержать встроенный обработчик исключений. Это особенно важно для тех операторов DML, которые изменяют состояние базы данных. Один из способов достижения этой цели предусматривает применение оператора `rollback()` при возникновении ошибки, что позволяет отменить все частичные изменения.

Для перехвата ошибок можно использовать метод `SQLException()`. В программах Java для генерирования (или выдачи) исключения используют блок кода `try`, а блок `catch` будет "перехватывать" выданное подобным образом исключение. Пример блока кода Java, который иллюстрирует эти концепции, приведен в листинге 11.9.

Листинг 11.9. Обработка ошибок в программе Java

```
try { conn.setAutoCommit(false);
      stmt.executeUpdate(" " +
        "(Disney World', 'MickeyMouse', 2.00)");
      conn.commit();
      conn.setAutoCommit(true);
    }
catch(SQLException ex) {
      system.err.println("SQLException: " + ex.getMessage());
      conn.rollback();
      conn.setAutoCommit(true);
    }
```

Полная программа

Пример программы, в которой все концепции, рассмотренные в предшествующих разделах, сведены воедино, приведен в листинге 11.10. Вначале эта программа регистрирует тонкий драйвер Oracle и с его помощью подключается к базе данных. Затем она обновляет некоторые строки в таблице и применяет результирующий набор для вывода данных.

Листинг 11.10. Полная программа Java, использующая JDBC

```
/* импорт пакетов java */
import java.sql.*;
public class accessDatabase{
   public static void main(String[] args)
     throws SQLException {
     stringfirst_name,last_name ;
     number salary ;
     connection c = null;
/* регистрация драйвера Oracle */
     try {
       class.forName("oracle.jdbc.driver.OracleDriver");
       c = DriverManager.getConnection(
          "jdbc:oracle:thin:@prod1:1521:finprod",
          "user", "user_passwd");
```

```
/* создание объекта statement */
          statement s = c.createStatement();
          c.setAutoCommit(false);
          s.executeUpdate("CREATE TABLE employees " +
            "(first_name VARCHAR2(30), last_name VARCHAR2(20),salary NUMBER)");
          s.executeUpdate("INSERT INTO employee VALUES " +
            "('nicholas', 'Alapati', 50000 )");
          c.commit();
          c.setAutoCommit(true);
/* результирующий набор */
          resultSet rs = s.executeQuery("SELECT * FROM Employees");
          while( rs.next() ){
            first_name = rs.getString("first_name");
            last_name = rs.getString("last_name");
            salary = rs.getFloat("salary");
            System.out.println(first_name + last_name + " works for " +
               Manager + " salary is:" + salary");
          }
/* обработчик исключений */
     } catch (ClassNotFoundException ex){
          system.out.println(ex);
     } catch (SQLException ex){
          if ( c != null ){
            c.rollback();
            c.setAutoCommit(true);
          }
          system.out.println("SQLException caught");
          system.out.println("---");
          while (ex != null){
            system.out.println("Message : " + ex.getMessage());
            system.out.println("SQLState : " + ex.getSQLState());
            system.out.println("ErrorCode : " + ex.getErrorCode());
            system.out.println("---");
            ex = ex.getNextException();
          }
       }
     }
   }
```

Поскольку в данном случае нас интересует администрирование базы данных Oracle, а не особенности программирования, эта простая программа Java, в основном, призвана привлечь внимание к аспектам соединения с базами данных Oracle.