## 1、安装

作为安装流程的一部分，安装程序还会运行网络配置助手（Net Configuration Assistant）以及数据库配置助手（Database Configuration Assistant）程序。这样，当Oracle数据库安装流程结束时，数据库实例已经开始运行。

Oracle完整的安装过程既包括安装软件，也包括配置和启动关键性的服务。

比较重要的数据库命名包括：

* 磁盘或装载点名称。
* Oracle软件和数据库文件的目录结构。
* 数据库文件名：控制文件、数据库文件以及redo日志文件。

### 1.1最优灵活体系结构（OFA）

最优灵活体系结构是数据库文件命名的基础。

OFA中有清晰的标准来处理同一台机器上部署多个或者不同版本Oracle实例的情况。它还包含有关装载点、目录结构、文件名以及脚本技术的建议。

## 2、创建数据库

对于任何新创建的数据库，用户都应该采取两步走的方法。首先要理解数据库的目的，其次是要使用适当的参数来创建数据库。

### 2.1规划数据库

用户应该在亲自创建数据库之前花一些时间来掌握Oracle数据库的目的。思考一下，数据库将应用在什么样的环境下，它将会容纳多少数据。用户需要了解将要使用的底层硬件——CPU的数目和类型、内存的大小、磁盘的个数以及磁盘控制器等。数据库是存储在磁盘上，因而对磁盘容量和I/O子系统做合适规划可以避免许多调整问题。

规划数据库及相关硬件的工作需要深入了解未来工作负荷的大小以及系统将要执行的任务类型。在设计数据库及其硬件配置的时候，需要考虑以下几点：

1. 数据库将会有多少用户：有多少用户会同时访问数据库？有多少用户会并发执行事务或查询？
2. 数据库是否需要支持OLTP应用或者数据仓库：这个因素的不同为数据库服务器带来的活动类型及规模也是不同的。例如，在线事务处理（OLTP）系统通常会有大量的用户在执行小型事务，而数据仓库通常用少量的用户在执行规模较大的查询。
3. 预计数据库对象会有多大？数量有多少：这些对象初始有多大？预计以后会有多大的增长率？
4. 各种数据库对象的访问模式是什么样的：相比其他对象而言，有些对象可能被频繁访问。理解数据库活动的规模和类型对规划数据库调整数据库非常关键。有些人会采用一种叫做CRUD的矩阵来表示数据库的创建、读、更新和删除活动，甚至用它来估计一个商业事务对每个关键对象的操作数量。这些估计可能以分钟为单位，以小时为单位，以天为单位，或者以系统中的某个周期为单位。例如下表中列出了一个简单的员工更新事务的CRUD矩阵，其中的数字指明了事务对某个兑现执行某个操作的次数。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 对象 | 创建 | 读 | 更新 | 删除 |
| EMP | √ | √ |  |  |
| DEPT |  | √ |  |  |
| SALARY |  | √ | √ |  |

1. 现在有多少硬件？随着数据库的扩张，应该加多少：Oracle 9i引入了可恢复的控件分配这样一个概念。当一个操作遇到了额空间不足的情况，如果这个会话的可恢复语句选项已经被打开，这个操作将会被悬挂一定的事件，等待操作者纠正空间不足的这种情况。用户甚至可以使用这个选项来创建一个AFTER SUSPEND触发器，当操作被悬挂的时候，触发器执行。借由Oracle 10g引入的自动化存储管理（ASM）特性，用户可以在不中断数据库业务的情况下添加额外的磁盘空间或者拆掉磁盘。有了它，即便初期对存储需求的判断有误，由此带来的惩罚（即数据库宕机时间）也会被降到最少。
2. 可能性需求是什么样的：冗余的措施之一就是添加备份的磁盘驱动器，为了按要求提供冗余性，你还需要提供哪些元素的冗余？ASM还提供数据的自动化镜像，它可以帮助提供数据的弹性存储。
3. 性能需求是什么样的：用户期望的响应时间是多少，能够给他们响应时间是多少？你是否根据平均响应时间、最大响应时间、负荷高峰期响应时间、总的吞吐量或者平均负荷来衡量性能。
4. 有什么样的安全要求：应用程序、操作系统、或者Oracle数据库（或者三者之中的任意结合）是否实施了安全策略？

1）评估的价值

即便不确定诸如大小和用途之类的属性，也要好好猜测一下初始值和增长率，并将猜测记录下来。随着数据库的发展，可以将初始估计和最新的情况比较以下，从而对目前状况作出及时的响应并对下一步作出有效的规划。例如假设你估计某个表初始大小5GB就可以了，每年将会增加3GB，但是当实际开始运行的时候，你可能发现这个表只有3GB，六个月之后，有发现这个表已经增长到8GB了。这个时候，你需要修改数据库规划，满足其快速增长的需求，从而避免磁盘空间不足的问题，另外，有必要将数据库当前的运行状况与当初数据库大小、增长率以及使用类型等方面做出的预估进行比较。将早期对数据库的推测记录在案对后来的实际运维非常有用。

对于可用性和性能这样的关键需求而言，上面讲述的方法也是行之有效的。如果没有确切的需求，那就做一些假设，并把这些假设记录归档。这些核心的需求经会很大程度上影响你在冗余和容量方面做出的决定。随着系统的演进以及需求的日益清晰，之前基类的这些关键决策标准对于理解自己已做出的选择和将要做出的选择具有决定性意义。

自动化工作负荷仓库（AWR）最初是在Oracle 10g上使用，它维护着工作负荷以及性能相关的历史数据。自动化数据库诊断监视器（ADDM）将会利用这些信息监测系统性能异常情况。用户也可以利用AWR来跟踪工作负荷正在发生的变化。

### 2.3创建数据库的工具

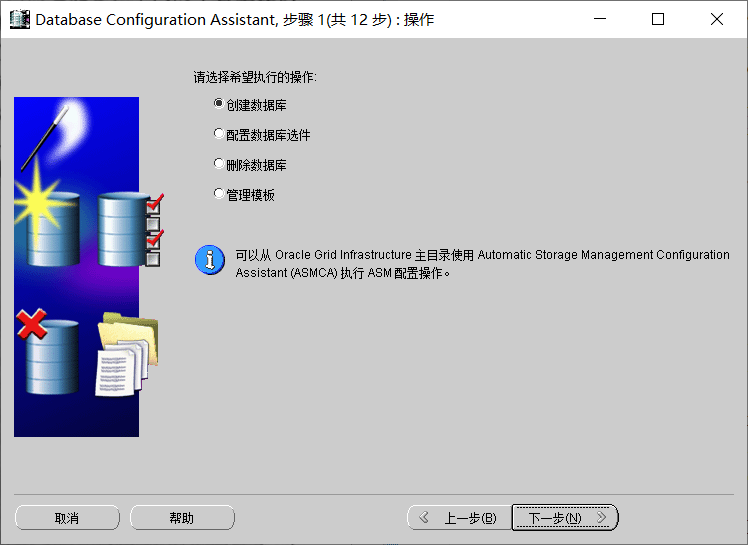
创建一个Oracle数据库有两种基本的方法：

1. 使用图形化Oracle数据库配置助手。
2. 运行字符模式的脚本。

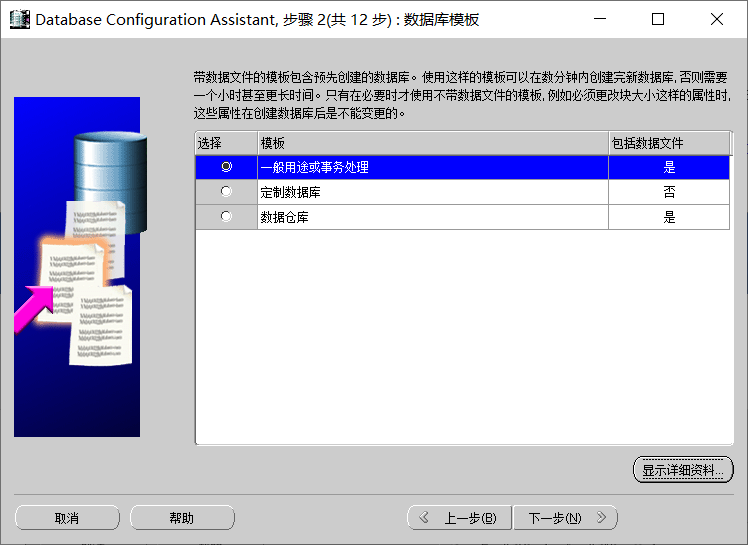
Oracle的发布中包括一个名叫Oracle数据库配置助手的图像化工具。它可以独立运行，也可以由Oracle安装程序运行。由于是用Java编写的，因而它能够在不同的平台上提供相同的外观和感观。使用这个助手工具可以方便快捷地创建，修改，或删除数据库。利用它，用户可以创建典型的进行了预配置的数据库（仅需要用户最少的输入）

，也可以创建自己定制的数据库（需要用户做出一些选择，并回答一些问题）。标准的安装过程通常会自动调用数据库配置助手。





如果选择创建数据库，那么可以选择想创建的数据库类型，不同类型的数据库在创的时候使用不同的默认配置。



创建数据库的另一张方法就是创建或编辑SQL脚本，并在里面包含各种需要执行的命令。很多Oracle DBA都有自己首选的脚本，需要的时候，随时进行编辑。在Oracle 7和Oracle 8中，用户可以在一个名为Server Manager的字符模式工具中执行这些脚本；从Oracle 8i开始，用户可以使用SQL PLUS。Oracle软件光盘还包含一个名为BUILD\_DB.SQL的示例脚本，Oracle文档对该脚本也进行了描述。不过，今天大多数用户都会选择使用图形化的标准安装程序来创建数据库。

## 3、配置Oracle Net

Oracle Net是一个软件层，不同的物理机器可以借助这个软件层实现相互间的通信，具体而言就是实现对Oracle数据库的远程访问。

Oracle Net运行在客户端或者数据库服务器上，为客户端和服务器提供基于常见网络协议的通信服务。Oracle Net能够支持不同网络协议之前的转换。例如，借助它，使用LU6.2通信协议的客户端可以和使用TCP/IP通信协议的服务器通信。

Oracle Net能够提供位置透明性——即客户端应用程序不需要事先知道服务器的物理位置。Oracle Net软件层处理了所有的通信工作，这意味着用户可以将数据迁移到另外一台机器上，只需简单地根据需要更新Oracle Net的配置细节。这个时候，客户端程序仍然能够连接到数据库，不需要做任何更改。

Oracle Net支持服务名或者别名。客户端通过提供服务名称或者Oracle Net别名来指定它所期望连接的数据库，而不需要指定实际的机器或者数据库实例。Oracle Net利用用户提供的服务名称来查找实际的机器和Oracle实例，并透明地将客户端的消息路由到合适的数据库。

### 3.1解析Oracle Net服务名称

下面的Oracle Net配置选项能够将用户指定的服务名称解析为对应服务器主机和数据库实例：

* 本地名字解析：用户需要为本地名字解析在每个客户机器上安装一个TNSNAMES.ORA文件。这个文件列出了每个Oracle Net别名对应的主机和Oracle实例；如果实际数据库的位置发生变化，你必须手工维护客户机器上这个文件。实际的网络拓扑肯定会随着时间的变化而发生变化，因而使用这个选项会带来日益繁杂的维护负担。如果你使用Oracle互联网目录，就不再需要TNSNAMES.ORA这个文件。
* Oracle名字服务：Oracle早期的发布就已经支持Oracle名字服务了，这个服务取代了必须安装在每个客户端上的TNSNAMES.ORA文件。以上是它的优点。它的缺点是Oracle名字服务是私有解决方案。由于Oracle互联网目录能够提供这个功能，而且是基于相关标准的，因而自Oracle 9i发布之后，Oracle宣布将Oracle名字服务作废。
* Oracle互联网目录：包括Oracle环境在内的许多应用场合都需要一个集中的名字服务。实际上，目前已有定义好的标准可用于访问此类信息，这个标准就是轻量级目录访问协议（LDAP）。就Oracle 11g而言，Oracle互联网目录（OID）是Fusion中间件的一部分。OID是一个具有LDAP功能的目录，它可以取代之前使用的Oracle名字服务。OID还可以用于其他目的，例如，它可以用于实现Oracle应用服务器门户产品的单点登录功能。从Oracle 10g开始，可以从目录服务中导出目录项用于创建本地TNSNAMES.ORA文件；这个文件可以用于没有使用或者无法使用目录服务的客户端。
* 主机名字：客户端可以使用实例所运行的的主机的名字来定位实例。TCP/IP网络提供了一种有效的机制来提供将主机名解析为IP地址的服务。举个例子，域名服务（DNS）可以将主机名翻译成IP地址，完成Oracle名字类似的功能。自从Oracle 10g开始，用户可以通过主机名、主机域名或者IP地址来定位数据库实例；不过这种方法建立的连接并不支持注入连接池等高级服务。
* 第三方名字服务：Oracle Net提供了与外部或者第三方名字服务和鉴权服务，例如，Kerberos、Radius等交互的接口。使用这些服务可能需要Oracle高级安全特性。

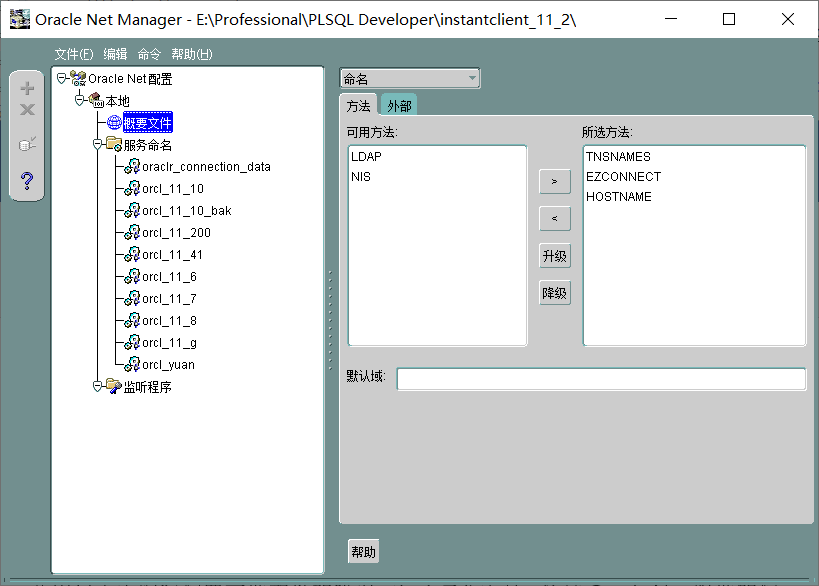
这些名字解析选项并不相互排斥。例如，用户可以同时使用Oracle互联网目录和本地名字解析（TNSNAMES.ORA）。在这种情况下，用户需要在SQLNET.ORA文件中指定Oracle使用这些选项进行名字解析的顺序（例如，首先查看OID，如果无法解析服务名字，则查看TNSANMES.ORA文件）。通常，用户会将OID用于标准的共同的数据库服务，如email，而将TNSNAMES.ORA用于客户端特定的数据库服务，如某个特定的开发数据库。

用户也可以选择Oracle称之为快捷连接的名字方法直接连接Oracle数据库。这个方法使用的是主机名字或者Oracle服务器的TCP/IP标识符和Oracle数据库实例的名字。它只适用于TCP/IP网络，仅被推荐用于主机表示很少变动的小型安装中。

### 3.2Oracle Net管理器

和数据库配置助手一样，Oracle Net管理器是用Java语言编写的，在不同的平台中能够提供相同的外观和感观，通常也是由安装程序首先调用它。Oracle Net配置文件语法非常特殊，由多级相嵌的括号组成。使用Oracle Net管理器可以防范用户手工编写配置文件常见的错误。有了这个工具，各种Oracle Net组件的配置可以自动完成。





1）自动发现及代理

从Oracle 7.3开始提供自动发现特性。Oracle能够自动找到新的数据库。从Oracle 8i开始，通过安装程序和Oracle Net管理器能够平稳的协同工作，自动完成Oracle Net的网络配置。

调试网络问题：

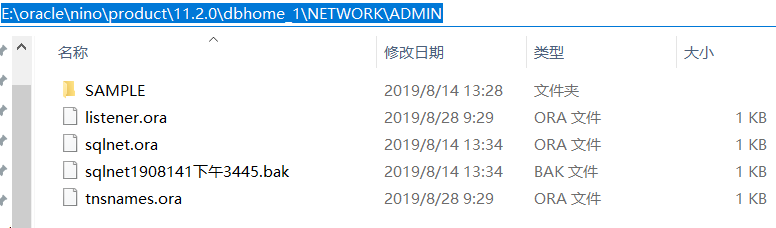
如果网络出现问题，调试首先要做的步骤之一就是查看自动生成的Oracle Net文件，而不是手工编写的文件。如果有疑惑，可以备份当前的配置文件，并使用Oracle Net管理器重新生成这些文件。事实上，当Oracle支持人员来帮助客户解决Oracle Net问题的时候，他们首先会问的问题之一就是这些文件是否是手工生成的。

Oracle智能代理是Oracle Net具有自动发现特性的关键组件。它作为一个功能代理来完成发现数据库并与数据库协同工作的功能。例如，这个代理了解主机上各种Oracle实例，并能够处理关键性的管理工作，如监控数据库中的某个时间和正在运行的作业，它提供了自动发现的一个中央点：Oracle Net通过询问代理发现实例和数据库。

### 3.3Oracle Net配置文件

Oracle Net需要好几个配置文件来配置网络。这些文件的位置如下：

* Windows中，目录为ORACLE\_HOME\NETWORK\ADMIN。
* UNIX上，目录为ORACLE\_HOME\NETWORK\ADMIN。



用户可以将这些文件存放在其他位置，不过必须通过环境或者系统变量TNS\_ADMIN来指定这个非默认的位置。Oracle会使用TNS\_ADMIN来定位这些文件。

配置文件有以下几个：

* listener.ora：包含配置Oracle Net监听器的详细信息，例如，监听器服务的实例或者服务。正如名字所暗示的那样，监听器负责监听希望通过网络访问Oracle数据库的客户连接请求。
* tnsnames.ora：负责将服务名称翻译成特定的机器地址和Oracle实例。（如果是使用Oracle名字服务或者OID，则不需要这个文件）这个文件对Oracle Net的位置透明特性非常关键。如果用户将数据库从一台机器移到另一台机器，对各种客户端上的tnsnames.ora文件进行更新，将其服务名对应的机器地址修改为最新的地址，即可。
* sqlnet.ora：提供默认的配置信息和其他琐碎的配置细节。例如，sqlnet.ora包含用户网络的默认域名。
* ldap.ora：对于Oracle 8i及以后的发布而言，ldap.ora这个文件包含了LDAP目录服务如Oracle互联网目录相关的配置信息。具体来说，这个信息包括LDAP目录服务器的位置以及服务器默认的管理上下文。从Oracle 10g开始，Oracle数据库不再要求将LDAP服务器注册到域名服务器（DNS）上。

Oracle 9i增加了服务器参数文件SPFILE。这个文件记录着Oracle 9i运行期间用户使用ALTER SYSTEM命令修改过的系统参数。有了SPFILE，新设置的参数值的保存，在下次用户重启Oracle实例之后，这些参数值得到应用。用户可以指定对系统参数的改动是持久的（如是持久的，那么它将被保存在SPFILE文件中），还是临时的。

SPFILE文件作为一个二进制文件被保存在服务器上。在默认情况下，Oracle 9i及之后的数据库实例在启动时都会先查找SPFILE文件，再查找INIT.ORA文件的实例。

SPFILE也可以被保存在共享磁盘中，这样它可以用于初始化Oracle实时应用程序集群配置中的多个实例。

## 4、启动数据库

在Windows上，只需启动Oracle服务（或者将Oracle服务指定为在及其启动时自动启动）；

在UNIX以及Linux上，只需从SQL Plus或者企业管理器上执行STARTUP命令。

虽然启动数据库看起来是一个单独的动作，但它涉及在几个不同阶段对数据库实例和数据库的操作。下面将介绍数据库启动过程中自动执行的操作：

1）启动数据库实例

首先，Oracle从服务器上的SPFILE或者INIT.ORA文件中读取实例初始化参数。

其次，Oracle为系统全局内存区分配内存并启动实例的后台进程。

到目前为止，数据库中的物理文件都没有被打开，实例处于NOMOUNT（非装载）状态。（注意：在Oracle 10g和Oracle 11g中，初始安装阶段，SPFILE文件中需要定义的参数个数已经大大减少了）

有些问题会影响数据库实例的正常启动。例如，初始化文件中可能存在错误，或者操作系统可能无法安装要求的大小，为SGA分为共享内存。用户要启动数据库实例可能需要操作系统或者密码文件授予其特殊的权限SYSOPER或者SYSDBA。

2）装载数据库

数据库实例负责打开数据库控制文件。初始化参数CONTROL\_FILES为实例指出了控制文件的位置。

到目前为止，仅有控制文件被打开。这个时候，实例处于MOUNT状态，仅有数据库管理员可以访问数据库。在这种状态下，DBA仅能够执行某种类型的数据库管理任务。例如，DBA可以移动或者重命名某个数据库文件。控制文件中列出的数据文件在MOUNT状态下并没有被打开。DBA可以执行ALTER DATABASE命令来重命名数据文件。这个命令将会用新的数据文件名来更新控制文件。

3）打开数据库

数据库实例利用控制文件中的信息打开redo日志文件和数据文件。这个时候，数据库已经完全打开，用户可以访问。

## 5、关闭数据库

从逻辑上来讲，关闭数据库使之停止服务所涉及的步骤和动作与之前启动数据库步骤和动作相反：

1）关闭数据库

Oracle迅速将所有改动过尚未从SGA高速缓存写回数据文件的数据块写回数据文件。同时，Oracle将保留在redo日志缓存中的任何相关信息写入redo日志文件。

接着会对数据文件做检查点操作，将数据文件的头部标记上数据库关闭时间。

随后关闭数据文件和redo日志文件。

这个时候，任何用户都不能访问数据库。

2）卸载数据

Oracle实例负责卸载数据库。Oracle会更新控制文件中的相关项以记录一个干净的关闭操作，随后关闭这些控制文件。

这个时候，整个数据库被关闭了。只有数据库实例在运行。

3）关闭实例

Oracle软件停止实例后台进程的运行，释放用于SGA的共享内存。

在某些情况下（例如，出现机器故障或者DBA异常关闭数据库实例），数据库可能无法干净的关闭。如果这种情况发生，Oracle已经没有机会将修改的数据块从SGA写回数据文件。当Oracle重启之后，实例将会检测到曾经发生过宕机，并利用redo日志自动执行宕机恢复操作。宕机操作可以保证所有提交的事务能够被完成，所有未提交的或者正在进行中的事务将会被清除。未提交的事务在redo日志被应用之后会被确定下来，并自动执行回滚。

## 6、访问数据库

前面描述了数据库启动和关闭的流程。但是数据库仅是完整系统的一部分——用户需要用客户端进程来访问数据库，即便是客户端进程与数据库在同一台物理机器上，也是这样。

### 6.1服务器进程和客户端

为了访问数据库，用户需要连接到数据库实例。数据库实例提供了对数据库的访问。访问数据库的程序实际上由两个不同的部分组成——客户端和服务器进程——服务器进程负责连接到Oracle实例。例如，运行Oracle字符模式工具SQL Plus，我们会发现它有两个进程：

1. SQL Plus进程本身，它作为客户端。
2. Oracle服务器进程，有时候又叫做影子进程，它提供了Oracle实例的连接。

1）服务器进程

Oracle服务器进程总是与数据库实例进程运行在同一台计算机上。服务器进程关联了SGA共享内存，它可以从中读取数据，也可以向其中写数据。

服务器进程是为客户端进程而工作——它代表客户端执行读取和回传请求数据，接收和应用数据库变更等操作。例如，当客户端期望从待定的数据块中读取一行数据的时候，服务器进程负责找到这个数据块，并从数据库高速缓存中获取它，或者从正确的数据文件中获取它并将它载入数据库高速缓存。如果用户请求变化，服务器进程会修改高速缓存中相关的数据块，生成必要的redo信息，并将这些信息存储在SGA的redo日志缓存中。不过，服务器进程并不会将日志缓存中的redo信息写入redo日志文件，也不会将高速缓存中的redo信息写入redo日志文件。这些工作分别由日志写进程（LGWR）和数据库写进程（DBWR）来完成。

2）客户端进程

客户端可以与数据库实例运行在同一台机器上，也可以单独运行。计算机之间的网络连接为客户端和数据库实例之间的通信提供了一条渠道。两种情况下通信的基本概念是相同的——客户端进程和服务器进程都会参与搭建客户端与数据库的这个通信渠道。

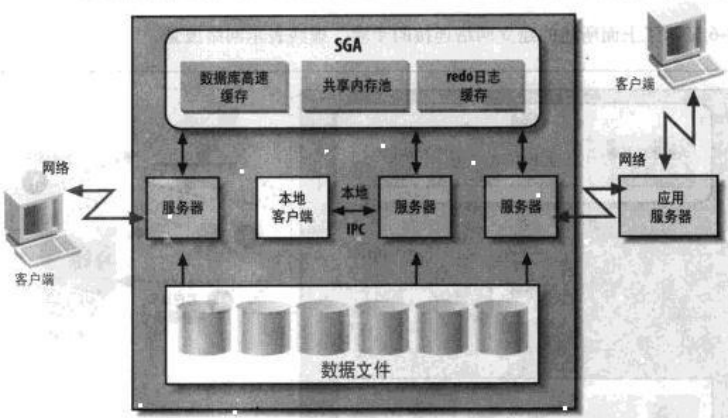
* 当这两个进程运行在同一台机器上，Oracle会使用基于进程间通信（IPC，Inter-Process Communication）的本地通信；
* 当这两个进程运行在不同的机器上，Oracle会使用Oracle Net实现这两台机器上的网络通信。

### 6.2作为客户端的应用服务器和Web服务器

不要以为Oracle是严格意义上的客户端/服务器模型的数据库。Oracle过去曾经是客户机/服务器计算模型的先锋。但当我们开始思考设计Web和应用服务器的多层计算模型的时候，这个客户端的概念有所变化。“客户端”进程变成了中间层，即应用服务器。我们可以从逻辑上把任何连接到数据库实例的进程看作是客户端，因为它们都接受数据库服务。请不要把这里使用的客户端概念与多层配置中的客户端相混淆。在多层模型中最终的客户端就是提供用户界面的某种程序——例如，运行Java的浏览器。

Oracle应用服务器是整个Oracle平台的一部分。根据设计，它可以用作中间层。Oracle应用服务器和Oracle数据库能够无缝地协同工作，并共享这些技术。

下图显示了用户连接到Oracle实例访问数据库的两种方式，即两层模型和三层模型；这里还涉及本地和网络两种通信方式。这幅图突出了服务器进程连接模型，淡化了后台进程之间的交换。其中，左侧是一个传统的客户端/服务器两层连接模型，右侧是一个使用了应用服务器的三层连接模型，中间是一个本地客户端连接模型。两层模型和三层模型使用网络与数据库通信，而本地客户端使用本地IPC与数据库通信。



### 6.3Oracle Net与建立网络连接

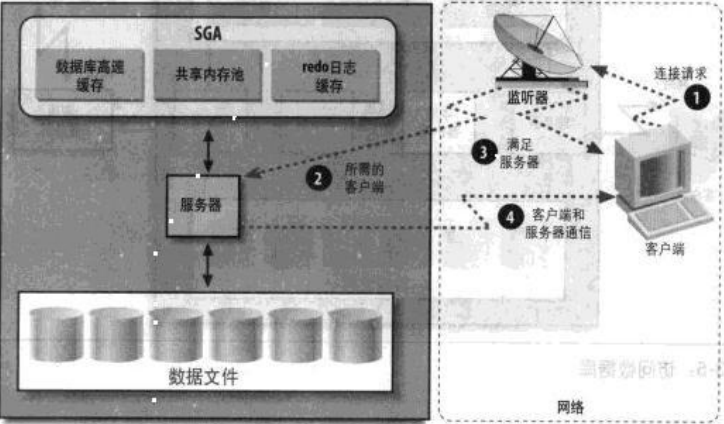
客户端进程借助某种类型的网络建立到服务器进程间的连接。客户端进程具体是如何与Oracle服务器进程建立通信的呢？

负责将Oracle客户端进程与服务器进程建立通信联系的组件是Oracle Net监听器。该监听器负责“监听”从外界进来流向一个或多个数据库实例的连接请求。监听器组件本身并非Oracle实例的一部分——它只负责将连接请求交给数据库实例。监听器的启动和停止独立于数据库实例。如果监听器停止运行而数据库实例仍在运行，那么远端客户就会因为没有监听器的指引而无法找到数据库实例。如果监听器在运行而数据库实例已经停止运行，那么监听器则无法传递客户请求。

监听器的简单功能：

1. 客户端通过网络联系监听器。
2. 监听器侦测到达的请求，并将侦测到的请求交送给Oracle服务器进程。
3. 监听器将服务器和客户端的网络地址告诉对方。
4. 客户端和服务器直接通信，监听器不干预。

下图显示了建立网络连接的步骤。虚线表示网络流量。



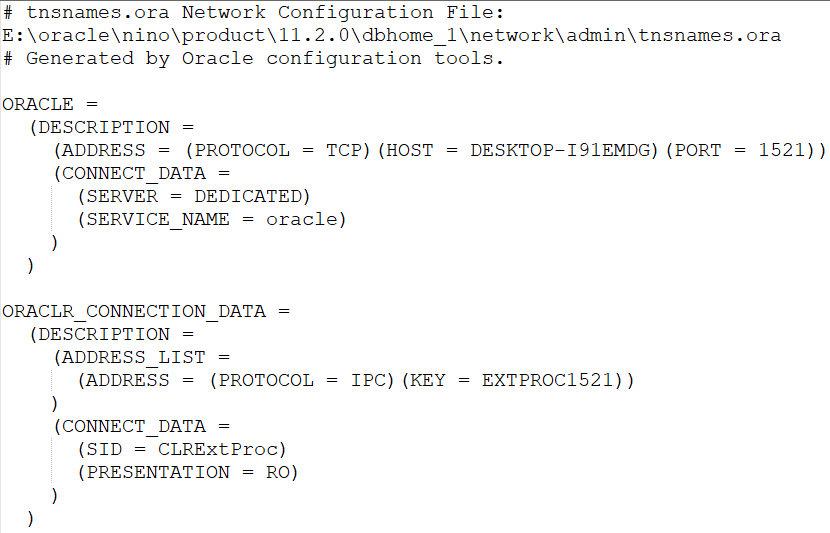
### 6.4共享服务器/多线程服务器

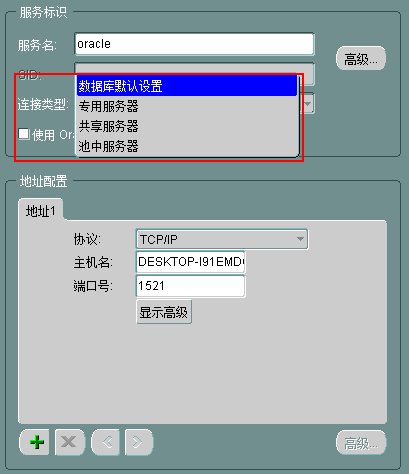
上图所示的服务器进程是专用进程，它们仅服务于一个客户端进程。就这样的应用模型而言，如果应用程序由1000个客户端，那么Oracle实例就需要1000个对应的服务器进程。这些进程每个都要使用内存、CPU等系统资源。当用户数量变得很大的时候，就会消耗大量的系统内存。为了支持日益增长的扩展需求，Oracle 7引入可多线程服务器，在Oracle 9i中又叫做共享服务器。

共享服务器使得Oracle实例可以在大量的用户间共享一组服务器进程。每个进程都使用共享服务器，而不是专用服务器，从而极大地降低了大量用户对系统整体资源的需求。

在许多系统中，客户端并不会持续地使用它们的服务器进程，这样的事常常发生，例如用户正在从数据库中读取数据。在专用服务器模型中，当客户端不再使用它的服务器进程的时候，这个服务器进程即便是不做任何有用的工作仍然占有着系统资源。而在共享服务器模型中，共享服务器可以利用非活跃客户端的资源来服务其他活跃的客户端进程。

用户可以不必为一个Oracle实例在共享服务器模型和专用服务器模型之间做互斥选择。Oracle能够混合使用这两种模型，客户端可以连接到一个或者其他的服务器进程。用户可以在Oracle Net配置文件中选择服务器模型，其中一个服务器名称代表使用专用服务器，另一个代表使用共享服务器。





客户端使用的服务器进程类型对其自身来说是透明的。从客户端的角度来说，多线程服务器/共享服务器运行在数据库服务器的内部。监听器能够同时处理到专用的服务器和到共享服务器的连接请求。

监听器在建立到共享服务器的连接的时候，采用的步骤略有不同，它会涉及数据库实例派遣器后台进程以及共享服务器本身：

派遣器：

客户端必须依赖始终有效的服务器进程实现数据库连接的建立。由于共享服务器进程可能需要服务其他客户端，而派遣器专门负责接收客户的请求，这个时候客户端可以连接到派遣器。每种网络协议都会有单独的派遣器（例如，TCP/IO派遣器）。对客户端而言，派遣器就是服务器进程的代理。每个客户端都会直接连接到派遣器而不是服务器。派遣器接收来自客户端的请求，并把它们放在请求队列。这个队列就是SGA中的一个内存结构，每个数据库实例都有一个。

共享服务器：

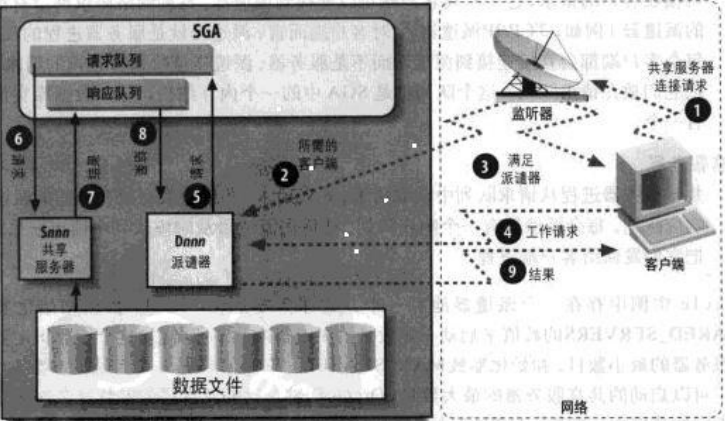
共享服务器进程从请求队列中读取请求，处理请求，并把结构存放在合适的派遣器响应队列。每个派遣器有一个响应队列。然后由派遣器从响应队列中读出结果，并把它们发回给客户端进程。

Oracle实例中存在一个派遣器池和一个共享服务池。Oracle根据初始化参数SHARED\_SERVERS的数值来启动一定数量的派遣器和共享服务器。这个参数指定了共享服务器的最小数目。初始化参数MAX\_SHARED\_SERVERS是一个可选参数，它指定可可以启动的共享服务器的最大数目。请求负荷重的时候，Oracle会多启动一些进程；请求负荷降低的时候，Oracle会逐步将进程数目减少到SHARED\_SERVERS指定的最小数目。

客户端和服务器进程是如何建立连接的，共享服务器进程和专用服务器进程在使用上的不同之处：

1. 客户端通过网络联系监听器。
2. 监听器侦测到达的请求，并根据Oracle Net的配置确定这是一个到共享服务器的请求。接下来，监听器将客户请求交送给网络协议特定的派遣器，而不是专用服务器。
3. 监听器把客户端和派遣器的地址告诉对方。
4. 一旦客户端和派遣器知道如何找到对方，它们就开始直接通信，不再需要监听器。客户端将所有的工作请求直接发给派遣器。
5. 派遣器将客户请求存放在SGA中的请求队列里。
6. 可用的共享服务器读取请求队列中的请求，并完成指定工作。
7. 共享服务器将请求处理的结果放回提交请求的那个派遣器响应队列。
8. 派遣器从队列中读取结果。
9. 派遣器将结果发送给客户端。

共享服务器的使用步骤。虚线表示网络流量：



### 6.5共享服务器进程和专用服务器进程的会话内存

Oracle中存在会话内存或者状态这样一个概念。状态信息是描述Oracle会话当前状态的基本数据。例如，状态信息包括有关会话执行SQL语句的信息。在使用专用服务器的时候，这个状态信息被存储在专用服务器使用的私有内存区中。由于专用服务器仅服务于一个客户端，因而这种做法是可能的。私有内存区的专业术语叫做程序全局内存区（PGA）。

然而，在使用共享服务器的时候，任何服务器都可以服务特定的客户端。会话状态不能存储在共享服务器的PGA中。由于会话会在不同的共享服务器间迁移，因而所有服务器都必须能够访问会话状态。为此，Oracle将这些状态信息存放在系统全局内存区（SGA）中。

所有的服务器都能够从SGA中读取信息。将状态信息存放在SGA中使得会话及其状态能够从一台共享服务器迁移到另外一台共享服务器。共享服务器从请求队列中读取请求，从SGA中读取会话状态信息，根据处理需要更新会话状态，并在处理结束时把结果放回SGA中。

请求队列和响应队列，以及会话状态，需要在SGA中分配另外的内存区，因而在较早的Oracle发布中，如果要使用共享服务器，就必须手动地多分配一些内存。在默认情况下，用于存储共享服务器会话状态的内存区是从共享内存池中分配的。或者，也可以将大型内存池配置成单独用于共享服务器的内存区。使用大型内存池可以避免与共享SQL、字典缓存以及其他共享内存功能协同使用内存的开销。这也使得我们可以借助大型内存池的内存管理功能避免在共享内存的分配与访问方面产生竞争。从Oracle 10g开始，Oracle数据库默认自动管理共享内存。当你设置了初始化参数MEMORY\_TARGET时，Oracle 11g默认自动管理SGA和PGA的大小。

### 6.6有关共享服务器的数据字典信息

Oracle数据字典视图包括四个层次，分别为X$内部表、基础数据字典表、数据字典视图和动态性能视图。其中几个视图包含了与MTS操作相关的信息：

V$SHARED\_SERVER\_MONITOR

这个视图包含与共享服务器相关的动态信息，例如，连接的最大个数，负荷变化时启动和停止服务器的个数。

V$DISPATCHER

这个视图包含共享服务器使用的派遣器进程的详细信息。它可以确定派遣器当前有多忙。

V$SHARED\_SERVER

这个视图包含共享服务器使用的共享服务器进程的详细信息。它可以确定共享服务器当前有多忙，从而帮助用户设置合适的谷值和峰值。

V$CIRCUIT

用户可以把客户端到派遣器，又从派遣器到共享服务器的路由想象成一条虚拟电路。这个视图详细描述了用户连接的虚拟电路。

## 7、Oracle的运行

这一节通过一个例子来考察Oracle数据库是如何响应客户端的请求。这个例子描述了用户向数据库中添加新的信息（执行事务）的流程。

### 7.1Oracle与事务

事务是客户端插入、更新或者删除数据的工作请求。这些操作语句是SQL语言的子集，也叫做数据操作语言（DML）。事务的处理必须保证它们的完整性。

基本概念：

事务是逻辑的和完整的

在数据库术语中，事务是由一个或多个数据变更操作组成的逻辑单元。事务可能由多个INSERT、UPDATE和DELETE语句组成，可能影响多个表格中的数据。这些变动操作作为一个完整的工作单元，要么成功地被执行，要么失败。事务从第一个DML语句开始到一个提交命令或者一个回滚命令结束。

（注意：Oracle还支持自治事务。所谓自治事务是说存在于一个更大的事务内，

它的工作能够被提交或或者回滚。这个概念非常重要，有了它，可以在更大的事务内提交事务，而不用毁坏这个更大的事务。）

提交或者回滚

一旦用户为自己的事务提供了数据，他可以提交这个事务，使相关的变动永久生效；也可以回滚事务，撤销相关的变动。

系统变动序号（SCN，System Change Number）

保持数据库完整性的一个关键因素就是清楚事务的顺序。例如，Oracle为了防止后来的事务不经意覆盖了前面事务的变动，它必须知道哪个事务先发生。为此，Oracle使用了一个逻辑时间戳（即系统变动序号）来跟踪事务发生的顺序。另外，SCN还可用于实时多版本的读一致性。

回滚段

回滚段是Oracle数据库中的结构，存储着事务用于回滚的“撤销”消息。借助这些信息，数据块能够恢复到事务启动前的状态。当事务开始修改数据块中的某些数据的时候，它首先将数据块中旧的数据写入回滚段。回滚段与redo日志不同。redo日志用于记录数据库中所有事务，以备系统发生故障时恢复数据库之用；而回滚段用于事务的回滚和保证多版本读操作的一致性。

回滚段与redo日志完全不一样。redo日志用来将事务记录到数据库，并且出现系统错误时恢复数据库；而回滚段为事务和读操作一致性提供回滚。

回滚段和表格与索引一样，数据块都缓存在SGA中。如果一段时间内回滚没有被使用，它们在缓存中会过期，进而被写入磁盘。

Oracle 9i引入了回滚段自动管理功能。在Oracle数据库的早期版本中，DBA必须显示地创建和管理回滚段。Oracle 9i中，由于可以使用撤销表空间，因而用户能够选择自动化管理所有的回滚段。有了自动化撤销管理，用户还可以指定保存撤销信息的时间长度；如果你计划使用flashback查询，那么这个特性会非常有用。Oracle 10g增加了撤销管理保存时间顾问工具。

5）快速提交

由于撤销信息是在用户提交Oracle事务时被写入日志，因而他们可用于加速数据库的操作。当用户提交了一个事务，Oracle通过下面两种方法中的一种将数据变动写入磁盘中的数据库：

* 将事务修改过的数据块全部写回各自数据文件。
* 仅是将redo信息写回磁盘，这种方法涉及的I/O操作通常比写回整个数据块要少得多。记录变动的数据足以在故障发生时恢复变动。

为了提供最大的性能而不用冒险破坏事务完整性，Oracle只会写入redo信息。当用户提交了事务，Oracle便要保证变动相关的redo信息必须写入磁盘中的redo日志。实际被修改的数据块之后会被写入数据文件。如果故障发生在变动数据块从缓存写入数据文件之前，redo日志将用于重新产生这些变动。由于计算机系统最慢的部分是物理磁盘，Oracle的快速提交机制最大程度地降低了事务提交的开销，并能够提供最大限度的无风险性能。

1）Flashback

回滚段可以用于实现Flashback查询这一特性。回滚段是用来提供Oracle数据库过去某个时间点的数据镜像。有了Flashback查询，用户可以从Oracle数据库获取指定时间点的SQL查询结果。例如，用户可以请求两个小时之前数据库的查询结果。借助Oracle体系结构中的核心特性回滚，Flashback提供了额外的功能。

由于使用了回滚段，因而用户最多只能回闪当前回滚段中的信息。这个要求将Flashback的范围限制在一个比较小的时间段——用户通常无法回滚几天的数据，因为Oracle数据库并不能保存太多的回滚信息。除了这个限制之外，用户能够在某些环境中有效地使用Flashback查询，例如，利用这个技术，用户可以回到导致数据丢失的错误发生之前。

随着Oracle数据库增加了更多的Flashback功能，Flashback技术越来越被广泛使用。Oracle 10g在很大程度上扩展了Flashback功能，具体如下：

* Flashback 数据库，用于将整个数据库回滚到一致的状态。
* Flashback 表，用于回滚特定的表。
* Flashback Drop，用于回滚DROP操作。
* Flashback 版本查询，用于获取一行或多行数据的改动。

Oracle 11g增加了一个新的Flashback扩展，即Flashback事务特性。它可用于撤销一个事务以及依赖这个事务的其他任何事务的影响。

2）一步一步讲述事务

根据一个例子讲述事务的完整流程。在这个例子中，员工信息保存在EMP表中，它是Oracle数据库用于测试的传统数据库模式的一部分。下面考虑某个HR职员需要更新员工名字的情况。该职员从数据库中获取某个员工的信息，更新他的名字，并提交这个事务。

这个例子只有一位用户试图对数据库中的某行数据进行更新。基于这种假设，这里并不包含Oracle为防止其他用户修改当前事务而采取的步骤。

这名HR职员已经可以在电脑屏幕上看到那名员工的信息，因而此时含有员工信息的数据块已经被存储在数据库告诉缓存中。之后步骤如下：

1. 用户修改屏幕上的员工名字，客户端程序通过网络将SQL UPDATE语句发送到服务器进程；
2. 服务器进程在共享内存区的共享SQL区查找相同的语句。如果找到，就重新使用。否则，服务器进程将检查它的语法，并对之进行评估，从而确定最佳的执行方法。对SQL语句这样一种处理流程叫做解析与优化。SQL语句一旦经过处理之后，便被存储在共享SQL区。
3. 服务器进程把即将被修改的员工数据以前的信息拷贝到回滚段和redo段。回滚段保存的变动信息只是redo段保存的变动信息的一部分。这里看起来可能会让人觉得有些奇怪，不过，请记住redo信息包含了事务产生的所有变动信息。
4. 一旦服务器进程完成了上面的工作，它将会修改数据块以改变员工的名字。此时数据块存储在数据库的缓存中。
5. 这位HR职员提交事务。
6. 日志写进程（LGWR）将redo日志缓存中事务完整的redo信息写入磁盘中当前redo日志文件。当操作系统确认对redo日志文件的写操作已经成功完成时，才认为这个事务已经提交。
7. 服务器进程向客户端发送消息，告诉它事务已提交。

除了提交事务，用户可能会取消或者回滚事务。这个时候，服务器进程会使用回滚段中保存的员工数据以前的信息来撤销事务对数据块的修改。

（虚线代表网络流量）

