大型和中型公司的数据通常分布在多个数据库里，这些数据所在的机器通常运行着不同的操作系统和不同的数据库管理系统。服务于特定业务的数据可能会被多台服务器访问。一个用户可能需要同时向几台服务器请求这些数据，或者将这些数据移动到本地的服务器上。这个时候需要对分布式数据库执行数据的插入、更新和删除操作。

处理分布式数据库中的数据有两种方法：

1. 将分布式体系结构做成透明的，把数据的处理作为单个分布式的实体的一部分；
2. 使用各种复制技术在多个位置为数据创建拷贝。

## 1、把多个数据库作为单个实体来访问

用户有时候需要查询或操作驻留在多个Oracle数据库中的数据，或者是驻留在Oracle数据库与非Oracle数据库中的数据。下面将会讲述分布式环境中数据交互的技术和体系结构。

### 1.1跨多个Oracle数据库的分布式数据访问

许多年来，Oracle一直提供对驻留在不同系统或不同节点上多个Oralce数据库中的分布式数据的访问。用户不需要了解数据在分布式数据库中的位置。数据仍然是通过特定表的唯一标识符来访问。管理员只需创建一个简单的标识符，单个机器上的Oracle表就可以被用户看成是某个逻辑数据库的一部分。

开发人员在使用SQL创建数据库链接的时候会在多个数据库之间创建连接。这些连接就构成了分布式数据库。

CREATE PUBLIC DATABASE LINK employees.northpole.bigtoyco.com

这条语句会创建一条到达包含Bigtoyco’s North Pole员工信息的远程数据库的连接。本地源数据库上的应用程序或用户可以在SQL查询、插入、更新、删除等语句中使用全局访问名（employees.northpole.bigtoyco.com）来访问远程的North Pole数据库。Oracle Net负责与各种网络协议交互，以保证用户能够透明地与远程数据库进行通信。

数据库链接使得用户可以透明地访问数据，不过Oracle仍然提供采用不同的方式来处理与分布式数据库的交互。下面简单看一下分布式Oralce数据库的查询和更新操作与单个Oracle数据库的不同。在查询分布式数据库时，首要关注的是优化数据查询。通常会使用基于成本的优化器对单个Oracle数据库的查询性能进行优化（第4章曾经讨论过这个内容。Oracle 7增加了基于成本的全局优化器，用于改善分布式数据库的查询性能。例如，在选择计划时，基于成本的优化器会考虑使用远程数据库中的索引，而基于规则的优化器则不会这样。另外，基于成本的优化器还会考虑使用远程数据库中的统计。Oracle 8i对优化器进行了改进，其中包括对多个节点上的关联和设置操作进行优化，以便在提供最高性能的同时最小化系统间的数据传输。从Oracle 10g开始，基于成本的优化器是Oracle唯一推荐用于单独数据库和分布式数据库的优化器）。

当用户准备把数据写回分布式数据库时，问题就变得有些复杂。正如我们前面所提到的那样，事务是由一条或多条SQL语句组成的一个原子逻辑工作单元。这些语句负责将数据写入数据库，它们作为整体被提交或者被回滚。分布式事务通常会跨多个数据库服务器执行。当分布式事务通过SQL COMMIT语句提交时，Oracle会使用两阶段提交协议来确保多个系统的完整性和一致性。

### 1.2访问非Oracle数据库与接收非Oracle数据库的访问

Oracle透明网关（如图）是Oracle的一个软件产品，它为用户提供了通过Oracle SQL访问非Oracle数据库的能力。透明网关可以部署在Oracle数据库中的两层体系结构中，也可以部署在中间层（即Oracle应用服务器上）。Oracle的SQL语句会被自动转换成目标数据库的SQL语句，这样Oralce应用程序就可以用于非Oralce数据库。还可以使用目标数据库自己的内部SQL语言，这样的SQL语句可以直接发送到目标数据库，而不需要转换。诸如NUMBER、CHAR以及DATE等Oracle数据库类型都可以转换成目标数据库中的类型。异构的数据库也可以通过数据库链接和Oracle数据库链接在一起形成一个分布式数据库。



目前有四种基本类型的数据库连接：

开放数据库连接：

透明网关：

过程化网关：

访问管理器：

## 2、两阶段提交

分布式数据库的使用存在一个最大问题，就是分布式数据库的更新要保证同等层次的数据完整性是很困难的。由于事务在向数据库写数据时必须依靠网络传递消息，因而相对于运行在单个机器上的单独Oracle实例而言，它本身就存在更容易丢失信息的缺陷。

这类问题的标准解决方案就是在事务提交过程中使用消息传递两个阶段；因而，这里使用的协议叫做两阶段提交。

主数据库首先征询每个参与者，确定它们是否准备好，如果准备好，事务的更新

就暂时送给它们。

在第二阶段，如果索引的参与者都确认已经正确地收到消息，这个时候才提交这些变化。如果事务中的某个节点没有确认收到这些变化，事务将会被回滚到它们的最初状态。

举个例子，如果某个事务涉及到A，B，C三台机器上的数据库，在提交操作的第一阶段，事务更新被发送到每一个数据库。如果每个数据库都确认自己收到这个更新，在第二阶段直接执行COMMIT命令。通过将更新数据的传输与实际的COMMIT操作分开，两阶段提交可以大大降低分布式数据丢失完整性的概率。

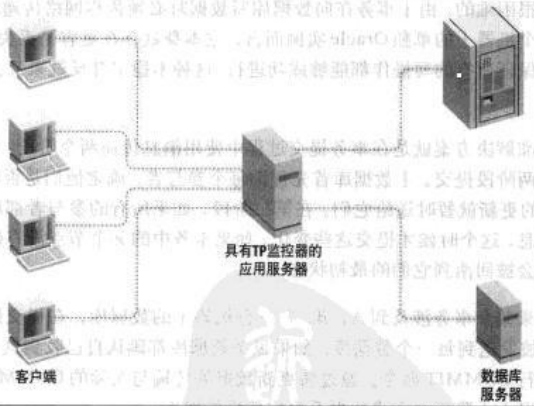
可以拿这个方法和单阶段更新做个比较；在单阶段的方法中，COMMIT命令与事务更新信息一起发送。这样就没有办法知道是否所有的机器都收到更新信息；这个过程发生任何的中断都会导致数据状态的不一致。尤其当事务涉及多个机器时，丢失更新消息的情况更是频繁；反过来，这就要求必须使用两阶段提交协议。当然，由于两阶段提交协议要求在机器间传递更多的消息，因而它比标准的提交提交方式要慢很多；不过，相对于保证重要数据的完整性而言，性能损失点并不算什么。

## 3、事务处理监控器

1991年，X/Open标准组定义了一个开发系统标准接口：通过该接口，事务处理（TP）监控器可以和Oracle RDBMS以及其他XA顺从的资源管理器进行通信。目前支持XA的流行商用TP监控器有BEA公司的Tuxdo以及IBM公司的CICS和Encina。

Oracle 8i增加了用于对Windows NT上的微软事务服务器（MTS）进行管理的Oracle管理器。那个时候，微软公司正在使用.NET体系结构取代COM体系结构。Oracle 9i增加了对.NET支持，使得.NET事务应用程序可以使用Oracle作为资源管理器。

我们曾经在前面提到过TP监控器在在线事务处理中的角色。TP监控器还有一些其他职责，其中之一就是确保多个应用程序和资源之间的事务能够正确地完成。正如前面提到的那样，Oracle为分布式事务提供了自己的两阶段提交协议，过去只有TP监控器才具有这种能力。现在这个功能已经被集成到中间层应用程序，很少会使用单独的TP监控器来进行工作负荷管理（如图）。



只有在下面的情景中才可能需要考虑使用TP监控器：

* 遗留应用程序向UNIX或Windows NT上的CICS迁移。
* 需要在Oracle和其他XA顺从的数据库之间使用两阶段提交。

## 4、在分布式系统间迁移数据

前面讨论了用户可以将多个数据库服务器作为一个逻辑数据库来使用的问题。下面的情形要求对数据库的内容进行复制并在系统之间进行迁移：

* 当在本地保存可用数据可以清除网络带宽问题和系统资源竞争时。
* 当移动数据库用户需要携带自己的数据库并在断开网络的情况下使用时。
* 当冗余数据库有助于增强高可靠性时，每个数据库都可以作为其他数据库的备份。

在许多网格的实现中，共享资源往往需要将数据复制到网格中的多个服务器上。

用户在使用多个相同或类似数据库时遇到的最大问题就是如何在数据变更时保持这些数据库中的数据同步。当用户在对其中一个数据库中的数据进行插入、更新或者删除操作时，必须采取某种方法将新的数据同步到其他数据库中。另外，当分布用户之间的修改产生竞争时，必须处理由此而可能产生的数据完整性问题。Oracle为解决这种问题提供了许多策略。Oracle 9i将这些策略集成到一个单独组件中，这个组件就是Oracle流。不过，Oracle流中的每种策略都有自己的特点。

### 4.1高级复制

复制就是对分布式系统中多个Oracle数据库的表进行拷贝和维护。任何本地站点的变化将会自动传递到所有的本地站点。这种变化包括对数据的更新以及对数据库模式的修改。复制通常用于远程站点的本地用户快速访问数据，也可用于在主站点丢失时提供灾难恢复站点。Oracle高级复制特性不仅支持异步复制也支持同步复制。Oracle还在大型机集成网关中绑定了复制服务（Replication Services），用于支持到DB2的异构复制。

Oracle 8将复制服务触发器的执行移到了数据库的内核，并且能够自动并行处理数据的复制以改进性能。Oracle 8i增加了复制触发条件，当一个表中被选择的行或列被修改时就触发复制。Oracle 9i复制增加了对于对象数据类型和多层可更新物化视图的支持。Oracle 9i通过Oracle流特性增加了基于日志的恢复。虽然Oracle在新的数据库版本中仍然支持上一代的高级复制功能，这里我们将描述复制的基础知识和高级复制的某些特性。

异步复制就是先将变化存储在本地，随后再提交给远程站点。某些类型的异步复制包含对单独可更新主表的只读复制快照以及可更新的快照。可更新快照在失去连接时，仍可以被更新。

在Oracle标准版中，只有一个主站点，负责将变化复制到各个子站点。在企业版中，可以存在多个主站点，每个主站点都可以执行更新。对这些站点的更新必须同步，即只有当所有目标站点都被更新之后，更新操作才被认为完成；否则，就会产生无法解决的冲突。

在多个站点在一个复制间隔内更新相同数据元素时就会发生冲突。在使用基于事件的延迟远程过程调用（RPC）时，或者在连接可以或通信成本最小的时候，更新将得到传播。

Oracle企业版提供的几种解决冲突的例程可用于自动解决复制冲突。管理人员可以简单地为特定复制选择自己希望的冲突解决策略。对于影响一个列或多个列的更新而言，可以使用下面标准的冲突解决方案：

覆盖和丢弃值：

当单个主站点需要用一个新值更新目的站点中的当前数据时。应用此方案。

最小值和最大值：

在源站点的新值和目标站点的当前值之间进行比较，如果它小于当前值则使用新值作为最小值，如果它大于当前值则使用新值作为最大值。

最早和最晚事件戳值（由类型为DATE的列指示）：

最早是指当有多个新值时，用于更新的数据将位于具有最早时间戳的行中。

最晚是指当有多个新值时，用于更新的数据将位于具有最晚时间戳的行中。

含有单个数字列的列组的相加值和平均值：

相加值就是先获取源站点的新值和旧值的差值，再把这些差值加入到目的站点的当前值。

平均值就是把目的炸点的当前值和源站点的新值相加再除以2，得到的新值。

优先组合站点优先级：

为列分配优先级，当出现多个新值时，低优先级的列将被高优先级的列更新。

另外还可以使用内置的唯一性冲突解决例程来解决由主键和唯一性约束的分布式使用而造成的冲突。这些内置例程如下所示：

为复制值附加站点名：

为复制的列图加上源站点的全局数据名。

为复制值附加序列：

为列值附加生成的序列号。

丢弃复制值：

丢弃导致错误的源站点的行。

当标准例程无法满足特定业务需求时，也可以编写自己的冲突解决例程并做相应的分配。

### 4.2管理高级复制

可以使用Oracle企业管理器来管理复制。管理员可以配置需要被复制的数据库对象，安排复制进度，调试发生错误的情况，并通过中心接口查看每个位置上的延迟事务队列。该队列包含了需要复制到子站点的事务。

例如，为了建立一个典型的多主系统复制，你必须首先在每个数据库中定义需要复制的主组、表以及对象。

可以定义一个到达主定义站点的连接，然后创建一个或多个主组，并将表和对象复制到多个主站点。接着，为每个主组中的复制表分配冲突解决例程。最后，为希望访问这些站点数据的应用程序用户分配适当的访问权限。

## 5、高级队列（Advanced Queue，AQ）

20世纪80年代，面向消息的中间件（MOM）获得了广泛的应用。MGM借助消息在系统间传递消息。由于MOM本身就能够保证所有消息的可靠传输，因而它可以省去两阶段提交的开销。诸如IBM的MQSeries之类的产品可以存储控制信息（消息目的地、时间期限、优先级以及接收者）和基于文件的队列中的消息内容。只有目的地可达，消息才会被传递过去，否则消息将一直存储在队列中，这样做保证了消息传递的可靠性。

Oracle高级队列特性首先由Oracle 8企业版引入，目前已成为Oracle流特性的一部分。它通过将队列存储在Oracle关系型数据库中来提供完整的队列环境。

高级队列就是Oracle数据库的表，不过它支持特殊的排队操作——入队列的操作创建消息，出队列的操作消息消费。这些消息可能是无结构的（原始的），也可能是有结构的（就像第14章描述的Oralce对象），它们分别对应于表中的一行。存储在正常队列的消息用于正常的消息处理，如果由于某种原因消息无法被索，则被存储在异常队列中。

### 5.1队列的创建和管理

可以通过PL/SQL命令或者Java API来创建队列。管理员创建队列的步骤如下：

1. 创建对列表。
2. 创建队列，并为队列命名。
3. 指定队列是作为正常队列还是异常队列。
4. 指定消息在队列中的保存时间，它可以是一个固定长度的时间，用于表示重试之间的超时时间，也可以基于重试的次数来确定。

管理员可以启动和停止队列，也可以授予用户使用队列的必要权限，并在必要时收回用户的权限。

消息的生产者指定队列的名字、入队选项、消息属性以及放入队列的有效载荷；然后由生产者代理处理。消费者代理负责监听一个或多个队列中的消息，然后将消息从队列中取出供用户使用。队列中消息存在的通知可以通过Oracle调用接口（OCI）注册回调或者通过监听调用来完成；应用程序可以使用监听调用来监控多个队列中的消息。

由于消息是持久化存储在数据库的队列中，因而可以采用许多消息管理特性。每条消息都携带了历史相关的信息，其中包括新消息的位置和状态、被访问的节点以及前面的接收者。在指定时间内没有到达预订者的消息将被移送到异常队列中，以便日后跟踪。成功到达预订者的消息在消费之后仍然可能被保留下来，用于日后分析，其中包括入队和出队的时间。由于消息之间可能是相关的（例如，两个消息的成功执行可能会导致产生一个新的消息），因而保留消息也可用于跟踪消息的顺序。

基于数据的队列提供了持久化的特性；在不需要该特性的场合，为了提高性能，Oracle 10g引入了非持久化消息队列。

Oralce企业管理器提供的队列管理功能包括：

* 创建、删除、启动以及通知队列。
* 添加和删除预订者。
* 调度消息从本地传播到远端队列。
* 显示队列的统计信息，包括队列的平均长度、处于等待状态的消息数量、处于准备状态的消息数量以及过期消息的数量。

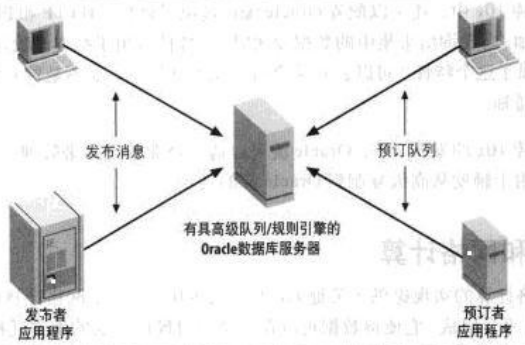
Oracle 9i引入了几种新的AQ功能：

* 使用HTTP协议的基于XML的消息机制，支持穿越防火墙，请求消息可能使用基于XML的互联网文档访问协议（iDAP）。
* 可以使用动态服务（Dynamic Service）来定义AQ策略和服务。
* 可以使用Oracle互联网目录（OID）来定义和管理AQ代理。

Oracle 9i及之后的AQ为PL/SQL和XSLT提供了内置的消息转换功能。消息网关还可用于向MQSeries和TIBCO等其他系统传播消息。

### 5.2发布和预定功能

Oracle 8i企业版在AQ中引入了发布和订阅功能。如图，发布者将消息放入队列中；预订者从队列中接收消息。发布者和预订者各自独立地与队列交互，互不知道对方的存在。发布者负责决定什么时候发布，如何发布，以及发布什么；而预订者只需表达出自己的兴趣。消息可以通过主题名或者内容（通过过滤规则）进行发布和预定。预订者注册回调函数后，可以得到异步的通知。



可以把高级队列及其发布——预定特性用于附加的数据库事件通知，它可以改善数据库和业务应用程序的管理。注入DMI（插入、更新、删除）和系统事件（启动、关闭等等）等数据库事件都可以被发布和预定。举个例子，可以构建这样一个应用程序，当某些高价值客户需要配送货物时，自动通知预订者，预订者然后就知道他应该跟踪发货的进程并提示客户正在运送货物。

Oracle 11g对消息服务做了几点改进，以提高服务器的性能和可靠性。

## 6、Oracle流

Oralce 9i引入了Oracle流，该特性将高级复制和高级队列整合到单个产品中，并增加了在数据库或数据库之间共享数据的功能。Oracle流通过捕捉——应用（capture-and-applay）进程（包括Oracle变化数据的捕捉）来传播变化。变化可以在Oracle实例之间传播，可以从Oracle实例传递到非Oracle实例（经过透明网关），也可以从非Oracle数据库传递到Oracle数据库（可以借助组消息网关并在非Oracle数据源加上定制代码来收集变化）。Oracle流使用基于日志的进程来捕捉DML或DDL变化或者同步捕捉DML变化，然后在分段处理过程中使用队列过程。用户提供的“应用”规则定义了消费过程。

当通过Oracle源数据库redo日志捕捉变化或者通过同步捕捉来收集行变化时，数据库后台进程将创建逻辑变化记录（LRC）。LRC和用户消息事件被加入Oracle流队列。事件从源队列传播到目标队列，然后由目标数据库中的后台进程将事件从队列中取出，并应用它。从10g开始，Oracle数据库支持变化的下游捕捉以及批量消息的入队和出队。

在Oralce 10g中，还可以配置Oracle流通过电子邮件、HTTP和PL/SQL来提供数据库变化通知。当查询结果集中的数据变化时，该特性可用于给客户发送通知。Oracle 11g加强了这个特性，可以针对单个行变化发出通知，而不是针对结果集中的行变化发送单个通知。

在Oracle 10g中，Oracle流可以通过企业管理器来管理。另外，还有一个迁移佛扭矩可用于辅助从高级复制到Oracle流的迁移。

### 6.1Oracle流和网格计算

Oracle流为网格计算的实现提供了关键功能。从本质上来说，网格计算由广泛分布的数据、用户以及平台组成。（它使得数据可以在需要的时候向需要的地方迁移；并为事件、消息以及数据库变化的预定过程提供消息共享、通知用户过程调用等功能；还提供与其他平台进行互操作的功能。）流数据库可以通过创建操作数据存储将处理工作交给复制数据库，也可以创建数据库复制，并将复制的变化以及数据的转换应用到产品数据库中。

在Oracle 11g中，Oracle流可以挖掘DML和DDL活跃的在线日志文件，从而实现RAC实例间低延迟的变化传播。Oracle流运行在单个RAC实例上，该实例被标识为主要实例，负责队列和处理。还可以标识一个次要实例，用于提供更高的可用性解决方案。

Oralce流还能够用于数据库向网格计算和Oracle新版本的迁移。当新数据库被安装好，原来的数据库仍在运行时，Oracle流可用于捕捉原来数据库的变化，当迁移接近完成时，可以把这些变化应用到新的额数据库中。

## 7、可传输表空间

前面的几节专注于分布式数据库之间活跃数据的共享——即确保变化可以实时传播到其他的数据库中。可传输表空间提供了一种在表空间不活跃的情况下快速完成所有表空间在多个数据库间分布的方法。

可传输表空间由Oralce 8i企业版引入，用于快速地在数据库实例之间拷贝和分布表空间。在以前，实现这样的操作需要先从源数据库中导出表空间，然后再导入到目标数据库（即卸载和加载的过程）。可传输表空间使得拷贝可以简单地通过文件的传输命令（如ftp）来完成迁移。

在对表空间的拷贝进行拷贝和迁移之前，首先应该把表空间设为只读，以避免不经意的修改。数据字典信息需要在传输之前先从源数据库中导出，然后再导入到目标数据库中。

使用可传输表空间最常见的原因包括：

* 把表空间快速地从企业数据仓库拷贝到数据中心。
* 将表空间从操作系统拷贝到操做数据存储，用于生成统一报表。
* 将表空间通过CD-ROM发布。
* 使用备用拷贝进行快速的及时点表空间恢复。

在Oracle 9i之前，源数据库和目的数据库的块大小必须相同；在Oracle 9i中取消了这个限制。Oracle 10g则取消了源数据库和目的数据库必须运行在相同的操作系统平台上的限制。