附录C　24×7可用性的重新设计

当修改系统以支持更大的用户群时，对系统的可用性要求可能提高。例如，如果你的公司并入一个国际公司，你的财务系统可能就要一天提供 24小时服务 —— 这个要求可能超出了你原来的技术要求。可用性要求增加的另一个原因是基于因特网的用户群。一旦你允许用户通过因特网执行你的数据库中的事务，用户将希望你的数据库一天 24小时不间断地可使用。

一天24小时、一周7天（24×7）可用是一个重要的技术要求，通常是在一项应用程序被投入产品环境后才被认识到。改进现有应用程序的可用性是困难的，而且可能要求重新设计系统。在本附录中，你将看到解决24×7要求时所遵循原则的简要介绍。为了取得最好的结果，应把24×7当作一个业务过程要求，而不是技术要求来处理。一旦把注意力放到正在处理的业务过程上，可使用的技术方案的数量将明显地增加。在本附录的第一节，你将看到高可用性数据库应用程序的一套技术手段，从这些手段中可看到一套高可用性的技术解决办法。

C.1　技术手段

根据应用程序的生命周期，你也许不能影响应用程序所采用的技术手段。如果你能影响应用程序的技术设计，应该遵循以下准则：

- 限制数据库访问。
- 限制数据库大小。
- 排除故障点。

在以下几节，你将看到这些手段的例子。

C.1.1　限制数据库访问

在某些情况下，你的数据库可能不可用 —— 如升级、冷备份或由于电源故障。这时，应尽量减少数据库执行中断对业务过程的影响。为了尽量减少停机影响，应尽量减少数据库被应用程序访问的次数。

例如，如果在批事务处理中你的应用程序通过 ODBC写入数据库，可将那些ODBC事务写入文件，然后，通过 SQL*Loader将这些文件装入到 Oracle中。如果数据库执行中断，那么处理事务的SQL*Loader部分不能完成，但是应用程序的 ODBC部分仍能够成功地执行。

如果用户重复执行同样的报表，可以预先生成这些报表并进行分发（或者通过因特网或本地网络进行公布），用户不需直接访问数据库就能查看他们需要的数据。如果数据库只被预先生成的报表不能解决的少数特别查询直接访问，那么数据库执行中断对业务过程的冲击就可以减至最小。

这两种方法通过将批事务处理用于数据输入和数据查看来解决可用性问题。在每种情况下，你都必须确定允许的数据延迟 —— 在不影响业务操作的前提下，数据可以是多旧的。如果用户可以使用8小时以前的数据作出有效的业务决定，那么就可以利用这个空隙管理数据库，而不需打断业务过程。

如果用户交互输入数据，那么在数据输入期间，应当限制查询数据库的次数。许多传统的OLTP应用程序查询数据库要求执行代码值的查找。例如，当输入一个购物申请号时，可对照一组有效的购物申请号来使它生效。但是，如果想限制数据库访问，可以在交互插入期间用不同的方法使数据生效。例如，可假设用户将正确地完成他们的工作，不需要对插入的每一个字段执行数据生效步骤；相反，你可让数据库中已规定的引用完整性使数据生效。可以通过训练和利用合法值的下拉列表自定义数据输入屏幕来帮助用户。

这些变化包括改变数据输入、接受、处理和显示的方式。除非由于24×7要求的原因，应用程序被重新设计，这类重大改变都很难实现。如果不能实现这些改变，那么你就应考虑采取其他技术手段和解决办法。

C.1.2 限制数据库大小

如果一个数据库必须是24×7可用的，那么应该将数据库做得越小越好。与业务服务无直接关系的任何数据都应从数据库中移走。如果只有一个小的事务系统需要是24×7可用的，那么就不要将一个大的数据仓库置于同样的情况中。数据库越小，就越容易达到高可用性要求。

将不同实例中的数据分开也许有帮助——数据可能有明显不同的增长和使用特性。但是，将不同实例的数据分开将使整个数据库环境的管理更加复杂。在一个分布性越大的环境中，就有越多的问题：管理数据库链接、调整远程查询和使复制的数据保持同步。

再重申一次，当你要你的数据库24小时可用时，使它越小越好。一旦应用程序在你的数据库中生成，要想在不同实例间分离其对象就更困难了。如果应用程序已经完成且没有明显的分隔，你就需要采取其他技术手段和方法。

C.1.3 排除故障点

在数据库环境中，应该排除将引起数据库或硬件不可用的任何已知方面。排除故障点可包括以下内容：

- 实现RAID和硬件镜像系统。如第4章中所述，这些系统可保护可用性，防止单张盘丢失造成的损失。
- 实现容错和冗余硬件。许多制造商都提供冗余硬件部件的解决方法，允许在破坏事件中立即修复故障。有些制造商提供的解决方法是自动地将一组盘上的数据拷贝到另一组盘上，提供磁盘级复制。
- 使用可自动扩展的数据文件。必须保证所使用的硬盘有足够的可用空间，以便支持数据文件扩展。
- 将MAXDATAFILES参数设置为操作系统的最大值。增加MAXDATAFILES将要求关闭数据库并重新创建控制文件（参见第1章），而这正是在24×7环境中需要避免的情况。
- 使用无限盘区。拥有太多的盘区将影响DDL操作的性能。但是，如果你的目标是高可用性，那么就应避免由于对象达到其最大盘区数目而引起的失败。应当适当规定对象的大小（参见第5章），避免获取大量的盘区。
- 创建一个大的SGA。如果不能关闭数据库，就不能增加数据块高速缓存或SQL共享池

的大小。当应用程序第一次进入产品阶段时，必须估计到未来的要求。

- 如果使用ARCHIVELOG方式（参见第10章），那么就给归档重做日志文件保留一个大的可用空间。监控重做日志文件目标区域的大小并使这些区域保持所需的足够的自由空间。如果ARCH进程不能将新的归档重做日志文件写入目标区域，那么数据库将停止执行，直至空间可用。
- 在数据库中，给回滚段、临时段和数据字典分配额外的空间。

这些方法将消除数据库中大多数普通的故障点。如果你的应用程序对故障点特别敏感，那么就应该尽量减少它的影响。例如，如果你的应用程序的数据经常改变，就需要安排好分析表的时间，不能影响用户。可以通过使用估计选项、分析分区而不是表及安排分析活动时间避免与用户使用这些表的时间相冲突等方法减少分析命令的影响。

这些技术方法可显著增强支持24×7业务过程的能力。如在这些方法中所提到的，必需能够控制数据的大小、访问数据的方式和在其环境中的故障点。在下面一节中，将看到用于24×7环境中的管理和恢复的技术解决方案。

C.2 技术解决方案

为了有效地管理一个高可用性数据库，必须能快速执行数据库管理操作，且尽量不妨碍业务进程。前面部分的技术手段着重于应用程序的重新设计，试图减少对数据库的可用性要求。如果不能修改应用程序的设计，那么就需要采取一种技术解决方法来解决业务问题。任何技术解决方法都不可能使一个本没有将24×7当成技术设计要求完成的系统完全达到24×7可用性。

任何达到高可用性的技术解决方法都应包括下面几个部分：

- 对数据库管理活动的广泛测试。
- 非常快速的管理过程。
- 非常短的恢复时间要求。

除了这些解决方法之外，还需要一个自动快速通知失败的机制。例如，可以配置一个SNMP启动的监控程序，以便在数据库发生问题自动通知DBA。在失败的情况下，迅速通知可以减少问题发生后修复所需的时间。也可以将监控程序配置成自动执行操作，诸如重新启动数据库。

在下面各节中，将看到对每种解决方法的说明。

C.2.1 对数据库管理活动的测试

如果不能按照本附录前半部分所述的技术手段重新设计应用程序和数据库，那么就需要对数据库执行定期的管理活动。例如，可能需要增加数据文件，重新确定表的大小或在表空间之间移动对象。

在进行这些改变之前，必须能够在一个模仿的产品环境中对它们进行有效的测试。可以利用测试结果来确定升级和修改时遵循的最佳过程。在这些结果的基础上，可以估计出变化对应用程序的影响并适当地安排修改。测试不可能发现每一个问题，但它将使你预见到主要问题。如果测试中发现了问题，就可以减少带入产品环境中的错误数量，从而也就提高了数据库的可用性。

C.2.2 快速管理

对数据库管理过程的选择将受到对已有数据库的高可用性要求的影响。要考虑到三种DBA活动：数据库软件升级、数据在数据库间的移动和数据库备份。

1. 数据库软件升级

如果数据库不能长时间不可用，那么数据库升级就很可能不涉及使用 Export和Import在分离的实例之间移动数据。相反，很可能使用第 2 章中所述的增量升级技术。如果操作系统能支持多种版本的Oracle软件，那么使Oracle数据库升级的过程可能就只是短时间地中断一下数据库。数据库将指向它的新 Oracle软件主目录，其listener.ora和/etc/oratab项可迅速被升级。在新的监听程序被启动且数据库被打开后，可迅速地执行 Oracle软件主目录下的/rdbms/admin目录中提供的数据字典目录脚本文件。专用的版本升级的具体细节由 Oracle软件主目录下的/rdbms/doc目录中的README.doc文件提供。

在对数据库软件进行升级之前，应该对现有数据库及其软件进行一次备份。在 24×7环境中，必需实现为你的备份需要而特别设计的技术解决方法。关于数据库备份的进一步说明请参见本附录中的“数据库备份”小节。

虽然用于数据库升级的目录升级方法对大多数版本升级都是有效的，但是主要版本之间的升级将包括更复杂的过程，为了帮助主要版本之间的升级，如从 Oracle7到Oracle8，Oracle提供了一些移植实用程序。使用这些移植实用程序的升级应在实现产品环境之前通过 DBA测试环境。还必须估计移植实用程序是否有附加性能、可用性成本是否随升级而增加。

2. 数据库间的数据移动

如果需要在数据库间进行大量的数据移动，有一些选项。在 Oracle最近的每一个版本中，Oracle 都引入了改进大型数据装载性能的新特性。例如 ，可以使用 nologging参数、SQL Loader Direct Path装载、parallel选项和APPEND提示等来改进数据装载操作。

如果数据在一个数据库中且需要迅速转移到另一个数据库中，应当试验使用可移动的表空间。如第12章中所述，你可以利用可移动表空间将一个表空间的自包含集从一个数据库转移到另一个数据库。在目标数据库中，只输入表空间的元数据，而不是它的数据。结果，数据转移的性能将明显地好于所有表的全导入。

注意 为了改进数据移动操作的性能，应当尽量减少要移动的数据总量和表数目。

如果源数据库和目标数据库都使用 Oracle8i，而且如果操作环境完全相同，那么可以只使用可移动表空间选项。作为进一步的限制，移动集中的表空间必须是自包含的，对其他表空间中的对象没有依赖性。为了利用可移动的表空间，必须为数据库对象重新设计表空间布局。

3. 数据库备份

直接影响可用性的唯一一种数据库备份是脱机文件系统备份。为了迅速执行脱机备份，必须按照以下要求配置操作环境。最理想的是，建立的操作环境对生成的数据有三份拷贝：数据库、镜像集和第二镜像集。要执行脱机备份，应按照以下步骤进行：

1) 关闭数据库。
2) 将第二镜像集从配置中分开。
3) 启动数据库。
4) 将第二镜像中的文件备份到备份介质上。

5) 使第二镜像集重新同步。

在此过程中，只有在将第二镜像集从配置中分离的一段时间中数据库是不可用的。第 5步的重新同步过程是在操作系统级完成的；许多硬件销售商都支持因镜像集被分开而引起变化的重新同步过程。如果配置要求完全重新生成镜像集，那么在设计中就必需注意重新生成镜像集对性能的影响。

如果使用Export备份数据，那么应当尽量减少导出对产品系统的影响。为了提高导出的执行速度，应使用 DIRECT=Y选项。直接导出执行明显地快于传统导出。也应估计导出对CONSISTENT参数的要求。一个一致性的导出执行比一个 CONSISTENT=N导出慢得多。如果在导出中使用CONSISTENT=N，导出执行速度将得到提高。但是，导出文件中的数据可能违反为数据库定义的引用完整性规则。对这些参数的详细说明请参见第 10章。

C.2.3 迅速恢复

除了设计一种将停机时间减至最小的备份方法之外，一个 24×7数据库还必须有一个最小化恢复平均时间(MTTR)的恢复策略。在计划备份和恢复策略时，为了给备份和恢复要求适当的权重，需要用户的输入。为了减少可能发生的影响数据库可用性的硬件失败，用户可能选择购买冗余硬件。如果硬件环境不是失败的根源，且所有改变在应用到产品中之前都经过严格地测试，那么就不需进行许多次恢复。如果情况如此 ，那么就可改变备份和恢复计划，在较快地进行备份的同时，延长了 MTTR时间。所做出的关于备份和恢复过程的决定将直接影响业务进程。因此，必须和你的用户一起确定正确的方向。

有四种减少MTTR的方法：

- 镜像环境。
- 备用数据库。
- 复制。
- Oracle Parallel Server（Oracle并行服务器）。

无论选择什么方法，都必须确保所选方法对应用程序的性能和可用性没有明显地影响。在下面，你将看到这几种解决方法的说明。

1.镜像环境

如本附录中“数据库备份”小节所述，可以利用镜像环境显著减少执行备份所需的时间。也可利用镜像环境减少恢复所需的时间。可以将前一夜的镜像集用作数据库文件的联机拷贝，在恢复过程中立即就可用。例如，假设一组数据文件有两个镜像集，在脱机备份期间，备份过程可能是这样的：

1) 关闭数据库。
2) 将第二镜像集从配置中分开。
3) 启动数据库。
4) 将第二镜像集中的文件备份到备份介质上。

使第二镜像集保持分开，而不使其重新同步。镜像集成为自上次备份以来数据库的磁盘拷贝，在恢复过程中立即可用。在第二夜的脱机备份中，备份过程可能是这样的：

1) 重新同步第二镜像。
2) 关闭数据库。

3) 将第二个镜像集从配置中分开。

4) 启动数据库。

5) 将第二镜像集中的文件备份到备份介质上。

这样，第二镜像集成为一个永久备份集。由于它被配置成一个备份集，如果主镜像集遭受介质失败，那么可以将它当作主镜像集重新同步。

如果使用联机备份，那么备份方法明显不同。在联机备份中，可以将磁盘的第二镜像集用作一个独立集，而不是一个镜像集；在备份期间，可执行从主磁盘组到第二磁盘组的数据文件盘到盘备份。因此在发生介质故障时，可以有一个镜像磁盘集，在发生较严重的故障时，将有一份备份的联机拷贝。

2. 备用数据库

如第10章中所述，可以有一个备用数据库用于灾难时的快速恢复。备用数据库有一份处于永久恢复状态的产品数据库的拷贝。如果产品数据库发生灾难，只需要少量的恢复就可以打开备用数据库。备用数据库提供很短的 MTTR，其代价是可能丢失产品数据库中的联机重做日志文件内容。

潜在丢失可能比联机重做日志文件本身更多。如果主数据库和备用数据库之间有一大段距离，那么将归档的重做日志文件从主服务器传输到备用服务器所需要的时间可能是至关重要的。如果在传输归档的重做日志文件过程中，主系统有了灾难，那么数据丢失将包括联机重做日志文件和至少一个归档的重做日志文件。根据重做日志文件转换之间的时间间隙，潜在的数据丢失可能对进行的业务进程有不利的影响。在恢复过程中，需要打开备用数据库，有效地切断它与产品实例的联系。如果能重新生成在故障恢复操作中丢失的事务，备用数据库就可能是适宜的。还要注意，一旦备用数据库被激活，就不再拥有备用数据库。一旦作为主数据库激活备用数据库，为了保持数据的可恢复性，必须生成一个新的备用数据库。必须执行更多的数据管理活动，但是， MTTR将减少。

3. 复制

能在数据库级和操作系统级复制数据。在数据库级，可使用 Oracle的复制方法（参见第10章）生成主表的只读拷贝。在恢复情况下，可以将用户连接到复制实例并执行事务处理。

利用复制方法的快速恢复要求仔细地进行计划。一旦允许用户输入数据到复制环境，就可能和主产品环境发生冲突。必须决定是使用只读复制还是更困难的管理多主复制模式。最后，为了支持MTTR需求，需要安排并监控复制调度。

一个成功的复制方案复制尽可能少的数据。如果不能限制数据库的大小，应限制复制的表和行数量。复制不必是对产品实例的完全拷贝 —— 只需要在恢复主数据库时有足够的数据处理新行的输入。必须和应用程序维护人员一起工作 ，弄清复制环境的最低要求。

规划复制环境时，应当考虑到操作系统级复制。有些销售商提供在操作系统级复制数据的复制服务。如果将一个新记录插入一个表，操作系统级复制将把变化复制到受插入影响的数据文件中。由于这种复制发生在操作系统级，可能比数据库级复制高效得多。将进行操作系统级复制和一个在失败时支持事务处理的小数据库相结合，可提供很短的 MTTR且没有潜在的数据丢失。

4. Oracle并行服务器

如第2章中所述，Oracle并行服务器（OPS）可支持访问同一数据文件集中的单个簇上的

多实例。如果簇中的某一个服务器失败了，用户可以从第二个实例访问数据文件。在这种情况下就不再需要恢复时间——数据库始终都是可用的。

在选择一个OPS解决方法之前，应先分析在你的环境中引起服务中断的原因。如果有一个磁盘不易失败的簇环境，那么OPS是有效的解决方法。在OPS环境中，磁盘失败将造成服务器停止执行，因为实例访问同一数据文件集。如果磁盘文件比服务器更容易失败，就应改进磁盘冗余方法或另外选择一种恢复策略。

当将一个双实例OPS策略用于快速恢复时，除非第一个实例失败，不应允许用户访问第二个实例。如果允许用户访问两个实例，将遇到在不同的实例中竞争同样的数据库块而引起的性能问题。管理一个OPS环境并不琐碎，但要仔细安排好锁分配和init.ora参数。为了有效地实现OPS，需要重新设计应用程序的表布局和数据访问方式。

C.3 建立业务应急计划

24×7可用性设计要求进行调整，如备份一额外的磁盘作为镜像集需要的时间。应当建立一个强大的测试环境，以支持那些为技术决定提供依据的测试。在这些测试结果的基础上，可选择使用不同于在非24×7数据库中使用的数据库管理和数据移动过程。在最坏情况的方案中，可以证明，为了支持24×7可用，应用程序需要明显地修改。可能需要将应用程序分段，允许某些部件故障的MTTR长于应用程序的其余部分。

如果应用程序不能被修改，那么它就可能失败。应当记住，始终都有应用程序结构的某些部分在你的控制之下。如果用户通过因特网访问数据库，那么他们的连接就和他们的客户机、电话线、ISP与你的服务连接一样可靠。24×7可用性设计的目标是尽量减少任何中断对业务过程的有效执行的影响。因此，你的设计必须计算在系统重新可用之前所有这些组成部分的总的失败率——甚至是潜在的。你需要和用户一起实现一种能满足高可用性要求的技术手段和解决方案。

对于每一个业务过程，都有一个匹配的应急计划，说明业务过程的技术部件失败时的动作要求。可以将本附录中提出的原则和方法用作数据库环境中的应急计划的组成部分。数据库的应急计划技术方案必须是整个技术结构应急计划的组成部分。一旦实现了一种技术手段，就能和用户建立服务级协议，而且就能进一步地接近24×7可用性。