IBM DB2 10.1
for Linux, UNIX, and Windows

数据恢复及高可用性指南与参考

更新时间 2013 年 2 月

S151-1755-01

IBM DB2 10.1
for Linux, UNIX, and Windows

数据恢复及高可用性指南与参考

更新时间 2013 年 2 月

S151-1755-01

注意

使用此信息及其支持的产品前，请先阅读第 435 页的附录 B，『声明』下的常规信息。

修订版声明

此文档包含 IBM 的所有权信息。它在许可协议中提供，且受版权法的保护。本出版物中包含的信息不包括对任何产品的保证，且提供的任何语句都不需要如此解释。

您可在线或通过当地的 IBM 代表处订购 IBM 出版物。

- 要在线订购出版物，请转至 IBM 出版物中心，网址为: http://www.ibm.com/shop/publications/order
- 要查找当地的 IBM 代表处，请转至 IBM 全球联系人目录，网址为: http://www.ibm.com/planetwide/

要从美国或加拿大的 DB2 市场和销售部订购 DB2 出版物，请致电 1-800-IBM-4YOU（426-4968）。

您发送信息给 IBM 后，即授予 IBM 非独占权限，IBM 可以按它认为适当的任何方式使用或分发您所提供的任何信息而无须对您承担任何责任。

© Copyright IBM Corporation 2001, 2013.

目录

© Copyright IBM Corp. 2001, 2013

关于本书

《*数据恢复及高可用性指南与参考*》描述了如何使 DB2® for Linux, UNIX, and Windows 数据库解决方案高度可用以及如何避免数据丢失。

《*数据恢复及高可用性指南与参考*》具有两部分:

- 第一部分: 高可用性。它描述了用于帮助使数据库解决方案高度可用的策略和DB2 数据库功能部件和功能。
- 第二部分: 数据恢复。它描述了如何使用 DB2 备份和复原功能以避免数据丢失。

© Copyright IBM Corp. 2001, 2013

第 1 部分 高可用性

数据库解决方案的可用性是对用户应用程序执行所需数据库任务的效果进行衡量的一种方法。

如果用户应用程序无法连接到数据库，或者如果其事务因错误而失败或因系统上的负载而超时，那么该数据库解决方案可用性不高。如果用户应用程序能成功连接到数据库并正常工作，那么该数据库解决方案具有高可用性。

设计高可用性数据库解决方案或者提高现有解决方案的可用性需要了解访问数据库的应用程序的需求。要从额外的存储空间、更快的处理器或更多的软件许可证获得最大利益，重点是使数据库解决方案在企业最重要的应用程序最需要它时尽可能可用。

意外中断

可能影响数据库解决方案的用户可用性的意外系统故障包括：电源中断、网络中断、硬件故障、操作系统错误或其他软件错误，以及出现灾难事件后的系统崩溃。如果出现这样的故障后用户希望仍可使用数据库，高可用性数据库解决方案必须执行以下任务:

- 对用户应用程序屏蔽该故障，使之不受该故障影响。例如，如果某个数据库服务器出现故障，DB2 数据服务器可以将数据库客户机连接重新路由至备用数据库服务器。
- 响应故障以抑制其影响。例如，如果集群中的一台机器出现故障，那么集群管理器可从集群中除去该机器，从而不会再有事务路由至故障机器上进行处理。
- 从故障中恢复以使系统回复正常运行。例如，如果主数据库出现故障之后由备用数据库接管其数据库操作，而原来的主数据库可能会重新启动并恢复过来，再次成为主数据库，重新接管其操作。

执行以上 3 项任务时，要尽量不影响解决方案提供给用户应用程序的可用性。

计划的中断

在高可用性数据库解决方案中，维护活动对用户应用程序的数据库可用性的影响也必须尽量降低。

例如，如果数据库解决方案用于营业时间为上午 9 点至下午 5 点的传统商店柜台，那么维护活动可以在营业时间外脱机进行，而不影响用户应用程序的数据库可用性。如果数据库解决方案用于期望一天 24 小时供客户通过互联网访问的联机银行业务，那么维护活动必须联机运行，或者避开活动高峰期，以尽量降低对客户的数据库可用性的影响。

当您就数据库解决方案的可用性作出商务决策或设计选择时，必须权衡以下两个因素:

- 数据库业务对客户不可用的代价
- 实现某种程度的可用性的成本

© Copyright IBM Corp. 2001, 2013

例如，假定某一基于互联网的企业的数据库解决方案每向客户服务一小时可获取特定收益 X。每年节省 10 小时停机时间的高可用性策略每年可为企业获得 10X 额外收益。如果实现此高可用性策略的成本低于预期的额外收益，那么就值得实现。

第 1 章 中断

中断是对数据库解决方案为用户应用程序服务的功能的任何破坏。中断可以分类为下列两组：意外中断和计划的中断。

意外中断

意外中断的示例包括:

- 系统的一个组件的故障，其中包括硬件或软件故障。
- 无效管理或用户应用程序操作，例如，无意中删除对业务至关重要的事务所需的表。
- 由于非最佳配置或者硬件或软件不足导致的性能低下。

计划的中断

计划的中断的示例包括:

- 维护。一些维护活动要求您执行完全中断；其他维护活动可以在不停止数据库的情况下执行，但可对性能产生不良影响。后者是计划的中断的最常见类型。
- 升级。升级软件或硬件有时可能需要执行部分或完全中断。

在有关可用性的讨论中，焦点通常集中在灾难情况或组件故障上。但是，为了设计可靠的高可用性解决方案，需要讨论所有这些类型的中断。

中断特征符

中断特征符是作为中断特征的症状和行为的集合。根据导致最终用户使站点故障结束的响应时间很长的临时性能问题，中断的特征符可能会有变化。

当设计用于避免和最小化中断以及从中进行恢复的策略时，请考虑这些变化会如何影响业务。

断电

当系统完全不可用于其最终用户时遇到断电类型中断。此类型中断可能由于硬件、操作系统或数据库级别的问题导致。当断电发生时，必须立即识别中断的作用域。中断完全发生在数据库级别上吗？中断发生在实例级别上吗？或者，它发生在操作系统或硬件级别上吗？

欠压

当系统性能低下至最终用户无法高效地完成其工作时遇到欠压类型中断。系统作为整体可能还处于接通和运行状态，但基本上，从最终用户角度来看，它没有正常运行。在系统维护和高峰使用时间段期间，可发生此类型中断。通常，在此类中断期间，CPU 和内存都接近容量。未适当调整或已过度使用的服务器经常导致欠压。

中断的频率和持续时间

© Copyright IBM Corp. 2001, 2013

在有关数据库可用性的对话中，焦点通常集中在给定时间段的总量或停机时间百分比（或相反，数据库系统可用的时间量）。但是，计划的中断或意外中断的频率和持续时间在这些中断对业务的影响方面有重大差别。

请考虑以下情况：必须对数据库系统进行将用七小时来执行的一些升级，并且可以选择在用户活动很少的时间段期间每天使数据库系统脱机一小时，也可以选择在最忙日的最忙时间期间使数据库脱机七小时。显然，若干次短时间中断将比一次中断七小时的成本要低并且对业务活动的损害也要小。现在，请考虑以下情况：发生了间歇的网络故障（每星期可能总计几分钟）这导致少量事务以固定频率失败。这些很短时间的中断会使您损失大量收益，并危及客户对您业务的信任而难以再次恢复，这又会对将来的收益造成更大损失。

请不要仅关注总的中断（或可用）时间。当对维护活动进行决策或响应意外中断时，请对成本较少时间较长的中断与成本数倍时间较短的中断进行权衡。在中断期间，进行此类决定可能很困难，因此，请创建公式或方法来使用这些中断特征符计算业务成本，以便可进行最佳选择。

多个和级联故障

当设计数据库解决方案来避免和最小化中断以及从中断进行恢复时，请注意多个组件同时发生故障或甚至一个组件的故障导致另一个组件发生故障的可能性。

中断成本

根据业务不同，中断成本会变化。作为最佳实践，每项业务都应该分析其任务的关键业务流程中断的成本。此分析的结果用来制订复原计划。

如果标识了多个流程，那么此计划包括复原活动之间的优先级排序。

中断成本

可以估计客户面对不可用来处理客户事务的数据库系统时业务的成本。例如，可以计算在该数据库系统不可用期间每小时或每分钟的失销收益中的平均成本。计算收益中因客户信任减少而导致的预计损失则要困难得多，但是当评估业务的可用性要求时，应该考虑此成本。

还要考虑内部数据库系统不可用于业务流程时的成本。简单如电子邮件或日历软件的一些内容不可用一小时会使业务逐渐停止，因为职员无法完成其工作。

中断容错

根据业务不同，中断容错会变化。作为最佳实践，每项业务都应该分析其任务的关键业务流程中断的影响。此分析的结果用来制订复原计划。

如果标识了多个流程，那么此计划包括复原的优先级顺序。

中断容错

确定可用性需要时的关键因素是了解业务容错性如何，或对于业务中的特定系统，则是中断的发生。例如，一个主要运行 Web 站点以发布菜单信息的饭店将不会由于服务器的偶尔中断损失很多收益。另一方面，记录交易的股票交易所服务器的任何中断都

将导致重大损失。因此，使用大量资源来确保饭店服务器的可用性为 99.99% 不具成本效益，但对于股票交易所则是高成本效益做法。

在讨论容错时，应该注意两个概念：恢复时间和恢复点。

恢复时间是使业务流程或系统恢复为联机状态所需的时间。

恢复点是对业务流程或系统进行复原的历史点。在数据库术语中，计划将权衡以下两种复原的优点：丢失一些事务的快速复原与不丢失任何事务但用较长时间来执行的完整复原。

恢复和避免策略

当就可用性方面考虑选择购买还是对系统进行设计时，很容易陷入查阅高可用性功能和技术的冗长列表的困境。但是，有关使系统具有并保持高可用性的最佳实践与下列任务的最佳实践一样多，因为它们与购买技术有关：进行很好的设计和配置选择以及设计并实行合理的管理过程和应急计划。

通过首先确定最适合业务需求的高可用性策略，您将为您的投资获取最全面的可用性。然后，可以选择最适合的技术来实现您的策略。

当设计或配置数据库解决方案以获取高可用性时，请考虑可如何避免中断、最大程度降低其影响以及使系统快速恢复。

避免中断

请尽可能避免中断。例如，除去单个故障点以避免意外中断，或研究用于执行联机维护活动的方法以避免计划的中断。监视数据库系统以识别对问题进行指示的系统行为中的趋势，并在这些问题导致中断之前将其解决。

最小化中断的影响

可以设计并配置数据库解决方案，以最小化计划的中断和意外中断的影响。例如，对数据库解决方案进行分配，以便组件和功能局部化，从而允许某些用户应用程序甚至在一个组件脱机时都继续处理事务。

快速地从意外中断进行恢复

制定恢复计划：创建清楚且存档完好的过程，在发生意外中断的情况下，管理员可以方便快速地遵循这些过程；创建对所涉及系统的所有组件进行描述且层次清楚的文档；就近放置服务协议以及组织良好的联系信息。虽然，快速地进行恢复非常重要，但是还请了解要收集哪些诊断信息，以便确定导致中断的根本原因并避免将来发生此中断。

第 2 章 高可用性策略

对于用户而言，数据库请求失败的原因并不重要。无论是因性能不佳而导致事务超时，还是解决方案的组件出现故障，还是管理员已使数据库脱机以执行维护，对用户而言，结果相同。

数据库不可用来处理请求。

提高数据库解决方案可用性的策略包括:

冗余 具有解决方案的每个组件的辅助副本，在出现故障时可以接管工作负载。

系统监视
收集有关解决方案的组件的统计信息，以帮助均衡工作负载或检测组件是否出现故障。

负载均衡
将部分工作负载从超负荷的解决方案组件转移到负载较轻的另一解决方案组件。

故障转移
将所有工作负载从出现故障的解决方案组件转移到辅助组件。

使性能最大化
降低事务花很长时间才完成或超时的机率。

尽量降低维护的影响
安排自动维护活动和手动维护活动，尽可能降低对用户应用程序的影响。

通过冗余获取高可用性

保持高可用性的重要策略是具有冗余组件。如果某个组件出现故障，那么该组件的辅助副本或备份副本可以接管该组件的工作负载，从而使数据库保持对用户应用程序可用。

如果系统的某个组件不是冗余的，那么该组件可能是系统的单一故障点。

冗余在系统设计中很常见:

- 不间断或备用的电源
- 每个组件之间的多个网络纤程
- 网卡的结合度或负载均衡
- 冗余阵列中的多个硬盘驱动器
- CPU 集群

如果系统的这些组件有任何一个不是冗余的，那么该组件将是整个系统中的单一故障点。

© Copyright IBM Corp. 2001, 2013

可以在数据库级别创建冗余，这需要两个数据库：一是主数据库，通常情况下处理所有或大部分应用程序工作负载；二是辅助数据库，在主数据库出现故障时可接管其工作负载。在 DB2 高可用性灾难恢复 (HADR) 环境中，此辅助数据库被称为备用数据库。

对于 DB2 Connect™ 客户机，DB2 for z/OS® 服务器上的综合系统 (sysplex) 工作负载均衡功能为直接连接至数据共享组的客户机应用程序提供高可用性。综合系统 (sysplex) 工作负载均衡功能提供了工作负载均衡和无缝客户机自动重新路由功能。此支持对使用 Java™ 客户机（JDBC、SQLJ 或 pureQuery）或其他客户机（ODBC、CLI、.NET、OLE DB、PHP、Ruby 或嵌入式 SQL）的应用程序可用。

通过故障转移获取高可用性

故障转移是指在主系统出现故障时将工作负载从主系统转移到辅助系统。当工作负载以这种方式转移后，我们称辅助系统已接管故障主系统的工作负载。

示例 1

在集群环境中，如果集群中的一个机器出现故障，集群管理软件可以将在故障机器上运行的进程移至集群中的另一机器。

示例 2

在具有多个 IBM® 数据服务器的数据库解决方案中，一旦某个数据库不可用，那么不再可用的数据库服务器上所连接的数据库应用程序可由数据库管理器重新路由至辅助数据库服务器。

市场上最常见的两种故障转移策略称为空闲备用和相互接管:

空闲备用

在这种配置中，主系统处理所有工作负载，而辅助或备用系统则处于空闲或备用方式，准备在主系统上出现故障时接管工作负载。在高可用性灾难恢复 (HADR) 设置中，您可以具有最多三个备用系统，并且您可以配置每个备用系统以允许只读工作负载。

相互接管

在这种配置中，有多个系统，每个系统都是另一系统的指定辅助系统。一旦某个系统出现故障，该系统所指定的辅助系统必须为出现故障的系统接管工作负载，同时继续处理自己的工作负载，这对总体性能有负面影响。

通过集群获取高可用性

一个集群就是一组互相连接的机器，可一起工作，相当于单个系统。当集群的一个机器出现故障时，集群管理软件会将故障机器上的工作负载转移到其他机器上。

脉动信号监视

要检测集群中一台机器上的故障，故障转移软件可以在机器之间使用脉动信号监视或保持活动包来确定可用性。脉动信号监视需要用到一些系统服务，以在集群中所有机器之间维持连续通信。如果未检测到脉动信号，就会对备份机器启动故障转移。

IP 地址接管

当集群中的一台机器上出现故障时，集群管理器可以通过将 IP 地址从一台机器转移到另一台机器上将工作负载从一台机器上转移到另一机器上。这被称为 IP 地址接管或 IP 接管。客户机应用程序是看不到这种转移的，所以仍会继续使用原来的 IP 地址，而并不知道与该 IP 地址映射的物理机器已更改。

DB2 High Availability Feature允许 IBM DB2 服务器与集群管理软件之间进行集成。

数据库日志记录

数据库日志记录是高可用的数据库解决方案设计的重要部分，因为从故障中恢复过来，以及同步主数据库和辅助数据库都需要用到数据库日志。

所有数据库都有相关的日志。这些日志保存了有关数据库更改的记录。如果需要将数据库复原到上一完整、脱机备份之前的一个点，那么需要日志才能将数据前滚至故障点。

支持以下两种类型的数据库日志记录: *循环*和*归档*。每种类型都提供了不同水平的恢复功能:

- 『循环日志记录』
- 第 10 页的『归档日志记录』

选择归档日志记录的好处是前滚恢复可以使用归档日志和活动日志将数据库复原到日志的结尾或特定时间点。归档日志文件可用来恢复备份完成之后所进行的更改。这不同于循环日志记录，在其中只能恢复至备份时间，并且在该时间之后进行的所有更改都会丢失。

循环日志记录

当创建新数据库时，循环日志记录是缺省行为。（将 **logarchmeth1** 和 **logarchmeth2** 数据库配置参数设置为 OFF。）

对于这种类型的日志记录，只允许完整的脱机数据库备份。进行完整备份时，数据库必须脱机（用户不可访问）。

正如它的名称所表示的那样，循环日志记录使用一个联机日志*环*，提供对事务故障和系统崩溃的恢复。仅使用和保留日志到确保当前事务的完整性这样一个程度。循环日志记录不允许将数据库在上次完整备份操作后执行的事务中前滚。上次备份操作后发生的所有更改都将丢失。因为这种类型的复原操作将数据恢复至进行完整备份的特定时间点，所以它称为*版本恢复*。

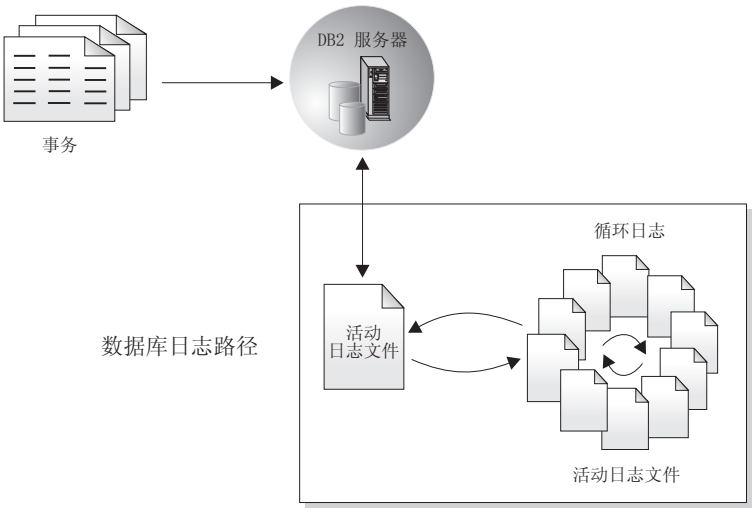


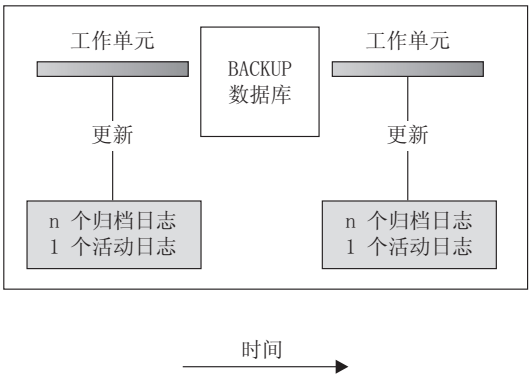
图 1. 循环日志记录

崩溃恢复期间，使用*活动*日志来防止故障（系统电源或应用程序错误）使数据库处于不一致的状态。活动日志位于数据库日志路径目录中。

归档日志记录

归档日志记录专门用于前滚恢复。已归档日志是指已从当前日志路径或镜像日志路径复制到其他位置的日志文件。

可使用 **logarchmeth1** 数据库配置参数和/或 **logarchmeth2** 数据库配置参数来允许您或数据库管理器管理日志归档进程。



日志在备份操作之间使用，以跟踪对数据库的更改.

图 2. 前滚恢复中活动的和已归档的数据库日志. 在运行时间较长的事务中，可以有多个活动日志。

仅当为数据库配置归档日志记录后，才支持进行联机备份。在联机备份操作期间，将记录对数据库的所有活动。完成联机备份后，数据库管理器会强制当前活动的日志关闭并因此对该日志归档。此过程确保联机备份有一组完整的已归档日志可用于恢复。复原联机备份映像时，必须至少将日志前滚至完成备份操作的时间点。为了便于执行此操作，在复原数据库时必须使已归档日志可用。

可使用 **logarchmeth1** 和 **logarchmeth2** 数据库配置参数来指定已归档日志的存储位置。可使用 **logarchmeth1** 参数对 **logpath** 配置参数设置的活动日志路径中的日志文件归档。可使用 **logarchmeth2** 参数将活动日志路径中的日志文件的其他副本归档至另一位置。如果未配置镜像日志记录，那么将从 **logarchmeth1** 参数使用的同一日志路径中获取其他副本。如果配置了镜像日志记录（通过 **mirrorlogpath** 配置参数），那么 **logarchmeth2** 配置参数将改为对镜像日志路径中的日志文件归档，这可以在前滚恢复期间提高弹性。**newlogpath** 参数影响活动日志的存储位置。

在某些方案中，可以压缩已归档日志文件以帮助减少与这些文件相关联的存储开销。如果 **logarchmeth1** 和 **logarchmeth2** 配置参数设置为 DISK、TSM 或 VENDOR，那么可通过将 **logarchcompr1** 和 **logarchcompr2** 配置参数设置为 ON 来启用已归档日志文件压缩。如果以动态方式设置 **logarchcompr1** 和 **logarchcompr2**，那么不会对已归档的日志文件进行压缩。

如果使用 LOGRETAIN 选项来指定用于指示您想要管理活动日志的值，那么数据库管理器会在对活动日志路径中的日志文件归档并且崩溃恢复不再需要这些文件后重命名这些文件。如果启用了无限日志记录，那么需要为更多活动日志文件提供额外空间，这样数据库服务器会在归档日志文件后对其重命名。

日志控制文件

当数据库在出现故障后重新启动时，数据库管理器应用存储在日志文件中的事务信息将数据库返回至一致状态。要确定需要将日志文件中的哪些记录应用于该数据库，数据库管理器使用日志控制文件中记录的信息。

数据库弹性冗余

数据库管理器维护每个成员的日志控制文件的两个副本（SQLOGCTL.LFH.1 和 SQLOGCTL.LFH.2）以及全局日志控制文件的两个副本（SQLOGCTL.GLFH.1 和 SQLOGCTL.GLFH.2），以便当一个副本损坏时，数据库管理器还可使用另一个副本。

性能注意事项

应用日志控制文件中包含的事务信息会增加出现故障后重新启动数据库的开销。通过使用数据库管理概念和配置参考中的“softmax - 恢复范围和软检查点时间间隔配置参数”，可配置数据库管理器向磁盘写入缓冲池页的频率，以降低崩溃恢复期间需要处理的日志记录数。

第 3 章 DB2 服务器的高可用性

IBM DB2 服务器包含支持许多高可用性策略的功能。

客户机自动重新路由路线图

客户机自动重新路由是一个 IBM DB2 服务器功能，它将故障服务器中的客户机应用程序重定向至备用服务器，以便这些应用程序可继续工作并尽量减少中断。仅当在故障前已指定了备用服务器时，才能实现客户机自动重新路由。

表 1 列示了每个类别中的相关主题。

表 1. 客户机自动重新路由信息指南

类别	相关主题
一般信息	• 第 22 页的『客户机自动重新路由的限制』 • 第 19 页的『描述和设置客户机自动重新路由』 • 《*DB2 Connect 安装和配置 DB2 Connect 服务器*》中的“客户机自动重新路由描述和设置”
配置	• 第 22 页的『标识用于客户机自动重新路由的备用服务器』 • Developing Java Applications 中的“为 Java 客户机配置适用于 Linux、UNIX 和 Windows 的 DB2 数据库高可用性支持”
示例	• 第 203 页的『客户机自动重新路由示例』
与其他 DB2 功能部件进行交互	• 第 31 页的『配置客户机自动重新路由和高可用性灾难恢复 (HADR)』 • Developing Java Applications 中的“为 Java 客户机配置适用于 Linux、UNIX 和 Windows 的 DB2 数据库高可用性支持”
故障诊断	• 第 21 页的『用于客户机连接分发器技术的客户机自动重新路由配置』

注: IBM 数据服务器客户机和非 Java IBM 数据服务器驱动程序中也提供了用于 DB2 z/OS 版综合系统 (sysplex) 的客户机自动重新路由功能。借助此支持，访问 DB2 z/OS 版综合系统 (sysplex) 的应用程序可以使用客户机提供的客户机自动重新路由功能，而不需要检查 DB2 Connect 服务器。有关此功能的更多信息，请参阅“*DB2 信息中心*”内有关客户机自动重新路由（客户端）功能的主题。

用于 Linux 和 UNIX 的 DB2 故障监视器工具

（仅在基于 UNIX 的系统上可用）DB2 故障监视器设施通过监视 DB2 数据库管理器实例并重新启动任何过早退出的实例来使 IBM DB2 服务器数据库启动并保持运行。

故障监视器协调程序 (FMC) 是在 UNIX 引导序列中启动的故障监视器设施的进程。`init` 守护程序启动 FMC，并在 FMC 异常终止时重新启动 FMC。FMC 对每个 DB2 实例启动一个故障监视器。每个故障监视器都作为一个守护进程来运行，并且具有与 DB2 实例相同的用户特权。

© Copyright IBM Corp. 2001, 2013

一旦启动了故障监视器，就会监视 DB2 实例以确保它不会过早退出。如果故障监视器发生故障，那么将通过 FMC 将它重新启动。每个故障监视器将依次负责监视一个 DB2 实例。如果 DB2 实例过早退出，那么故障监视器会将其重新启动。仅当发出 **db2stop** 命令时，故障监视器才变为不活动状态。如果 DB2 实例以任何其他方式关闭，故障监视器都会将其再次启动。

DB2 故障监视器限制

如果要使用高可用性集群产品（例如，IBM Tivoli® System Automation for Multiplatforms (SA MP) 或 IBM PowerHA® SystemMirror for AIX®），那么必须关闭故障监视器设施，因为实例启动和关闭由集群产品控制。

DB2 故障监视器与 DB2 运行状况监视器之间的差别

运行状况监视器和故障监视器是作用于单个数据库实例的工具。运行状况监视器使用*运行状况指示器*来评估数据库管理器性能或数据库性能特定方面的运行状况。运行状况指示器测量特定种类数据库对象（例如表空间）某些方面的运行状况。可以针对特定条件评估运行状况指示器以确定该类数据库对象的运行状况。此外，当某个运行状况指示器超出阈值或指示某个数据库对象处于非正常状态时，运行状况指示器可以生成警报以通知您。

与之对比，故障监视器仅仅负责保持它所监视的实例正常运行。如果它所监视的 DB2 实例意外终止，那么故障监视器就会重新启动该实例。故障监视器在 Windows 上不可用。

高可用性灾难恢复 (HADR)

高可用性灾难恢复 (HADR) 功能提供针对部分站点故障和整个站点故障的高可用性解决方案。HADR 通过将数据更改从源数据库（称为*主数据库*）复制到一个或多个目标数据库（称为*备用数据库*）来防止数据丢失。

部分站点故障可能是由硬件、网络或软件（DB2 数据库系统或操作系统）故障引起的。如果没有 HADR，发生部分站点故障时就需要重新启动数据库所在的数据库管理系统（DBMS）服务器。重新启动数据库和数据库所在的服务器所需的时间长度是不可预测的。可能在几分钟时间后，数据库才会恢复为一致状态并可用。使用 HADR 时，备用数据库可在数秒内接管。另外，还可以通过使用客户机自动重新路由功能，或重试应用程序中的逻辑，将使用原始主数据库的客户机重定向至新的主数据库。

当由于灾难（例如，火灾）而导致整个站点被破坏时，就可能会发生整个站点故障。但是，因为 HADR 使用 TCP/IP 在主数据库和备用数据库之间进行通信，所以数据库可以位于不同位置。例如，主数据库可能位于某个城市的总部，而备用数据库位于另一城市的销售办事处。如果在主要站点发生了灾难，那么可以通过让远程备用数据库接管具有所有 DB2 功能的主数据库来维护数据可用性。执行接管操作之后，可以备份原始主数据库，并将其返回至主数据库状态；这即是所谓的*故障回退*。如果您可以使旧的主数据库与新的主数据库保持一致，那么可以启动故障回退。旧的主数据库作为备用数据库重新集成到 HADR 设置之后，可以切换数据库角色，以再次将原始主数据库启用为主数据库。

使用 HADR 时，针对潜在的数据丢失的保护级别取决于您的配置和拓扑选择。您必须做出的一些关键选择如下所示:

您将要使用什么同步级别?

通过在主数据库上生成并递送到备用数据库的日志数据，备用数据库与主数据库保持同步。通过日志，备用数据库不断地前滚。您可以从四种不同同步方式中选择。这些同步方式为 SYNC、NEARSYNC、ASYNC 和 SUPERASYNC（按照保护级别从高到低排列）。有关更多信息，请参阅第 50 页的『高可用性灾难恢复 (HADR) 同步方式』。

您将使用对等时间吗?

对等时间功能指定在对等状态下，如果主数据库丢失 HADR 连接，那么主数据库和备用数据库在配置的一段时间内表现得就像它们仍处于对等状态一样。如果主数据库在对等状态或此“断开连接的对等”状态下发生故障，那么故障转移备用数据库将发生零数据丢失。此功能提供了最佳保护。有关更多信息，请参阅第 39 页的『设置 hadr_timeout 和 hadr_peer_window 数据库配置参数』。

您将会部署多少个备用数据库?

使用 HADR，您可以使用单备用数据库方式或多备用数据库方式。在多备用数据库方式下，您可以通过单一技术同时达到高可用性和灾难恢复目标。有关更多信息，请参阅第 171 页的『HADR 多备用数据库』。

除了将 HADR 备用数据库或其他备用数据库用于其 HA 或 DR 用途之外，您还可以将它们用于许多其他用途。

在备用数据库上读取

您可以使用“在备用数据库上读取”功能将只读工作负载导向到一个或多个备用数据库，但不影响备用数据库的 HA 或 DR 职责。此功能可帮助减轻主数据库上的工作负载，但不影响备用数据库的主要职责。有关该主题的更多信息，请参阅第 191 页的『HADR“在备用数据库上读取”功能』。

除非您已启用“在备用数据库上读取”，否则应用程序只能访问当前主数据库。如果您已启用“在备用数据库上读取”，那么可将只读应用程序重定向至备用数据库。如果发生故障转移，那么连接至备用数据库的应用程序不影响备用数据库的可用性。

延迟重放

您可以使用延迟重放来指定使备用数据库上的日志重放保留在早于主数据库的某个时间点。如果主数据库上发生数据丢失或损坏，那么您可以在时间延迟备用数据库上恢复此数据。有关更多信息，请参阅第 165 页的『HADR 延迟重放』。

滚动更新和升级

使用 HADR 设置，您可以在不停机的情况下对数据库进行各种类型的升级和 DB2 修订包更新。如果正在使用多备用数据库方式，那么您可以在执行升级的同时保留 HADR 所提供的保护。有关更多信息，请参阅第 148 页的『在 DB2 高可用性灾难恢复 (HADR) 环境中执行滚动更新和升级』。

如果数据库中的大多数或所有数据需要保护，或者如果执行必须在备用数据库上自动复制的 DDL 操作时，那么 HADR 可能是最佳选择。但是，HADR 只是 DB2 产品系列中提供的若干复制解决方案之一。InfoSphere® Federation Server 软件和 DB2 数据库系统包括 SQL 复制和 Q 复制解决方案，在某些配置中也可以使用这些解决方案来提供高可用性。这些解决方案在多个位置维护逻辑上一致的数据库表副本。另外，它们还提供灵活性和复杂功能，如支持列和行过滤、数据变换、任何表副本的更新。您还可以在分区数据库环境中使用这些解决方案。

在 IBM Data Studio V3.1 或更高版本中，可以使用以下工具的任务助手：设置 HADR. 任务助手可以指导您执行以下过程：设置选项、查看自动生成的命令以执行任务以及运行这些命令。有关更多详细信息，请参阅使用任务助手管理数据库。

DB2 High Availability Feature

DB2 High Availability Feature允许 IBM DB2 服务器与集群管理软件之间进行集成。

当在集群环境中停止数据库管理器实例时，必须使集群管理器知道实例已停止。如果集群管理器不知道实例已停止，那么集群管理器可能尝试操作，例如对已停止的实例进行故障转移。DB2 High Availability Feature提供了基础结构，在实例配置更改（例如，停止数据库管理器实例）需要更改集群时，使数据库管理器能与集群管理器通信。

每当实例更改需要集群更改时，如果数据库管理器与集群管理器通信，那么您在执行实例配置更改后无需执行单独的集群操作。

DB2 High Availability Feature由以下元素组成:

- IBM Tivoli System Automation for Multiplatforms (SA MP)（作为 DB2 High Availability Feature的一部分）与 AIX 和 Linux 上的 DB2 服务器捆绑到一起，并与 DB2 安装程序集成。可使用 DB2 安装程序或 DB2 服务器安装介质中包括的 **installSAM** 和 **uninstallSAM** 脚本来安装、升级或卸载 SA MP。
- 在集群环境中，某些数据库管理器实例配置和管理操作需要相关集群配置更改。每当您执行某些数据库管理器实例配置和管理操作时，DB2 High Availability Feature (HA) 功能部件使数据库管理器能够自动请求集群管理器配置更改。 请参阅：第 73 页的『使用 DB2 高可用性 (HA) 功能部件来自动配置集群』
- DB2 高可用性实例配置实用程序 (db2haicu) 是基于文本的实用程序，您可以使用它在集群环境中配置和管理高可用性数据库。 请参阅：第 81 页的『DB2 高可用性实例配置实用程序 (db2haicu)』

通过日志装入获取高可用性

日志装入是将整个日志文件复制到备用机器上的过程，可以从归档设备上复制，也可以通过对主数据库运行的用户出口程序来复制。

备用数据库根据生产机器生成的日志文件不断前滚。当生产机器发生故障时，发生故障转移并出现下列情况:

- 余下日志被传送到备用机器。
- 备用数据库前滚至日志的末尾，然后停止。
- 客户机与备用数据库重新连接，并继续运行。

备用机器有它自己的资源（例如磁盘），但必须具有与生产数据库相同的物理和逻辑定义。如果使用此方法，那么应通过使用复原实用程序（通过主数据库的备份）或者使用分割镜像功能（如果可用）来创建初始备用数据库。

要确保能够在灾难恢复情况下恢复数据库，请考虑下列事宜:

- 归档位置应该在地理上独立于主位置。
- 在备用数据库站点远程镜像日志。

- 使用同步镜像，以避免失去支持。可以使用现代磁盘子系统（例如 ESS 和 EMC）或其他远程镜像技术来完成此任务。同时建议使用 NVRAM 高速缓存（本地和远程）来最小化灾难恢复情况下对性能的影响。

如果希望控制在备用机器上要前滚哪些日志文件，那么可通过将 **NORETRIEVE** 选项与 **ROLLFORWARD DATABASE** 命令配合使用来禁用归档日志的检索。其优点如下:

- 通过控制要前滚的日志文件，可确保备用机器落后生产机器 *X* 小时，以避免同时影响这两个系统。
- 如果备用系统没有对归档的访问权（例如，如果 TSM 为归档，那么它只允许原始机器检索这些文件）。
- 还有可能出现以下情况：在生产系统对某个文件进行归档期间，备用系统正在对此文件进行检索，可能获得不完整的日志文件。**NORETRIEVE** 将解决此问题。

注:

1. 当备用数据库处理指示在主数据库上发生索引重建的日志记录时，备用服务器上的索引将不会自动重建。在备用服务器脱离前滚暂挂状态之后，与数据库的首次连接或首次尝试访问索引时，将在备用服务器上重建该索引。如果在主服务器上重建了任何索引，那么建议将备用服务器与主服务器重新同步。如果设置了 **logindexbuild** 数据库配置参数，那么可在前滚操作期间重建索引。
2. 如果在指定了 **COPY YES** 选项的情况下，装入实用程序在主数据库上运行，那么备用数据库必须对副本映像具有访问权。
3. 如果在指定了 **COPY NO** 选项的情况下，装入实用程序在主数据库上运行，那么应该重新使备用数据库同步，否则会将表空间置于复原暂挂状态。
4. 有两种方法来初始化备用机器:
   a. 通过从备份映像复原到备用机器。
   b. 通过创建生产系统的分割镜像并发出带有 **STANDBY** 选项的 **db2inidb** 命令。

   只有在初始化备用机器之后，才能在备用系统上发出 **ROLLFORWARD DATABASE** 命令。
5. 未记录的操作将不会在备用数据库上重放。因此，建议在执行这类操作后重新同步备用数据库。可以通过联机分割镜像和暂挂 I/O 支持来完成此任务。

日志镜像

IBM DB2 服务器支持在数据库级别进行日志镜像。镜像日志文件有助于防止数据库无意中删除活动日志以及硬件故障导致的数据损坏。

如果担心活动日志可能已损坏（由磁盘崩溃所导致），应该考虑使用 **mirrorlogpath** 配置参数指定数据库辅助路径来管理活动日志副本，并镜像存储这些日志的卷。

mirrorlogpath 配置参数允许数据库将日志文件完全相同的第二份副本写至另一路径。建议您将辅助日志路径设置到在物理上独立的磁盘（最好该磁盘也在另一磁盘控制器上）。在那种情况下，磁盘控制器不能是单个故障点。

如果是第一次对 **mirrorlogpath** 配置参数指定值，那么直到下次数据库启动时 DB2 才会使用该值。此行为与 **newlogpath** 配置参数类似。

如果写至活动日志路径或镜像日志路径时出错，数据库会将有问题的路径标记为“坏”，并将消息写至到管理通知日志，然后仅将后续的日志记录写至其他“好”日志路径

中。DB2 将不会再次尝试使用“坏”路径，直到当前日志文件已满或被截断为止。当 DB2 需要打开下一个日志文件时，将会验证此路径是否有效，如果有效就开始使用。如果无效，DB2 将不会尝试再次使用该路径，直到下一个日志文件被第一次访问时为止。不会尝试使日志路径同步，但 DB2 保留了关于发生的访问错误的信息，因此在归档日志文件时可使用正确的路径。如果在写至余下“好”路径时出现故障，那么数据库关闭。

通过暂挂 I/O 和联机分割镜像支持获取高可用性

IBM DB2 服务器暂挂 I/O 支持允许您分割主数据库的镜像副本而不必使数据库脱机。利用这种方法，可以在主数据库出现故障时迅速创建一个备用数据库来接管操作。

磁盘镜像是将数据同时写入两个单独的硬盘中的进程。一个数据副本是另一个数据副本的镜像。分割镜像是分离两个副本的进程。

可以使用磁盘镜像来维护主数据库的辅助副本。可使用 DB2 服务器暂挂 I/O 功能来分割数据库的主镜像副本和辅助镜像副本而不必使数据库脱机。分割主数据库副本和辅助数据库副本后，在主数据库出现故障时，辅助数据库可以接管操作。

如果不想使用 DB2 服务器备份实用程序来备份大型数据库，那么可通过使用暂挂 I/O 和分割镜像功能来通过镜像映像建立副本。此方法还可以:

- 消除来自生产机器的备份操作开销
- 提供了克隆系统的一个快捷方式
- 提供了对空闲备用故障转移的快速实现。不需要初始复原操作，如果证实前滚操作太慢或遇到了错误，重新初始化会非常快。

db2inidb 命令初始化分割镜像，因此可用来:

- 作为克隆数据库
- 作为备用数据库
- 作为备份映像

只能对分割镜像发出此命令，必须首先运行该命令才能使用分割镜像。

在分区数据库环境中，不必同时对所有数据库分区暂挂 I/O 写操作。可以暂挂由一个或多个数据库分区组成的数据库分区子集来创建用于执行脱机备份的分割镜像。如果该子集包括目录分区，那么它必须是要暂挂的最后一个数据库分区。

在分区数据库环境中，必须先对每个数据库分区运行 **db2inidb** 命令，然后才能使用根据任何这些数据库分区创建的分割映像。可以使用 **db2_all** 命令来同时对所有数据库分区运行此工具。但是，如果要使用 RELOCATE USING 选项，那么不能使用 **db2_all** 命令同时在所有数据库分区上运行 **db2inidb**。必须为每个数据库分区提供独立的配置文件，该配置文件包含所更改的数据库分区的 NODENUM 值。例如，如果要更改数据库的名称，这将影响每个数据库分区，并且必须在每个数据库分区上都有独立配置文件的情况下运行 **db2relocatedb** 命令。如果要移动属于单一数据库分区的容器，那么只需要在该数据库分区上运行一次 **db2relocatedb** 命令。

注: 确保分割镜像包含组成数据库的所有容器和目录（包括卷目录）。要收集此信息，请参阅 DBPATHS 管理视图，该视图显示了需要分割的数据库的所有文件和目录。

第 4 章 为获取高可用性进行配置

要配置 DB2 数据库解决方案以获取高可用性，您必须：安排数据库维护活动；配置主数据库服务器和备用数据库服务器，使它们相互了解并了解在出现故障时各自的角色；以及配置任何集群管理软件以从故障集群节点转移工作负载。

开始之前

在配置数据库解决方案之前:

- 组合和安装组成解决方案的底层硬件和软件组件。这些底层组件可能包括：电源；网络连接；网卡；磁盘或其他存储设备；操作系统；以及集群管理软件。
- 在尝试使用这些底层组件进行数据库负载均衡、故障转移或恢复操作之前，在没有任何数据库工作负载的情况下测试这些底层组件，以确保它们工作正常。

关于此任务

冗余是高可用性解决方案的重要组成部分。但是，如果您不明智地安排维护，如果用完了恢复日志所需的存储空间，或者如果集群管理软件未正确配置，那么当用户需要使用数据库完成关键工作时，解决方案可能会不可用。

过程

为获取高可用性进行配置包括:

- 配置客户机重新路由
- 配置故障监视器
- 配置 DB2 高可用性灾难恢复
- 安排维护活动
- 配置日志记录
- 配置集群管理软件

描述和设置客户机自动重新路由

客户机自动重新路由功能的主要目标是使 IBM Data Server Client应用程序能够恢复通信，以便应用程序可以继续工作，并将中断减至最低。

顾名思义，支持连续操作的核心在于重新路由。但是只有存在已向客户机连接标识的备用位置时，才能进行重新路由。

如果服务器是 DB2 for Linux, UNIX, and Windows，那么可以在下列可配置环境中使用客户机自动重新路由功能:

1. 具备 DB2 Database Partitioning Feature的 DB2 Enterprise Server Edition
2. 具备 IBM DB2 pureScale® Feature的 DB2 Enterprise Server Edition
3. InfoSphere Replication Server
4. IBM PowerHA SystemMirror for AIX
5. 高可用性灾难恢复 (HADR)

© Copyright IBM Corp. 2001, 2013

客户机自动重新路由与 HADR 或 DB2 pureScale Feature配合工作，允许客户机应用程序在访问数据库故障转移之后继续其工作，并将中断减至最少。

如果数据库服务器在 System i® 或 System z® 上，那么会在以下配置中使用无缝客户机自动重新路由功能:

1. IBM 数据服务器客户机 数据服务器客户机通过具有备用服务器的 DB2 Connect 服务器连接至 z/OS 或 i5/OS® 系统。将在 IBM Data Server Client与两台 DB2 Connect 服务器之间使用客户机自动重新路由功能。
2. 用于访问 DB2 for z/OS Parallel Sysplex® 数据共享环境的 DB2 Connect 客户机或服务器产品。会在 DB2 Connect 与 z/OS Parallel Sysplex 系统间使用客户机自动重新路由。客户机自动重新路由功能支持 DB2 Connect 特许客户机与 Parallel Sysplex 之间的无缝故障转移。有关无缝故障转移的更多信息，请参阅“*DB2 信息中心*”内有关客户机自动重新路由（客户端）功能的主题。

对于 DB2 Connect 服务器及其备用项，因为不需要使本地数据库同步，所以只需要确保原始和备用 DB2 Connect 服务器按某种方式编目目标主机或 System i 数据库，在此方式下，可使用完全相同的数据库别名来访问该主机或该数据库。

要使 DB2 数据库系统能够恢复通信，必须在通信中断之前指定备用服务器位置。使用 **UPDATE ALTERNATE SERVER FOR DATABASE** 命令来定义特定数据库上的备用服务器位置。

在服务器实例的特定数据库上指定备用服务器位置后，备用服务器位置信息将在连接过程中返回至 IBM 数据服务器客户机。如果在 DB2 Connect 客户机或服务器产品与主机或 System i 数据库服务器之间使用客户机自动重新路由，那么远程服务器必须为其自身提供一个或多个备用地址。对于 DB2 for z/OS，如果数据库是系统复用数据共享环境，那么多个地址已知。因此，不需要在 DB2 Connect 上编目备用服务器。如果客户机与服务器之间的通信因某种原因而中断，那么 IBM Data Server Client将使用备用服务器信息尝试重新建立连接。IBM 数据服务器客户机将尝试重新连接至某个数据库服务器，该数据库服务器可能是原始数据库、服务器上的数据库目录文件中列示的备用服务器或 z/OS Parallel Sysplex 系统返回的服务器列表中的备用服务器。最初可能尝试非常快速地重新建立连接，随后两次尝试之间的时间间隔逐渐增加，所以这些尝试的计时不同。

连接成功后，会返回 SQL30108N 以指示通信失败后已重新建立数据库连接。将返回主机名或 IP 地址以及服务名称或端口号。如果不能重新建立客户机与原始服务器或备用服务器之间的通信，那么 IBM 数据服务器客户机仅向应用程序返回表示原始通信故障的错误。

在 V10.1 FP2 和更高版本的修订包中，在启用了工作负载均衡 (WLB) 功能的情况下连接至 DB2 for z/OS 数据共享组时，非无缝 ACR 功能行为已更改:

- 在连接失败时，CLI 驱动程序不会立即查找新的传输。如果该应用程序重新提交 SET 语句（专用寄存器）或者 SQL 语句，那么 CLI 驱动程序将分配传输。但是，在启用了非无缝 ACR 功能并禁用了 WLB 功能的情况下，CLI 驱动程序将立即查找新的传输并重新连接至下一个可用成员。
- 如果 CLI 驱动程序未能重新连接至主组的成员，并且必须切换至备用组，那么会向该应用程序返回 SQL30108N 两次。如果在 `db2dsdriver.cfg` 文件中指定了备用组，并且 **alternategroup** 参数和 **enableAlternateGroupSeamlessAcr** 设置为 `FALSE`，那么会返回该错误两次。与当前组中的成员的现有连接失败时，会返回第一个 SQL30108N 错误，原因码为 2。尝试与现有主组中的所有成员进行的所有连接都失败

时，会返回第二个 SQL30108N 错误，原因码为 4。如果能够保证重新连接至备用组，那么该应用程序可以再次重新提交 SET 语句或 SQL 语句。返回 ACR 连接错误 (SQL30108N) 时，CLI 驱动程序将跟踪同一连接句柄上的失败成员，以避免向失败成员重新提交该语句。

注: 在下列情况下，SQL30108N 未返回两次:

– 使用 DB2 Connect 服务器作为网关时。

– 显式启用了 ACR 功能，但未启用 WLB 功能时。

连接至 DB2 for z/OS 数据共享组时，不应禁用无缝 ACR 功能和 WLB 功能，除非 IBM 支持机构指示您这样做。

请注意涉及 DB2 Connect 服务器环境中备用服务器连接的以下注意事项:

- 使用 DB2 Connect 服务器来提供代表远程客户机和本地客户机对主机或 System i 数据库的访问时，可能会引起有关系统数据库目录条目中的备用服务器连接信息的混乱。为使混乱程度降至最低，请考虑在系统数据库目录中编目两个条目来表示同一主机或 System i 数据库。为远程客户机编目一个条目，为本地客户机编目另一个条目。
- 从目标 DB2 for z/OS 服务器返回的所有 Parallel Sysplex 信息仅保留在 DB2 Connect 服务器上的高速缓存中。只有一个备用服务器写入磁盘。存在多个备用服务器或活动服务器时，仅在内存中维护信息并且这些信息在进程终止时将丢失。

通常情况下，如果指定了备用服务器，那么当检测到通信错误时将启用客户机自动重新路由。在高可用性灾难恢复 (HADR) 环境中，如果从 HADR 备用服务器返回了 SQL1776N，那么也会启用客户机自动重新路由。

工作负载均衡和客户机自动重新路由要求客户机在 `/etc/hosts` 文件中提供集群内每个成员的条目。例如:

```
10.10.10.1 hostname01.linux hostname01
10.10.10.2 hostname02.linux hostname02
```

用于客户机连接分发器技术的客户机自动重新路由配置

在主数据库服务器出现故障时，分发器或分派器技术（如 WebSphere® Edge Server Load Balancer）会将客户机应用程序重新连接请求分发至一组已定义的系统。

如果要将分发器技术与 DB2 客户机自动重新路由配合使用，那么必须将分发器本身标识为 DB2 客户机自动重新路由的备用服务器。

可以在类似下面的环境中使用分发器技术:

客户机 -> 分发器技术 ->（DB2 Connect 服务器 1 或 DB2 Connect 服务器 2）->DB2 for z/OS

其中:

- 分发器技术组件具有 TCP/IP 主机名 DThostname
- DB2 Connect 服务器 1 具有 TCP/IP 主机名 GWYhostname1
- DB2 Connect 服务器 2 具有 TCP/IP 主机名 GWYhostname2
- DB2 for z/OS 服务器具有 TCP/IP 主机名 zOShostname

将使用 **DThostname** 对客户机进行编目，以便利用分发器技术来访问任何一个 DB2 Connect 服务器。加入分发器技术就可以决定是使用 **GWYhostname1** 还是 **GWYhostname2**。一旦作出了决定，客户机就可与这两个 DB2 Connect 网关的其中一个网关进行直接套接字连接。一旦与所选择的 DB2 Connect 服务器建立了套接字连接，您就建立了客户机至 DB2 Connect 服务器、该服务器至 DB2 for z/OS 的典型连接。

例如，假定分发器选择 **GWYhostname2**。这将生成以下环境:

客户机 -> DB2 Connect 服务器 2 -> DB2 for z/OS

如果发生任何通信故障，分发器不会重试任何连接。如果要在这样的环境中对数据库启用客户机自动重新路由功能，那么应将 DB2 Connect 服务器（DB2 Connect 服务器 1 或 DB2 Connect 服务器 2）中相关联的一个或多个数据库的备用服务器设置为分发器 (DThostname)。然后，如果 DB2 Connect 服务器 1 由于任何原因而锁定，那么将触发客户机自动重新路由，并且在分发器同时作为主服务器和备用服务器的情况下重试客户机连接。此选项允许您使用 DB2 客户机自动重新路由功能来组合和维护分发器功能。将备用服务器设置为除了分发器主机名以外的主机仍然会为客户机提供客户机自动重新路由功能。但是，客户机将与已定义的备用服务器建立直接连接，从而未使用分发器技术，这就消除了分发器及其所带来的价值。

客户机自动重新路由功能拦截下列 SQL 代码:

- SQL20157N
- SQL1768N（原因码: 7）

注: 如果“TCP Keepalive”操作系统配置参数的设置太高，那么可能无法将套接字故障及时通知客户机重新路由。（注意，此配置参数的名称随平台而改变。）

标识用于客户机自动重新路由的备用服务器

每当 DB2 服务器或 DB2 Connect 服务器崩溃时，与该服务器连接的每台客户机都会接收到通信错误，该错误将终止连接，并归结为应用程序错误。

如果可用性极其重要，那么应实现冗余设置或具备将服务器故障转移到备用节点的能力。在上述任一情况下，DB2 客户机节点尝试重新建立与原始服务器的连接，此服务器可能正在故障转移节点上运行（IP 地址也进行故障转移），或者建立与新服务器的连接。

过程

要定义新服务器或备用服务器，请使用 **UPDATE ALTERNATE SERVER FOR DATABASE** 或 **UPDATE ALTERNATE SERVER FOR LDAP DATABASE** 命令。
这些命令将在系统数据库目录中更新数据库别名的备用服务器信息。

客户机自动重新路由的限制

在设计高可用性 DB2 数据库解决方案时，请考虑 DB2 数据库客户机重新路由限制。

以下是 DB2 数据库客户机自动重新路由功能的限制列表:

- 仅当用于连接 DB2 数据库服务器或 DB2 Connect Server服务器的通信协议为 TCP/IP 时，才支持客户机自动重新路由。这意味着，如果连接使用的协议不是 TCP/IP，将不

会启用客户机自动重新路由功能。即使设置 DB2 数据库以便进行回送，仍然必须使用 TCP/IP 通信协议以适应客户机自动重新路由功能。

- 在 DB2 Connect 客户机或服务器产品与主机或 System i 数据库服务器之间使用自动重新路由功能时，如果您处于下列情况中，那么将受到一些相关联的影响:
  - 使用 DB2 Connect Server 代表远程客户机和本地客户机访问主机或 System i 数据库时，系统数据库目录条目中的备用服务器连接信息可能会变得混乱。为使混乱程度降至最低，请考虑在系统数据库目录中编目两个条目来表示同一主机或 System i 数据库。为远程客户机编目一个条目，为本地客户机编目另一个条目。
  - 从目标 DB2 for z/OS 服务器返回的任何综合系统 (sysplex) 信息仅保存在 DB2 Connect Server 的高速缓存中。只有一个备用服务器写入磁盘。存在多个备用服务器或活动服务器时，仅在内存中维护信息并且这些信息在进程终止时将丢失。
- 如果在替代服务器位置重新建立连接，相同数据库别名的任何新连接将连接至替代服务器位置。如果要在解决原始位置上的问题之后建立与其的任何新连接，下列几个选项可供选择:
  - 需要将替代服务器脱机并允许连接转移回原始服务器。（这假定已使用 **UPDATE ALTERNATE SERVER** 命令对原始服务器进行了编目，以便将其设置为备用服务器的替代位置。）
  - 可以对新连接要使用的新数据库别名进行编目。
  - 可以对数据库条目取消编目，然后再次对其进行编目。
- 如果客户机和服务器都支持客户机自动重新路由功能，那么 DB2 for Linux, UNIX, and Windows同时对客户机和服务器支持客户机自动重新路由功能。其他 DB2 数据库产品系列当前不支持此功能。
- 客户机自动重新路由功能的行为与在 DB2 for z/OS 综合系统 (sysplex) 环境中客户机自动重新路由的行为有所不同。尤其表现在下列方面:
  - 客户机自动重新路由功能需要主服务器指定一个备用服务器。这可通过在主服务器中发出 **UPDATE ALTERNATE SERVER FOR DATABASE** 或 **UPDATE ALTERNATE SERVER FOR LDAP DATABASE** 命令来完成。此命令将更新具有备用服务器信息的本地数据库目录，以便同一客户机中的其他应用程序可以访问此信息。相比之下，用于 DB2 for z/OS 的数据共享综合系统 (sysplex) 会维护客户机可以在内存中连接的一个或多个服务器的列表。如果发生通信故障，那么客户机使用该服务器列表来确定适当的备用服务器的位置。
  - 在使用客户机自动重新路由功能的情况下，每次专用寄存器设置更改时，服务器都会告知客户机专用寄存器的最新设置。这将允许客户机在进行重新路由之后尽可能地重新建立运行时环境。相比之下，用于 DB2 for z/OS 的综合系统 (sysplex) 在落实边界将专用寄存器设置返回给客户机，因此需要重放在已重新路由的工作单元内更改的任何专用寄存器。所有其他专用寄存器将自动重放。

  仅在 Linux、UNIX 或 Windows 客户机与 Linux、UNIX 或 Windows 服务器之间提供完全客户机自动重新路由支持。在 Linux、UNIX 或 Windows 客户机与任何受支持版本的 DB2 for z/OS 综合系统 (sysplex) 服务器之间不提供此支持；仅支持重新路由功能。
- 当进行比较时，备用主机服务器中安装的 DB2 数据库服务器的版本必须与原始主机服务器中安装的 DB2 数据库实例的版本相同，但是备用主机服务器可以具有更高版本的修订包。

- 无论您是否具有在客户机上更新数据库目录的权限，替代服务器信息始终保留在内存中。换而言之，如果没有更新数据库目录的权限（或者因为它是只读数据库目录），其他应用程序将不能确定和使用替代服务器，原因是内存未在应用程序之间共享。
- 所有替代位置使用相同的认证方法。这意味着，如果替代位置的认证类型与原始位置的认证类型不同，客户机将无法重新建立数据库连接。
- 当发生通信故障时，所有会话资源（如用于联合处理和专用寄存器的全局临时表、标识、顺序、游标、服务器选项（SET SERVER OPTION））均会丢失。应用程序负责重新建立会话资源以便继续处理工作。因为 DB2 数据库重放在通信错误之前发出的专用寄存器语句，因此重新建立连接之后，无需运行任何专用寄存器语句。但是，某些专用寄存器不能重放。它们是:
  - SET ENCRYPTPW
  - SET EVENT MONITOR STATE
  - SET SESSION AUTHORIZATION
  - SET TRANSFORM GROUP

  如果您在使用 DB2 Connect 时遇到问题，那么您应该参阅特定于数据服务器上的 DB2 Connect 产品的受限制专用寄存器列表。
- 如果在通信故障后重新建立了连接，并且客户机正在使用 CLI 或 JCC 2 类或 4 类驱动程序，那么将通过新服务器重新隐式预编译那些已对原始服务器预编译的 SQL 和 XQuery 语句。但是，将不通过新服务器重新预编译嵌入式 SQL 例程（例如，SQC 或 SQX 应用程序）。
- 不要对支持客户机重新路由功能的数据库别名运行高可用性灾难恢复 (HADR) 命令（**START HADR**、**STOP HADR** 或 **TAKEOVER HADR**）。实现了 HADR 命令来使用数据库别名标识目标数据库。因此，如果为目标数据库定义了替代数据库，HADR 命令就难以确定命令实际上作用于哪个数据库。虽然客户机可能需要使用支持客户机重新路由功能的别名进行连接，但 HADR 命令必须应用于特定的数据库。为了适应这种情况，可以定义特定于主数据库和备用数据库的别名，并且只对那些别名运行 HADR 命令。
- 因为只能对每个数据库服务器定义一个备用服务器，如果具有一个 HADR 多备用数据库设置，那么需要选择一个备用数据库（可以是主体备用数据库）作为主数据库的备用服务器。

另一种实现客户机自动重新路由的方法是使用 DNS 条目来指定 DNS 条目的替代 IP 地址。其思想是在 DNS 条目中指定另一个 IP 地址（备用服务器位置）；客户机虽然不了解备用服务器，但 DB2 数据库系统进行连接时将在此 DNS 条目的 IP 地址之间进行切换。

配置 TCP/IP 保持活动参数

客户机与服务器之间的 DB2 连接使用 TCP/IP 协议进行通信。为避免 TCP/IP 层内的超时导致的潜在故障转移问题，需要在客户机上调整 TCP/IP 保持活动参数。

降低客户机上的保持活动值可更加及时地检测服务器故障。

有一些不同方法可用来更新客户机 TCP/IP 保持活动参数。您选择的方法取决于客户机连接是否基于 IBM Data Server Driver for JDBC and SQLJ。

为高可用性客户机 (JDBC) 配置 TCP/IP 保持活动参数

对于使用 IBM Data Server Driver for JDBC and SQLJ 的客户机系统，TCP/IP 保持活动设置是通过调整某些参数在操作系统级别设置的。

关于此任务

这些命令中提供的值是建议值，但您应根据特定网络和服务器功能对这些设置进行微调。

注: 通过在操作系统级别改变这些设置，会影响客户机上的所有 TCP/IP 通信。

过程

1. **更新 AIX**

   对于 AIX 客户机，需要更改三个操作系统保持活动参数:

   - **tcp_keepidle** - 要使空闲 TCP 连接保持活动的时间长度
   - **tcp_keepintvl** - 发送包以验证 TCP 连接的时间间隔
   - **tcp_keepcnt** - 终止连接前要发送的保持活动探测器的数目

   在 AIX 操作系统上，请使用“network option”命令来更新这些参数:

   ```
   no -o tcp_keepidle=12
   no -o tcp_keepintvl=2
   no -o tcp_keepcnt=10
   ```

   tcp_keepidle 和 **tcp_keepintvl** 值以半秒为单位表示。

2. **更新 Linux**

   对于 Linux 客户机，需要更改四个操作系统保持活动参数:

   - **tcp_keepalive_probes** - 客户机认为连接断开并通知应用层之前发送并未获确认的探测器数。
   - **tcp_keepalive_time** - 发送的最后一个数据包与第一个保持活动探测器之间的时间间隔
   - **tcp_keepalive_intvl** - 发送后续保持活动探测器的时间间隔
   - **tcp_retries2** - 放弃之前重新传输包的最大次数

   在 Linux 操作系统上，请使用“echo”命令来更新这些参数:

   ```
   echo "6" > /proc/sys/net/ipv4/tcp_keepalive_time
   echo "1" > /proc/sys/net/ipv4/tcp_keepalive_intvl
   echo "10" > /proc/sys/net/ipv4/tcp_keepalive_probes
   echo "3" > /proc/sys/net/ipv4/tcp_retries2
   ```

   tcp_keepalive_time 和 **tcp_keepalive_intvl** 值以秒为单位表示。要在系统重新启动后保留这些值，必须将它们添加至 `/etc/sysctl.conf file`。

下一步做什么

对于其他客户机平台，请参阅有关设置 TCP/IP 保持活动值的方式的操作系统文档。

相关信息:

AIX network option 命令

在 Linux 下使用 TCP/IP 保持活动

Microsoft Windows TCP/IP 注册表项

为非高可用性客户机（AIX、HP-UX、Linux 和 Windows）配置 TCP/IP 保持活动参数

在该客户机上设置保持活动参数的建议方法是在 db2dsdriver.cfg 配置文件中使用 **keepAliveTimeout** 参数。

关于此任务

这些命令中提供的值是建议值，但您应根据特定网络和服务器功能对这些设置进行微调。

过程

有两个方法可用来更新未使用 IBM Data Server Driver for JDBC and SQLJ的客户机的 TCP/IP 保持活动参数:

- 修改 db2dsdriver.cfg 文件。

  要设置此参数，请编辑 db2dsdriver.cfg 文件并将 **keepAliveTimeout** 行放到 <acr> 段外部，但仍保留在 <databases> 父段中。例如:

```
<configuration>
  <dsncollection>
    <dsn alias="D3D" name="D3D" host="DB2PS-member0" port="5912" />
  </dsncollection>
  <databases>
    <database name="D3D" host="DB2PS-member0" port="5912">
    <parameter name="keepAliveTimeout" value="20"/>
    <acr>
      <parameter name="enableAcr" value="true"/>
      <parameter name="enableSeamlessAcr" value="true"/>
      <parameter name="affinityFailbackInterval" value="15"/>
  </databases>
...
</configuration>
```

  建议使用此方法，因为它可在没有实例的情况下用于基于实例的客户机和驱动程序。此外，通过使用 db2dsdriver.cfg 文件，每个数据库都可有不同的 **keepAliveTimeout** 设置。

- 修改 **DB2TCP_CLIENT_KEEPALIVE_TIMEOUT** 参数。

  更新此类型的客户机上的保持活动参数的第二个方法是设置 **DB2TCP_CLIENT_KEEPALIVE_TIMEOUT** 参数以检测 TCP/IP 通信层中的故障。

  要更新此参数，请在客户机上的命令窗口或终端中发出以下命令:

```
db2set DB2TCP_CLIENT_KEEPALIVE_TIMEOUT=20
```

  此值以秒为单位指定。

注: 虽然实例连接也支持 TCP/IP 超时保持活动，但只能使用第二种方法（对 **DB2TCP_CLIENT_KEEPALIVE_TIMEOUT** 参数指定值）来设置 TCP/IP 超时保持活动。请注意，客户机自动重新路由 (ACR) 在实例连接的情况下不适用。

DB2 故障监视器注册表文件

在启动故障监视器守护程序时，会为每台物理机器上的每个 DB2 数据库管理器实例创建故障监视器注册表文件。此文件中的关键字和值指定故障监视器的行为。

可在 /sqllib/ 目录中找到故障监视器注册表文件，文件名为 fm.*machine_name*.reg。通过使用 **db2fm** 命令可以改变此文件。

如果故障监视器注册表文件不存在，那么将使用缺省值。

以下是故障监视器注册表文件的内容示例:

```
FM_ON = no
FM_ACTIVE = yes
START_TIMEOUT = 600
STOP_TIMEOUT = 600
STATUS_TIMEOUT = 20
STATUS_INTERVAL = 20
RESTART_RETRIES = 3
ACTION_RETRIES = 3
NOTIFY_ADDRESS = instance_name@machine_name
```

故障监视器注册表文件关键字

FM_ON

指定是否应启动故障监视器。如果该值设置为 NO，那么将不启动故障监视器守护程序，或者如果已经启动，那么将关闭该程序。缺省值是 NO。

FM_ACTIVE

指定故障监视器是否处于活动状态。仅当 **FM_ON** 和 **FM_ACTIVE** 都设置为 YES 时，故障监视器才执行操作。如果 **FM_ON** 设置为 YES 并且 **FM_ACTIVE** 设置为 NO，那么将启动故障监视器守护程序，但故障监视器不活动。这意味着如果 DB2 关闭，那么不会尝试将它重新联机。缺省值为 YES。

START_TIMEOUT

指定故障监视器必须在其内启动它监视的服务的时间长短。缺省值为 600 秒。

STOP_TIMEOUT

请指定一个时间段，故障监视器必须在该时间段内关闭它所监视的服务。缺省值为 600 秒。

STATUS_TIMEOUT

指定故障监视器必须在其内获取它监视的服务的状态的时间长短。缺省值为 20 秒。

STATUS_INTERVAL

指定用来获取受监视的服务的状态的两次连续调用之间的最小时间。缺省值为 20 秒。

RESTART_RETRIES

指定在尝试失败之后故障监视器将尝试获取受监视的服务的状态的次数。一旦达到此数字，故障监视器将执行操作以使服务回到联机状态。缺省值为 3。

ACTION_RETRIES

指定故障监视器将尝试使服务回到联机状态的次数。缺省值为 3。

NOTIFY_ADDRESS

指定故障监视器将通知消息发送至的电子邮件地址。缺省值为 *instance_name*@*machine_name*。

使用 db2fm 命令来配置 DB2 故障监视器

可以使用 **db2fm** 命令来改变 DB2 故障监视器注册表文件。

以下是使用 **db2fm** 命令来更新故障监视器注册表文件的一些示例:

示例 1: 更新 START_TIMEOUT

要将实例 DB2INST1 的 START_TIMEOUT 值更新为 100 秒，请从 DB2 数据库命令窗口输入以下命令:

```
db2fm -i db2inst1 -T 100
```

示例 2: 更新 STOP_TIMEOUT

要将实例 DB2INST1 的 STOP_TIMEOUT 值更新为 200 秒，请输入以下命令:

```
db2fm -i db2inst1 -T /200
```

示例 3: 更新 START_TIMEOUT 和 STOP_TIMEOUT

要将实例 DB2INST1 的 START_TIMEOUT 值更新为 100 秒，并将 STOP_TIMEOUT 值更新为 200 秒，请输入以下命令:

```
db2fm -i db2inst1 -T 100/200
```

示例 4: 打开故障监视

要对实例 DB2INST1 打开故障监视，请输入以下命令:

```
db2fm -i db2inst1 -f yes
```

示例 5: 关闭故障监视

要对实例 DB2INST1 关闭故障监视，请输入以下命令:

```
db2fm -i db2inst1 -f no
```

在 UNIX 系统上，要确认已经不再为 DB2INST1 运行故障监视器，请输入以下命令:

```
ps -ef|grep -i fm
```

在 Linux 上，输入以下命令:

```
ps auxw|grep -i fm
```

显示了 **db2fmd** 和 DB2INST1 的条目表示仍在该实例上运行故障监视器。要关闭故障监视器，请作为实例所有者输入以下命令:

```
db2fm -i db2inst1 -D
```

使用 db2fmcu 命令和系统命令来配置 DB2 故障监视器

可使用 DB2 故障监视控制器命令 **db2fmcu** 或系统命令来配置 DB2 故障监视器。

以下是使用 **db2fmcu** 和系统命令来配置故障监视器的一些示例:

示例 1: 防止 FMC 启动

可使用 DB2 故障监视控制器命令来阻止启动故障监视控制器 (FMC)。 必须以 root 用户身份运行 **db2fmcu** 命令，因为它会访问系统的 inittab 文件。要禁止 FMC 运行，请以 root 用户身份输入以下命令:

```
db2fmcu -d
```

注: 如果应用 DB2 数据服务器修订包，那么此命令将重置以便将 inittab 再次配置为包括 FMC。在应用修订包后，要防止 FMC 启动，必须重新发出此示例中显示的命令。

示例 2: 包括要启动的 FMC

要撤销 **db2fmcu -d** 命令并将 inittab 重新配置为包括 FMC，请输入以下命令:

```
db2fmcu -u -p fullpath
```

其中 *fullpath* 是 **db2fmcd** 对象的完整路径，如 /opt/IBM/db2/bin/db2fmcd。

示例 3: 自动启动 DB2 数据库管理器实例

也可以启用 FMC 以便在系统第一次引导时自动启动实例。要对 DB2INST1 实例启用此功能，请输入以下命令:

```
db2iauto -on db2inst1
```

注: 在 Red Hat Enterprise Linux 6 (RHEL6) 系统上，DB2 故障监视器协调程序守护程序 (db2fmcd) 在系统重新启动之后不会重新启动，因此，即使将 DB2 实例正确配置为自动启动，这些实例也不会重新启动。请查阅以下技术说明以启动故障监视器，以便 db2fmcd 在 RHEL6 系统上自动启动: http://www-01.ibm.com/support/docview.wss?uid=swg21497220。

示例 4: 禁止自动启动实例

要关闭自动启动行为，请输入以下命令:

```
db2iauto -off db2inst1
```

示例 5: 防止故障监视器进程启动

通过针对系统上特定实例更改全局注册表记录字段，可以针对该实例阻止故障监视器进程启动。要更改全局注册表字段以便对 DB2INST1 实例禁用故障监视器，请以 root 用户身份输入以下命令:

```
db2greg -updinstrec instancename=db2inst1!startatboot=0
```

要对 DB2INST1 实例撤销此命令并重新启用故障监视器，请以 root 用户身份输入以下命令:

```
db2greg -updinstrec instancename=db2inst1!startatboot=1
```

初始化高可用性灾难恢复 (HADR)

使用此过程在“单个备用数据库”方式下为 DB2 高可用性灾难恢复 (HADR) 设置和初始化主数据库和备用数据库。

关于此任务

可以通过命令行处理器 (CLP) 或通过调用 db2HADRStart API 来初始化 HADR。

过程

要使用 CLP 以在系统上首次初始化 HADR:

1. 确定每个 HADR 数据库的主机名、主机 IP 地址和服务名称或端口号。

   如果主机使用多个网络接口，请确保 HADR 主机名或 IP 地址映射至预期的接口。您需要在 /etc/services 中为每个受保护数据库分配不同的 HADR 端口。这些端口不能与分配给实例的端口相同。主机名只能映射至一个 IP 地址。

   注: 主数据库和备用数据库的实例名称无需相同。
2. 通过复原备份映像或初始化分割镜像，根据要作为主数据库的现有数据库来创建备用数据库。

   在以下示例中，**BACKUP DATABASE** 和 **RESTORE DATABASE** 命令用于将数据库 SOCKS 初始化为备用数据库。在此情况下，NFS 已安装文件系统在这两处都是可访问的。

   在主数据库上发出以下命令:

   ```
   backup db socks to /nfs1/backups/db2/socks
   ```

   在备用数据库上发出以下命令:

   ```
   restore db socks from /nfs1/backups/db2/socks replace history file
   ```

   以下示例举例说明，如何通过主数据库的分割镜像来使用 **db2inidb** 实用程序来初始化备用数据库。此过程是上述备份和复原过程的替代方案。

   在备用数据库上发出以下命令:

   ```
   db2inidb socks as standby
   ```

   注:

   a. 主数据库和备用数据库的数据库名称必须相同。

   b. 在执行复原操作或分割镜像初始化后，不要在备用数据库上发出 **ROLLFORWARD DATABASE** 命令。使用前滚操作的结果可能与在备用数据库上使用 HADR 重放日志略有不同。如果数据库不相同，那么尝试启动备用数据库将会失败。

   c. 将 REPLACE HISTORY FILE 选项用于 **RESTORE DATABASE** 命令。

   d. 使用 **RESTORE DATABASE** 命令创建备用数据库时，请确保备用数据库仍处于前滚暂挂方式或正在前滚方式。这表示不能发出带有 COMPLETE 选项或 STOP 选项的 **ROLLFORWARD DATABASE** 命令。在停止前滚后，如果尝试在数据库上执行带有 AS STANDBY 选项的 **START HADR** 命令，那么将返回错误。

   e. 设置备用数据库时，应该避免使用下列 **RESTORE DATABASE** 命令选项: TABLESPACE、INTO、REDIRECT 和 WITHOUT ROLLING FORWARD。

f. 使用 **db2inidb** 实用程序来设置备用数据库时，不要使用 SNAPSHOT 或 MIRROR 选项。可以指定 RELOCATE USING 选项以更改下列其中一个或多个配置属性：实例名称、日志路径和数据库路径。但是，切勿更改数据库名或表空间容器路径。

3. 在主数据库和备用数据库上设置下列 HADR 配置参数:
   - **hadr_local_host**
   - **hadr_local_svc**
   - **hadr_remote_host**
   - **hadr_remote_svc**
   - **hadr_remote_inst**

   必须在创建备用数据库后设置这些配置参数。如果在创建备用数据库之前设置了备用数据库上的设置，这些设置将反映在主数据库上设置的内容。

   注: 这是一般的 HADR 设置；要了解更高级的配置选项和设置，请参阅下列链接。
4. 与备用实例连接，并在备用数据库上启动 HADR，如以下示例中所示:

   ```
   START HADR ON DB SOCKS AS STANDBY
   ```

   注: 通常，先启动备用数据库。如果先启动主数据库，那么当未在 **hadr_timeout** 数据库配置参数指定的时间段内启动备用数据库时，此启动过程将失败。
   备用数据库在启动后会进入*本地同步复制*状态，在此状态下，将读取和重放本地的可用日志文件。在备用数据库重放所有本地日志后，它将进入*远程同步复制暂挂*状态。
5. 与主实例连接，并在主数据库上启动 HADR，如以下示例中所示:

   ```
   START HADR ON DB SOCKS AS PRIMARY
   ```

   在主数据库启动后，备用数据库将进入*远程同步复制*状态，它将在此状态下接收来自主数据库的日志页并重放这些页面。在备用数据库已重放主数据库机器的磁盘上的所有日志文件后，两个数据库都将进入*对等*状态。

配置客户机自动重新路由和高可用性灾难恢复 (HADR)

可以将客户机自动重新路由功能与高可用性灾难恢复 (HADR) 功能配合使用，以便将失败的数据库服务器发出的客户机应用程序请求传送至备用数据库服务器。

限制

- 仅当在服务器上指定了替代数据库位置时，才有可能重新路由。
- 只有使用 TCP/IP 协议，才支持客户机自动重新路由。

配置详细信息

- 使用 **UPDATE ALTERNATE SERVER FOR DATABASE** 命令来启用客户机自动重新路由。
- 客户机自动重新路由不使用 **hadr_remote_host** 和 **hadr_remote_svc** 数据库配置参数。
- 在多备用数据库方式下，仅可为客户机自动重新路由指定一个备用数据库。
- 替代主机位置存储在服务器的系统数据库目录文件中。

- 如果未启用客户机自动重新路由，客户机应用程序将接收错误消息 SQL30081N，并且不会进一步尝试建立与服务器的连接。

使用 UPDATE ALTERNATE SERVER FOR DATABASE 命令来设置具有 HADR 的客户机自动重新路由

系统设置如下:

- 您具有一台客户机，其中的数据库 MUSIC 的编目方式与位于主机 HORNET 上的数据库的编目方式相同。
- 数据库 MUSIC 是主数据库，它的相应备用数据库（也是 MUSIC）位于主机 MONTERO 上并且端口号为 456，该端口号是由 **svcename** 配置参数指定的。

要启用客户机自动重新路由，为主机 HORNET 上的数据库 MUSIC 更新备用服务器:

```
db2 update alternate server for database music using hostname montero port 456
```

发出此命令后，客户机必须与主机 HORNET 成功连接以获取备用服务器信息。然后，如果客户机和主机 HORNET 上的数据库 MUSIC 之间发生通信错误，客户机将首先尝试重新与主机 HORNET 上的数据库 MUSIC 连接。如果失败，随后客户机将尝试与主机 MONTERO 上的备用数据库 MUSIC 建立连接。

索引记录和高可用性灾难恢复 (HADR)

您应该考虑为高可用性灾难恢复 (HADR) 数据库设置数据库配置参数 **logindexbuild** 和 **indexrec**。

使用 logindexbuild 数据库配置参数

建议: 对于 HADR 数据库，将 **logindexbuild** 数据库配置参数设置为 ON，以确保为索引的创建、重新创建和重组记录完整的信息。虽然，这意味着在主系统上的索引构建可能需要更长时间和更多日志空间，但是在 HADR 日志重放期间，将在备用系统上重建索引，而且当发生故障转移时索引仍可用。否则，由于日志记录没有包含足够信息来填充新索引，因此，重放索引构建或重建事件时，备用系统会将索引标记为无效。如果不记录主系统上的索引构建，而且发生故障转移，那么在故障转移完成之后所剩的任何无效索引都必须重建，然后才能对其进行访问。当重新创建索引时，任何应用程序都无法对它们进行访问。

注: 如果 LOG INDEX BUILD 表属性设置为其缺省值 NULL，那么 DB2 会使用为 **logindexbuild** 数据库配置参数指定的值。如果 LOG INDEX BUILD 表属性设置为 ON 或 OFF，会忽略为 **logindexbuild** 数据库配置参数指定的值。

可能会选择在一个或多个表上将 LOG INDEX BUILD 表属性设置为 OFF，原因如下:

- 没有足够的活动日志空间来支持索引构建的日志记录。
- 索引数据非常大而且不会经常访问表；因此，它适用于要在接管操作结束时重新创建的索引。在此情况下，将 **indexrec** 配置参数设置为 RESTART。因为不经常访问表，所以此设置会导致系统在接管操作结束时，重新创建索引，而不是等待执行接管操作之后第一次访问表时创建。

如果 LOG INDEX BUILD 表属性在一个或多个表上设置为 OFF，那么这些表上的任何索引构建操作都可能导致在执行接管操作时，随时重新创建索引。类似地，如果 LOG INDEX BUILD 表属性设置为其缺省值 NULL，而且 **logindexbuild** 数据库配置参数

设置为 OFF，那么表上的任何索引构建操作都可能导致在执行接管操作时，随时重新创建该表上的索引。可通过执行下列其中一个操作来阻止重新创建索引:

- 在新的主数据库上重新创建所有无效索引之后，获得数据库的备份，并将其应用于备用数据库。执行此操作的结果是，备用数据库不必对用于在主数据库上重新创建无效索引的日志进行应用，该数据库将这些索引标记为需要在备用数据库上重建。
- 在备用数据库上，将 LOG INDEX BUILD 表属性设置为 ON，或者将 LOG INDEX BUILD 表属性设置为 NULL，并将 **logindexbuild** 配置参数设置为 ON，以确保将记录索引的重新创建。

使用 indexrec 数据库配置参数

建议: 在主数据库和备用数据库上，将 **indexrec** 数据库配置参数设置为 RESTART（缺省值）。这会导致完成接管操作之后重建无效索引。如果未记录任何索引构建，那么此设置允许 DB2 检查无效索引并重建这些索引。此进程在后台进行，而且在接管操作成功完成之后，数据库可访问。

在通过后台重新创建索引进程重建索引之前，如果事务访问具有无效索引的表，那么会通过第一个访问该表的事务来重建这些无效索引。

高可用性灾难恢复 (HADR) 的数据库配置

可以使用数据库配置参数来帮助获得 DB2 HADR 的最佳性能。

大多数情况下，在主数据库和备用数据库所在的系统上，应使用相同的数据库配置参数设置和数据库管理器配置参数设置。如果备用数据库上的配置参数的设置与主数据库上的设置不同，那么可能会产生下列问题:

- 重放主数据库提供的日志文件时，可能会为备用数据库返回错误消息。
- 执行接管操作之后，新的主数据库可能无法处理工作负载，这将导致性能问题，或者，在接收错误消息的应用程序中，这些应用程序在与原始主数据库连接后无法接收到此类错误消息。

对主数据库的配置参数所作的更改不会自动传播至备用数据库。您必须对备用数据库手动执行这些更改。对于动态配置参数来说，更改会立即生效，而不需要先关闭然后重新启动数据库管理系统 (DBMS) 或数据库。对于非动态配置参数来说，更改会在重新启动备用数据库之后生效。

以下是有关 HADR 的特定配置主题的章节:

- 『备用数据库上的日志文件大小配置参数』
- 第 34 页的『备用数据库上的日志接收缓冲区大小』
- 第 34 页的『装入操作和 HADR』
- 第 35 页的『DB2_HADR_PEER_WAIT_LIMIT 注册表变量』
- 第 36 页的『HADR 配置参数』

备用数据库上的日志文件大小配置参数

前一段落中描述的配置参数行为的一种例外情况是 **logfilsiz** 数据库配置参数的行为。尽管不会将此参数的值复制到备用数据库中，但是为了保证两个数据库具有完全相同的日志文件，将忽略备用数据库的 **logfilsiz** 配置参数的设置。备用数据库将创建其大小与主数据库上的日志文件大小相匹配的本地日志文件。

完成接管之后，原始备用数据库（也就是新的主数据库）将使用您对原始主数据库设置的 **logfilsiz** 参数值，直到您重新启动该数据库为止。这时，新的主数据库将恢复为使用您在本地设置的值。此外，在新的主数据库上，当前日志文件会被截断，并且会调整预先创建的任何日志文件的大小。

如果由于执行非强制接管而导致数据库不断切换角色，并且未取消激活任何一个数据库，那么所使用的日志文件大小始终是原始主数据库中的日志文件大小。然而，如果取消激活原始备用数据库（也就是新的主数据库），然后将其重新启动，那么新的主数据库将使用您在本地配置的日志文件大小。如果再次接管原始主数据库，仍将继续使用此日志文件大小。只有将原始主数据库取消激活并接着重新启动它之后，日志文件大小才会恢复为原始主数据库上的设置。

备用数据库上的日志接收缓冲区大小

缺省情况下，备用数据库上的日志接收缓冲区大小是您对主数据库的 **logbufsz** 配置参数指定的值的两倍。此大小可能不够大。例如，考虑在 HADR 同步方式设置为 ASYNC，并且主数据库和备用数据库处于对等状态时可能发生的情况。如果主数据库也正在处理大量事务负载，那么备用数据库的日志接收缓冲区可能会达到最大容量，而主数据库中的日志装入操作可能会停止。要管理这些临时高峰负载，您可以进行下列任一配置更改:

- 通过修改 **DB2_HADR_BUF_SIZE** 注册表变量的值来增大备用数据库上的日志接收缓冲区大小。
- 通过设置 **hadr_spool_limit** 配置参数对备用数据库启用日志假脱机。

装入操作和 HADR

如果您对主数据库发出附带 **COPY YES** 参数的 **LOAD** 命令，那么将对主数据库执行此命令；如果通过此命令所指定的路径或设备可以访问装入副本，那么会将数据复制到备用数据库中。如果从备用数据库无法访问装入副本数据，那么在备用数据库中就会将用来存储表的表空间标记为无效。会跳过将来的任何与此表空间有关的日志记录。要确保装入操作可以访问备用数据库上的装入副本，请使用共享位置来存储 **COPY YES** 参数产生的输出文件。此外，您可以在对主数据库执行装入操作时取消激活备用数据库，将输出文件的副本放到备用数据库路径中，然后激活备用数据库。

如果您对主数据库发出附带 **NONRECOVERABLE** 参数的 **LOAD** 命令，那么将对此主数据库执行此命令，并将备用数据库中的表标记为无效。会跳过将来的任何与此表有关的日志记录。您可以发出附带 **COPY YES** 和 **REPLACE** 参数的 **LOAD** 命令来恢复此表，也可以删除此表以恢复空间。

因为 HADR 不支持执行附带 **COPY NO** 参数的装入操作，所以会自动将此操作转换为附带 **NONRECOVERABLE** 参数的装入操作。要允许将指定了 **COPY NO** 参数的装入操作转换为指定了 **COPY YES** 参数的装入操作，请在主数据库上设置 **DB2_LOAD_COPY_NO_OVERRIDE** 注册表变量。会对备用数据库忽略此注册表变量。请确保备用数据库可以通过使用相同的路径、设备或装入库来访问您为主数据库指定的设备或目录。

如果您正在使用 Tivoli Storage Manager (TSM) 软件来执行附带 **COPY YES** 参数的装入操作，那么可能需要对主数据库和备用数据库设置 **vendoropt** 配置参数。根据 TSM 的配置方式不同，主数据库和备用数据库上的值可能不同。并且，当使用 TSM 来执行附带 **COPY YES** 参数的装入操作时，必须发出附带 **GRANT** 参数的 **db2adutl** 命令，以使备用数据库对所装入的文件具有读访问权。

如果通过附带 **COPY YES** 参数的装入操作复制了表数据，那么将按如下所示复制索引:

- 如果使用 **LOAD** 命令指定 REBUILD 建立索引方式选项，并且将 LOG INDEX BUILD 表属性设置为 ON （使用 ALTER TABLE 语句），或者将表属性设置为 NULL 并将 **logindexbuild** 数据库配置参数设置为 ON，主数据库将在副本文件中包括重建索引对象（即所有在表中定义的索引），以启用备用数据库来复制索引对象。 如果在执行装入操作之前备用数据库上的索引对象已标记为无效，那么在执行装入操作后，由于重建了索引，所以此索引对象将再次可用。
- 只要备用数据库上的索引对象在装入操作之前未标记为无效，如果使用 **LOAD** 命令指定 INCREMENTAL 建立索引方式选项，并且将 LOG INDEX BUILD 表属性设置为 ON （使用 ALTER TABLE 语句），或者将表属性设置为 NULL 并将主数据库上的 **logindexbuild** 数据库配置参数设置为 ON，索引对象就会更新。否则，在备用数据库上将索引标记为无效。

DB2_HADR_PEER_WAIT_LIMIT 注册表变量

限制: 在多备用数据库方式下，本节都不适用于辅助备用数据库，这是因为它们处于 SUPERASYNC 同步方式，因此它们绝不会进入对等状态。

如果您设置 **DB2_HADR_PEER_WAIT_LIMIT** 注册表变量，并且由于将日志复制到备用数据库而导致将 HADR 主数据库的日志记录阻塞了所指定的秒数，那么该主数据库将脱离对等状态。达到此限制时，主数据库将与备用数据库断开连接。如果您通过将 **hadr_peer_window** 配置参数设置为 0 来禁用对等时间，那么主数据库将进入断开连接状态，并且恢复日志记录。如果您启用对等时间，那么主数据库将进入断开连接的对等状态，在这种状态下将继续阻塞日志记录。在重新连接或对等时间到期时，主数据库将脱离断开连接的对等状态。在主数据库脱离断开对等状态后，将恢复日志记录。

注: 如果设置了 **DB2_HADR_PEER_WAIT_LIMIT**，请使用最小值 10 来避免触发假警报。

在数据库脱离对等状态时执行对等时间转换可确保对等时间语义在所有情况下都适用于安全接管。如果主数据库在转换期间失败，那么一般对等时间保护仍然适用: 如果备用数据库仍然处于断开连接的对等状态，那么可以从备用数据库安全接管。

对于备用数据库，在断开连接后，它将继续重放已接收到的日志。在重放接收到的日志后，备用数据库将重新连接至主数据库。重放接收到的日志之后，备用数据库将重新连接至主数据库。重新连接后会进行一般状态过渡: 首先处于远程同步复制状态，然后处于对等状态。

与 hadr_timeout 数据库配置参数的关系

如果在阻塞期间主数据库不断地接收来自备用数据库的脉动信号消息，那么 **hadr_timeout** 数据库配置参数不会使主数据库脱离对等状态。**hadr_timeout** 数据库配置参数指定 HADR 网络层的超时值。如果 HADR 数据库在 **hadr_timeout** 配置参数所指定的时间段内未从其伙伴数据库接收到任何消息，那么该数据库将与其伙伴数据库断开连接。此超时不控制更高层操作（例如，日志装入和 ACK（确认）信号）的超时。如果备用数据库的日志重放在执行大型操作（例如，装入或重组）时被阻塞，那么 HADR 组件仍会按正常时间表将脉动信号消息发送至主数据库。在这样的情况下，除非您设置 **DB2_HADR_PEER_WAIT_LIMIT** 注册表变量，否则只要备用数据库重放被阻塞，主数据库也会被阻塞。

无论连接状态如何，**DB2_HADR_PEER_WAIT_LIMIT** 注册表变量都将取消阻塞主数据库日志记录。即使您未设置 **DB2_HADR_PEER_WAIT_LIMIT** 注册表变量，在检测到网络错误或者连接被关闭时（可能是由于 **hadr_timeout** 配置参数而导致），主数据库也始终会脱离对等状态。

HADR 配置参数

某些 HADR 配置参数是静态参数，例如，**hadr_local_host** 和 **hadr_remote_host**。会在数据库启动时装入静态参数，并且在运行时会忽略更改。完成 **START HADR** 命令时也会装入 HADR 参数。在主数据库中，HADR 可以动态启动和停止，并且数据库保持联机状态。因此，在不关闭数据库的情况下刷新 HADR 配置参数的有效值的一种方法是停止并重新启动 HADR。相比之下，**STOP HADR** 会关闭备用数据库，因此，在数据库处于联机状态时无法刷新备用数据库的参数。

主机名参数以及服务和端口名参数（“单个备用数据库”方式）

一个 HADR 对具有五个您应该设置的相互关联的配置参数:

- **hadr_local_host**
- **hadr_remote_host**
- **hadr_local_svc**
- **hadr_remote_svc**
- **hadr_remote_inst**

主数据库与备用数据库之间的通信使用 TCP 连接。“local”参数指定本地地址，而“remote”参数指定远程地址。主数据库将侦听其本地地址，以监测是否有新连接。未连接至主数据库的备用数据库将重试连接至其远程地址。

备用数据库也会侦听其本地地址。在某些情况下，另一个 HADR 数据库可以在此地址与备用数据库联系并向其发送消息。

除非设置了 **HADR_NO_IP_CHECK** 注册表变量，否则 HADR 会对连接的本地地址和远程地址执行下列交叉检查:

```
my local address = your remote address
```

以及

```
my remote address = your local address
```

使用 IP 地址和端口号来完成此检查，而不使用配置参数中的文字串。您需要在 NAT（网络地址转换）环境中设置 **HADR_NO_IP_CHECK** 注册表变量以绕过此检查。

您可以配置 HADR 数据库以使用 IPv4 或 IPv6 来找到其伙伴数据库。如果主机服务器不支持 IPv6，那么您必须使用 IPv4。如果服务器支持 IPv6，那么数据库是使用 IPv4 还是 IPv6 取决于您为 **hadr_local_host** 和 **hadr_remote_host** 配置参数指定的地址的格式。数据库会尝试将这两个参数解析为相同的 IP 格式，并在可能时使用 IPv6。下表说明了如何确定支持 IPv6 的服务器的 IP 方式:

用于 hadr_local_host 参数的 IP 方式	用于 hadr_remote_host 参数的 IP 方式	用于 HADR 通信的 IP 方式
IPv4 地址	IPv4 地址	IPv4
IPv4 地址	IPv6 地址	Error

用于 **hadr_local_host** 参数的 **IP** 方式	用于 **hadr_remote_host** 参数的 **IP** 方式	用于 **HADR** 通信的 **IP** 方式
IPv4 地址	主机名，仅映射至 IPv4	IPv4
IPv4 地址	主机名，仅映射至 IPv6	Error
IPv4 地址	主机名，映射至 IPv4 和 IPv6	IPv4
IPv6 地址	IPv4 地址	Error
IPv6 地址	IPv6 地址	IPv6
IPv6 地址	主机名，仅映射至 IPv4	Error
IPv6 地址	主机名，仅映射至 IPv6	IPv6
IPv6 地址	主机名，映射至 IPv4 和 IPv6	IPv6
主机名，仅映射至 IPv4	IPv4 地址	IPv4
主机名，仅映射至 IPv4	IPv6 地址	Error
主机名，仅映射至 IPv4	主机名，仅映射至 IPv4	IPv4
主机名，仅映射至 IPv4	主机名，仅映射至 IPv6	Error
主机名，仅映射至 IPv4	主机名，映射至 IPv4 和 IPv6	IPv4
主机名，仅映射至 IPv6	IPv4 地址	Error
主机名，仅映射至 IPv6	IPv6 地址	IPv6
主机名，仅映射至 IPv6	主机名，仅映射至 IPv4	Error
主机名，仅映射至 IPv6	主机名，仅映射至 IPv6	IPv6
主机名，仅映射至 IPv6	主机名，映射至 IPv4 和 IPv6	IPv6
主机名，映射至 IPv4 和 IPv6	IPv4 地址	IPv4
主机名，映射至 IPv4 和 IPv6	IPv6 地址	IPv6
主机名，映射至 IPv4 和 IPv6	主机名，仅映射至 IPv4	IPv4
主机名，映射至 IPv4 和 IPv6	主机名，仅映射至 IPv6	IPv6
主机名，映射至 IPv4 和 IPv6	主机名，映射至 IPv4 和 IPv6	IPv6

仅当主数据库与备用数据库使用同一种 IPv4 或 IPv6 格式时，它们才能建立 HADR 连接。如果一台服务器支持 IPv6（但是也支持 IPv4），而另一台服务器仅支持 IPv4，那么 IPv6 服务器上的 **hadr_local_host** 和 **hadr_remote_host** 参数中至少有一个参数必须指定 IPv4 地址，以强制此服务器上的数据库使用 IPv4。

您可以将 HADR 本地服务参数和远程服务参数（即，**hadr_local_svc** 和 **hadr_remote_svc**）设置为端口号或服务名称。您指定的值必须映射至任何其他服务（其中包括其他 DB2 组件或其他 HADR 数据库）都未使用的端口。特别是，您无法将任何一个参数值设置为服务器在等待来自远程客户机的通信时使用的 TCP/IP 端口（**svcename** 数据库管理器配置参数的值）或下一个端口（**svcename** 参数的值加 1）。

如果主数据库与备用数据库在不同服务器上，那么它们可以使用同一端口号或服务名称；否则，它们必须具有不同的值。

主机名、服务名称或端口名以及目标列表参数（“多个备用数据库”方式）

在“多个备用数据库”方式下，您仍然应该设置 **hadr_local_host**、**hadr_local_svc**、**hadr_remote_host**、**hadr_remote_host** 和 **hadr_remote_inst** 配置参数。如果这些参数设置不正确，那么在主数据库连接至备用数据库之后

会使用 **hadr_target_list** 配置参数的设置来自动更新这些参数。此参数指定所有备用数据库的主机名和端口名。您在目标列表中指定的第一个备用数据库被认为是*主要 HADR 备用数据库*。

在“多个备用数据库”方式下，您仍然应该设置 **hadr_local_host**、**hadr_local_svc**、**hadr_remote_host**、**hadr_remote_host** 和 **hadr_remote_inst** 配置参数。**hadr_local_host** 和 **hadr_local_svc** 参数与在“单个备用数据库”方式下具有相同的含义。在主数据库中，设置 **hadr_remote_host**、**hadr_remote_host** 和 **hadr_remote_inst** 以指示其主要备用数据库。使用新参数 **hadr_target_list** 来列示所有备用数据库，第一个条目为备用数据库。在备用数据库中，设置“remote”参数以指示主数据库。在某些情况下，可以（在主数据库和备用数据库上）自动更新“remote”参数。有关更多信息，请参阅第 176 页的『多个 HADR 备用数据库的数据库配置』中的“自动重新配置 HADR 参数”部分。

同步方式

在“单个备用数据库”方式下，您使用 **hadr_syncmode** 配置参数指定的同步方式必须在主数据库和备用数据库上完全相同。当 HADR 数据库对建立连接时，会检查此配置参数的值的一致性。

在“多个备用数据库”方式下，同步方式不必相同。所有备用数据库都具有由它们所属的备用数据库类型确定的*有效同步方式*。主要备用数据库使用主数据库的同步方式，而辅助备用数据库使用 SUPERASYNC。所有备用数据库都具有*已配置的同步方式*，这是 **hadr_syncmode** 的显式设置，并在备用数据库成为新的主数据库的情况下使用。

有关更详细的信息，请参阅“DB2 高可用性灾难恢复 (HADR) 同步方式”。

HADR 超时和对等时间

您使用 **hadr_timeout** 配置参数指定的超时时间段必须在主数据库和备用数据库上完全相同。当 HADR 数据库对建立连接时，会检查这些配置参数的值的一致性。

有一种例外情况，当主数据库启动时，它会等待备用数据库与其连接，等待时间为以下两个时间段中的较长时间段:

- 最短时间 30 秒
- 由 **hadr_timeout** 数据库配置参数所指定的秒数。

如果备用数据库未在指定的时间内建立连接，启动将失败。当您发出附带 **BY FORCE** 参数的 **START HADR** 命令时将不具有此行为。在这种情况下，主数据库启动时不会等待备用数据库与其建立连接。

在“多个备用数据库”方式下，主数据库仅等待主要备用数据库与其连接；是否连接至辅助备用数据库是可选的。

在 HADR 数据库对建立连接之后，它们将交换脉动信号消息。根据诸如 **hadr_timeout** 和 **hadr_peer_window** 配置参数等因素来计算脉动信号间隔。由 MON_GET_HADR 表函数中的 HEARTBEAT_INTERVAL 字段来报告脉动信号间隔。如果一个数据库在 **hadr_timeout** 配置参数所指定的秒数内未从另一数据库接收到任何消息，那么它会启动断开连接操作。此行为意味着 HADR 数据库最多耗用由 **hadr_timeout** 配置参数所指定的秒数来检测其伙伴数据库的故障或者这两个数据库之间的网络故障。如果将 **hadr_timeout** 配置参数设置得太小，那么您将接收到虚假的警报，并且将频繁地断开连接。

如果您将 **hadr_peer_window** 配置参数设置为非零值，并且主数据库在对等状态下失去与备用数据库的连接，那么主数据库要在恢复与备用数据库的连接或者经过了 **hadr_peer_window** 配置参数指定的时间值（以这两种情况中先发生者为准）之后才会落实事务。

为了获得最大的可用性，**hadr_peer_window** 数据库配置参数的缺省值为 0。当此参数设置为 0 时，主数据库将脱离对等状态，以避免阻塞事务。连接可能会因为备用数据库关闭连接、检测到网路错误或者已达到超时时间而关闭。为了增强数据一致性（但是会降低可用性），您可以将 **hadr_peer_window** 数据库配置参数设置为非零值。

有关更多信息，请参阅“设置 **hadr_timeout** 和 **hadr_peer_window** 数据库配置参数”。

以下是主数据库和备用数据库的样本配置:

主数据库:

```
HADR_LOCAL_HOST           host1.ibm.com
HADR_LOCAL_SVC            hadr_service
HADR_REMOTE_HOST          host2.ibm.com
HADR_REMOTE_SVC           hadr_service
HADR_REMOTE_INST          dbinst2
HADR_TIMEOUT              120
HADR_SYNCMODE             NEARSYNC   HADR_PEER_WINDOW          120
```

备用数据库:

```
HADR_LOCAL_HOST           host2.ibm.com
HADR_LOCAL_SVC            hadr_service
HADR_REMOTE_HOST          host1.ibm.com
HADR_REMOTE_SVC           hadr_service
HADR_REMOTE_INST          dbinst1
HADR_TIMEOUT              120
HADR_SYNCMODE             NEARSYNC   HADR_PEER_WINDOW          120
```

设置 hadr_timeout 和 hadr_peer_window 数据库配置参数

可以配置 **hadr_timeout** 和 **hadr_peer_window** 数据库配置参数以便对连接故障作出最佳响应。

hadr_timeout 数据库配置参数

如果 HADR 数据库在 **hadr_timeout** 数据库配置参数指定的时间内未接收到来自其伙伴数据库的任何通信，那么该数据库将断定与伙伴数据库的连接已断开。如果连接断开时数据库处于对等状态，那么数据库将进入断开连接的对等状态（如果 **hadr_peer_window** 数据库配置参数大于零），或进入远程同步复制暂挂状态（如果 **hadr_peer_window** 数据库配置参数小于或等于零）。状态更改应用于主数据库和备用数据库。

hadr_peer_window 数据库配置参数

hadr_peer_window 配置参数不会替换 **hadr_timeout** 配置参数。**hadr_timeout** 配置参数确定 HADR 数据库在认为其与伙伴数据库的连接失败之前所等待的时间长度。**hadr_peer_window** 配置参数确定数据库在连接断开后是否进入断开连接的对等状态以及数据库应保持此状态的时间长度。HADR 会在 TCP 套接字上进行发送、接收或轮询期间检测到网络错误时立即断开连接。HADR 每 100 毫秒对套接字进行一次轮询。这使其能够对操作系统检测到的网络错误迅速作出响应。仅在最坏的情况下，HADR 会等待直到超时再断开坏连接。在此情况

下，故障发生时正在运行的数据库应用程序被阻塞的时间长度等于 **hadr_timeout** 与 **hadr_peer_window** 数据库配置参数之和。

设置 hadr_timeout 和 hadr_peer_window 数据库配置参数

建议将数据库等待时间设置为最小值。将 **hadr_timeout** 和 **hadr_peer_window** 配置参数设置为较小值将减少当 HADR 备用数据库丢失与主数据库的连接时，数据库应用程序必须等待的时间。但是，在选择指定给 **hadr_timeout** 和 **hadr_peer_window** 配置参数的值时还必须考虑另外两个细节问题:

- **hadr_timeout** 数据库配置参数应设置为足够长的值，以避免因短时、临时网络中断而误报 HADR 连接问题。例如，**hadr_timeout** 的缺省值为 120 秒，这在许多网络上都是一个合理的值。
- **hadr_peer_window** 数据库配置参数应设置为足够长的值，以允许系统执行自动故障响应。如果 HA 系统（例如，集群管理器）在断开连接的对等状态结束之前检测到主数据库故障，那么将对备用数据库执行故障转移。在故障转移过程中，不会发生数据丢失，因为已将所有数据从旧主数据库复制到新主数据库。如果 **hadr_peer_window** 太短，那么 HA 系统可能没有足够时间来检测故障并作出响应。

  注: 在 HADR 多备用数据库方式下，主体备用数据库使用主数据库的 **hadr_peer_window**（*有效对等窗口*）设置。任何辅助备用数据库的 **hadr_peer_window** 设置都没有意义，因为该类型的备用数据库总是在 SUPERASYNC 方式下运行。

HADR 日志假脱机

高可用性灾难恢复 (HADR) 日志假脱机功能支持主数据库上的事务在无需等待备用数据库上的日志重放的情况下继续进行。

启用此功能时，主数据库发出的日志数据将*假脱机*或写至备用数据库上的磁盘，以后日志重放将读取该日志数据。

通过设置 **hadr_spool_limit** 数据库配置参数启用的日志假脱机是对 HADR 功能的改进。当重放缓慢时，主数据库上的新事务可能会被阻塞，因为当缓冲区中没有可用于接收数据的可用空间时，主数据库将不能把日志数据发送给备用系统。日志假脱机功能意味着备用数据库不受其缓冲区的大小限制。当接收的数据增加，缓冲区无法容纳时，日志重放将从磁盘读取数据。这使得系统可以更好地容忍主数据库上的事务量峰值，或备用数据库上的日志重放减慢（因重放特定类型的日志记录导致）。

此功能可能会导致备用数据库上接收到的日志的日志位置与备用数据库上的日志重放位置之间存在较大的间隔，而这可能会导致更长的接管时间。您应该谨慎考虑假脱机限制设置，因为直到假脱机的日志重放结束为止，旧的备用数据库将无法作为新的主数据库启动并接收事务。

DB2 高可用性灾难恢复 (HADR) 的日志归档配置

要将日志归档与 DB2 高可用性灾难恢复 (HADR) 配合使用，请将主数据库和备用数据库配置为能够从所有日志归档位置自动检索日志。对于多备份系统，在主数据库和所有备用数据库上配置归档。

只有当前主数据库能够执行日志归档。如果对主数据库和备用数据库设置了不同的归档位置，那么会只将日志归档到主数据库的归档位置。执行接管时，备用数据库成为

新的主数据库，从该刻起归档的任何日志都将被保存到原始备用数据库的归档位置。在此配置中，日志被归档到其中一个位置，而不是同时被归档到两个位置；有一种例外情况，即执行接管后，新的主数据库可能会归档一小部分已被原始主数据库归档的日志。在多备用系统中，归档日志文件会分散到所有数据库（主数据库和备用数据库）的归档设备。最好选择共享归档，因为所有文件都存储在一个位置。

许多操作都需要检索归档日志文件。这些操作包括：数据库前滚、在远程同步复制中 HADR 主数据库检索日志文件以发送给备用数据库，以及复制程序（例如，Q 复制）读取日志。因此，最好选择 HADR 系统的共享归档，否则，所需的文件会分散到多个归档设备，需要用户干预来找到所需的文件并将它们复制到请求的数据库。建议的复制目的地是归档设备。如果不能复制到归档，那么将日志复制到溢出日志路径。只有在万不得已时，才应将它们复制到日志路径中（但是您应了解存在损坏有效日志文件的风险）。DB2 不会自动删除溢出日志路径和活动日志路径中的用户复制的文件，因此在任何 HADR 备用数据库或任何应用程序不再需要这些文件时，您应该手动将它们除去。

有一个特殊情况，就是多备用数据库方式下的接管。在接管后，新主数据库可能不具有其他备用数据库所需的所有日志文件（因为某个备用数据库位于较旧的日志位置）。如果主数据库无法找到请求的日志文件，那么它将通知该备用数据库，而该备用数据库会关闭连接并在若干秒后重试连接。重试持续时间被限制为若干分钟。当重试时间耗尽时，备用数据库将关闭。在此情况下，您应该将这些文件复制到主数据库，以确保其具有从第一个丢失的文件到其当前日志文件的所有文件。复制文件后，需要时请重新启动备用数据库。

备用数据库自动管理其日志路径中的日志文件。在主数据库通知备用数据库它已归档一个日志文件之前，备用数据库不会从它的本地日志路径中删除该日志文件。此行为防止了日志文件的丢失。如果主数据库发生故障，并且它的日志磁盘已损坏，但尚未在主数据库上归档特定日志文件，备用数据库就不会从它自己的磁盘中删除该日志文件，这是因为它尚未接收到指示主数据库已成功归档该日志文件的通知。如果备用数据库作为新的主数据库进行接管，那么它就会先归档该日志文件，然后再复用该文件。如果同时使用了 **logarchmeth1** 和 **logarchmeth2** 配置参数，那么在主数据库使用这两种方法归档日志文件前，备用数据库不会复用该日志文件。

除了前面列出的好处之外，共享日志归档设备通过以下方式改进同步复制过程：使备用数据库能够直接从处于本地同步复制状态的归档设备检索旧日志文件，而不是间接通过处于远程同步复制的主数据库检索这些文件。但是，建议您不要将串行归档设备（例如，磁带机）用于 HADR 数据库。使用串行设备，由于混合读写操作，主数据库和备用数据库上的性能可能会降低。主数据库在归档日志文件时向该设备写入，而备用数据库从设该备读取以重放日志。即使该设备未配置为共享设备也会对性能产生影响。

Tivoli Storage Manager 上的共享日志归档

将共享日志归档与 IBM Tivoli Storage Manager (TSM) 配合使用将允许一个或多个节点对于 TSM 服务器表现为单个节点，这在任一机器在任何时候都可以作为主服务器的 HADR 环境中特别有用。

要设置共享日志归档，您需要使用代理节点；代理节点允许 TSM 客户机节点对 TSM 服务器上的集中式名称空间执行数据保护操作。目标客户机节点拥有数据，代理节点代表目标节点来管理备份数据。代理节点目标是在 TSM 服务器上定义的、与备份版本的分布式数据相关联的节点名。在 TSM 服务器上的单个名称空间中管理此数据，就像它

完全是此节点的数据一样。代理节点目标名称可以是实节点（例如，一个应用程序主机），也可以时虚拟节点名称（即，没有相应的物理节点）。要创建虚拟代理节点名称，请在 TSM 服务器上使用下列命令:

```
Grant proxynode target=virtual-node-name agent=HADR-primary-name
Grant proxynode target=virtual-node-name agent=HADR-standby-name
```

接下来，您需要在主数据库和备用数据库上将这些数据库配置参数设置为 *virtual-node-name*:

- **vendoropt**
- **logarchopt**

在多备用数据库设置中，您需要评估代理节点对于 TSM 服务器上的所有机器的访问权，并配置所有备用数据库的 **vendoropt** 和 **logarchopt** 配置参数。

高可用性灾难恢复 (HADR) 的性能

配置数据库系统的各个方面（包括网络带宽、CPU 电源和缓冲区大小）可以提高 DB2 高可用性灾难恢复 (HADR) 数据库的性能。

网络是 HADR 设置的关键部分，因为将数据库更改从主数据库复制到备用数据库（以使这两个数据库保持同步）需要网络连接。

关于最大程度地提高网络性能的一些建议:

- 确保网络带宽大于数据库日志生成速率。
- 在选择 HADR 同步方式时考虑网络延迟。网络延迟只有在 SYNC 和 NEARSYNC 方式下才会影响主数据库。

  使用 SYNC 方式对系统性能产生的影响明显大于其他同步方式的影响。在 SYNC 方式下，主数据库只有成功地将日志页写入主数据库日志磁盘后才会将日志页发送至备用数据库。为了保护系统完整性，主数据库先等待来自备用数据库的确认消息，然后再将事务已准备或落实完毕的情况通知应用程序。备用数据库只有在将接收到的日志页写入备用数据库磁盘后才会发送确认消息。性能开销等于在备用数据库上写入日志页所需的时间加上将消息发送回主数据库所需时间。

  在 NEARSYNC 方式下，主数据库以并行方式写日志页并发送它们。然后，主数据库等待来自备用数据库的确认消息。备用数据库一旦将日志页接收到它的内存中，就发出确认消息。在快速网络中，此工作方式能够将主数据库上的开销降至最低。当主数据库完成本地日志写操作时，确认消息可能已到达。

  对于 ASYNC 方式来说，日志的写操作和发送操作是并行进行的；但是，在此方式下，主数据库不等待来自备用数据库的确认消息。因此，不存在网络延迟问题。与 NEARSYNC 方式相比，ASYNC 方式的性能开销更低。

  对于 SUPERASYNC 方式，事务永远不会由于网络中断或拥塞而受到阻塞或经历较长的响应时间。在将先前提交的事务写入主数据库后，马上就可以处理新事务。因此，不存在网络延迟问题。因为主数据库与备用数据库之间的日志间隔可能相对较大，所以完成非强制接管操作的耗用时间可能长于其他方式。
- 考虑调整 **DB2_HADR_SOSNDBUF** 和 **DB2_HADR_SORCVBUF** 注册表变量。

HADR 日志装入工作负载、网络带宽和传输延迟是调整 TCP 套接字缓冲区大小时要考虑的重要因素。两个注册表变量 **DB2_HADR_SOSNDBUF** 和 **DB2_HADR_SORCVBUF** 只允许您为 HADR 连接调整 TCP 套接字发送和接收缓冲区大小。这两个变量的值介于 1024 至 4294967295 范围内，并且是操作系统的套接字缓冲区大小的缺省设置，具体情况因操作系统的不同而异。强烈建议您对 **DB2_HADR_SOSNDBUF** 和 **DB2_HADR_SORCVBUF** 设置使用最小值 16384 (16 K)。一些操作系统将自动对值进行四舍五入或以静默方式对用户指定的值设置上限。

可以使用 HADR 模拟器（一个用于生成模拟 HADR 工作负载的命令行工具）来测量网络性能和试验各种网络调整选项。可以从以下网址下载该模拟器：http://www.ibm.com/developerworks/wikis/display/data/HADR_sim。

网络阻塞

对于在主数据库上写的每个日志，还将相同的日志页发送至备用数据库。每一次写操作都称为*清空*。清空大小不能超过主数据库上的日志缓冲区大小，后者由数据库配置参数 **logbufsz** 控制。每次清空的确切大小是不确定的。即使日志缓冲区较大，也不一定表示清空大小较大。

如果备用数据库重放日志页的速度太慢，它的日志接收缓冲区就可能会填满，从而导致缓冲区无法接收更多的日志页。在 SYNC 和 NEARSYNC 方式下，如果主数据库再次清空其日志缓冲区，数据就有可能缓存在由主数据库机器、网络和备用数据库组成的网络管道中。由于备用数据库没有可用缓冲区来接收数据，因此它无法进行确认，从而导致主数据库在等待备用数据库确认时处于阻塞状态。

在 ASYNC 方式下，主数据库将持续发送日志页直到管道填满为止，之后，它就无法发送更多的日志页。这种情况称为*拥塞*。拥塞由 **hadr_connect_status** 监视元素报告。对于 SYNC 和 NEARSYNC 方式来说，管道通常可以吸收一次清空内容，因此不会发生拥塞。然而，执行清空操作时，主数据库在等待来自备用数据库的确认消息时将处于阻塞状态。

如果备用数据库长时间重放日志记录，例如数据库或表重组日志记录，也可能会发生拥塞。

在 SUPERASYNC 方式下，因为主数据库上的事务落实操作不会受到相对较慢的 HADR 网络或备用 HADR 服务器的影响，所以主数据库与备用数据库之间的日志间隔可能会继续增大。监视日志间隔很重要，这是因为在主系统上发生真正的灾难时，日志间隔是可能丢失的事务数的间接度量。在灾难恢复方案中，日志间隔期间落实的任何事务都对备用数据库不可用。因此，请使用 **hadr_log_gap** 监视元素来监视日志间隔；如果出现日志间隔不可接受的情况，请调查网络中断次数或备用 HADR 服务器的相对速度，并采取纠正措施以减小日志间隔。

关于最大程度地减小网络阻塞的一些建议:

- 备用数据库的功能应该足够强大，从而以主数据库生成数据库操作记录的速度来重放那些记录。建议主数据库与备用数据库使用完全相同的硬件。
- 考虑使用 **DB2_HADR_BUF_SIZE** 注册表变量来调整备用数据库日志接收缓冲区的大小。

  较大的缓冲区有助于减小拥塞，尽管它并不能消除所有拥塞原因。缺省情况下，备用数据库日志接收缓冲区大小是主数据库日志写缓冲区大小的两倍。数据库配置参数 **logbufsz** 指定主数据库日志写缓冲区的大小。

可以使用带有 -hadr 选项的 **db2pd** 命令来确定备用数据库的日志接收缓冲区是否不够大。如果 StandByRcvBufUsed（用于指示已使用的备用数据库日志接收缓冲区的百分比）的值接近 100，那么您应该增大 **DB2_HADR_BUF_SIZE**。

- 考虑设置 **DB2_HADR_PEER_WAIT_LIMIT** 注册表变量，该变量使您能够防止主数据库日志记录由于备用数据库速度较慢或处于阻塞状态而受到阻塞。

  当设置了 **DB2_HADR_PEER_WAIT_LIMIT** 注册表变量时，如果由于将日志复制到备用数据库而导致将 HADR 主数据库上的日志记录阻塞了指定的秒数，那么该主数据库将脱离对等状态。达到此限制时，主数据库将断开与备用数据库的连接。如果对等时间被禁用，那么主数据库将进入断开状态，并且记录将继续进行。如果对等时间已启用，那么主数据库将进入断开对等状态，此时记录继续被阻塞。在重新连接或对等时间到期时，主数据库将脱离断开对等状态。在主数据库脱离断开对等状态后，记录就立即继续进行。

  注: 如果设置了 **DB2_HADR_PEER_WAIT_LIMIT**，请使用最小值 10 来避免触发假警报。

  在脱离对等状态时执行对等时间转换可确保对等时间语义在所有情况下都适用于安全接管。如果主数据库在转换期间失败，那么一般对等时间保护仍适用（只要备用数据库仍处于断开对等状态，就可以从备用数据库安全接管）。
- 在大部分系统中，记录功能不会达到其极限。即使在 SYNC 方式下，在主数据库上也不会有明显的性能下降情况。例如，如果启用 HADR 后记录限制是每秒 40 Mb，但系统在启用 HADR 前的运行速度仅仅是每秒 30 Mb，那么您可能看不出整体系统性能有任何变化。
- 要加速同步复制进程，可以使用共享日志归档设备。但是，如果共享设备是串行设备（如，磁带机），由于混合读写操作，主数据库和备用数据库上的性能可能会降低。
- 如果要使用“在备用数据库上读取”功能，那么备用数据库必须有资源可接纳这个额外的工作。
- 如果要使用“在备用数据库上读取”功能，请在主数据库上配置缓冲池，且该信息将通过日志递送至备用数据库。
- 如果要使用“在备用数据库上读取”功能，请在备用数据库上调整 **pckcachesz**、**catalogcache_sz**、**applheapsz** 和 **sortheap** 配置参数。

可以访问以下 Web 站点以了解关于 HADR 性能调整的最新更新: http://www.ibm.com/developerworks/wikis/display/data/HADR_tune。

集群管理器和高可用性灾难恢复 (HADR)

可以在集群中的节点上实现 DB2 高可用性灾难恢复 (HADR) 数据库，并使用集群管理器来提高数据库解决方案的可用性。

可以让同一个集群管理器同时管理主数据库和备用数据库，也可以让不同集群管理器来管理主数据库和备用数据库。

在同一集群管理器维护的主数据库和备用数据库中设置 HADR 数据库对

此配置最适合两种环境，即主数据库和备用数据库位于相同站点，以及要求尽可能最快速的故障转移环境。这些环境受益于使用 HADR 而不是使用崩溃恢复或其他恢复方法来维持 DBMS 可用性。

可以使用集群管理器来快速发现问题并启动接管操作。因为 HADR 要求单独存储 DBMS，所以应该使用单独卷控件配置集群管理器。在备用系统上使用 DBMS 之前，此配置防止集群管理器等待故障转移出现在卷上。可以使用客户机自动重新路由功能以将客户机应用程序重定向到新的主数据库。

在并非由同一集群管理器维护的主数据库和备用数据库中设置 HADR 数据库对

此配置最适合两种环境，即主数据库和备用数据库位于不同站点，以及整个站点发生故障时灾难恢复要求高可用性的环境。可以通过多种方法实现此配置。当 HADR 主数据库或备用数据库是集群的一部分时，可以使用两种可能的故障转移方案。

- 如果发生部分站点故障，并且 DBMS 可以将故障转移到其中的节点仍然可用，那么可以选择执行集群故障转移。在此情况下，使用集群管理器来执行 IP 地址和卷故障转移；HADR 不受影响。
- 如果在主数据库所处的位置发生整个站点故障，可以通过启动接管操作来使用 HADR 以保持 DBMS 可用性。如果在备用数据库所处的位置发生整个站点故障，可以修复站点或将备用数据库移至其他站点。

初始化备用数据库

要让数据库解决方案具有高可用性，一种策略是同时维护一个主数据库和一个辅助或备用数据库，前者响应用户应用程序请求，后者在前者出现故障时可以接管其数据库操作。

初始化备用数据库需要将主数据库复制到备用数据库。

过程

有几种方法可用来初始化备用数据库。例如:

- 使用磁盘镜像来复制主数据库，并使用 DB2 数据库暂挂的 I/O 支持来分割镜像以创建另一个数据库。
- 创建主数据库的备份映像并将该映像恢复到备用数据库。
- 使用 SQL 复制来从主数据库捕获数据并将该数据应用到备用数据库。

下一步做什么

初始化备用数据库后，必须对数据库解决方案进行配置，使主数据库和备用数据库同步，从而当主数据库失败时，备用数据库可以接管主数据库。

将分割镜像用作备用数据库

在 DB2 pureScale 环境外部，使用以下过程来创建数据库的分割镜像，以用作备用数据库。

如果主数据库发生故障且变得无法访问，那么可以使用备用数据库来接管主数据库。

关于此任务

如果配置主数据库以进行日志归档，那么备用数据库将共享相同的日志归档配置。如果备用数据库可以访问日志归档目标，那么备用数据库会在前滚操作期间，自动从该目标检索日志文件。但是，一旦数据库不处于前滚暂挂状态，那么备用数据库会尝试将日志文件归档到主数据库所使用的相同位置。当备用数据库最初使用与主数据库不同的日志链时，主数据库最终会使用与备用数据库相同的日志链值。这可能会导致主数据库将日志文件归档到备用数据库所归档的日志文件的顶部（或者相反），并可影响这两个数据库的可恢复性。应该先将备用数据库的日志归档目标更改为与主数据库不同的目标，以避免可恢复性问题。

过程

要将分割镜像用作备用数据库:

1. 使用以下命令来连接至主数据库:

   ```
   db2 connect to db_name
   ```

2. 使用以下命令来暂挂主数据库上的 I/O 写操作:

   ```
   db2 set write suspend for database
   ```

   注: 当数据库处于暂挂状态时，您不应该运行其他实用程序或工具。您只应生成数据库副本。您可以选择在发出 **SET WRITE** SUSPEND 之前使用 FLUSH BUFFERPOOLS ALL 语句，以使备用数据库的恢复时间降到最低。

3. 使用相应的操作系统级别和储存器级别命令从主数据库创建一个或多个分割镜像。

   注:

   - 您务必复制整个数据库目录（其中包括卷目录）。还必须复制日志目录和存在于数据库目录之外的任何容器目录。要收集此信息，请参阅 DBPATHS 管理视图，该视图显示了需要分割的数据库的所有文件和目录。
   - 如果您对 **SET WRITE** 命令指定了 EXCLUDE LOGS，请勿将日志文件包括在副本中。

4. 使用以下命令来恢复主数据库上的 I/O 写操作:

   ```
   db2 set write resume for database
   ```

5. 在辅助系统上对镜像数据库进行编目。

   注: 缺省情况下，镜像数据库与主数据库不能存在于同一个系统上。它必须位于具有相同目录结构的辅助系统上并使用与主数据库相同的实例名。如果镜像数据库与主数据库必须存在于同一个系统上，那么可使用 **db2relocatedb** 实用程序或 **db2inidb** 命令的 RELOCATE USING 选项来实现这一点。

6. 使用以下命令来启动辅助系统上的数据库实例:

   ```
   db2start
   ```

7. 通过使用以下命令使辅助系统上的镜像数据库处于前滚暂挂状态，来初始化该镜像数据库:

   ```
   db2inidb <database_alias> as standby
   ```

   如果需要，请指定 **db2inidb** 命令的 RELOCATE USING 选项来重定位备用数据库:

   ```
   db2inidb <database_alias> as standby relocate using relocatedbcfg.txt
   ```

   其中 relocatedbcfg.txt 文件包含重定位数据库所需的信息。

注: 如果您具有 DMS 表空间（数据库管理的空间）或自动存储器表空间，那么可以使用分割镜像建立完整的数据库备份。使用分割镜像建立备份可减少在生产数据库上建立备份的开销。此类备份会被视为联机备份并将包含未完成事务，但不能包括备用数据库中的日志文件。复原此类备份时，必须至少前滚至备份的末尾，然后才能发出带 STOP 选项的 **ROLLFORWARD** 命令。因为该备份将不包含任何日志文件，所以在发出 **SET WRITE** SUSPEND 命令时主数据库中正在使用的日志文件必须可用，否则前滚操作将无法到达备份末尾。

8. 通过配置备用数据库上的日志归档参数或将日志装入至备用数据库，来让备用数据库可以使用主数据库中的归档日志文件。
9. 将数据库前滚至日志结尾或某个时间点。
10. 继续检索日志文件，并通过日志前滚数据库，直到到达日志结尾或备用数据库要求的时间点。
11. 通过发出指定了 STOP 选项的 **ROLLFORWARD** 命令来使备用数据库联机。

注:

- 在使镜像数据库脱离前滚暂挂状态之后，就不能对其应用来自主数据库的日志。
- 如果为主数据库配置了日志归档功能，那么备用数据库将共享相同的日志归档配置。如果备用数据库可以访问日志归档目标，那么备用数据库会在执行前滚时自动从该目标检索日志文件。然而，一旦备用数据库脱离前滚暂挂状态，那么它会尝试将日志文件归档到主数据库所使用的相同位置。虽然备用数据库最初将使用与主数据库不同的日志链，但是主数据库最终会使用与备用数据库相同的日志链值。这可能会导致主数据库将日志文件归档到备用数据库所归档的日志文件的顶部，或者反过来。这会影响这两个数据库的可恢复性。应该将备用数据库的日志归档目标更改为与主数据库的日志归档目标不同，以避免这些问题。

在 DB2 pureScale环境中将分割镜像用作备用数据库

在 DB2 pureScale镜像中，使用以下过程来创建数据库的分割镜像，以用作备用数据库。如果主数据库发生故障且变得无法访问，那么可以使用备用数据库来接管主数据库。

关于此任务

如果配置主数据库以进行日志归档，那么备用数据库将共享相同的日志归档配置。如果备用数据库可以访问日志归档目标，那么备用数据库会在前滚操作期间，自动从该目标检索日志文件。但是，一旦数据库不处于前滚暂挂状态，那么备用数据库会尝试将日志文件归档到主数据库所使用的相同位置。当备用数据库最初使用与主数据库不同的日志链时，主数据库最终会使用与备用数据库相同的日志链值。这可能会导致主数据库将日志文件归档到备用数据库所归档的日志文件的顶部（或者相反），并可影响这两个数据库的可恢复性。应该先将备用数据库的日志归档目标更改为与主数据库不同的目标，以避免可恢复性问题。

过程

要将分割镜像用作备用数据库:

1. 使用以下命令来连接至主数据库:

```
db2 connect to <db_namd>
```

2. 通过抽取并导入主集群的设置，在辅助集群上配置 General Parallel File System (GPFS™)。在主集群上，运行以下 GPFS 命令:

   ```
   mmfsctl <filesystem> syncFSconfig -n <remotenodefile>
   ```

   其中 *<remotenodefile>* 是辅助集群中主机的列表。
3. 使用以下命令来列示集群管理器域:

   ```
   db2cluster -cm -list -domain
   ```
4. 使用以下命令来停止集群中每台主机上的集群管理器:

   ```
   db2cluster -cm -stop -host <host> -force
   ```

   注: 您关闭的最后一台主机必须是您将从中发出此命令的主机。
5. 使用以下命令来停止辅助系统上的 GPFS 集群:

   ```
   db2cluster -cfs -stop -all
   ```
6. 使用以下命令来暂挂主数据库上的 I/O 写操作:

   ```
   db2 set write suspend for database
   ```

   注: 当数据库处于暂挂状态时，您不应该运行其他实用程序或工具。您只应生成数据库副本。可选择在发出 **SET WRITE SUSPEND** 之前清空所有缓冲池，以将恢复时间减到最少。可以使用 FLUSH BUFFERPOOLS ALL 语句来清空所有缓冲池。
7. 使用以下命令来确定必须暂挂和复制的文件系统:

   ```
   db2cluster -cfs -list -filesystem
   ```
8. 使用以下命令来暂挂每个包含数据或日志数据的 GPFS 文件系统:

   ```
   /usr/lpp/mmfs/bin/mmfsctl <filesystem> suspend
   ```

   其中 *<filesystem>* 表示包含数据或日志数据的文件系统。

   注: 当 GPFS 文件系统处于暂挂状态时，将同时阻塞读操作和写操作。在此期间，您应该只能执行分割镜像操作，以使阻塞读操作的时间量降到最低。
9. 使用相应的操作系统级别和储存器级别命令从主数据库创建一个或多个分割镜像。

   注:

   - 您务必复制整个数据库目录（其中包括卷目录）。还必须复制日志目录和存在于数据库目录之外的任何容器目录。要收集此信息，请参阅 DBPATHS 管理视图，该视图显示了需要分割的数据库的所有文件和目录。
   - 如果您对 **SET WRITE** 命令指定了 EXCLUDE LOGS，请勿将日志文件包括在副本中。
10. 对每个已暂挂的文件系统使用以下命令来恢复已暂挂的 GPFS 文件系统:

    ```
    /usr/lpp/mmfs/bin/mmfsctl <filesystem> resume
    ```

    其中 *filesystem* 表示包含数据或日志数据的暂挂文件系统。
11. 使用以下命令来恢复主数据库上的 I/O 写操作:

    ```
    db2 set write resume for database
    ```
12. 使用以下命令来启动辅助系统上的 GPFS 集群:

    ```
    db2cluster -cfs -start -all
    ```
13. 使用以下命令来启动集群管理器

    ```
    db2cluster -cm -start -domain <domain>
    ```

14. 在辅助系统上对镜像数据库进行编目。

    注: 缺省情况下，镜像数据库与主数据库不能存在于同一个系统上。它必须位于具有相同目录结构的辅助系统上并使用与主数据库相同的实例名。如果镜像数据库与主数据库必须存在于同一个系统上，那么可使用 **db2relocatedb** 实用程序或 **db2inidb** 命令的 **RELOCATE USING** 选项来实现这一点。

15. 使用以下命令来启动辅助系统上的数据库实例:

    ```
    db2start
    ```

16. 在辅助系统上，通过将数据库置于前滚暂挂状态将其初始化:

    ```
    db2inidb <database_alias> as standby
    ```

    如果需要，请指定 **db2inidb** 命令的 RELOCATE USING 选项来重定位数据库:

    ```
    db2inidb database_alias as standby relocate using relocatedbcfg.txt
    ```

    其中 relocatedbcfg.txt 包含重定位数据库所需的信息。

    注: 如果您具有 DMS 表空间（数据库管理的空间）或自动存储器表空间，那么可以使用分割镜像建立完整的数据库备份。使用分割镜像建立备份可减少在生产数据库上建立备份的开销。此类备份会被视为联机备份并将包含未完成事务，但不能包括备用数据库中的日志文件。复原此类备份时，必须至少前滚至备份的末尾，然后才能发出 **ROLLFORWARD** STOP 命令。因为该备份将不包含任何日志文件，所以在发出 **SET WRITE** SUSPEND 命令时主数据库中正在使用的日志文件必须可用，否则前滚操作将无法到达备份末尾。

17. 通过配置备用数据库上的日志归档参数或将日志装入至备用数据库，来让备用数据库可以使用主数据库中的归档日志文件。

18. 将数据库前滚至日志结尾或某个时间点。

    注: 执行前滚操作时，可能会遇到 SQL1273 错误。如果分割数据库时未从主系统复制某些日志文件，或者如果某个成员生成日志文件的速度快于其他成员，那么会预期发生这些错误。前滚操作必须停止以保持数据内容一致性时，有时会生成 SQL1273 错误，原因是日志文件的内容取决于来自其他成员的不可用日志文件的内容。如果备用数据库配置为检索由主数据库归档的日志文件，那么您可等待主系统归档必要的日志文件，也可对主系统使用 **ARCHIVE LOG** 命令来强制归档该日志文件。否则，必须将必需的日志文件装入至备用数据库。在备用数据库上提供必需的日志文件后，前滚操作可进一步向前读取日志，尽管可能再次遇到 SQL1273 错误（如果某些成员仍在以快于其他成员的速度生成日志文件）。有关更多信息，请参阅“DB2 pureScale 环境中的备份和复原操作”信息中心主题中的“通过在 DB2 pureScale 环境中传送日志实现的灾难恢复和高可用性”部分。

19. 继续通过日志执行前滚操作，直到到达日志的末尾或备用数据库要求的时间点，以将新的日志文件按照需要装入备用数据库。

20. 通过发出指定了 STOP 选项的 **ROLLFORWARD DATABASE** 命令来使备用数据库联机。

    注:

    - 在使镜像数据库离开前滚暂挂状态之后，就不能对其应用来自主数据库的日志。
    - 如果为主数据库配置了日志归档功能，那么备用数据库将共享相同的日志归档配置。如果备用数据库可以访问日志归档目标，那么备用数据库会在执行前滚时自动从该目标检索日志文件。然而，一旦备用数据库脱离前滚暂挂状态，那

么它会尝试将日志文件归档到主数据库所使用的相同位置。虽然备用数据库最初将使用与主数据库不同的日志链，但是主数据库最终会使用与备用数据库相同的日志链值。这可能会导致主数据库将日志文件归档到备用数据库所归档的日志文件的顶部，或者反过来。这会影响这两个数据库的可恢复性。应该将备用数据库的日志归档目标更改为与主数据库的日志归档目标不同，以避免这些问题。

高可用性灾难恢复 (HADR) 同步方式

HADR 同步方式确定您的 DB2 高可用性灾难恢复 (HADR)数据库解决方案对事务损失的保护程度。同步方式决定主数据库服务器根据备用数据库上记录的状态，何时认为事务已完成。

同步方式配置参数值越严格，数据库解决方案就越能防止事务数据丢失，但事务处理速度也越慢。您必须在防止事务丢失的需求与对性能需求之间取得平衡。

图 3 显示了一些可用的 DB2 HADR 同步方式以及根据所选同步方式认为事务已落实的时间:

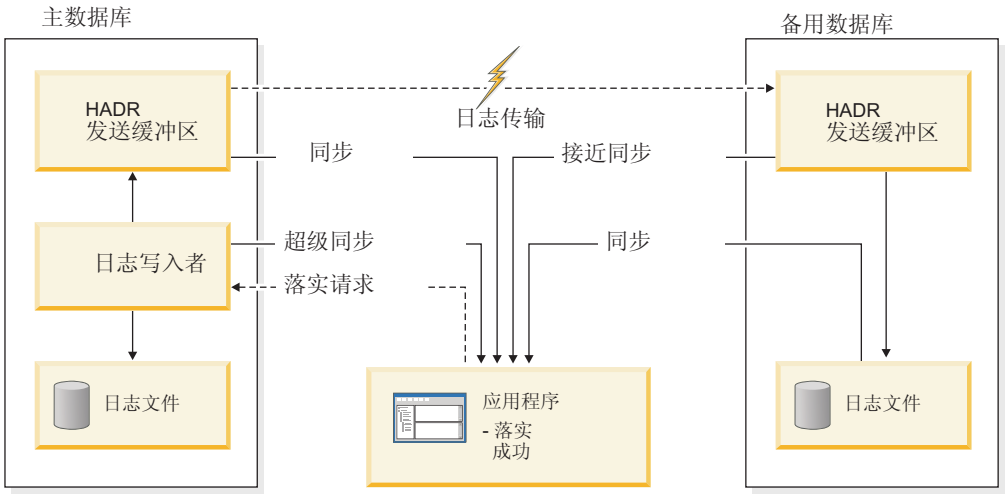


图 3. 高可用性和灾难恢复（HADR）的同步方式

在多备用数据库方式下，**hadr_syncmode** 的设置在主数据库和备用数据库上不必相同。在备用数据库上指定的任意 **hadr_syncmode** 设置将被视为其*已配置的同步方式*；此配置仅在备用数据库成为主数据库时具有相关性。反而系统会对备用数据库指定*有效同步方式*。对于任何辅助备用数据库，有效同步方式始终为 SUPERASYNC。对于主体备用数据库，有效同步方式为主数据库的 **hadr_syncmode** 设置。备用数据库的有效同步方式是任何监视界面显示的值。

使用 **hadr_syncmode** 数据库配置参数来设置同步方式。以下值是有效的:

SYNC（同步）

在四种方式中，此方式将最大可能地避免事务丢失，但使用此方式会导致事务响应时间最长。

在此方式中，仅当日志已写入主数据库上的日志文件，而且主数据库已接收到来自备用数据库的应答，确定日志也已写入备用数据库上的日志文件时，方才认为日志写入是成功的。保证日志数据同时存储在这两处。

如果备用数据库在重放日志记录之前崩溃，那么它下次启动时，可从其本地日志文件中检索和重放这些记录。如果主数据库发生故障，故障转移至备用数据库可以保证任何已在主数据库上落实的事务也在备用数据库上落实了。执行故障转移操作之后，当客户机重新连接至新的主数据库时，新的主数据库上可能会有一些已落实的事务，在原始主数据库上却从未对应用程序将这些事务报告为已落实。当主数据库在处理来自备用数据库的应答消息之前出现故障时，即会出现此种情况。客户机应用程序应考虑查询数据库以确定是否存在此类事务。

如果主数据库断开了与备用数据库的连接，那么接下来的出现的情况取决于 **hadr_peer_window** 数据库配置参数的配置。如果 **hadr_peer_window** 已设置为非零时间值，那么在断开与备用数据库的连接时，主数据库将进入断开连接的对等状态，并且会在落实事务之前继续等待来自备用数据库的确认。如果 **hadr_peer_window** 数据库配置参数已设置为零，那么将不再认为主数据库和备用数据库处于对等状态，并且在等待来自备用数据库的确认时不会暂挂事务。如果在数据库不处于对等或断开连接的对等状态期间进行了故障转移操作，那么将不能保证在主数据库上落实的所有事务都会出现在备用数据库上。

当数据库处于对等或断开连接的对等状态时，如果主数据库失败，那么它可以在完成故障转移操作之后，作为备用数据库重新加入 HADR 数据库对。因为在主数据库接收到来自备用数据库的应答，确认日志已写入备用数据库上的日志文件之前，不认为事务已落实，所以主数据库上的日志顺序将与备用数据库上的日志顺序相同。原始主数据库（现在是备用数据库）只需要通过重放自从执行故障转移操作以来，在新的主数据库上生成的新日志记录来进行同步复制。

如果主数据库发生故障时并未处于对等状态，那么其日志顺序可能与备用数据库上的日志顺序不同。如果必须执行故障转移操作，主数据库和备用数据库上的日志顺序可能不同，因为在故障转移之后，备用数据库启动自己的日志顺序。因为无法撤销某些操作（例如，删除表），所以不可能将主数据库回复到创建新的日志顺序的时间点。如果日志顺序不同，并且在原始主数据库发出了带有 **AS STANDBY** 参数的 **START HADR** 命令，那么将接收到命令成功的消息。但是，此消息是在试图重新集成之前发出的。如果重新集成失败，HADR 对验证消息将发送至主数据库和备用数据库的管理日志和诊断日志。被重新集成的备用数据库将保持等候，但在进行对验证期间，主数据库将拒绝备用数据库，导致备用数据库关闭。如果原始主数据库成功地重新加入 HADR 对，那么可以通过发出未指定 **BY FORCE** 参数的 **TAKEOVER HADR** 命令来完成数据库的故障回退。如果原始主数据库无法重新加入 HADR 对，那么可以通过复原新的主数据库的备份映像将其重新初始化为备用数据库。

NEARSYNC（接近同步）

此方式具有比同步方式更短的事务响应时间，但针对事务丢失提供的保护也较少。

在此方式中，仅当日志记录已写入主数据库上的日志文件，而且主数据库已接收到来自备用系统的应答，确定日志也已写入备用系统上的主存储器时，方才认为日志写入是成功的。仅当两处同时发生故障，并且目标位置未将接收到的所有日志数据转移至非易失性存储器时，才会出现数据的丢失。

如果备用数据库在其将日志记录从存储器复制到磁盘之前崩溃，那么备用数据库上将丢失日志记录。通常，当备用数据库重新启动时，它可以从主数据库中获取丢失的日志记录。但是，如果主数据库或网络上的故障使检索无法进行，

并且需要故障转移，那么日志记录将不会出现在备用数据库上，而且与这些日志记录相关联的事务将不会出现在备用数据库上。

如果事务丢失，那么在故障转移操作之后，新的主数据库与原始主数据库不相同。客户机应用程序应该考虑重新提交这些事务，以便使应用程序状态保持最新。

当主数据库和备用数据库处于对等状态时，如果主数据库发生故障，那么在没有使用完全复原操作重新初始化的情况下，原始主数据库可能无法作为备用数据库重新加入 HADR 对。如果故障转移涉及丢失的日志记录（因为主数据库和备用数据库已发生故障），主数据库和备用数据库上的日志顺序将会不同，并且在未首先执行复原操作的情况下，重新启动原始主数据库以作为备用数据库的尝试将会失败。如果原始主数据库成功地重新加入 HADR 对，那么可以通过发出未指定 **BY FORCE** 参数的 **TAKEOVER HADR** 命令来完成数据库的故障回退。如果原始主数据库无法重新加入 HADR 对，那么可以通过复原新的主数据库的备份映像将其重新初始化为备用数据库。

ASYNC（异步）

与 SYNC 和 NEARSYNC 方式相比，ASYNC 方式使事务响应时间更短，但在主数据库出现故障时，导致事务丢失的可能性更大。

在 ASYNC 方式下，仅当日志记录已写入主数据库上的日志文件，并且已传递到主系统的主机的 TCP 层时，才认为日志写入成功。因为主系统不会等待来自备用系统的确认，所以当事务仍处于正在传入备用数据库的过程中时，可能会认为事务已落实。

主数据库主机上、网络上或备用数据库上的故障可能导致传送中的日志记录丢失。如果主数据库可用，那么会在重新建立“ 对”连接时，将丢失的日志记录重新发送至备用数据库。但是，如果在有丢失日志记录的情况下需要执行故障转移操作，那么那些日志记录将不会到达备用数据库，从而导致在故障转移中丢失关联的事务。

如果事务丢失，那么在故障转移操作之后，新的主数据库与原始主数据库不是完全相同的。客户机应用程序应该考虑重新提交这些事务，以便使应用程序状态保持最新。

当主数据库和备用数据库处于对等状态时，如果主数据库发生故障，那么在没有使用完全复原操作重新初始化的情况下，原始主数据库可能无法作为备用数据库重新加入 HADR 对。如果故障转移涉及丢失的日志记录，主数据库和备用数据库上的日志顺序将会不同，并且重新启动原始主数据库以作为备用数据库的尝试将失败。因为，如果在异步方式中发生故障转移，日志记录更有可能丢失，所以主数据库将不能重新加入 HADR 对的可能性也更大。如果原始主数据库成功地重新加入 HADR 对，那么可以通过发出未指定 **BY FORCE** 参数的 **TAKEOVER HADR** 命令来完成数据库的故障回退。如果原始主数据库无法重新加入 HADR 对，那么可以通过复原新的主数据库的备份映像将其重新初始化为备用数据库。

SUPERASYNC（超级异步）

此方式具有最短的事务响应时间，但在主系统出现故障时，此方式导致事务丢失的可能性也最大。当您不希望事务由于网络中断或拥塞而受到阻塞或经历较长的响应时间时，此方式很有用。

在此方式下，HADR 对永远不会处于对等状态或断开连接的对等状态。只要日志记录已写入主数据库上的日志文件，就认为日志写入成功。由于主数据库不会等待来自备用数据库的确认，所以无论事务的复制状态如何，都会认为已落实该事务。

主数据库主机上、网络上或备用数据库上的故障可能导致传送中的日志记录丢失。如果主数据库可用，那么会在重新建立“ 对”连接时，将丢失的日志记录重新发送至备用数据库。但是，如果在有丢失日志记录的情况下需要执行故障转移操作，那么那些日志记录将不会到达备用数据库，从而导致在故障转移中丢失关联的事务。

如果事务丢失，那么在故障转移操作之后，新的主数据库与原始主数据库不是完全相同的。客户机应用程序应该考虑重新提交这些事务，以便使应用程序状态保持最新。

因为主数据库上的事务落实操作不会受到相对较慢的 HADR 网络或备用 HADR 服务器的影响，所以主数据库与备用数据库之间的日志间隔可能会继续增大。监视日志间隔很重要，这是因为在主系统上发生真正的灾难时，日志间隔是可能丢失的事务数的间接度量。在灾难恢复方案中，日志间隔期间落实的任何事务都对备用数据库不可用。因此，请使用 **hadr_log_gap** 监视元素来监视日志间隔；如果出现日志间隔不可接受的情况，请调查网络中断次数或备用数据库节点的相对速度，并采取纠正措施以减小日志间隔。

如果主数据库发生故障，那么在没有使用完全复原操作重新初始化的情况下，原始主数据库可能无法作为备用数据库重新加入 HADR 对。如果故障转移涉及丢失的日志记录，主数据库和备用数据库上的日志顺序将会不同，并且重新启动原始主数据库以作为备用数据库的尝试将失败。因为在超级异步方式下发生故障转移时，丢失日志记录的可能性更大，所以主数据库无法重新加入 HADR 对的可能性也更大。如果原始主数据库成功地重新加入 HADR 对，那么可以通过发出未指定 **BY FORCE** 参数的 **TAKEOVER HADR** 命令来完成数据库的故障回退。如果原始主数据库无法重新加入 HADR 对，那么可以通过复原新的主数据库的备份映像将其重新初始化为备用数据库。

高可用性灾难恢复 (HADR) 支持

为获得 DB2 数据库高可用性灾难恢复 (HADR) 功能的最佳性能，请在设计高可用性数据库解决方案时考虑系统要求和功能限制。

高可用性灾难恢复 (HADR) 的系统要求

为了获取高可用性灾难恢复 (HADR) 的最佳性能，请确保系统满足下列硬件、操作系统和 DB2 数据库系统要求。

建议：为获得更好的性能，应对主数据库所在的系统和备用数据库所在的系统使用相同的硬件和软件。如果备用数据库所在的系统比主数据库所在的系统拥有的资源少，那么备用数据库可能无法与主数据库所生成的事务装入保持一致。这会导致备用数据库落后，或者主数据库性能降低。在故障转移情况下，新的主数据库应该拥有足够的资源以服务于客户机应用程序。

如果启用“在备用数据库上读取”且使用备用数据库来运行部分只读工作负载，请确保备用数据库具有足够的资源。活动备用数据库需要使用额外的内存和临时表空间来支持事务、会话和新线程以及涉及排序和连接操作的查询。

硬件和操作系统要求

建议：HADR 主数据库和备用数据库使用相同的主机。即，它们应该来自相同的供应商，而且具有相同的体系结构。

主数据库和备用数据库上的操作系统版本（包括补丁）应该相同。在滚动升级过程中可以暂时违反此规则，但要极为谨慎对待。

TCP/IP 接口在 HADR 主机之间必须可用，并且建议使用高速、高容量网络。

DB2 数据库要求

主数据库和备用数据库的数据库系统的版本必须相同；例如，都必须为版本 8 或版本 9。在滚动更新期间，为了测试新的级别，备用数据库的数据库系统的修改级别（例如，修订包级别）在短时间内可以比主数据库的数据库系统的修改级别高。但是，不能长时间保持此配置。如果主数据库的数据库系统的修改级别比备用数据库的数据库系统的修改级别高，那么主数据库与备用数据库不会彼此连接。为了使用“在备用数据库上读取”功能，主数据库和备用数据库都必须是 V9.7 FP1。

主数据库和备用数据库的 DB2 数据库软件的位大小必须相同（32 位或 34 位）。主数据库和备用数据库上的表空间及其容器必须完全相同。必须相同的属性包括表空间类型（DMS 或 SMS）、表空间大小、容器路径、容器大小和容器文件类型（原始设备或文件系统）。在主数据库和备用数据库上，为日志文件分配的空间量也应该相同。

当您对主数据库发出表空间语句（如 CREATE TABLESPACE、ALTER TABLESPACE 或 DROP TABLESPACE）时，会对备用数据库重放此语句。在您对主数据库发出表空间语句之前，必须确保在这两个数据库上安装了所涉及的设备。

主数据库和备用数据库不需要相同的数据库路径。如果使用相对容器路径，那么同一个相对路径可能映射至主数据库和备用数据库上的不同绝对容器路径。

HADR 完全支持存储器组，包括 CREATE STOGROUP、ALTER STOGROUP 和 DROP STOGROUP 语句的复制。与表空间容器类似，存储路径在主数据库和备用数据库上都必须存在。

主数据库和备用数据库的数据库名必须相同。这表示它们必须在不同的实例中。

不支持重定向复原。即，HADR 不支持重定向表空间容器。但是，支持更改数据库目录和日志目录。使用相对路径创建的表空间容器将被复原到相对于新的数据库目录的路径。

缓冲池要求

因为也会在备用数据库上重放缓冲池操作，所以主数据库和备用数据库具有相同的内存量这一点至关重要。如果要使用“在备用数据库上读取”，那么您将需要在主数据库上配置缓冲池，以便活动备用数据库可以容纳日志回放和读取应用程序。

高可用性灾难恢复 (HADR) 的安装和存储要求

要使用高可用性灾难恢复 (HADR) 实现最佳性能，请确保系统满足以下安装和存储要求。

安装要求

对于 HADR，实例路径在主数据库和备用数据库上应该相同。使用不同的实例路径在某些情况下可能会导致出现问题，例如，如果 SQL 存储过程调用用户定义的函数 (UDF) 并且期望 UDF 对象代码对于主服务器和备用服务器都在同一目录下。

存储器要求

HADR 完全支持存储器组，包括 CREATE STOGROUP、ALTER STOGROUP 和 DROP STOGROUP 语句的复制。与表空间容器类似，存储路径在主数据库和备用数据库上都必须存在。可以使用符号链接来创建完全相同的路径。主数据库和备用数据库可以在同一计算机上。即使它们的数据库存储器以相同的路径开头，也不会发生冲突，这是因为使用的实际目录在目录中嵌入了实例名（由于主数据库与备用数据库的数据库名必须相同，所以它们必须在不同的实例中）。存储路径的格式为 `storage_path_name/inst_name/dbpart_name/db_name/tbsp_name/container_name` 。

主数据库和备用数据库上的表空间及其容器必须完全相同。属性必须完全相同，包括：表空间类型（DMS 或 SMS）、表空间大小、容器路径、容器大小和容器文件类型（原始设备或文件系统）。存储器组及其存储路径必须相同。这包括路径名和每个路径上专用于每个存储器组的空间量。在主数据库和备用数据库上，为日志文件分配的空间量也应该相同。

当您对主数据库发出表空间语句（如 CREATE TABLESPACE、ALTER TABLESPACE 或 DROP TABLESPACE）时，会对备用数据库重放此语句。在您对主数据库发出表空间语句之前，必须确保在这两个数据库上安装了所涉及的设备。

如果主数据库和备用数据库上的表空间设置不同，那么对备用数据库重放日志时可能会遇到错误，如 OUT OF SPACE 或 TABLE SPACE CONTAINER NOT FOUND。同样，如果主数据库和备用数据库上存储器组的存储路径设置不同，那么不会重放与 CREATE STOGROUP 或 ALTER STOGROUP 相关联的日志记录。因此，现有存储路径可能会过早用完备用系统上的空间，并且自动存储器表空间无法增加大小。如果发生任一上述情况，就会将受影响的表空间置于前滚暂挂状态并在后面重放日志时忽略该表空间。如果执行接管操作，那么该表空间对应用程序不可用。

如果在接管之前发现备用系统上有问题，那么解决方法是在解决存储器问题时重新建立备用数据库。执行此操作的步骤包括:

- 取消激活备用数据库。
- 删除备用数据库。
- 确保必需的文件系统存在并具有足够的可用空间用于后续复原和前滚。
- 使用主数据库的最新备份来复原备用系统上的数据库（或者，使用 **db2inidb** 命令通过分割镜像或快速复制来重新初始化）。不应该在复原期间重新定义存储器组存储路径。另外，不应将表空间容器作为复原的一部分进行重定向。
- 在备用系统上重新启动 HADR。

但是，如果在执行接管之后才发现备用数据库有问题（或者如果选择了直到此时才解决存储器问题），那么解决方案取决于所遇到问题的类型。

如果数据库启用了自动存储器，但在与备用数据库相关联的存储路径上没有可用空间，请遵循以下步骤:

1. 通过扩展文件系统或除去文件系统上不必要的非 DB2 文件以在存储路径上提供空间。
2. 执行表空间前滚至日志末尾。

如果日志重放期间未能添加或扩展容器，并且必需的备份映像和日志文件归档可用，那么您可能能够恢复表空间，方法是先发出带 IGNORE ROLLFORWARD CONTAINER OPERATIONS 选项的 SET TABLESPACE CONTAINERS 语句，然后发出 **ROLLFORWARD** 命令。

主数据库和备用数据库不需要相同的数据库路径。如果使用相对容器路径，那么同一个相对路径可能映射至主数据库和备用数据库上的不同绝对容器路径。因此，如果主数据库和备用数据库在同一台计算机上，那么必须使用相对路径来定义所有表空间容器，以使它们映射到主数据库和备用数据库的不同路径。

HADR 和网络地址转换 (NAT) 支持

因为 NAT（它在 HADR 环境中受支持）会隐藏服务器的实地址，所以通常将它用于防火墙和安全性。

在 HADR 设置中，会对主节点和备用节点上的本地主机配置和远程主机配置进行交叉检查以确保这些配置都正确。在 NAT 环境中，主机通过特定 IP 地址为其本身所知，但是通过另一个 IP 地址为其他主机所知。除非您将 **DB2_HADR_NO_IP_CHECK** 注册表变量设置为 ON，否则此行为会导致 HADR 主机交叉检查失败。使用此设置将导致忽略交叉检查，从而使主节点和备用节点能够在 NAT 环境中进行连接。

如果您不是在 NAT 环境中工作，那么对 **DB2_HADR_NO_IP_CHECK** 注册表变量使用缺省设置 OFF。禁用交叉检查会削弱 HADR 对配置的验证。

有关 HADR 多备用数据库方式的注意事项

在具有“多个备用数据库”设置的 NAT 环境中，每个备用数据库的 **hadr_local_host** 和 **hadr_local_svc** 设置必须仍然列示在主数据库的 **hadr_target_list** 中，否则主数据库不会接受来自该备用数据库的连接。

通常，在多备用数据库方式下，在启动时，备用数据库将检查其 **hadr_remote_host** 和 **hadr_remote_svc** 设置在其 **hadr_target_list** 中，以确保在进行角色切换时旧的主数据库可以成为新的备用数据库。在 NAT 方案中，除非 **DB2_HADR_NO_IP_CHECK** 注册表变量为 ON，否则该检查将失败。因为已忽略此检查，所以备用数据库将等到它连接至主数据库时才会检查主数据库的 **hadr_local_host** 和 **hadr_local_svc** 是否位于备用数据库的 **hadr_target_list** 中。此检查仍然可确保在此对数据库之间可以成功地进行角色切换。

注：如果 **DB2_HADR_NO_IP_CHECK** 注册表变量设置为 ON，那么不会自动更新 **hadr_remote_host** 和 **hadr_remote_svc**。

在“多个备用数据库”设置中，应该对所有可能跨越 NAT 边界与另一个数据库建立连接的数据库设置 **DB2_HADR_NO_IP_CHECK**。如果数据库决不会跨越 NAT 边界以连接至另一个数据库（即，未配置这样的链路），那么您不应对该数据库设置此注册表变量。设置了 **DB2_HADR_NO_IP_CHECK** 之后，它会阻止备用数据库在进行接管之后自动发现新的主数据库，您必须手动重新配置此备用数据库才能将其连接至新的主数据库。

高可用性灾难恢复 (HADR) 的限制

为了获取高可用性灾难恢复 (HADR) 的最佳性能，请在设计高可用性 DB2 数据库解决方案时考虑 HADR 的限制。

下表对高可用性灾难恢复 (HADR) 的限制进行了概述:

- 在分区数据库环境中，不支持 HADR。
- HADR 在 DB2 pureScale环境中不受支持。
- 主数据库和备用数据库的操作系统版本和 DB2 数据库系统版本都必须相同，但滚动升级期间可有短时不同。
- 主数据库和备用数据库的 DB2 数据库系统软件的位大小必须相同（32 位或 64 位）。
- 除非已启用“在备用数据库上读取”，否则客户机无法与备用数据库连接。“在备用数据库上读取”可让客户机连接至活动备用数据库及发出只读查询。
- 如果已启用“在备用数据库上读取”，那么不允许在备用数据库上执行写入日志记录的操作；只有读取客户机可以连接至活动备用数据库。
- 如果已启用“在备用数据库上读取”，那么不允许在备用数据库上执行修改数据库内容的写操作。尝试修改数据库对象的任何异步线程（例如实时统计信息收集和自动重建索引）和实用程序都将不受支持。实时统计信息收集和自动重建索引不应在备用数据库上运行。
- 日志文件只能由主数据库进行归档。
- 只能在当前主数据库上运行自调整内存管理器 (STMM)。在主数据库启动或备用数据库通过接管而转换为主数据库后，直到第一个客户机连接生效，STMM EDU 才可能启动。
- 在备用数据库上不支持备份操作。
- 未进行日志记录的操作（例如，对数据库配置参数和恢复历史记录文件的更改）以及具有 `NOT LOGGED` 选项的 LOB 表列不会复制到备用数据库。
- 不支持指定了 **`COPY`** `NO` 选项的装入操作。
- HADR 不支持对数据库日志文件使用原始 I/O（直接磁盘访问）。如果 HADR 是通过 **`START HADR`** 命令启动的，或者在配置了 HADR 的情况下激活（重新启动）数据库，并且检测到原始日志，那么相关联的命令将失败。
- 联合服务器未完全支持 HADR。对于一阶段落实，HADR 数据库可以充当联合服务器（事务管理器）或数据源（资源管理器），这需要客户机重新路由配置。对于两阶段落实，HADR 数据库只能充当数据源；HADR 数据库不能充当联合服务器（因为存在数据一致性限制）。
- HADR 不支持无限日志记录。
- HADR 主数据库的系统时钟必须与 HDAR 备用数据库的系统时钟同步。

为获取高可用性安排维护

DB2 数据库解决方案需要定期维护。您将需要执行诸如以下的维护: 软件或硬件升级、数据库性能调整、数据库备份、统计信息收集和商业目的的监视。您必须尽量降低这些维护活动对数据库解决方案可用性的影响。

开始之前

在安排维护活动之前，必须先确定您必须对数据库解决方案执行的维护活动。

过程

要安排维护，请执行下列步骤:

1. 确定数据库活动较少的期间。

   最好是将维护活动安排在使用率低的时间进行（对数据库系统发出请求的用户应用程序最少的时间段。）根据您创建的业务应用程序类型，甚至可能有没有用户应用程序访问数据库的时间段。
2. 按照以下准则对必须执行的维护活动分类:
   - 维护可以自动进行
   - 执行维护时必须将数据库解决方案脱机
   - 可以在数据库解决方案联机时执行维护
3. 对于那些可以自动进行的维护活动，使用下列方法中的一种来配置自动维护:
   - 使用 **auto_maint** 配置参数
   - 使用名为 AUTOMAINT_SET_POLICY 和 AUTOMAINT_SET_POLICYFILE 的系统存储过程之一
4. 如果必须执行的任何维护活动需要数据库服务器脱机，请安排在使用率较低的时间进行这些脱机维护活动。
5. 对于那些可以在数据库服务器联机时执行的维护活动:
   - 确定运行这些联机维护活动对可用性的影响。
   - 按照尽可能降低运行这些维护活动对数据库系统可用性的影响的原则安排这些联机维护活动。

   例如: 安排在使用率低的时间进行联机维护活动; 以及使用调速机制来均衡维护活动使用的系统资源量。

使用 SYSPROC.AUTOMAINT_SET_POLICY 或 SYSPROC.AUTOMAINT_SET_POLICYFILE 来配置自动维护策略

可以使用系统存储过程 AUTOMAINT_SET_POLICY 和 AUTOMAINT_SET_POLICYFILE 来为数据库配置自动维护策略。

过程

要配置数据库的自动维护策略，请执行下列步骤:

1. 连接至数据库
2. 调用 AUTOMAINT_SET_POLICY 或 AUTOMAINT_SET_POLICYFILE
   - AUTOMAINT_SET_POLICY 所需的参数有:
     a. 维护类型，指定要配置的自动维护活动的类型。
     b. 指向以 XML 格式指定自动维护策略的 BLOB 的指针。
   - AUTOMAINT_SET_POLICYFILE 所需的参数有:
     a. 维护类型，指定要配置的自动维护活动的类型。

b. 指定自动维护策略的 XML 文件的名称。

有效维护类型值有:

- AUTO_BACKUP - 自动备份
- AUTO_REORG - 表和索引自动重组
- AUTO_RUNSTATS - 自动表 **RUNSTATS** 操作
- MAINTENANCE_WINDOW - 维护窗口

下一步做什么

可以使用系统存储过程 AUTOMAINT_GET_POLICY 和 AUTOMAINT_GET_POLICYFILE 来检索为数据库配置的自动维护策略。

AUTOMAINT_SET_POLICY 或 AUTOMAINT_SET_POLICYFILE 的自动维护策略规范 XML 样本

无论是使用 AUTOMAINT_SET_POLICY 还是使用 AUTOMAINT_SET_POLICYFILE 来指定自动维护策略，都必须使用 XML 来指定策略。有一些样本文件说明了如何以 XML 形式指定自动维护策略。在 Linux 和 UNIX 操作系统中，可在 SQLLIB/samples/automaintcfg 目录中找到这些样本文件。在 Windows 操作系统中，可在 SQLLIB\samples\automaintcfg 目录中找到这些样本文件。

传递到系统存储过程 AUTOMAINT_SET_POLICY 的第二个参数是包含 XML 的 BLOB，它指定所需的自动维护策略。传递到系统存储过程 AUTOMAINT_SET_POLICYFILE 的第二个参数是指定所需自动维护策略的 XML 文件的名称。传递到 AUTOMAINT_SET_POLICY 的 BLOB 中的有效 XML 元素与传递到 AUTOMAINT_SET_POLICYFILE 的 XML 文件中的有效 XML 元素相同。

在 samples 目录（在 Linux 和 UNIX 环境中为 SQLLIB/samples/automaintcfg，在 Windows 环境中为 SQLLIB\samples\automaintcfg）中，有四个 XML 文件包含自动维护策略规范示例:

DB2MaintenanceWindowPolicySample.xml

演示如何指定数据库管理器应在其中安排自动维护的维护窗口。

DB2AutoBackupPolicySample.xml

演示指定数据库管理器应如何执行自动备份。

DB2AutoReorgPolicySample.xml

演示指定数据库管理器应如何执行表和索引自动重组。

DB2DefaultAutoRunstatsPolicySample.xml

演示指定数据库管理器应如何执行自动表 **runstats** 操作。

可以通过从这些文件复制 XML 并根据系统要求修改该 XML 来创建自己的自动维护策略规范 XML。

配置数据库日志记录选项

使用数据库日志记录配置参数来指定数据库的数据日志记录选项，例如要使用的日志记录的类型、日志文件的大小以及日志文件的存储位置。

开始之前

要配置数据库日志记录选项，您必须拥有 SYSADM、SYSCTRL 或 SYSMAINT 权限。

关于此任务

可以通过在命令行处理器 (CLP) 中使用 **UPDATE DATABASE CONFIGURATION** 命令或通过调用 db2CfgSet API 来配置数据库日志记录选项。

过程

- 要通过在命令行处理器中使用 **UPDATE DATABASE CONFIGURATION** 命令来配置数据库日志记录选项:
  1. 指定是使用循环日志记录功能还是归档日志记录功能。如果要使用循环日志记录功能，那么必须将 **logarchmeth1** 和 **logarchmeth2** 数据库配置参数设置为 OFF。此设置是缺省值。要使用归档日志记录功能，必须至少将这些数据库配置参数中的一个参数设置为非 OFF 值。例如，如果要使用归档日志记录功能，并且要将归档的日志保存到磁盘，请发出以下命令:

     ```
     db2 update db configuration for mydb using logarchmeth1
        disk:/u/dbuser/archived_logs
     ```

     归档的日志将放置在名为 /u/dbuser/archived_logs 的目录中。
  2. 根据需要，对其他数据库日志记录配置参数指定值。下列其他配置参数适用于数据库日志记录:
     - **archretrydelay**
     - **blk_log_dsk_ful**
     - **failarchpath**
     - **logarchcompr1**
     - **logarchcompr2**
     - **logarchmeth1**
     - **logarchmeth2**
     - **logarchopt1**
     - **logarchopt2**
     - **logbufsz**
     - **logfilsiz**
     - **logprimary**
     - **logsecond**
     - **max_log**
     - **mirrorlogpath**
     - **newlogpath**
     - **mincommit**
     - **numarchretry**
     - **num_log_span**
     - **overflowlogpath**

有关这些数据库日志记录配置参数的更多信息，请参阅『用于数据库日志记录的配置参数』。

- 要使用 IBM Data Studio 配置数据库日志记录选项，请对 **UPDATE DATABASE CONFIGURATION** 命令使用任务助手。

用于数据库日志记录的配置参数

任何高可用性策略的一个关键元素是数据库日志记录。可以使用数据库日志来记录事务信息、使主数据库与辅助（备用数据库）同步并前滚已接管出现故障的主数据库的辅助数据库。

要配置这些数据库日志记录活动，必须设置各种数据库配置参数。

归档重试延迟 (archretrydelay)

指定在上一次尝试失败之后，归档日志文件尝试之间等待的时间量（以秒计）。缺省值为 20。

日志磁盘已满时挂起 (blk_log_dsk_ful)

可以设置此配置参数以防止当 DB2 数据库管理器不能在活动日志路径中创建新日志文件时发生“磁盘已满”错误。DB2 数据库管理器将改为每隔五分钟就尝试创建一次日志文件，直至成功。每次尝试之后，DB2 数据库管理器都会将一条消息写至管理通知日志。确认应用程序因为日志磁盘已满情况而挂起的唯一的方法就是监视管理通知日志。在成功创建日志文件之前，尝试更新表数据的所有用户应用程序都不能落实事务。只读查询可能不会直接受影响；但是，如果查询需要访问被更新请求锁定的数据或者由更新应用程序在缓冲池中修正的数据页时，只读查询也将被阻塞。

如果将 **blk_log_dsk_ful** 设置为 YES，那么会导致应用程序在 DB2 数据库管理器遇到日志磁盘已满错误时挂起。于是您就能够解决错误，而应用程序可以继续运行。磁盘已满情况可以通过将旧的日志文件移至另一文件系统、增加文件系统的大小以使挂起应用程序能够完成或调查并解决任何日志归档失败来解决。

如果将 **blk_log_dsk_ful** 设置为 NO，那么接收到日志磁盘已满错误的事务将失败并被回滚。

故障转移归档路径 (failarchpath)

如果常规归档路径存在问题（例如，如果该路径无法访问或已满），那么会为归档日志文件指定备用目录。在失败的日志归档方法再次可用之前，此目录是日志文件的临时存储区，日志归档方法再次可用后日志文件将从此目录中移至原始日志归档中所指定的路径。通过将日志文件移动至此临时位置，有助于避免日志目录发生已满情况。此参数必须是一个标准现有目录。

主日志归档压缩 (logarchcompr1) 和辅助日志归档压缩 (logarchcompr2)

在某些情况下，这些参数控制数据库管理器是否压缩归档日志文件。如果对日志归档文件进行压缩，那么可以减少与存储这些文件相关联的开销。

这些参数的有效值如下所示:

OFF 此值指定不压缩日志归档文件。缺省值为 OFF。

ON 此值指定压缩日志归档文件。如果以动态方式设置，那么不会对已归档的日志文件进行压缩。

注:

1. 如果将 **logarchmeth1** 配置参数设置为 DISK、TSM 或 VENDOR 以外的值，那么无论 **logarchcompr1** 配置参数设置为何，对日志归档压缩都没有影响。
2. 如果将 **logarchmeth2** 配置参数设置为 DISK、TSM 或 VENDOR 以外的值，那么无论 **logarchcompr2** 配置参数设置为何，对日志归档压缩都没有影响。

日志归档方法 1 (logarchmeth1)、日志归档方法 2 (logarchmeth2)

这些参数使数据库管理器将日志文件归档至活动日志路径之外的位置。如果同时指定这两个参数，那么由 **logpath** 配置参数设置的活动日志路径中的每个日志文件均会进行两次归档。这意味着将在两个不同目标位置具有该日志路径中的已归档日志文件的两个相同副本。如果通过使用 **mirrorlogpath** 配置参数指定了镜像日志记录，那么 **logarchmeth2** 配置参数将归档镜像日志路径中的日志文件，而不是归档活动日志路径中日志文件的其他副本。这意味着将在两个不同目标位置归档日志文件的两个不同副本: 一个副本来自活动日志路径，另一个副本来自镜像日志路径。

这些参数的有效值如下所示:

OFF 此值指定不使用日志归档方法。如果将 **logarchmeth1** 和 **logarchmeth2** 配置参数都设置为 OFF，那么认为数据库正在使用循环日志记录，且不可前滚恢复。缺省值为 OFF。

LOGRETAIN

指定活动日志文件文件保留并成为联机归档日志文件以用于前滚恢复。

USEREXIT

指定执行日志保留日志记录并应使用用户出口程序来归档和检索这些日志文件。日志文件是在变满时归档的。前滚实用程序必须使用日志文件来复原数据库时会检索这些日志文件。

DISK 此值后必须紧跟冒号 (:)，然后是现有标准路径名，日志文件将在其中归档。例如，如果将 **logarchmeth1** 配置参数设置为 DISK:/u/dbuser/archived_logs，那么归档日志文件将放入 /u/dbuser/archived_logs/*INSTANCE_NAME*/*DBNAME*/NODE*xxxx*/LOGSTREAM*xxxx*/C*xxxxxxx* 目录。

注: 如果正在归档至磁带，那么可以使用 **db2tapemgr** 实用程序来存储和检索日志文件。

TSM 如果指定不带有任何附加配置参数，那么此值指示应该使用缺省管理类，将日志文件归档在本地 Tivoli Storage Manager (TSM) 服务器上。如果此值后紧跟冒号（:）和 TSM 管理类，那么使用指定的管理类来归档日志文件。

VENDOR

指定将使用供应商库来归档日志文件。此值后必须紧跟冒号（:）和库的名称。库中提供的 API 必须使用备份并复原供应商产品的 API。

注:

1. 如果将 **logarchmeth1** 或 **logarchmeth2** 设置为 OFF 以外的值，那么必须配置数据库以进行前滚恢复。

日志归档选项 1 (logarchopt1) 和日志归档选项 2 (logarchopt2)

指定传递至 TSM API 或供应商 API 的字符串。

对于 TSM 环境，使用此参数来允许数据库检索在不同 TSM 节点或通过不同 TSM 用户或使用代理节点在 TSM 环境中（例如在 DB2 pureScale环境中）生成的日志。必须以下例其中一种格式来提供字符串:

- 要在 TSM 服务器未配置为支持代理节点客户机时，检索不同 TSM 节点上生成的日志:

  "-fromnode=*nodename*"

- 当未将 TSM 服务器配置成支持代理节点客户机时，对于检索通过不同的 TSM 用户生成的日志:

  "-fromowner=*ownername*"

- 要在 TSM 服务器未配置为支持代理节点客户机时，检索在不同 TSM 节点上生成的日志以及由不同 TSM 用户生成的日志:

  "-fromnode=*nodename* -fromowner=*ownername*"

- 对于检索在客户机代理节点配置中（例如在 DB2 pureScale环境中，其中有多个成员处理相同数据）生成的日志:

  "-asnodename=*proxynode*"

  其中 *nodename* 是最初归档日志文件的 TSM 节点的名称，*ownername* 是最初归档日志文件的 TSM 用户的名称，*proxynode* 是共享的 TSM 目标代理节点的名称。每个日志归档选项字段对应于一个日志归档方法: **logarchopt1** 与 **logarchmeth1** 配合使用，**logarchopt2** 与 **logarchmeth2** 配合使用。

注:

- 使用 -asnodename TSM 选项时，不使用每个成员的节点名 (nodename) 来存储数据。将改为使用 DB2 pureScale实例内所有成员使用的共享 TSM 目标节点的名称来存储数据。
- -fromnode 选项与 -fromowner 选项和 -asnodename 选项不兼容，因此不能同时使用。请将 -asnodename 选项用于使用代理节点的 TSM 配置，而将另外两个选项用于其他类型的 TSM 配置。有关更多信息，请参阅第 373 页的『配置 Tivoli Storage Manager 客户机』。

日志缓冲区 (logbufsz)

此参数允许您指定在将日志记录写至磁盘之前用作这些记录的缓冲区的内存量。当发生下列任何一项事件时会将日志记录写入磁盘:

- 事务落实
- 日志缓冲区已满
- 发生了某些其他的内部数据库管理器事件。

增加日志缓冲区的大小可使与日志记录关联的输入/输出 (I/O) 活动更有效，因为将日志记录写到磁盘中的频率更低，而每次写入的记录却更多。但是，如果日志缓冲区大小值较大，执行恢复所需的时间就会较长。此外，您也能够使用较高的 **logbufsz** 设置来减少从日志磁盘的读取次数。（要确定系统是否从中受益，请使用 **log_reads** 监视元素来检查读取日志磁盘的次数是否很多。

日志文件大小 (logfilsiz)

此参数以 4 KB 的页数指定每个配置日志的大小。

对可配置的每个日志流的总活动日志空间有 1024 GB 的逻辑限制。此限制源自每个日志文件的上限（即 4 KB）以及最大主日志文件数与最大辅助日志文件数之和（即 256）。

日志文件的大小对性能有直接的影响。从一个日志切换至另一个日志需要付出性能代价。因此，从纯性能角度来说，日志文件大小越大越好。此参数还指示要归档的日志文件大小。这种情况下，日志文件大小越大并不一定越好，因为较大的日志文件大小增加了故障或导致日志装入方案中的延迟的发生机率。当考虑活动日志空间时，最好有较多的较小日志文件。例如，如果有 2 个很大的日志文件，并且事务启动接近一个日志文件的末尾，那么仅有一半日志空间仍然可用。

每当数据库取消激活（与该数据库的所有连接都终止）时，就截断当前正写入的日志文件。因此，如果某个数据库被频繁取消激活，那么最好不要选择较大的日志文件大小，这是因为 DB2 数据库管理器将仅创建将会截断的较大文件。可使用 **ACTIVATE DATABASE** 命令来避免此成本，因为它会阻止最后一个客户机与数据库断开连接时数据库自动取消激活。

假定应用程序将数据库保持为打开以使打开数据库时的处理时间最短，日志文件大小应由建立脱机归档日志副本所花的时间确定。

将日志文件的丢失降低至最小程度，也是设置日志大小时的一个重要注意事项。归档一次针对一整个日志文件进行操作。如果配置较大的日志文件，那么会增加归档之间的时间。如果介质包含日志故障，某些事务信息将可能丢失。减小日志文件大小会增加归档的频率，但可以减少由于介质故障而丢失的信息量，因为平均而言，在任何给定的时间点处，尚未归档的日志数据较少。

每个事务的最大日志 (max_log)

此参数指示一个事务可以消耗的主日志空间的百分比。该值是为 **logprimary** 配置参数指定的值的百分比。

如果该值设置为 0，那么对一个事务可以消耗的总的主日志空间的百分比没有限制。如果应用程序违反了 **max_log** 配置，那么将强制该应用程序与数据库断开连接并且事务将回滚。

可以通过将 **DB2_FORCE_APP_ON_MAX_LOG** 注册表变量设置为 FALSE 来重设此行为。这将导致违反了 **max_log** 配置的事务失败。该应用程序仍然可以落实在工作单元中由先前语句完成的工作，它也可以回滚已完成的工作以撤销该工作单元。

当启用了无限活动日志空间时，此参数和 **num_log_span** 配置参数非常有用。如果打开了无限记录（即，**logsecond** 为 -1），那么事务数不受日志文件数的上限 (**logprimary** + **logsecond**) 限制。当达到 **logprimary** 的值时，DB2 数据库管理器将开始归档活动日志，而不是使事务失败。这样可能会导致问题，例如，有一个长期运行的事务，但一直未落实它（可能是由于具有逻辑错误的应用程序导致的）。如果出现这种情况，那么活动日志空间会不断增长，从而可能使得崩溃恢复性能很差。为了防止这样，可以为 **max_log** 和/或 **num_log_span** 配置参数指定值。

注：**max_log** 配置参数施加的限制不适用于以下 DB2 命令：**ARCHIVE LOG**、**BACKUP DATABASE**、**LOAD**、**REORG**、**RESTORE DATABASE** 和 **ROLLFORWARD DATABASE**。

镜像日志路径 (mirrorlogpath)

要防止主日志路径上的日志发生磁盘故障或被无意中删除的情况，可以指定在辅助（镜像）路径上维护完全相同的一组日志。要执行此操作，将此配置参数

的值更改为指向另一目录。如果数据库被配置为进行前滚恢复，那么不要将当前存储在镜像日志路径目录中的归档日志移至新位置。

mirrorlogpath 参数也会影响日志归档行为，您可以使用此参数进一步提高前滚恢复期间的弹性：设置了 **mirrorlogpath** 和 **logarchmeth2** 时，**logarchmeth2** 将归档镜像日志路径中的日志文件，而不是归档活动日志路径中日志文件的其他副本。可以使用此日志归档行为来提高弹性，这是因为镜像日志路径中可用的已归档日志文件可能仍然可用以继续执行数据库恢复操作，即使在归档之前主日志文件由于磁盘故障而毁坏。

因为可以更改日志路径位置，因此前滚恢复所需的日志可以存在于不同的目录中。在前滚操作期间可更改此配置参数的值，以允许您从其他镜像日志路径访问日志文件。

必须跟踪这些日志的位置。

直到数据库处于一致状态时才会应用所作的更改。配置参数 **database_consistent** 返回数据库的状态。

要关闭此配置参数，将它的值设置为 DEFAULT。

注:

1. 如果主日志路径是原始设备，那么此配置参数不受支持。
2. 对此参数指定的值不能是原始设备。
3. 在 DB2 pureScale环境中，连接至数据库或激活数据库的第一个成员会处理对此日志路径参数的配置更改。DB2 数据库管理器会验证路径是否存在，以及它对该路径是否具有读和写访问权。它还会为日志文件创建特定于成员的子目录。如果其中任何一个操作失败，那么 DB2 数据库管理器会拒绝指定的路径，并使用旧路径让数据库联机。如果接受了指定的路径，那么会将新值传播给每个成员。如果某个成员在尝试切换至新路径时失败，那么后续尝试激活它或连接至它都将失败 (SQL5099N)。所有成员都必须使用相同的日志路径。

新日志路径 (newlogpath)

数据库日志最初创建于下列目录：*db_path*/*instance_name*/*dbname*/NODE0000/LOGSTREAM0000。通过将此配置参数的值更改为指向另一目录或另一设备，可以更改放置活动日志文件（将来会放置以后的归档日志）的位置。如果数据库被配置为进行前滚恢复，那么不要将当前存储在数据库日志路径目录中的归档日志移至新位置。

因为可以更改该日志路径的位置，因此前滚恢复所需的日志可以存在于不同的目录中或不同的设备上。在前滚操作期间可更改此配置参数的值，以允许您访问位于多个位置的日志。

必须跟踪这些日志的位置。

直到数据库处于一致状态时才会应用所作的更改。配置参数 **database_consistent** 返回数据库的状态。

注: 在 DB2 pureScale环境中，连接至数据库或激活数据库的第一个成员会处理对此日志路径参数的配置更改。DB2 数据库管理器会验证路径是否存在，以及它对该路径是否具有读和写访问权。它还会为日志文件创建特定于成员的子目录。如果其中任何一个操作失败，那么 DB2 数据库管理器会拒绝指定的路径，并使用旧路径让数据库联机。如果接受了指定的路径，那么会将新值传播给每

个成员。如果某个成员在尝试切换至新路径时失败，那么后续尝试激活它或连接至它都将失败 (SQL5099N)。所有成员都必须使用相同的日志路径。

对组的落实次数 (mincommit)

此参数允许您延迟将日志记录写入磁盘，直到执行了最小数目的落实为止。此延迟可有助于减少与写入日志记录关联的数据库管理器开销，这样如果您有多个应用程序对数据库运行，且在很短的时间内该应用程序请求了许多落实，那么可改进性能。

仅当此参数的值大于 1，且多个应用程序大约同时尝试落实其事务时，才会对落实进行这种分组。落实组合生效时，保持应用程序落实请求，直到经过 1 秒钟或落实请求数等于此参数的值为止。

出错时的归档重试次数 (numarchretry)

指定在日志文件归档到 **failarchpath** 配置参数指定的路径之前，使用配置的日志归档方法归档日志文件的尝试次数。如果设置了 **failarchpath** 配置参数，那么只能使用该参数。缺省值为 5。

事务可以跨越的活动日志数 (num_log_span)

此参数指示一个活动事务可以跨越的活动日志文件数。如果该值设置为 0，那么对单个事务可以跨越的日志文件数没有限制。

如果应用程序违反了 **num_log_span** 设置，那么将强制该应用程序与数据库断开连接。

当启用了无限活动日志空间时，此参数和 **max_log** 配置参数非常有用。如果打开了无限记录（即，**logsecond** 为 -1），那么事务数不受日志文件数的上限 (**logprimary** + **logsecond**) 限制。当达到 **logprimary** 的值时，DB2 数据库管理器将开始归档活动日志，而不是使事务失败。这样可能会导致问题，例如，有一个长期运行的事务，但一直未落实它（可能是由于具有逻辑错误的应用程序导致的）。如果出现这种情况，那么活动日志空间会不断增长，从而可能使得崩溃恢复性能很差。为了防止这样，可以为 **max_log** 和/或 **num_log_span** 配置参数指定值。

注: 由 **num_log_span** 配置参数施加的限制不适用于下列 DB2 命令: ARCHIVE LOG、BACKUP DATABASE、LOAD、REORG、RESTORE DATABASE 和 ROLLFORWARD DATABASE。

溢出日志路径 (overflowlogpath)

此参数可以用于几种函数，这取决于日志记录要求。可以指定一个位置来让 DB2 数据库管理器查找前滚操作需要的日志文件。它与 ROLLFORWARD 命令的 OVERFLOW LOG PATH 选项相似，但是，不需要对发出的每个 ROLLFORWARD 命令指定 OVERFLOW LOG PATH 选项，可以只设置此配置参数一次。如果同时使用了这两个选项，那么 OVERFLOW LOG PATH 选项将覆盖该前滚操作的 **overflowlogpath** 配置参数。

如果 **logsecond** 设置为 -1，那么可以指定一个目录来让 DB2 数据库管理器存储从归档中检索到的活动日志文件。（如果活动日志文件不再存在于活动日志路径中，那么必须检索它们以用于回滚操作）。

如果未指定 **overflowlogpath**，那么 DB2 数据库管理器会将日志文件检索到活动日志路径中。通过指定此参数，可以提供其他存储器资源让 DB2 数据库管理器放置检索到的日志文件。好处包括将 I/O 成本分布到不同的磁盘上，以及允许将更多的日志文件存储在活动日志路径中。

例如，如果将 db2ReadLog API 用于复制，那么可以使用 **overflowlogpath** 来指定一个位置让 DB2 数据库管理器搜索此 API 所需的日志文件。如果找不到日志文件（在活动日志路径或溢出日志路径中）并且已配置数据库进行日志归档，那么 DB2 数据库管理器将检索日志文件。还可以使用此参数来指定一个目录来让 DB2 数据库管理器存储检索到的日志文件。好处包括降低活动日志路径上的 I/O 成本以及允许将更多的日志文件存储在活动日志路径中。

当配置无限日志记录（即，将 **logsecond** 设置为 -1）时，设置 **overflowlogpath** 非常有用。DB2 数据库管理器可以将从归档中检索的活动日志文件存储在此路径中。（使用无限日志记录，如果活动日志文件不再在活动日志路径中，那么可能需要从归档检索活动日志文件，以进行回滚或崩溃恢复操作。）

如果将原始设备配置为活动日志路径，那么在想要将 **logsecond** 设置为 -1 或想要使用 **db2ReadLog** API 时，必须配置 **overflowlogpath**。

要设置 **overflowlogpath**，指定一个最长 242 个字节的字符串。该字符串必须指向路径名，并且它必须为标准路径名，而不是相对路径名。该路径名必须是目录，而不是原始设备。

注: 在分区数据库环境中，数据库分区号自动追加在路径后面。这样做是为了维护多逻辑节点配置中路径的唯一性。

主日志文件 (logprimary)

此参数指定将创建的大小为 **logfilsiz** 的主日志数。

主日志文件，无论是空的还是满的，都需要相同的磁盘空间容量。因此，如果配置的日志多于需要的日志，将会不必要地占用磁盘空间。如果配置的日志太少，可能会遇到日志满载的情况。当选择要配置的日志数时，必须考虑建立的每个日志的大小，以及应用程序是否可以处理日志满载的情况。对活动日志空间的总日志文件大小限制为 256 GB。

如果对现有数据库启用前滚恢复，请将主日志数更改为主日志和辅助日志之和，再加 1。

辅助日志 (logsecond)

此参数指定创建并用于恢复（如果需要）的辅助日志文件的数目。

如果主日志文件已满，可按需要一次分配一个辅助日志文件（大小为 **logfilsiz**），最多可分配由此参数指定的最大数目。如果此参数设置为 -1，那么将数据库配置为无限活动日志空间。对在数据库上运行的未完成事务的大小或数量没有任何限制。在必须容纳大型作业的环境中（这些作业需要的日志空间比通常分配给主日志的空间多），无限活动日志记录功能非常有用。

注:

1. 要将 **logsecond** 设置为 -1，必须启用日志归档。
2. 如果此参数设置为 -1，那么崩溃恢复时间可能会延长，这是因为 DB2 数据库管理器可能需要检索已归档的日志文件。

使用 NOT LOGGED INITIALLY 参数减少日志记录

如果应用程序根据主表来创建并填充工作表，那么您可以创建工作表并对 CREATE TABLE 语句指定 NOT LOGGED INITIALLY 参数。

如果您不在意这些工作表的可恢复性（因为根据主表很容易重新创建这些工作表），那么此选项很有用。指定 NOT LOGGED INITIALLY 参数可以减少日志记录并提高性能。

使用 NOT LOGGED INITIALLY 参数的优点是，不记录在创建表的同一个工作单元中对该表所作的任何更改（包括插入、删除、更新或创建索引操作）。这不仅降低了日志记录工作量，还能提高应用程序性能。还可以对现有表使用带 NOT LOGGED INITIALLY 参数的 ALTER TABLE 语句，来获得同样的结果。

注:

1. 可以在同一个工作单元中使用 NOT LOGGED INITIALLY 参数创建多个表。
2. 仍会记录对目录表和其他用户表的更改。

因为不记录表的更改，所以当决定使用 NOT LOGGED INITIALLY 表属性时应该考虑下列事宜:

- 落实时将对表的*所有*更改清仓至磁盘。这意味着落实操作可能会耗用更长时间。
- 如果激活了 NOT LOGGED INITIALLY 属性且发生了未记录的活动，那么当语句失败或执行 ROLLBACK TO SAVEPOINT 时将回滚整个工作单元 (SQL1476N)。
- 如果正在使用高可用性灾难恢复 (HADR)，那么不应该使用 NOT LOGGED INITIALLY 表属性。在指定了 NOT LOGGED INITIALLY 选项的情况下，在主数据库上创建的表不会被复制到备用数据库上。在活动备用数据库上或因为接管操作而使备用数据库成为主数据库后，尝试访问此类表将导致错误 (SQL1477N)。
- 执行前滚操作时，不能恢复这些表。如果前滚操作遇到使用 NOT LOGGED INITIALLY 选项创建或改变的表，那么将该表被标记为不可用。在恢复了数据库之后，访问该表的任何尝试都将返回 SQL1477N。

  注: 当创建了一个表时，将会保持对目录表的行锁定，直到执行 COMMIT 为止。要利用不记录日志行为，必须在创建该表的同一个工作单元中填充该表。这就潜在地存在并行。

使用已声明的临时表减少日志记录

如果计划使用已声明的临时表作为工作表，注意下列各项:

- 已声明的临时表不是在目录中创建的。因此，不挂起锁定。
- 不对已声明的临时表执行日志记录，甚至在第一个 COMMIT 之后也如此。
- 使用 ON COMMIT PRESERVE 选项来使各行在 COMMIT 之后仍留在表中；否则，将删除所有行。
- 只有创建已声明的临时表的应用程序才能访问表的那个实例。
- 当删除应用程序与数据库的连接时，隐式删除该表。
- 在活动备用数据库上，不能创建或访问创建临时表 (CGTT) 和已声明临时表 (DGTT)。
- 工作单元中使用一个已声明临时表的操作中的错误不会导致该工作单元完全回滚。但是，更改已声明的临时表内容的语句中的操作错误将删除该表中的所有行。回滚工作单元（或保存点）将删除已声明的临时表中被该工作单元（或保存点）修改过的所有行。

日志目录已满时分块事务

如果由于没有足够的空间用于新文件而导致 DB2 数据库管理器无法在有效日志路径中创建日志文件，那么将会产生一些错误，指示磁盘已满。

如果设置了 **blk_log_dsk_ful** 数据库配置参数，那么 DB2 数据库管理器将重复尝试创建日志文件，直到成功创建了该文件为止，而不是返回“磁盘已满”错误。

如果设置了 **blk_log_dsk_ful** 数据库配置参数，那么 DB2 数据库管理器将尝试每隔 5 分钟就创建一次日志文件，直至成功。如果指定了日志归档方法，那么 DB2 数据库管理器还会检查日志文件归档操作是否已完成。如果归档日志文件已成功归档，那么 DB2 数据库管理器将不活动日志文件重命名为新的日志文件名并继续。每次尝试之后，DB2 数据库管理器都会将一条消息写至管理通知日志。可以确认应用程序因为日志磁盘已满情况而挂起的唯一方法就是监视管理通知日志。

在成功创建日志文件之前，尝试更新表数据的任何用户应用程序都不能落实事务。只读查询可能不会直接受影响；但是，如果查询需要访问被更新请求锁定的数据或者由更新应用程序在缓冲池中修正的数据页时，只读查询看起来也像已挂起一样。

通过日志归档管理日志文件

DB2 服务器日志文件归档因为各种操作系统文件处理问题和安排问题而变得很复杂。例如，如果磁盘在 DB2 数据库管理器对日志文件队列归档时失效，那么这些日志文件及以及它们包含的事务数据可能会丢失。

正确配置数据库记录可以防止这些类型的问题损害可用性和恢复策略。

以下是适用于所有日志归档方法的一般注意事项:

- **logarchcompr1** 数据库配置参数指定数据库管理器是否压缩 **logarchmeth1** 中设置的位置中包含的日志文件。如果 **logarchmeth1** 配置参数为 DISK、TSM 或 VENDOR 以外的值，那么日志归档压缩将没有影响，无论 **logarchcompr1** 配置参数设置为何。
- **logarchcompr2** 数据库配置参数指定数据库管理器是否压缩 **logarchmeth2** 中设置的位置中包含的日志文件。如果 **logarchmeth2** 配置参数为 DISK、TSM 或 VENDOR 以外的值，那么日志归档压缩将没有影响，无论 **logarchcompr2** 配置参数设置为何。
- **logarchmeth1** 数据库配置参数导致数据库管理器对日志文件归档或在数据库前滚恢复期间检索日志文件（通过使用您指定的方法）。当 ROLLFORWARD 实用程序需要一个在日志路径目录中找不到的日志文件时，会发出检索日志文件的请求。系统从 **logpath** 配置参数指定的路径对日志文件归档。

  logarchmeth2 数据库配置参数导致数据库管理器对日志文件的其他副本归档。如果配置了镜像日志记录，那么系统将从镜像日志路径中获取已归档至 **logarchmeth2** 参数指定的路径的日志文件。如果未配置镜像日志记录，那么系统将从当前日志路径获取已归档至 **logarchmeth2** 参数指定的路径的日志文件。
- 如果您要使用下列任何功能，那么不应该使用本地连接的磁带机来存储日志文件:
  - 无限日志记录
  - 在表空间级别进行联机恢复
  - 复制
  - 异步读取日志 API (db2ReadLog)
  - 高可用性灾难恢复 (HADR)

任何这些功能都会导致检索日志文件，从而与日志归档操作相冲突。此外，您不能在 DB2 pureScale 环境中使用以本地方式连接的磁带机，因为执行日志合并操作的成员必须检索其他成员的日志。

- 如果正在使用日志归档，那么当活动日志写完时，日志管理器将尝试将它们归档。在某些情况下，如果数据库在日志管理器能够成功记录归档之前被取消激活，那么日志管理器可能会在该数据库被激活时尝试再次归档日志。因此，一个日志文件会多次归档。
- 如果使用归档，那么一个日志文件满时，即使该日志文件仍是活动的且需要用于正常的处理，系统仍会将它传送至日志管理器。此过程允许数据的副本尽快从易丢失数据的介质移出。传递至日志管理器的日志文件保留在日志路径目录中，直到不再需要它用于正常处理为止。这样就复用了磁盘空间。
- 如果日志文件已归档且不包含任何已打开事务，那么 DB2 数据库管理器不会删除该文件但在需要此类文件时将其重命名为下一个日志文件。此过程可改进性能，因为创建新的日志文件（而不是重命名文件）需要写出所有页以保证有必要磁盘空间或有其他存储空间可用。
- 在崩溃恢复的成员崩溃恢复期间（在 DB2 pureScale 环境中）或运行时回滚期间，DB2 数据库管理器不会检索日志文件，除非您将 **logsecond** 数据库配置参数设置为 -1（即，如果您启用无限日志记录）。在 DB2 pureScale 环境中，在组崩溃恢复期间，数据库管理器可能必须检索已归档日志，即使您未启用无限日志记录也是如此。
- 配置日志归档不保证前滚恢复至故障点，而是仅尝试缩小故障范围。日志文件填满后，日志管理器会对这些日志异步归档。如果日志文件填满之前包含该日志的磁盘失效，那么该日志文件中的数据会丢失。而且，因为这些文件排队等待归档，所以磁盘可能会在复制所有文件之前失效，从而导致队列中的任何日志文件丢失。

  为有助于避免日志路径所在的磁盘或设备失效而导致日志文件永久丢失，您可以使用 **mirrorlogpath** 数据库配置参数来确保将这些日志写入第二个路径。如果第二个路径未与主磁盘或设备同时失效，那么可以使用这些日志文件进行恢复。

  如果设置了 **mirrorlogpath** 和 **logarchmeth2** 配置参数，那么 **logarchmeth2** 配置参数会对镜像日志路径中的日志文件归档（而不是对当前日志路径中的日志文件的其他副本归档）。在前滚恢复期间，可使用此日志归档行为来提高弹性。原因在于，从镜像日志路径归档的可用日志文件可能仍可用于继续数据库恢复操作，即使当前日志路径中的主日志文件在归档前因为磁盘失效已损坏也是如此。
- 每个日志文件的已配置大小对日志归档有直接影响。如果每个日志文件都非常大，磁盘发生故障时将会丢失大量数据。如果将数据库配置为使用较小日志文件，那么日志管理器会以更高频率对日志归档。

  但是，如果您要将数据移到慢速设备（如磁带）上，您可能希望有较大的日志文件以防构建该队列。如果归档每个文件需要大量开销（例如，回绕磁带设备或建立与归档介质的连接），也建议您使用较大日志文件。
- 如果使用日志归档，那么日志管理器会尝试在主日志填满时对其归档。在某些情况下，日志管理器将在日志变满之前对其归档。如果日志文件因为数据库取消激活、您发出 **ARCHIVE LOG** 命令、到达联机备份结尾或您发出 **SET WRITE SUSPEND** 命令而截断，那么会发生此情况。

  注： 要释放未使用的日志空间，应在对日志文件归档前将其截断。

- 如果要将日志和备份映像归档至磁带机，那么必须确保同一磁带机并非同时是备份映像和已归档日志的目标。因为某些日志归档可能在备份操作进行时发生，所以当这两个进程尝试同时写至同一磁带机时可能发生错误。

在调用用户出口程序或供应商程序来归档或检索日志文件时应注意以下注意事项:

- DB2 数据库管理器在启动用户出口程序来归档日志文件时，以读方式打开该文件。在某些操作系统上，这可防止用户出口程序能删除日志文件。其他操作系统（例如，AIX 操作系统）允许进程（包括用户出口程序）删除日志文件。用户出口程序在日志文件归档后永远不能删除它，因为该文件可能仍是活动的并且是崩溃恢复所需的。DB2 数据库管理器管理它在对日志文件归档时重复使用的磁盘空间。
- 用户出口或供应商程序可能接收到对不存在的文件进行归档的请求，因为存在许多归档请求并且第一次成功归档操作后该文件会被删除。用户出口或供应商程序还可能接收到检索不存在的文件的请求，因为该文件位于另一目录或到达日志结尾。在两种情况下，用户出口或供应商程序都应忽略此请求并传递指示成功的返回码。
- 在 Windows 操作系统上，不能使用 REXX 用户出口来归档日志。
- 用户出口或供应商程序应允许时间点恢复后存在同名的不同日志文件。应编写用户出口或供应商程序以保存所有日志文件或将这些日志文件与正确恢复路径相关联。
- 如果对使用同一磁带设备对日志文件归档的两个或更多数据库启用用户出口或供应商程序，并且正在其中一个数据库上执行前滚操作，那么所有其他数据库都不应处于活动状态。如果另一数据库尝试对日志文件归档而前滚操作正在进行，那么可能会发生下列其中一种情况:
  - 可能找不到前滚操作所需的日志。
  - 归档至磁带设备的新日志文件可能会覆盖先前存储在该磁带设备上的日志文件。

  为避免发生任一情况，可执行下列其中一个步骤:
  - 可确保在前滚操作期间数据库分区上没有其他数据库调用打开的用户出口程序。
  - 可编写用户出口程序以处理此情况。

配置集群环境以获取高可用性

创建机器集群并使用集群管理软件来均衡这些机器上的工作负载是设计高可用性解决方案的一种策略。

如果在集群中的一台或几台机器上安装 IBM DB2 服务器，那么必须配置集群管理器以对影响数据库的故障作出正确反应。另外，还必须配置数据库管理器实例，使它在集群环境中正常工作。

关于此任务

手动配置和管理数据库实例与集群管理器是一项复杂、耗时和易于出错的工作。DB2 High Availability Feature提供了基础结构，在实例配置更改（例如，停止数据库管理器实例）需要更改集群时，使数据库管理器能与集群管理器通信。

过程

1. 安装集群管理软件。

   SA MP 与 DB2 Enterprise Server Edition、DB2 Advanced Enterprise Server Edition、DB2 Workgroup Server Edition、DB2 Connect Enterprise Edition 以及

AIX、Linux 和 Solaris SPARC 操作系统上的 DB2 Connect Application Server Edition 集成。在 Linux 操作系统上，它还与 DB2 Express-C 固定期限许可证 (FTL) 和 IBM DB2 Express® Edition 集成。在 Windows 操作系统上，SA MP 与所有这些 DB2 数据库产品和功能部件捆绑在一起，但是未与 DB2 安装程序集成在一起。

2. 为集群管理器配置 DB2 数据库管理器实例，并为 DB2 服务器配置集群管理器。

   DB2 高可用性实例配置实用程序 (db2haicu) 是基于文本的实用程序，您可以使用它在集群环境中配置和管理高可用性数据库。

3. 随着时间的推移，当您的数据库需要更改，并且您需要在集群环境中修改数据库配置时，请继续保持数据库管理器实例配置与集群管理器配置同步。

   DB2 High Availability Feature提供了基础结构，在实例配置更改（例如，停止数据库管理器实例）需要更改集群时，使数据库管理器能与集群管理器通信。

   不论您是将 **db2haicu** 与 SA MP 配合使用，还是使用另一个支持 DB2 集群管理器 API 的集群管理器，使用 DB2 HA 功能部件管理集群环境都会比分别维护数据库管理器配置和集群配置更为容易。

集群管理器与 DB2 High Availability Feature的集成

DB2 High Availability Feature允许 IBM DB2 服务器与集群管理软件之间进行集成。

当在集群环境中停止数据库管理器实例时，必须使集群管理器知道实例已停止。如果集群管理器不知道实例已停止，那么集群管理器可能尝试操作，例如对已停止的实例进行故障转移。DB2 High Availability Feature提供了基础结构，在实例配置更改（例如，停止数据库管理器实例）需要更改集群时，使数据库管理器能与集群管理器通信。

DB2 High Availability Feature由以下元素组成:

- IBM Tivoli System Automation for Multiplatforms (SA MP)（作为 DB2 High Availability Feature的一部分）与 AIX 和 Linux 上的 DB2 服务器捆绑到一起，并与 DB2 安装程序集成。可使用 DB2 安装程序或 DB2 服务器安装介质中包括的 **installSAM** 和 **uninstallSAM** 脚本来安装、升级或卸载 SA MP。
- 在集群环境中，某些数据库管理器实例配置和管理操作需要相关集群配置更改。每当您执行某些数据库管理器实例配置和管理操作时，DB2 High Availability Feature (HA) 功能部件使数据库管理器能够自动请求集群管理器配置更改。 请参阅: 第 73 页的『使用 DB2 高可用性 (HA) 功能部件来自动配置集群』
- DB2 高可用性实例配置实用程序 (db2haicu) 是基于文本的实用程序，您可以使用它在集群环境中配置和管理高可用性数据库。 请参阅: 第 81 页的『DB2 高可用性实例配置实用程序 (db2haicu)』

IBM Tivoli System Automation for Multiplatforms (SA MP) 基本组件

IBM Tivoli System Automation for Multiplatforms (SA MP) 为 AIX, Linux, Solaris SPARC 和 Windows 提供高可用性和灾难恢复功能。

在 AIX、Linux 和 Solaris SPARC 操作系统上， SA MP 与 DB2 Enterprise Server Edition、DB2 Advanced Enterprise Server Edition、DB2 Workgroup Server Edition、DB2

Connect Enterprise Edition 和 DB2 Connect Application Server Edition 集成在一起。它还与 Express 版集成在一起，以与 DB2 Express-C 固定期限许可证 (FTL) 和 DB2 High Availability Feature 配合使用。

在 Windows 操作系统上，SA MP 与所有这些 DB2 数据库产品和功能部件捆绑在一起，但是未与 DB2 数据库产品安装程序集成在一起。

可以使用 SA MP 的此副本来管理 DB2 数据库系统的高可用性。如果不购买 SA MP 许可证的升级版本，您不能使用 DB2 数据库系统以外的数据库系统。

在 AIX，Linux 和 Solaris SPARC 操作系统上，SA MP 是 IBM DB2 服务器集群环境中的缺省集群管理程序。

有关 SA MP的更多信息，请参阅 IBM Tivoli System Automation for Multiplatforms (SA MP)publib.boulder.ibm.com/tividd/td/IBMTivoliSystemAutomationforMultiplatforms3.1.html。在以下 Web 站点上也提供了受支持的操作系统的列表：http://www.ibm.com/software/tivoli/products/sys-auto-linux/platforms.html。

使用 DB2 高可用性 (HA) 功能部件来自动配置集群

在集群环境中，某些数据库管理器实例配置和管理操作需要相关集群配置更改。每当您执行某些数据库管理器实例配置和管理操作时，DB2 High Availability Feature (HA) 功能部件使数据库管理器能够自动请求集群管理器配置更改。

开始之前

要使数据库管理器能够执行数据库管理任务所需的集群配置，您必须为高可用性配置实例，方法是使用 **db2haicu** 来为该实例创建*集群域*。有关更多信息，请参阅：第 74 页的『使用 DB2 高可用性实例配置实用程序 (db2haicu) 来配置集群环境』。

过程

当执行下列数据库管理器实例配置和管理操作时，数据库管理器将为您自动执行相关的集群管理器配置:

- 使用 **START DATABASE** 或 **db2start** 来启动数据库。
- 使用 **STOP DATABASE** 或 **db2stop** 来停止数据库。
- 使用 **CREATE DATABASE** 来创建数据库。
- 使用 **CREATE TABLESPACE** 来添加存储器。
- 使用 **ALTER TABLESPACE DROP** 或 **DROP TABLESPACE** 来除去存储器。
- 使用 ALTER DATABASE 来添加或除去存储路径。
- 使用 **DROP TABLESPACE** 来删除数据库。
- 使用 **RESTORE DATABASE** 或 db2Restore 来复原数据库。
- 使用 **SET TABLESPACE CONTAINERS** 为重定向复原指定表空间容器。
- 使用 **ROLLFORWARD DATABASE** 或 db2Rollforward 前滚数据库。
- 使用 **RECOVER DATABASE** 或 db2Recover 来恢复数据库。
- 使用 **CREATE EVENT MONITOR** 来创建事件监视器。
- 使用 **DROP EVENT MONITOR** 来删除事件监视器。

- 使用以下各项来创建和改变外部例程:
  - CREATE PROCEDURE
  - CREATE FUNCTION
  - CREATE FUNCTION
  - CREATE METHOD
  - ALTER PROCEDURE
  - ALTER FUNCTION
  - ALTER METHOD
- 使用以下各项来删除外部例程:
  - DROP PROCEDURE
  - DROP FUNCTION
  - DROP METHOD
- 使用 **START HADR** 来为数据库启动 DB2 高可用性灾难恢复 (HADR) 操作。
- 使用 **STOP HADR** 来为数据库停止 HADR 操作。
- 使用 **TAKEOVER HADR** 来促使 HADR 备用数据库作为 HADR 主数据库接管工作。
- 设置数据库管理器配置参数 **diagpath** 或 **spm_log_path**。
- 设置数据库配置参数 **newlogpath**、**overflowlogpath**、**mirrorlogpath** 或 **failarchpath**。
- 使用 **db2idrop** 来删除数据库管理器实例。

结果

当数据库管理器为所列示的数据库管理任务协调集群配置更改时，您不必执行单独的集群管理器操作。

使用 DB2 高可用性实例配置实用程序 (db2haicu) 来配置集群环境

您可以在集群环境中使用 DB2 高可用性实例配置实用程序 (db2haicu)来配置和管理数据库。当为 **db2haicu** 指定数据库管理器实例配置详细信息时，**db2haicu** 向集群管理软件通知所需的集群配置详细信息。

开始之前

- 在使用 DB2 高可用性实例配置实用程序 (db2haicu) 之前，必须执行一组任务。有关更多信息，请参阅: 第 112 页的『DB2 高可用性实例配置实用程序 (db2haicu) 先决条件』。

关于此任务

可以采用交互方式运行 **db2haicu** 或使用 XML 输入文件:

交互方式

当通过运行 **db2haicu** 命令而未使用 **-f** 参数指定 XML 输入文件来调用 DB2 高可用性实例配置实用程序 (db2haicu) 时，实用程序将以交互方式运行。在交互方式下，**db2haicu** 显示信息并询问您以获得基于文本格式的信息。 有关更多信息，请参阅: 第 83 页的『以交互方式运行 DB2 高可用性实例配置实用程序 (db2haicu)』

批处理方式和 XML 输入文件

您可以将 **-f** *input-file-name* 参数与 **db2haicu** 命令配合使用，以运行 DB2 高可用性实例配置实用程序 (db2haicu)，同时 XML 输入文件指定配置详细信息。当必须多次执行配置任务，例如具有多个要为高可用性而配置的数据库分区时，运行 **db2haicu** 和 XML 输入文件是非常有用的。 有关更多信息，请参阅：第 83 页的『使用 XML 输入文件来运行 DB2 高可用性实例配置实用程序 (db2haicu)』

有关将 **db2haicu** 与两种方法配合使用来设置 HADR 对的详细场景，请参阅“使用 IBM DB2 高可用性实例配置实用程序 (db2haicu) 来设置自动集群受控 HADR（高可用性灾难恢复）配置”。

限制

使用 DB2 高可用性实例配置实用程序 (db2haicu) 时存在一些限制。有关更多信息，请参阅：第 115 页的『DB2 高可用性实例配置实用程序 (db2haicu) 限制』。

过程

对每个数据库管理器实例执行下列步骤:

1. 创建新的集群域。

   当首次为数据库管理器实例运行 DB2 高可用性实例配置实用程序 (db2haicu) 时，**db2haicu** 会创建被称为*集群域*的集群模型。有关更多信息，请参阅：第 113 页的『使用 DB2 高可用性实例配置实用程序 (db2haicu) 来创建集群域』。
2. 继续优化集群域配置以及管理和维护集群域

   当您使用 **db2haicu** 来修改集群环境的集群域模型时，数据库管理器将相关更改传播到数据库管理器实例和集群配置。有关更多信息，请参阅：第 114 页的『使用 DB2 高可用性实例配置实用程序 (db2haicu) 来维护集群域』。

下一步做什么

DB2 高可用性实例配置实用程序 (db2haicu) 没有单独的诊断日志。可以使用数据库管理器诊断日志、**db2diag** 日志文件和 **db2pd** 工具来调查与诊断 **db2haicu** 错误。 有关更多信息，请参阅：第 115 页的『对 DB2 高可用性实例配置实用程序 (db2haicu) 进行故障诊断』

集群域

集群域是包含有关集群元素（如数据库、安装点和故障转移策略）的信息的模型。可以使用 DB2 高可用性实例配置实用程序 (db2haicu) 来创建集群域。

db2haicu 使用集群域中的信息来启用配置和维护集群管理任务。此外，作为 DB2 高可用性 (HA) 功能部件的一部分，数据库管理器还使用集群域中的信息来执行自动化集群管理任务。

如果将集群元素添加至集群域，那么任何后续 **db2haicu** 配置操作或由 DB2 HA 功能部件中数据库管理器执行的任何自动化集群管理操作都会包含该元素。如果从集群域中除去集群元素，那么 **db2haicu** 操作或数据库管理器自动化集群管理操作将不再包含该元素。**db2haicu** 和数据库管理器只能与使用 **db2haicu** 创建的集群域中集群元素的集群管理器协同操作。

可以使用 **db2haicu** 来创建和配置下列集群域元素:

- 计算机或机器（在集群域上下文中，它们称为*集群域节点*）
- 网络接口卡或 NIC（在 **db2haicu** 中称为*网络接口*、*接口*、*网络适配器*或*适配器*）
- IP 地址
- 数据库，包括高可用性灾难恢复 (HADR) 主/备用数据库对
- 数据库分区
- 安装点和路径，包括在出现故障时对故障转移不重要的那些路径
- 故障转移策略
- 定额设备

集群管理软件:

集群管理软件使一组计算机能够执行的工作量达到最大。集群管理器平衡工作负载以减少瓶颈、监视集群元素的运行状况并在元素出现故障时管理故障转移。

集群管理器还可以帮助系统管理员对集群中的元素执行管理任务（例如，将计算机中需要处理的工作负载重新路由到该计算机外）。

集群元素

要正常运行，集群管理器必须了解许多与集群元素相关的详细信息以及集群元素之间的关系。

以下是集群管理器必须了解的一些集群元素示例:

- 集群中的物理或虚拟计算机、机器或设备（在集群上下文中，它们称为*集群节点*）
- 连接集群节点的网络
- 将集群节点与网络连接在一起的网络接口卡
- 集群节点的 IP 地址
- 虚拟或服务 IP 地址

以下是集群管理器必须了解的一些关系示例:

- 安装了相同软件并且相互之间可以进行故障转移的集群节点对
- 具有相同属性并且相互之间可以进行故障转移的网络
- 当前与虚拟 IP 地址关联的集群节点

添加或修改集群元素

要让集群管理器了解集群元素和这些元素之间的关系，系统管理员必须向集群管理器注册这些元素。如果系统管理员更改集群元素，那么该管理员必须将所作的更改告知集群管理器。集群管理器具有可帮助完成这些任务的界面。

集群管理是一项富有挑战性的任务，因为可能的集群元素数量庞大且种类繁多。管理员必须精通集群节点的硬件和操作系统、网络协议和配置，以及集群节点上安装的软件（如数据库软件）。向集群管理软件注册集群元素或者在系统更改后更新集群管理器是一项复杂且耗时的工作。

使用 db2haicu 添加或修改集群元素

在 DB2 数据库解决方案中，可以使用 DB2 高可用性实例配置实用程序 (db2haicu) 向集群管理器注册集群元素，并且在对集群进行管理更改后更新集群管理器。使用 **db2haicu** 可简化这些任务，因为在您了解 **db2haicu** 用于封装集群元素和这些元素之间的关系的模型之后，您不必精通硬件、操作系统和集群管理器界面的特征即可执行这些任务。

资源和资源组:

*资源*是指任何已向集群管理器注册的集群元素，如集群节点、数据库、安装点或网络接口卡。如果某个元素未向集群管理器注册，那么集群管理器不会知道该元素并且不会将它包括在集群管理操作中。*资源组*是资源的逻辑集合。资源组是一种功能非常强大的结构，这是因为可以在资源组上定义关系和约束，而这些关系和约束能够简化对这些组中的资源执行的复杂管理任务。

当集群管理器将资源收集成组时，它可以统一对这些资源进行操作。例如，如果两个数据库 database-1 和 database-2 属于资源组 resource-group-A，那么当集群管理器对 resource-group-A 执行启动操作时，database-1 和 database-2 都会被这个集群管理器操作启动。

限制

- 资源组不能包含等*值*，等值也不能包含资源组（等*值*是指一组资源，它们彼此提供相同的功能，并且相互之间可以进行故障转移。）
- 一个资源只能位于一个资源组中
- 一个资源不能同时位于资源组和等值中
- 资源组可以包含其他资源组，但最大嵌套级别为 50
- 一个资源组中可以收集的最大资源数为 100

定额设备:

当集群管理器的一般决策过程不能作出明确选择时，*定额设备*可帮助集群管理器作出集群管理决策。

当集群管理器必须在多个潜在操作之间进行选择时，它计算每个潜在操作各有多少个集群域节点支持，然后选择大多数集群域节点支持的操作。如果正好有相同数目的集群域节点支持多个选项，那么集群管理器将引用定额设备来作出选择。

db2haicu 支持下表中列示的定额设备。

表 2. **db2haicu** *支持的定额设备类型*

定额设备	描述
网络	网络定额设备是每个集群域节点在任何时候都可连接至的 IP 地址。

集群域中的网络:

要配置与网络相关的集群域元素，可以使用 DB2 高可用性实例配置实用程序 (db2haicu) 在集群域中添加*物理网络*。物理网络由以下几部分组成：网络接口卡、IP 地址和子网掩码。

网络接口卡

网络接口卡 (NIC) 是使计算机（也称为*集群节点*）与网络连接在一起的硬件。NIC 有时称为*接口*、*网络适配器*或*适配器*。使用 **db2haicu** 在集群域中添加物理网络时，应指定至少一个 NIC，包括该 NIC 所属的计算机的主机名、该主机上 NIC 的名称以及 NIC 的 IP 地址。

IP 地址

因特网协议地址（IP 地址）是网络上的唯一地址。在 IP V4 中，IP 地址为 32 位，并且通常用点分十进制表示法表示如下：129.30.180.16。IP 地址由网络部分和主机部分组成。

db2haicu 不支持 IP V6。

子网掩码

使用*子网掩码*可以将一个网络分为多个逻辑子网。子网掩码是一种机制，用于将 IP 地址的主机部分中的某些位移至 IP 地址的网络部分。使用 **db2haicu** 在集群域中添加 IP 地址时，有时需要指定该 IP 地址的子网掩码。例如，使用 **db2haicu** 添加 NIC 时，必须指定该 NIC 的 IP 地址的子网掩码。

网络等值

*等值*是指一组资源，它们彼此提供相同的功能，并且相互之间可以进行故障转移。使用 **db2haicu** 创建网络时，该网络中的 NIC 之间可以进行故障转移。这种网络也称为*网络等值*。

网络协议

使用 **db2haicu** 在集群域中添加网络时，必须指定要使用的网络协议的类型。当前仅支持 TCP/IP 网络协议。

使用说明

虚拟 IP (VIP) 故障转移仅需要使用 **db2haicu** 配置的网络。因为 VIP 故障转移需要公共虚拟局域网，所以不能将不同子网（或不同虚拟局域网）中的网络适配器添加到相同网络。

集群域中的故障转移策略:

*故障转移策略*指定在网络接口卡或数据库服务器等集群元素出现故障时集群管理器应如何响应。通常，集群管理器会将工作负载从故障元素转移到备用元素，先前已向集群管理器将该备用元素标识为故障元素的适当替换元素。这种将工作负载从故障元素转移到另一个元素的操作称为*故障转移*。

循环故障转移策略

使用*循环故障转移策略*时，如果集群域中的一台计算机（也称为*集群域节点*或简单地称为*节点*）出现故障，那么在集群域中的任何其他节点上，数据库管理器将重新开始出现故障的集群域节点中的工作。

相互故障转移策略

要配置*相互故障转移策略*，应将集群域中的一个计算机对（也称为*集群域节点*或简单地称为*节点*）作为系统对关联。如果此计算机对中的一个节点出现故障，那么故障节点上的数据库分区将故障转移到该计算机对中的另一个节点。仅当具有多个数据库分区时，相互故障转移才可用。

N 加 M 故障转移策略

使用 *N 加 M 故障转移策略*时，如果集群域中的一台计算机（也称为*集群域节点*或简称为*节点*）出现故障，那么故障节点上的数据库分区将故障转移至集群域中的任何其他节点。如果启用了漫游 HA 故障转移，那么当该故障节点再次处于联机状态时，最近一次发生故障的节点将成为备用节点。N 加 M 故障转移策略的漫游 HA 故障转移仅在 M=1 时才受支持。仅当具有多个数据库分区时，N 加 M 故障转移才可用。

本地重新启动故障转移策略

使用*本地重新启动故障转移策略*时，如果集群域中的一台计算机（也称为*集群域节点*或简单地称为*节点*）出现故障，那么数据库管理器将在出现故障的那个节点上的适当位置（或本地）重新启动数据库。

HADR 故障转移策略

配置 *HADR 故障转移策略*时，您正在启用 DB2 高可用性灾难恢复 (HADR) 功能部件来管理故障转移。如果 HADR 主数据库出现故障，那么数据库管理器会将出现故障的数据库中的工作负载移至 HADR 备用数据库。

定制故障转移策略

配置*定制故障转移策略*时，您将在集群域中创建一列计算机（也称为*集群域节点*或简单地称为*节点*），数据库管理器可以故障转移到这些计算机上。如果集群域中的一个节点出现故障，那么数据库管理器会将故障节点中的工作负载移至您指定的列表中的一个节点。

在分区数据库环境中使用漫游高可用性 (HA) 故障转移:

在将 N 加 M 故障转移策略用于“N”个活动节点和一个备用节点时，可以启用漫游 HA 故障转移。

开始之前

集群中的每个节点必须都启用或禁用了漫游 HA 故障转移支持。

在未启用漫游 HA 故障转移的分区数据库环境中，指定的备用节点通常是唯一能够访问所有磁盘和卷组（包括这些卷组上的文件系统）的节点。在这些环境中，确保外部存储器 LUN 映射和集群中的 SAN 区域可以访问数据库实例中的所有磁盘。另外，还应该验证是否在所有集群节点上都导入了由集群控制的所有卷组。在导入这些卷组后，请禁用卷组的 auto-varyon 属性和所有活动集群节点上的文件系统的 auto-mount 属性。

如果要使用漫游 HA 故障转移，那么必须在应用新修订包后使用这些步骤再次启用漫游 HA 故障转移。

关于此任务

在将 *N 加 M* 故障转移策略用于“N”个活动节点和一个备用节点时，如果某个活动节点发生故障，那么将执行故障转移操作。然后备用节点开始托管故障节点的资源。当故障节点重新联机时，您通常必须使集群环境再次脱机，以便最初选择为备用节点的节点再次成为备用节点。可以将漫游 HA 故障转移配置为让集群中最近一次发生故障的节点成为备用节点，而不需要执行其他故障恢复操作。

过程

要启用漫游 HA 故障转移:

1. 确保未在执行任何故障转移操作。
2. 制作 `sqllib\samples\tsa` 目录中的 `db2V10_start.ksh` 脚本的备份副本。
3. 编辑 `db2V10_start.ksh` 脚本。 找出以下行:

   ```
   ROVING_STANDBY_ENABLED=false
   ```

   并作出下列更改:

   ```
   ROVING_STANDBY_ENABLED=true
   ```
4. 保存您的更改。

结果

该更改将在下次执行故障转移操作时生效。

下一步做什么

如果要禁用漫游 HA 故障转移支持，请在每个节点上执行下列步骤:

1. 确保未在执行任何故障转移操作。
2. 编辑 `db2V10_start.ksh` 脚本。找出以下行:

   ```
   ROVING_STANDBY_ENABLED=true
   ```

   并作出下列更改:

   ```
   ROVING_STANDBY_ENABLED=false
   ```
3. 保存您的更改。该更改将在下次执行故障转移操作时生效。

集群域中的安装点:

安装文件系统后，可以使用 DB2 高可用性实例配置实用程序 (db2haicu) 将该安装点添加至集群域。

安装点

在 UNIX、Linux 和 AIX 操作系统上，安装文件系统意味着使该文件系统可用于操作系统。在安装操作期间，操作系统将执行一些任务（例如，读取索引或导航数据结构），并使一个目录路径与安装的文件系统关联。可用于访问安装的文件系统的关联目录路径称为安装点。

非重要安装点或路径

集群中可能存在出现故障时不需要进行故障转移的安装点或路径。可以使用 **db2haicu** 将这些非重要安装点或路径的列表添加至集群域。集群管理器不会将该非重要列表中的安装点或路径包括在故障转移操作中。

例如，请考虑以下情况：您在集群中的计算机 node1 上的 /mnt/driveA 中安装了一个硬盘驱动器。如果您认为让 /mnt/driveA 可用是至关重要的，那么在 node1 出现故障时集群管理器会进行故障转移以保持 /mnt/driveA 可用。但是，如果您认为可以接受在 node1 出现故障时 /mnt/driveA 不可用，那么您可以通过将 /mnt/driveA 添加至非重要路径列表来向集群管理器指示 /mnt/driveA 对于故障转移不是重要的。如果 /mnt/driveA 标识为对于故障转移不是重要的，那么在 node1 出现故障时该驱动器可能不可用。

DB2 高可用性实例配置实用程序 (db2haicu)

DB2 高可用性实例配置实用程序 (db2haicu) 是基于文本的实用程序，您可以使用它在集群环境中配置和管理高可用性数据库。

db2haicu 通过查询系统来收集有关数据库实例、集群环境和集群管理器的信息。您可通过参数向 **db2haicu** 调用、输入文件提供更多信息，或在运行时通过在 **db2haicu** 提示符处提供信息来实现上述目的。

语法

```
db2haicu [ -f XML-input-file-name ]
         [ -disable ]
         [ -delete [ dbpartitionnum db-partition-list |
                     hadrdb database-name ] ]
```

参数

传递到 **db2haicu** 命令的参数区分大小写，并且它们必须采用小写形式。

-f *XML-input-file-name*

可以使用 **-f** 参数来指定 XML 输入文件 *XML-input-file-name* 中的集群域详细信息。有关更多信息，请参阅：第 83 页的『使用 XML 输入文件来运行 DB2 高可用性实例配置实用程序 (db2haicu)』。

-disable

一旦使用 **db2haicu** 来为数据库管理器实例创建集群域，该实例被视为*已配置用于获取高可用性*。当配置数据库管理器实例以获取高可用性时，那么每当您执行需要相关集群配置更改的某些数据库管理器管理操作时，数据库管理器将向集群管理器通知那些集群配置更改。当数据库管理器将这些集群管理任务与集群管理器进行协调时，您不必为那些管理任务执行单独的集群管理器操作。数据库管理器与集群管理器之间的这一集成是 DB2 High Availability Feature的一个功能。

可以使用 **-disable** 参数来使数据库管理器实例停止以被配置用于获取高可用性。如果不再配置数据库管理器实例以获取高可用性，那么在您执行需要相关集群配置更改的任何数据库管理器管理操作时，数据库管理器将不与集群管理器进行协调。

要重新配置数据库管理器实例以获取高可用性，您可以再次运行 **db2haicu**。

-delete

可以使用 **-delete** 参数来删除当前数据库管理器实例的资源组。

如果不使用 **dbpartitionnum** 参数或 **hadrdb** 参数，那么 **db2haicu** 将除去与当前数据库管理器实例相关联的所有资源组。

dbpartitionnum *db-partition-list*

可以使用 **dbpartitionnum** 参数来删除与 *db-partition-list* 中列示的数据库分区相关联的资源组。*db-partition-list* 是用逗号分隔的编号列表，它标识了数据库分区。

hadrdb *database-name*

可使用 **hadrdb** 参数来删除与名为 *database-name* 的高可用性灾难恢复 (HADR) 数据库相关联的资源组。

如果在 **db2haicu** 除去资源组之后集群域中没有剩余资源组，那么 **db2haicu** 也将除去集群域。

运行带 **-delete** 参数的 **db2haicu** 将导致当前数据库管理器实例停止配置用于获取高可用性。如果不再配置数据库管理器实例以获取高可用性，那么在您执行需要相关集群配置更改的任何数据库管理器管理操作时，数据库管理器将不与集群管理器进行协调。

要重新配置数据库管理器实例以获取高可用性，您可以再次运行 **db2haicu**。

DB2 高可用性实例配置实用程序 (db2haicu) 启动方式:

首次为给定数据库管理器实例运行 DB2 高可用性实例配置实用程序 (db2haicu) 时，**db2haicu** 在启动方式下工作。

当运行 **db2haicu** 时，**db2haicu** 检查数据库管理器实例与系统配置，并且搜索现有*集群域*。集群域是包含有关集群元素（如数据库、安装点和故障转移策略）的信息的模型。可以使用 DB2 高可用性实例配置实用程序 (db2haicu) 来创建集群域。

为给定数据库管理器实例运行 **db2haicu** 且没有为该实例创建和配置集群域时，**db2haicu** 将立即开始创建和配置新集群域。**db2haicu** 通过提示您输入诸如新集群域名称与当前机器主机名之类的信息来创建新的集群域。

如果创建集群域，但未完成配置集群域任务，那么下次运行 **db2haicu** 时，**db2haicu** 将继续执行配置集群域任务。

在为数据库管理器实例创建和配置集群域后，**db2haicu** 将在维护方式下运行。

DB2 高可用性实例配置实用程序 (db2haicu) 维护方式:

当运行 DB2 高可用性实例配置实用程序 (db2haicu) 且存在为当前数据库管理器实例创建的集群域时，**db2haicu** 在维护方式下工作。

当 **db2haicu** 在维护方式下运行时，**db2haicu** 向您展示可执行的配置和管理任务的列表。

db2haicu 维护任务包括将诸如数据库或集群节点之类的集群元素添加到集群域，并且从集群域中除去元素。**db2haicu** 维护任务还包括修改集群域元素的详细信息，例如数据库管理器实例的故障转移策略。

当在维护方式下运行 **db2haicu** 时，**db2haicu** 向您展示可以对集群域执行的操作的列表:

- 添加或除去集群节点（主机名标识的机器）
- 添加或除去网络接口（网络接口卡）
- 添加或除去数据库分区（仅限于分区数据库环境）
- 添加或除去 DB2 高可用性灾难恢复 (HADR) 数据库
- 添加或除去高可用性数据库
- 添加或除去安装点
- 添加或除去 IP 地址
- 添加或除去非重要路径
- 移动数据库分区和 HADR 数据库以进行预定维护
- 更改当前实例的故障转移策略
- 为集群域创建新的定额设备
- 破坏集群域

以交互方式运行 DB2 高可用性实例配置实用程序 (db2haicu):

当通过运行 **db2haicu** 命令而未使用 **-f** 参数指定 XML 输入文件来调用 DB2 高可用性实例配置实用程序 (db2haicu) 时，实用程序将以交互方式运行。在交互方式下，**db2haicu** 显示信息并询问您以获得基于文本格式的信息。

开始之前

- 在使用 DB2 高可用性实例配置实用程序 (db2haicu) 之前，必须执行一组任务。有关更多信息，请参阅: 第 112 页的『DB2 高可用性实例配置实用程序 (db2haicu) 先决条件』。

关于此任务

以交互方式运行 **db2haicu** 时，您将看到信息和问题以文本格式出现在屏幕上。您可以在屏幕底部的提示符处输入 **db2haicu** 请求的信息。

过程

要以交互方式运行 **db2haicu**，调用 **db2haicu** 命令（不带 **-f** *input-file-name*）。

下一步做什么

DB2 高可用性实例配置实用程序 (db2haicu) 没有单独的诊断日志。可以使用数据库管理器诊断日志、**db2diag** 日志文件和 **db2pd** 工具来调查与诊断 **db2haicu** 错误。 有关更多信息，请参阅: 第 115 页的『对 DB2 高可用性实例配置实用程序 (db2haicu) 进行故障诊断』

使用 XML 输入文件来运行 DB2 高可用性实例配置实用程序 (db2haicu):

您可以将 **-f** *input-file-name* 参数与 **db2haicu** 命令配合使用，以运行 DB2 高可用性实例配置实用程序 (db2haicu)，同时 XML 输入文件指定配置详细信息。当必须多次执行配置任务，例如具有多个要为高可用性而配置的数据库分区时，运行 **db2haicu** 和 XML 输入文件是非常有用的。

开始之前

- 在使用 DB2 高可用性实例配置实用程序 (db2haicu) 之前，必须执行一组任务。有关更多信息，请参阅: 第 112 页的『DB2 高可用性实例配置实用程序 (db2haicu) 先决条件』。

关于此任务

在 sqllib 目录的 samples 子目录中存在一组您可以修改且与 **db2haicu** 配合使用的样本 XML 输入文件，可用来配置集群环境。 有关更多信息，请参阅: 第 103 页的『DB2 高可用性实例配置实用程序 (db2haicu) 的样本 XML 输入文件』

有关将 **db2haicu** 与样本 XML 输入文件配合使用来设置 HADR 对的详细场景，请参阅“使用 IBM DB2 高可用性实例配置实用程序 (db2haicu) 来设置自动集群受控 HADR（高可用性灾难恢复）配置”。

过程

1. 创建 XML 输入文件。如果您在配置数据库分区或 HADR 设置中的主实例和备用实例，那么您将使用同一 XML 文件。
2. 调用 **db2haicu**（带 **-f** *input-file-name*）。 在 HADR 设置中，
   a. 登录备用实例并发出此命令。
   b. **db2haicu** 存在后，登录主实例并发出此命令。

下一步做什么

DB2 高可用性实例配置实用程序 (db2haicu) 没有单独的诊断日志。可以使用数据库管理器诊断日志、**db2diag** 日志文件和 **db2pd** 工具来调查与诊断 **db2haicu** 错误。 有关更多信息，请参阅: 第 115 页的『对 DB2 高可用性实例配置实用程序 (db2haicu) 进行故障诊断』

DB2 高可用性实例配置实用程序 (db2haicu) 输入文件的 XML 模式定义:

DB2 高可用性实例配置实用程序 (db2haicu)输入文件的 XML 模式定义 (XSD) 用于定义可以在 **db2haicu** XML 输入文件中指定的集群域对象。此 **db2haicu** XSD 位于 sqllib/samples/ha/xml 目录中的文件 db2ha.xsd 中。

DB2ClusterType

db2haicu XML 模式定义 (XSD) 的根元素是 DB2Cluster，其类型为 DB2ClusterType。**db2haicu** XML 输入文件必须以 DB2Cluster 元素开头。

『XML 模式定义』
第 85 页的『子元素』
第 86 页的『属性』
第 86 页的『使用说明』

XML 模式定义

```
<xs:complexType name='DB2ClusterType'>
  <xs:sequence>
    <xs:element name='DB2ClusterTemplate'
                type='DB2ClusterTemplateType'
                minOccurs='0'
                maxOccurs='unbounded'/>
```

```
    <xs:element name='ClusterDomain'
                type='ClusterDomainType'
                maxOccurs='unbounded'/>
    <xs:element name='FailoverPolicy'
                type='FailoverPolicyType'
                minOccurs='0'/>
    <xs:element name='DB2PartitionSet'
                type='DB2PartitionSetType'
                minOccurs='0'
                maxOccurs='unbounded'/>
    <xs:element name='HADRDBSet'
                type='HADRDBType'
                minOccurs='0'
                maxOccurs='unbounded'/>
    <xs:element name='HADBSet'
                type='HADBType'
                minOccurs='0'
                maxOccurs='unbounded'/>
  </xs:sequence>
  <xs:attribute name='clusterManagerName' type='xs:string' use='optional'/>
</xs:complexType>
```

子元素

DB2ClusterTemplate

类型: DB2ClusterTemplateType

使用说明:

不要在 **db2haicu** XML 输入文件中包括 DB2ClusterTemplateType 元素。当前保留 DB2ClusterTemplateType 元素以供将来使用。

ClusterDomain

类型: ClusterDomainType

ClusterDomainType 元素包含关于下列内容的规范: 集群域中的机器或计算机（也称为*集群域节点*）、*网络等值*（相互之间可以进行故障转移的多组网络），以及*定额设备*（决定性机制）。

出现规则:

必须在 DB2ClusterType 元素中包括一个或多个 ClusterDomain 元素。

FailoverPolicy

类型: FailoverPolicyType

FailoverPolicyType 元素指定集群管理器应该对集群域使用的*故障转移策略*。

出现规则:

可以在 DB2ClusterType 元素中包括零个或一个 FailoverPolicy 元素。

DB2PartitionSet

类型: DB2PartitionSetType

DB2PartitionSetType 元素包含关于数据库分区的信息。DB2PartitionSetType 元素仅适用于分区数据库环境。

出现规则:

根据 **db2haicu** XML 模式定义，可以在 DB2ClusterType 元素中包括零个或零个以上 DB2PartitionSet 元素。

HADRDBSet

类型: HADRDBType

HADRDBType 元素包含一列高可用性灾难恢复 (HADR) 主/备用数据库对。

出现规则:

根据 **db2haicu** XML 模式定义，可以在 DB2ClusterType 元素中包括零个或零个以上 HADRDBSet 元素。

使用说明:

- 在分区数据库环境中不能包括 HADRDBSet。
- 如果包括 HADRDBSet，那么必须在 FailoverPolicy 元素中指定故障转移策略 HADRFailover。

HADBSet

类型: HADBType

HADBType 元素包含要包括在集群域中并使其具有高可用性的数据库的列表。

出现规则:

根据 **db2haicu** XML 模式定义，可以在 DB2ClusterType 元素中包括零个或零个以上 HADBSet 元素。

属性

clusterManagerName（可选）

clusterManagerName 属性指定集群管理器。

下表中指定了此属性的有效值:

表 3. clusterManager 属性的有效值

clusterManagerName 值	集群管理器产品
TSA	IBM Tivoli System Automation for Multiplatforms (SA MP)

使用说明

在单一分区数据库环境中，您通常只为每个数据库管理器实例创建单个集群域。

以下是多分区数据库环境的一种可能配置:

- 将 FailoverPolicy 元素设置为 Mutual
- 在 DB2PartitionSet 的 DB2Partition 子元素中，使用 MutualPair 元素指定位于一个集群域中的两个集群域节点

DB2 高可用性实例配置实用程序 (db2haicu) 输入文件的 ClusterDomainType XML 模式定义:

ClusterDomainType 元素包含关于下列内容的规范: 集群域中的机器或计算机（也称为*集群域节点*）、*网络等值*（相互之间可以进行故障转移的多组网络），以及*定额设备*（决定性机制）。

『超元素』
『XML 模式定义』
『子元素』
第 88 页的『属性』

超元素

下列类型的元素包含 ClusterDomainType 子元素:

- DB2ClusterType

XML 模式定义

```
<xs:complexType name='ClusterDomainType'>
  <xs:sequence>
    <xs:element name='Quorum'
                type='QuorumType'
                minOccurs='0'/>
    <xs:element name='PhysicalNetwork'
                type='PhysicalNetworkType'
                minOccurs='0'
                maxOccurs='unbounded'/>
    <xs:element name='ClusterNode'
                type='ClusterNodeType'
                maxOccurs='unbounded'/>
  </xs:sequence>
  <xs:attribute name='domainName'      type='xs:string' use='required'/>
</xs:complexType>
```

子元素

Quorum

类型: QuorumType

QuorumType 元素为集群域指定*定额设备*。

出现规则:

可以在 ClusterDomainType 元素中包括零个或一个 Quorum 元素。

PhysicalNetwork

类型: PhysicalNetworkType

PhysicalNetworkType 元素包含相互之间可以进行故障转移的网络接口卡。这种网络也称为*网络等值*。

出现规则:

可以在 ClusterDomainType 元素中包括零个或零个以上 PhysicalNetwork 元素。

ClusterNode

类型: ClusterNodeType

ClusterNodeType 元素包含关于集群中的特定计算机或机器（也称为*集群域节点*）的信息。

出现规则:

必须在 ClusterDomainType 元素中至少指定一个 ClusterNode 元素。

使用说明

IBM Tivoli System Automation for Multiplatforms (SA MP)最多支持 32

个集群域节点。如果集群管理器是 SA MP，那么最多可以在 ClusterDomainType 元素中包括 32 个 ClusterNode 元素。

属性

domainName（必需）

必须为 ClusterDomainType 元素指定唯一名称。

如果要使用可靠的可缩放集群技术 (RSCT) 来管理集群，那么下列限制适用于 domainName:

- domainName 只能包含字符 A-Z、a-z、数字 0-9、句号 (.) 和下划线 (_)
- domainName 不能是“IW”

以下示例针对 ClusterDomainType 元素:

```
<ClusterDomain domainName="hadr_linux_domain">
 <Quorum quorumDeviceProtocol="network" quorumDeviceName="9.26.4.5"/>
  <PhysicalNetwork physicalNetworkName="db2_public_network_0"
   physicalNetworkProtocol="ip">
   <Interface interfaceName="eth0" clusterNodeName="linux01">
    <IPAddress baseAddress="9.26.124.30" subnetMask="255.255.255.0"
     networkName="db2_public_network_0"/>
   </Interface>
   <Interface interfaceName="eth0" clusterNodeName="linux02">
    <IPAddress baseAddress="9.26.124.31" subnetMask="255.255.255.0"
     networkName="db2_public_network_0"/>
   </Interface>
  </PhysicalNetwork>
 <ClusterNode clusterNodeName="linux01"/>
 <ClusterNode clusterNodeName="linux02"/>
</ClusterDomain>
```

DB2 高可用性实例配置实用程序 (db2haicu) 输入文件的 QuorumType XML 模式定义:

QuorumType 元素为集群域指定*定额设备*。

『超元素』
『XML 模式定义』
第 89 页的『子元素』
第 89 页的『属性』

超元素

下列类型的元素包含 QuorumType 子元素:

- ClusterDomainType

XML 模式定义

```
<xs:complexType name='QuorumType'>
  <xs:attribute name='quorumDeviceProtocol'
                type='QuorumDeviceProtocolType'
                use='required'/>
  <xs:attribute name='quorumDeviceName'
                type='xs:string'
                use='required'/>
</xs:complexType>
```

子元素

无。

属性

quorumDeviceProtocol（必需）

quorumDeviceProtocol 指定要使用的定额类型。

*定额设备*可帮助集群管理器作出集群管理决策。

quorumDeviceProtocol 属性的类型为 QuorumDeviceProtocolType。

以下是 QuorumDeviceProtocolType 的 XML 模式定义:

```
<xs:simpleType name='QuorumDeviceProtocolType'>
  <xs:restriction base='xs:string'>
    <xs:enumeration value='disk'/>
    <xs:enumeration value='scsi'/>
    <xs:enumeration value='network'/>
    <xs:enumeration value='eckd'/>
    <xs:enumeration value='mns'/>
  </xs:restriction>
</xs:simpleType>
```

下表中指定了当前支持的此属性值:

表 4. quorumDeviceProtocol 属性的有效值

quorumDeviceProtocol 值	含义
网络	网络定额设备是每个集群域节点在任何时候都可连接至的 IP 地址。

quorumDeviceName（必需）

quorumDeviceName 的值取决于在 quorumDeviceProtocol 中指定的定额设备的类型。

下表中指定了此属性的有效值:

表 5. quorumDeviceName 属性的有效值

quorumDeviceProtocol 的值	quorumDeviceName 的有效值
网络	包含格式正确的 IP 地址的字符串。例如: 12.126.4.5 要使您指定的 IP 地址成为有效的网络定额设备，每个集群域节点都必须能够访问此 IP 地址（例如，使用 ping 实用程序）。

DB2 高可用性实例配置实用程序 (db2haicu) 输入文件的 PhysicalNetworkType XML 模式定义:

PhysicalNetworkType 元素包含相互之间可以进行故障转移的网络接口卡。这种网络也称为*网络等值*。

『超元素』
『XML 模式定义』
『子元素』
『属性』

超元素

下列类型的元素包含 PhysicalNetworkType 子元素:

- ClusterDomainType

XML 模式定义

```
<xs:complexType name='PhysicalNetworkType'>
  <xs:sequence>
    <xs:element name='Interface'
                type='InterfaceType'
                minOccurs='1'
                maxOccurs='unbounded'/>
    <xs:element name='LogicalSubnet'
                type='IPAddressType'
                minOccurs='0'
                maxOccurs='unbounded'/>
  </xs:sequence>
  <xs:attribute name='physicalNetworkName'
                type='xs:string'
                use='required'/>
  <xs:attribute name='physicalNetworkProtocol'
                type='PhysicalNetworkProtocolType'
                use='required'/>
</xs:complexType>
```

子元素

Interface

类型: InterfaceType

InterfaceType 元素由 IP 地址、网络中的计算机或机器（也称为*集群域节点*）的名称以及该集群域节点上的*网络接口卡* (NIC) 的名称组成。

出现规则:

必须在 PhysicalNetworkType 元素中指定一个或多个 Interface 元素。

LogicalSubnet

类型: IPAddressType

IPAddressType 元素包含一个 IP 地址的所有详细信息，例如，*基地址*、*子网掩码*和该 IP 地址所属的网络的名称。

出现规则:

可以在 PhysicalNetworkType 元素中包括一个或多个 LogicalSubnet 元素。

属性

physicalNetworkName（必需）

必须为每个 PhysicalNetworkType 元素指定唯一的 physicalNetworkName。

physicalNetworkProtocol（必需）

physicalNetworkProtocol 属性的类型为 PhysicalNetworkProtocolType。

以下是 PhysicalNetworkProtocolType 元素的 XML 模式定义:

```
<xs:simpleType name='PhysicalNetworkProtocolType'>
  <xs:restriction base='xs:string'>
    <xs:enumeration value='ip'/>
    <xs:enumeration value='rs232'/>
    <xs:enumeration value='scsi'/>
    <xs:enumeration value='ssa'/>
    <xs:enumeration value='disk'/>
  </xs:restriction>
</xs:simpleType>
```

下表中指定了当前支持的此属性值:

表 6. physicalNetworkProtocol 属性的有效值

physicalNetworkProtocol 值	**含义**
ip	TCP/IP 协议

DB2 高可用性实例配置实用程序 (db2haicu) 输入文件的 InterfaceType XML 模式定义:

InterfaceType 元素由 IP 地址、网络中的计算机或机器（也称为*集群域节点*）的名称以及该集群域节点上的*网络接口卡* (NIC) 的名称组成。

『超元素』
『XML 模式定义』
『子元素』
第 92 页的『属性』

超元素

下列类型的元素具有 InterfaceType 子元素:

- PhysicalNetworkType

XML 模式定义

```
<xs:complexType name='InterfaceType'>
  <xs:sequence>
    <xs:element name='IPAddress'       type='IPAddressType'/>
  </xs:sequence>
  <xs:attribute name='interfaceName'   type='xs:string' use='required'/>
  <xs:attribute name='clusterNodeName' type='xs:string' use='required'/>
</xs:complexType>
```

子元素

IPAddress

类型: IPAddressType

IPAddressType 元素包含一个 IP 地址的所有详细信息，例如，*基地址*、*子网掩码*和该 IP 地址所属的网络的名称。

出现规则:

必须在 InterfaceType 元素中正好指定一个 IPAddress。

属性

interfaceName（必需）

必须在 interfaceName 属性中指定 NIC 的名称。在 interfaceName 中指定的 NIC 必须存在于您在 clusterNodeName 属性中指定的集群域节点上。

clusterNodeName（必需）

必须指定位于您在 IPAddress 元素中指定的 IP 地址处的集群域节点的名称。

DB2 高可用性实例配置实用程序 (db2haicu) 输入文件的 IPAddressType XML 模式元素:

IPAddressType 元素包含一个 IP 地址的所有详细信息，例如，*基地址*、*子网掩码*和该 IP 地址所属的网络的名称。

『超元素』
『XML 模式定义』
『子元素』
『属性』

超元素

下列类型的元素具有 IPAddressType 子元素:

- PhysicalNetworkType
- InterfaceType
- DB2PartitionType

XML 模式定义

```
<xs:complexType name='IPAddressType'>
  <xs:attribute name='baseAddress' type='xs:string' use='required'/>
  <xs:attribute name='subnetMask'  type='xs:string' use='required'/>
  <xs:attribute name='networkName' type='xs:string' use='required'/>
</xs:complexType>
```

子元素

无。

属性

baseAddress（必需）

必须使用具有以下有效 IP 地址格式的字符串指定基本 IP 地址: 范围在 0 到 255 之间的四组数字，中间用句点分隔。例如:

```
162.148.31.101
```

subnetMask（必需）

必须使用具有有效 IP 地址格式的字符串指定基本 IP 地址。

networkName（必需）

此处对 networkName 指定的值必须与对包含此 IPAddress 元素的 PhysicalNetworkType 元素的 physicalNetworkName 属性所指定的值相同。

DB2 高可用性实例配置实用程序 (db2haicu) 输入文件的 ClusterNodeType XML 模式定义:

ClusterNodeType 元素包含关于集群中的特定计算机或机器（也称为*集群域节点*）的信息。

『超元素』
『XML 模式定义』
『子元素』
『属性』

超元素

下列类型的元素具有 ClusterNodeType 元素:

- ClusterDomainType

XML 模式定义

```
<xs:complexType name='ClusterNodeType'>
  <xs:attribute name='clusterNodeName' type='xs:string' use='required'/>
</xs:complexType>
```

子元素

无。

属性

clusterNodeName（必需）
必须指定集群域节点的名称。

DB2 高可用性实例配置实用程序 (db2haicu) 输入文件的 FailoverPolicyType XML 模式定义:

FailoverPolicyType 元素指定集群管理器应该对集群域使用的*故障转移策略*。

『超元素』
『XML 模式定义』
第 94 页的『子元素』
第 94 页的『可能的值』

超元素

下列类型的元素包含 InterfaceType 子元素:

- DB2ClusterType

XML 模式定义

```
<xs:complexType name='FailoverPolicyType'>
  <xs:choice>
    <xs:element name='RoundRobin'
                type='xs:string'
                minOccurs='0' />
    <xs:element name='Mutual'
                type='xs:string'
                minOccurs='0'
                maxOccurs='unbounded'/>
    <xs:element name='NPlusM'
                type='xs:string'
                minOccurs='0'
                maxOccurs='unbounded'/>
```

```
    <xs:element name='LocalRestart'
                type='xs:string'
                fixed=''/>
    <xs:element name='HADRFailover'
                type='xs:string'
                fixed=''/>
    <xs:element name='Custom'
                type='xs:string'
                minOccurs='0' />
  </xs:choice>
</xs:complexType>
```

子元素

无。

可能的值

选择下列其中一个选项，以指定在集群域中的任何地方出现故障时集群管理器应使用的故障转移策略的类型。

*故障转移策略*指定在网络接口卡或数据库服务器等集群元素出现故障时集群管理器应如何响应。通常，集群管理器会将工作负载从故障元素转移到备用元素，先前已向集群管理器将该备用元素标识为故障元素的适当替换元素。这种将工作负载从故障元素转移到另一个元素的操作称为*故障转移*。

RoundRobin

使用*循环故障转移策略*时，如果集群域中的一台计算机（也称为*集群域节点*或简单地称为*节点*）出现故障，那么在集群域中的任何其他节点上，数据库管理器将重新开始出现故障的集群域节点中的工作。

Mutual

要配置*相互故障转移策略*，应将集群域中的一个计算机对（也称为*集群域节点*或简单地称为*节点*）作为系统对关联。如果此计算机对中的一个节点出现故障，那么故障节点上的数据库分区将故障转移到该计算机对中的另一个节点。仅当具有多个数据库分区时，相互故障转移才可用。

NPlusM

使用 *N 加 M 故障转移策略*时，如果集群域中的一台计算机（也称为*集群域节点*或简称为*节点*）出现故障，那么故障节点上的数据库分区将故障转移至集群域中的任何其他节点。如果启用了漫游 HA 故障转移，那么当该故障节点再次处于联机状态时，最近一次发生故障的节点将成为备用节点。N 加 M 故障转移策略的漫游 HA 故障转移仅在 M=1 时才受支持。仅当具有多个数据库分区时，N 加 M 故障转移才可用。

LocalRestart

使用*本地重新启动故障转移策略*时，如果集群域中的一台计算机（也称为*集群域节点*或简单地称为*节点*）出现故障，那么数据库管理器将在出现故障的那个节点上的适当位置（或本地）重新启动数据库。

HADRFailover

配置 *HADR 故障转移策略*时，您正在启用 DB2 高可用性灾难恢复 (HADR) 功能部件来管理故障转移。如果 HADR 主数据库出现故障，那么数据库管理器会将出现故障的数据库中的工作负载移至 HADR 备用数据库。

Custom

配置*定制故障转移策略*时，您将在集群域中创建一列计算机（也称为*集群域节点*或简单地称为*节点*），数据库管理器可以故障转移到这些计算机上。如果集群域中的一个节点出现故障，那么数据库管理器会将故障节点中的工作负载移至您指定的列表中的一个节点。

DB2 高可用性实例配置实用程序 (db2haicu) 输入文件的 DB2PartitionSetType XML 模式定义:

DB2PartitionSetType 元素包含关于数据库分区的信息。DB2PartitionSetType 元素仅适用于分区数据库环境。

『超元素』
『XML 模式定义』
『子元素』
『属性』

超元素

InterfaceType 是下列各项的子元素:

- PhysicalNetworkType

XML 模式定义

```
<xs:complexType name='DB2PartitionSetType'>
  <xs:sequence>
    <xs:element name='DB2Partition'
                type='DB2PartitionType'
                maxOccurs='unbounded'/>
  </xs:sequence>
</xs:complexType>
```

子元素

DB2Partition

类型: DB2PartitionType

DB2PartitionType 元素指定数据库分区（包括该数据库分区所属的 DB2 数据库管理器实例）和数据库分区号。

出现规则:

必须在 DB2PartitionSetType 元素中指定一个或多个 DB2Partition 元素。

属性

无。

DB2 高可用性实例配置实用程序 (db2haicu) 输入文件的 DB2PartitionType XML 模式元素:

DB2PartitionType 元素指定数据库分区（包括该数据库分区所属的 DB2 数据库管理器实例）和数据库分区号。

『超元素』
『XML 模式定义』
『子元素』
第 97 页的『属性』

超元素

InterfaceType 是下列各项的子元素:

- DB2PartitionSetType

XML 模式定义

```
<xs:complexType name='DB2PartitionType'>
 <xs:sequence>
    <xs:element name='VirtualIPAddress'
                type='IPAddressType'
                minOccurs='0'
                maxOccurs='unbounded'/>
    <xs:element name='Mount'
                type='MountType'
                minOccurs='0'
                maxOccurs='unbounded'/>
    <xs:element name='HADRDB'
                type='HADRDBType'
                minOccurs='0'
                maxOccurs='unbounded'/>
    <xs:element name='MutualPair'
                type='MutualPolicyType'
                minOccurs='0'
                maxOccurs='1'/>
    <xs:element name='NPlusMNode'
                type='NPlusMPolicyType'
                minOccurs='0'
                maxOccurs='unbounded'/>
    <xs:element name='CustomNode'
                type='CustomPolicyType'
                minOccurs='0'
                maxOccurs='unbounded'/>
  </xs:sequence>
  <xs:attribute name='instanceName'     type='xs:string'  use='required'/>
  <xs:attribute name='dbpartitionnum'   type='xs:integer' use='required'/>
</xs:complexType>
```

子元素

VirtualIPAddress

类型: IPAddressType

IPAddressType 元素包含一个 IP 地址的所有详细信息，例如，*基地址*、*子网掩码*和该 IP 地址所属的网络的名称。

您可以不包括 VirtualIPAddress，也可以在 DB2PartitionType 元素中包括无数个 VirtualIPAddress 元素。

Mount 类型: MountType

MountType 元素包含关于*安装点*的信息，例如，标识所安装文件的位置的文件路径。

您可以不包括 Mount，也可以在 DB2PartitionType 元素中包括无数个 Mount 元素。

HADRDB

类型: HADRDBType

HADRDBType 元素包含一列高可用性灾难恢复 (HADR) 主/备用数据库对。

您可以不包括 HADRDB，也可以在 DB2PartitionType 元素中包括无数个 HADRDB 元素。

MutualPair

类型: MutualPolicyType

MutualPolicyType 元素包含关于相互之间可以进行故障转移的集群域节点对的信息。

您可以不包括 MutualPair，也可以在 DB2PartitionType 元素中正好包括一个 MutualPair 元素。

NPlusMNode

类型: NPlusMPolicyType

您可以不包括 NPlusMNode，也可以在 DB2PartitionType 元素中包括无数个 NPlusMNode 元素。

CustomNode

类型: CustomPolicyType

您可以不包括 CustomNode，也可以在 DB2PartitionType 元素中包括无数个 CustomNode 元素。

属性

instanceName（必需）

在 instanceName 属性中，必须指定与此 DB2PartitionType 元素关联的 DB2 数据库管理器实例。

dbpartitionnum（必需）

在 dbpartitionnum 属性中，必须指定唯一标识数据库分区的数据库分区号（例如，在 db2nodes.cfg 文件中指定的 dbpartitionnum 号码。）

DB2 高可用性实例配置实用程序 (db2haicu) 输入文件的 MountType XML 模式定义:

MountType 元素包含关于安装点的信息，例如，标识所安装文件的位置的文件路径。

『超元素』
第 98 页的『XML 模式定义』
第 98 页的『子元素』
第 98 页的『属性』

超元素

下列类型的元素包含 MountType 子元素:

- DB2PartitionType

XML 模式定义

```
<xs:complexType name='MountType'>
  <xs:attribute name='filesystemPath' type='xs:string' use='required'/>
</xs:complexType>
```

子元素

无。

属性

filesystemPath（必需）

指定在安装文件系统时与安装点关联的路径。

DB2 高可用性实例配置实用程序 (db2haicu) 输入文件的 MutualPolicyType XML 模式定义:

MutualPolicyType 元素包含关于相互之间可以进行故障转移的集群域节点对的信息。

『超元素』
『XML 模式定义』
『子元素』
『属性』

超元素

下列类型的元素包含 MutualPolicyType 子元素:

- DB2PartitionType

XML 模式定义

```
<xs:complexType name='MutualPolicyType'>
  <xs:attribute name='systemPairNode1' type='xs:string' use='required'/>
  <xs:attribute name='systemPairNode2' type='xs:string' use='required'/>
</xs:complexType>
```

子元素

无。

属性

systemPairNode1（必需）

在 systemPairNode1 中，必须指定可以故障转移到 systemPairNode2 中指定的集群域节点的集群域节点名称。

systemPairNode2（必需）

在 systemPairNode2 中，必须指定可以故障转移到 systemPairNode1 中指定的集群域节点的集群域节点名称。

DB2 高可用性实例配置实用程序 (db2haicu) 输入文件的 NPlusMPolicyType XML 模式定义:

NPlusMPolicy 表明如果集群域中的某台计算机发生故障，那么失败节点上的数据库分区将故障转移到同一集群域中的任何其他可用节点。XML 模式将定义与此 HADR 策略相关联的配置。

『超元素』
『XML 模式定义』
『子元素』
『属性』

超元素

下列类型的元素包含 NPlusMPolicyType 子元素:

- DB2PartitionType

XML 模式定义

```
<xs:complexType name='NPlusMPolicyType'>
  <xs:attribute name='standbyNodeName' type='xs:string' use='required'/>
</xs:complexType>
```

子元素

无。

属性

standbyNodeName（必需）

在 standbyNodeName 元素中，必须指定包含此 NPlusMPolicyType 元素的分区可以故障转移至的集群域节点的名称。

DB2 高可用性实例配置实用程序 (db2haicu) 输入文件的 CustomPolicyType XML 模式定义:

CustomPolicyType XML 模式定义定制 HADR 策略的配置设置。您可以定义在此模式中缺省情况下会故障转移到的节点。

『超元素』
『XML 模式定义』
『子元素』
第 100 页的『属性』

超元素

下列类型的元素包含 CustomPolicyType 子元素:

- DB2PartitionType

XML 模式定义

```
<xs:complexType name='NPlusMPolicyType'>
  <xs:attribute name='standbyNodeName' type='xs:string' use='required'/>
</xs:complexType>
```

子元素

无。

属性

customNodeName（必需）

在 customNodeName 元素中，必须指定包含此 CustomPolicyType 元素的分区可以故障转移至的集群域节点的名称。

DB2 高可用性实例配置实用程序 (db2haicu) 输入文件的 HADRDBType XML 模式定义:

HADRDBType 元素包含一列高可用性灾难恢复 (HADR) 主/备用数据库对。

『超元素』
『XML 模式定义』
『子元素』
第 101 页的『属性』
第 101 页的『使用说明』
第 101 页的『限制』

超元素

下列类型的元素包含 HADRDBType 子元素:

- DB2ClusterType
- DB2PartitionType

XML 模式定义

```
<xs:complexType name='HADRDBType'>
 <xs:sequence>
   <xs:element name='HADRDB' type='HADRDBDefn'  minOccurs='1'  maxOccurs='1'/>
   <xs:element name='VirtualIPAddress' type='IPAddressType'
minOccurs='0' maxOccurs='1'/>
 </xs:sequence>
</xs:complexType>
```

子元素

VirtualIPAddress

类型: IPAddressType

IPAddressType 元素包含一个 IP 地址的所有详细信息，例如，*基地址*、*子网掩码*和该 IP 地址所属的网络的名称。

出现规则:

可以在 HADRDBType 元素中包括零个或零个以上 VirtualIPAddress 元素。

HADRDB

类型: HADRDBDefn

HADRDBDefn 元素包含关于高可用性灾难恢复 (HADR) 主/备用数据库对的信息。

出现规则:

可以在 HADRDBType 元素中包括一个或多个 VirtualIPAddress 元素。

属性

无。

使用说明

如果在给定集群域的规范中包括 HADRDBType 元素，那么还必须在相同集群域规范中包括指定 HADRFailover 的 FailoverPolicy 元素。

限制

在分区数据库环境中不能使用 HADRDBType 元素。

以下示例针对 HADRDBType 元素:

```
<HADRDBSet>
 <HADRDB databaseName="HADRDB" localInstance="db2inst1"
  remoteInstance="db2inst1" localHost="linux01" remoteHost="linux02" />
 <VirtualIPAddress baseAddress="9.26.124.22" subnetMask="255.255.245.0"
  networkName="db2_public_network_0"/>
</HADRDBSet>
```

DB2 高可用性实例配置实用程序 (db2haicu) 输入文件的 HADRDBDefn XML 模式定义:

HADRDBDefn 元素包含关于高可用性灾难恢复 (HADR) 主/备用数据库对的信息。

『超元素』
『XML 模式定义』
『子元素』
『属性』

超元素

下列类型的元素包含 HADRDBDefn 子元素:

- HADRDBType

XML 模式定义

```
<xs:complexType name='HADRDBDefn'>
  <xs:attribute name='databaseName'      type='xs:string' use='required'/>
  <xs:attribute name='localInstance'     type='xs:string' use='required'/>
  <xs:attribute name='remoteInstance'    type='xs:string' use='required'/>
  <xs:attribute name='localHost'         type='xs:string' use='required'/>
  <xs:attribute name='remoteHost'        type='xs:string' use='required'/>
</xs:complexType>
```

子元素

无。

属性

databaseName（必需）
输入 HADR 数据库的名称。

localInstance（必需）
localInstance 是 HADR 主数据库的数据库管理器实例。

remoteInstance（必需）

remoteInstance 是 HADR 备用数据库的数据库管理器实例。

localHost（必需）

localHost 是 HADR 主数据库所在的集群域节点的主机名。

remoteHost（必需）

remoteHost 是 HADR 备用数据库所在的集群域节点的主机名。

DB2 高可用性实例配置实用程序 (db2haicu) 输入文件的 HADBType XML 模式定义:

HADBType 元素包含要包括在集群域中并使其具有高可用性的数据库的列表。

『超元素』
『XML 模式定义』
『子元素』
『属性』

超元素

下列类型的元素包含 HADBType 子元素:

- DB2ClusterType

XML 模式定义

```
<xs:complexType name='HADBType'>
  <xs:sequence>
    <xs:element name='HADB'         type='HADBDefn' maxOccurs='unbounded'/>
  </xs:sequence>
  <xs:attribute name='instanceName' type='xs:string' use='required'/>
</xs:complexType>
```

子元素

HADB

类型: HADBDefn

HADBDefn 元素描述要包括在集群域中并使其具有高可用性的数据库。

出现规则:

必须在 HADBType 元素中包括一个或多个 HADB 元素。

属性

instanceName（必需）

在 instanceName 属性中，必须指定 HADB 元素中指定的数据库所属的 DB2 数据库管理器实例。

DB2 高可用性实例配置实用程序 (db2haicu) 输入文件的 HADBDefn XML 模式元素:

HADBDefn 元素描述要包括在集群域中并使其具有高可用性的数据库。

『超元素』
『XML 模式定义』
『子元素』
『属性』

超元素

HADBDefn 是下列各项的子元素:

- HADRDBType

XML 模式定义

```
<xs:complexType name='HADBDefn'>
  <xs:attribute name='databaseName' type='xs:string' use='required'/>
</xs:complexType>
```

子元素

无。

属性

databaseName（必需）

必须在 databaseName 属性中正好指定一个数据库名称。

DB2 高可用性实例配置实用程序 (db2haicu) 的样本 XML 输入文件:

在 sqllib 目录的 samples 子目录中存在一组您可以修改且与 **db2haicu** 配合使用的样本 XML 输入文件，可用来配置集群环境。

db2ha_sample_sharedstorage_mutual.xml:

样本文件 db2ha_sample_sharedstorage_mutual.xml 是一个 XML 输入文件示例，您将它传递至 DB2 高可用性实例配置实用程序 (db2haicu)来指定新的*集群域*。db2ha_sample_sharedstorage_mutual.xml 位于 sqllib/samples/ha/xml 目录中。

功能部件

db2ha_sample_sharedstorage_mutual.xml 样本演示如何将 **db2haicu** 与 XML 输入文件配合使用来定义具有下列详细信息的集群域:

- 定额设备: 网络
- 集群中的计算机数（集群域节点数）: 两台
- 故障转移策略: 相互
- 数据库分区数: 一个
- 虚拟（服务）IP 地址数: 一个
- 用于故障转移的共享安装点数: 一个

XML 源代码

```
<!-- ================================================================ -->
<!-- = Use the DB2 High Availability Instance Configuration Utility  = -->
<!-- = (db2haicu) XML schema definition, db2ha.xsd, and specify      = -->
<!-- = IBM Tivoli System Automation for Multiplatforms (SA MP)       = -->
<!-- = Base Component as the cluster manager.                        = -->
<!-- ================================================================ -->
```

```
<DB2Cluster xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:noNamespaceSchemaLocation="db2ha.xsd"
            clusterManagerName="TSA"
            version="1.0">

  <!-- ================================================================= -->
  <!-- = Create a cluster domain named db2HAdomain.                    = -->
  <!-- ================================================================= -->
  <ClusterDomain domainName="db2HAdomain">

    <!-- ================================================================= -->
    <!-- = Specify a network quorum device (IP address: 19.126.4.5).   = -->
    <!-- = The IP must be pingable at all times by each of the cluster = -->
    <!-- = domain nodes.                                               = -->
    <!-- ================================================================= -->
    <Quorum quorumDeviceProtocol="network" quorumDeviceName="19.126.4.5"/>

    <!-- ================================================================= -->
    <!-- = Create a network named db2_public_network_0 with an IP      = -->
    <!-- = network protocol.                                           = -->
    <!-- = This network contains two computers: hasys01 and hasys02.   = -->
    <!-- = Each computer has one network interface card (NIC) called   = -->
    <!-- = eth0.                                                       = -->
    <!-- = The IP address of the NIC on hasys01 is 19.126.52.139       = -->
    <!-- = The IP address of the NIC on hasys02 is 19.126.52.140       = -->
    <!-- ================================================================= -->
    <PhysicalNetwork physicalNetworkName="db2_public_network_0"
                     physicalNetworkProtocol="ip">

      <Interface interfaceName="eth0" clusterNodeName="hasys01">
        <IPAddress baseAddress="19.126.52.139"
                   subnetMask="255.255.255.0"
                   networkName="db2_public_network_0"/>
      </Interface>

      <Interface interfaceName="eth0" clusterNodeName="hasys02">
        <IPAddress baseAddress="19.126.52.140"
                   subnetMask="255.255.255.0"
                   networkName="db2_public_network_0"/>
      </Interface>

    </PhysicalNetwork>

    <!-- ================================================================= -->
    <!-- = List the computers (cluster nodes) in the cluster domain.   = -->
    <!-- ================================================================= -->
    <ClusterNode clusterNodeName="hasys01"/>
    <ClusterNode clusterNodeName="hasys02"/>

  </ClusterDomain>


  <!-- ================================================================= -->
  <!-- = The failover policy specifies the order in which the cluster  = -->
  <!-- = domain nodes should fail over.                              = -->
  <!-- ================================================================= -->
  <FailoverPolicy>
     <Mutual />
  </FailoverPolicy>


  <!-- ================================================================= -->
  <!-- = Specify all the details of the database partition             = -->
  <!-- ================================================================= -->
  <DB2PartitionSet>

    <DB2Partition dbpartitionnum="0" instanceName="db2inst1">
```

```
      <VirtualIPAddress baseAddress="19.126.52.222"
                        subnetMask="255.255.255.0"
                        networkName="db2_public_network_0"/>
      <Mount filesystemPath="/home/db2inst1"/>
      <MutualPair systemPairNode1="hasys01" systemPairNode2="hasys02" />
    </DB2Partition>

  </DB2PartitionSet>

</DB2Cluster>
```

db2ha_sample_DPF_mutual.xml:

样本文件 db2ha_sample_DPF_mutual.xml 是一个 XML 输入文件示例，您将它传递至 DB2 高可用性实例配置实用程序 (db2haicu)来指定新的*集群域*。db2ha_sample_DPF_mutual.xml 位于 sqllib/samples/ha/xml 目录中。

功能部件

db2ha_sample_DPF_mutual.xml 样本演示如何将 **db2haicu** 与 XML 输入文件配合使用来定义具有下列详细信息的集群域:

- 定额设备: 网络
- 集群中的计算机数（集群域节点数）: 四台
- 故障转移策略: 相互
- 数据库分区数: 两个
- 虚拟（服务）IP 地址数: 一个
- 用于故障转移的共享安装点数: 两个
- 为实现高可用性配置的数据库数: 两个

XML 源代码

```
<!-- ******************************************************************** -->
<!-- * Use the DB2 High Availability Instance Configuration Utility  * -->
<!-- * (db2haicu) XML schema definition, db2ha.xsd, and specify      * -->
<!-- * IBM Tivoli System Automation for Multiplatforms (SA MP)       * -->
<!-- * Base Component as the cluster manager.                       * -->
<!-- ******************************************************************** -->
<DB2Cluster xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:noNamespaceSchemaLocation="db2ha.xsd"
            clusterManagerName="TSA"
            version="1.0">

  <!-- ******************************************************************** -->
  <!-- * Create a cluster domain named db2HAdomain.                     * -->
  <!-- ******************************************************************** -->
  <ClusterDomain domainName="db2HAdomain">

    <!-- ****************************************************************** -->
    <!-- * Specify a network quorum device (IP address: 19.126.4.5).   * -->
    <!-- * The IP must be pingable at all times by each of the cluster * -->
    <!-- * domain nodes.                                                * -->
    <!-- ****************************************************************** -->
    <Quorum quorumDeviceProtocol="network" quorumDeviceName="19.126.4.5"/>

    <!-- ****************************************************************** -->
    <!-- * Create a network named db2_public_network_0 with an IP       * -->
    <!-- * network protocol.                                            * -->
    <!-- * This network contains four computers: hasys01, hasys02,     * -->
    <!-- * hasys03, and hasys04.                                       * -->
    <!-- * Each computer has a network interface card called eth0.     * -->
```

```
<!-- = The IP address of eth0 on hasys01 is 19.126.124.30          = -->
<!-- = The IP address of eth0 on hasys02 is 19.126.124.31          = -->
<!-- = The IP address of eth0 on hasys03 is 19.126.124.32          = -->
<!-- = The IP address of eth0 on hasys04 is 19.126.124.33          = -->
<!-- ============================================================== -->
<PhysicalNetwork physicalNetworkName="db2_public_network_0"
                 physicalNetworkProtocol="ip">

  <Interface interfaceName="eth0" clusterNodeName="hasys01">
    <IPAddress baseAddress="19.126.124.30"
               subnetMask="255.255.255.0"
               networkName="db2_public_network_0"/>
  </Interface>

  <Interface interfaceName="eth0" clusterNodeName="hasys02">
    <IPAddress baseAddress="19.126.124.31"
               subnetMask="255.255.255.0"
               networkName="db2_public_network_0"/>
  </Interface>

  <Interface interfaceName="eth0" clusterNodeName="hasys03">
    <IPAddress baseAddress="19.126.124.32"
               subnetMask="255.255.255.0"
               networkName="db2_public_network_0"/>
  </Interface>

  <Interface interfaceName="eth0" clusterNodeName="hasys04">
    <IPAddress baseAddress="19.126.124.33"
               subnetMask="255.255.255.0"
               networkName="db2_public_network_0"/>
  </Interface>

</PhysicalNetwork>

<!-- ============================================================== -->
<!-- = Create a network named db2_private_network_0 with an IP     = -->
<!-- = network protocol.                                           = -->
<!-- = This network contains four computers: hasys01, hasys02,     = -->
<!-- = hasys03, and hasys04 (same as db2_public_network_0.)        = -->
<!-- = In addition to eth0, each computer has a network interface  = -->
<!-- = card called eth1.                                           = -->
<!-- = The IP address of eth1 on hasys01 is 192.168.23.101         = -->
<!-- = The IP address of eth1 on hasys02 is 192.168.23.102         = -->
<!-- = The IP address of eth1 on hasys03 is 192.168.23.103         = -->
<!-- = The IP address of eth1 on hasys04 is 192.168.23.104         = -->
<!-- ============================================================== -->
<PhysicalNetwork physicalNetworkName="db2_private_network_0"
                 physicalNetworkProtocol="ip">

  <Interface interfaceName="eth1" clusterNodeName="hasys01">
    <IPAddress baseAddress="192.168.23.101"
               subnetMask="255.255.255.0"
               networkName="db2_private_network_0"/>
  </Interface>

  <Interface interfaceName="eth1" clusterNodeName="hasys02">
    <IPAddress baseAddress="192.168.23.102"
               subnetMask="255.255.255.0"
               networkName="db2_private_network_0"/>
  </Interface>

  <Interface interfaceName="eth1" clusterNodeName="hasys03">
    <IPAddress baseAddress="192.168.23.103"
               subnetMask="255.255.255.0"
               networkName="db2_private_network_0"/>
  </Interface>
```

```
    <Interface interfaceName="eth1" clusterNodeName="hasys04">
      <IPAddress baseAddress="192.168.23.104"
                 subnetMask="255.255.255.0"
                 networkName="db2_private_network_0"/>
    </Interface>

  </PhysicalNetwork>

  <!-- ================================================================ -->
  <!-- = List the computers (cluster nodes) in the cluster domain.   = -->
  <!-- ================================================================ -->
  <ClusterNode clusterNodeName="hasys01"/>
  <ClusterNode clusterNodeName="hasys02"/>
  <ClusterNode clusterNodeName="hasys03"/>
  <ClusterNode clusterNodeName="hasys04"/>

</ClusterDomain>


<!-- ================================================================ -->
<!-- = The failover policy specifies the order in which the cluster  = -->
<!-- = domain nodes should fail over.                                = -->
<!-- ================================================================ -->
<FailoverPolicy>
   <Mutual />
</FailoverPolicy>


<!-- ================================================================ -->
<!-- = Specify all the details of the database partitions.           = -->
<!-- ================================================================ -->
<DB2PartitionSet>

  <DB2Partition dbpartitionnum="0" instanceName="db2inst1">
     <VirtualIPAddress baseAddress="19.126.124.251"
                       subnetMask="255.255.255.0"
                       networkName="db2_public_network_0"/>
     <Mount filesystemPath="/hafs/db2inst1/NODE0000"/>
     <MutualPair systemPairNode1="hasys01" systemPairNode2="hasys02" />
  </DB2Partition>

  <DB2Partition dbpartitionnum="1" instanceName="db2inst1">
     <Mount filesystemPath="/hafs/db2inst1/NODE0001"/>
     <MutualPair systemPairNode1="hasys02" systemPairNode2="hasys01" />
  </DB2Partition>

  <DB2Partition dbpartitionnum="2" instanceName="db2inst1">
     <Mount filesystemPath="/hafs/db2inst1/NODE0002"/>
     <MutualPair systemPairNode1="hasys03" systemPairNode2="hasys04" />
  </DB2Partition>

  <DB2Partition dbpartitionnum="3" instanceName="db2inst1">
     <Mount filesystemPath="/hafs/db2inst1/NODE0003"/>
     <MutualPair systemPairNode1="hasys04" systemPairNode2="hasys03" />
  </DB2Partition>

</DB2PartitionSet>


<!-- ================================================================ -->
<!-- = List of databases to be configured for High Availability      = -->
<!-- ================================================================ -->
<HADBSet instanceName="db2inst1">
  <HADB databaseName = "SAMPLE" />
  <HADB databaseName = "MYDB" />
```

```
  </HADBSet>

</DB2Cluster>
```

db2ha_sample_DPF_NPlusM.xml:

样本文件 db2ha_sample_DPF_NPlusM.xml 是一个 XML 输入文件示例，您将它传递至 DB2 高可用性实例配置实用程序 (db2haicu)来指定新的*集群域*。db2ha_sample_DPF_NPlusM.xml 位于 sqllib/samples/ha/xml 目录中。

功能部件

db2ha_sample_DPF_NPlusM.xml 样本演示如何将 **db2haicu** 与 XML 输入文件配合使用来定义具有下列详细信息的集群域:

- 定额设备: 网络
- 集群中的计算机数（集群域节点数）: 四台
- 故障转移策略: N 加 M
- 数据库分区数: 两个
- 虚拟（服务）IP 地址数: 一个
- 用于故障转移的共享安装点数: 四个

XML 源代码

```
<!-- ================================================================= -->
<!-- = Use the DB2 High Availability Instance Configuration Utility  = -->
<!-- = (db2haicu) XML schema definition, db2ha.xsd, and specify      = -->
<!-- = IBM Tivoli System Automation for Multiplatforms (SA MP)       = -->
<!-- = Base Component as the cluster manager.                        = -->
<!-- ================================================================= -->
<DB2Cluster xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:noNamespaceSchemaLocation="db2ha.xsd"
            clusterManagerName="TSA"
            version="1.0">

  <!-- ================================================================= -->
  <!-- = Create a cluster domain named db2HAdomain.                    = -->
  <!-- ================================================================= -->
  <ClusterDomain domainName="db2HAdomain">

    <!-- ================================================================= -->
    <!-- = Specify a network quorum device (IP address: 19.126.4.5).   = -->
    <!-- = The IP must be pingable at all times by each of the cluster = -->
    <!-- = domain nodes.                                               = -->
    <!-- ================================================================= -->
    <Quorum quorumDeviceProtocol="network" quorumDeviceName="19.126.4.5"/>

    <!-- ================================================================= -->
    <!-- = Create a network named db2_public_network_0 with an IP      = -->
    <!-- = network protocol.                                           = -->
    <!-- = This network contains four computers: hasys01, hasys02,     = -->
    <!-- = hasys03, and hasys04.                                       = -->
    <!-- = Each computer has a network interface card called eth0.     = -->
    <!-- = The IP address of eth0 on hasys01 is 19.126.124.30          = -->
    <!-- = The IP address of eth0 on hasys02 is 19.126.124.31          = -->
    <!-- = The IP address of eth0 on hasys03 is 19.126.124.32          = -->
    <!-- = The IP address of eth0 on hasys04 is 19.126.124.33          = -->
    <!-- ================================================================= -->
    <PhysicalNetwork physicalNetworkName="db2_public_network_0"
                     physicalNetworkProtocol="ip">
```

```
  <Interface interfaceName="eth0" clusterNodeName="hasys01">
    <IPAddress baseAddress="19.126.124.30"
               subnetMask="255.255.255.0"
               networkName="db2_public_network_0"/>
  </Interface>

  <Interface interfaceName="eth0" clusterNodeName="hasys02">
    <IPAddress baseAddress="19.126.124.31"
               subnetMask="255.255.255.0"
               networkName="db2_public_network_0"/>
  </Interface>

  <Interface interfaceName="eth0" clusterNodeName="hasys03">
    <IPAddress baseAddress="19.126.124.32"
               subnetMask="255.255.255.0"
               networkName="db2_public_network_0"/>
  </Interface>

  <Interface interfaceName="eth0" clusterNodeName="hasys04">
    <IPAddress baseAddress="19.126.124.33"
               subnetMask="255.255.255.0"
               networkName="db2_public_network_0"/>
  </Interface>

</PhysicalNetwork>

<!-- ============================================================= -->
<!-- = Create a network named db2_private_network_0 with an IP     = -->
<!-- = network protocol.                                           = -->
<!-- = This network contains four computers: hasys01, hasys02,     = -->
<!-- = hasys03, and hasys04 (same as db2_public_network_0.)        = -->
<!-- = In addition to eth0, each computer has a network interface  = -->
<!-- = card called eth1.                                           = -->
<!-- = The IP address of eth1 on hasys01 is 192.168.23.101         = -->
<!-- = The IP address of eth1 on hasys02 is 192.168.23.102         = -->
<!-- = The IP address of eth1 on hasys03 is 192.168.23.103         = -->
<!-- = The IP address of eth1 on hasys04 is 192.168.23.104         = -->
<!-- ============================================================= -->
<PhysicalNetwork physicalNetworkName="db2_private_network_0"
                 physicalNetworkProtocol="ip">

  <Interface interfaceName="eth1" clusterNodeName="hasys01">
    <IPAddress baseAddress="192.168.23.101"
               subnetMask="255.255.255.0"
               networkName="db2_private_network_0"/>
  </Interface>

  <Interface interfaceName="eth1" clusterNodeName="hasys02">
    <IPAddress baseAddress="192.168.23.102"
               subnetMask="255.255.255.0"
               networkName="db2_private_network_0"/>
  </Interface>

  <Interface interfaceName="eth1" clusterNodeName="hasys03">
    <IPAddress baseAddress="192.168.23.103"
               subnetMask="255.255.255.0"
               networkName="db2_private_network_0"/>
  </Interface>

  <Interface interfaceName="eth1" clusterNodeName="hasys04">
    <IPAddress baseAddress="192.168.23.104"
               subnetMask="255.255.255.0"
               networkName="db2_private_network_0"/>
  </Interface>

</PhysicalNetwork>
```

```
    <!-- ================================================================= -->
    <!-- = List the computers (cluster nodes) in the cluster domain.   = -->
    <!-- ================================================================= -->
    <ClusterNode clusterNodeName="hasys01"/>
    <ClusterNode clusterNodeName="hasys02"/>
    <ClusterNode clusterNodeName="hasys03"/>
    <ClusterNode clusterNodeName="hasys04"/>

  </ClusterDomain>


  <!-- ================================================================= -->
  <!-- = The failover policy specifies the order in which the cluster  = -->
  <!-- = domain nodes should fail over.                                = -->
  <!-- ================================================================= -->
  <FailoverPolicy>
     <NPlusM />
  </FailoverPolicy>


  <!-- ================================================================= -->
  <!-- = Specify all the details of the database partitions            = -->
  <!-- ================================================================= -->
  <DB2PartitionSet>

    <DB2Partition dbpartitionnum="0" instanceName="db2inst1">
       <VirtualIPAddress baseAddress="19.126.124.250"
                         subnetMask="255.255.255.0"
                         networkName="db2_public_network_0"/>
       <Mount filesystemPath="/ha_dpf1/db2inst1/NODE0000"/>
       <Mount filesystemPath="/hafs/NODE0000"/>
       <NPlusMNode standbyNodeName="hasys03" />
    </DB2Partition>

    <DB2Partition dbpartitionnum="1" instanceName="db2inst1">
       <Mount filesystemPath="/ha_dpf1/db2inst1/NODE0001"/>
       <Mount filesystemPath="/hafs/NODE0001"/>
       <NPlusMNode standbyNodeName="hasys04" />
    </DB2Partition>

  </DB2PartitionSet>

</DB2Cluster>
```

db2ha_sample_HADR.xml:

样本文件 db2ha_sample_DPF_HADR.xml 是一个 XML 输入文件示例，您将它传递至 DB2 高可用性实例配置实用程序 (db2haicu)来指定新的*集群域*。db2ha_sample_HADR.xml 位于 sqllib/samples/ha/xml 目录中。

功能部件

db2ha_sample_HADR.xml 样本演示如何将 **db2haicu** 与 XML 输入文件配合使用来定义具有下列详细信息的集群域:

- 定额设备: 网络
- 集群中的计算机数（集群域节点数）: 两台
- 故障转移策略: HADR
- 数据库分区数: 一个
- 虚拟（服务）IP 地址数: 无

- 用于故障转移的共享安装点数: 无

XML 源代码

```
<!-- ================================================================= -->
<!-- = DB2 High Availability configuration schema                    = -->
<!-- = Schema describes the elements of DB2 High Availability        = -->
<!-- = IBM Tivoli System Automation for Multiplatforms (SA MP)       = -->
<!-- = that are used in the configuration of a HA cluster            = -->
<!-- ================================================================= -->
<DB2Cluster xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="db2ha.xsd" cluster ManagerName="TSA" version="1.0">

<!-- ================================================================= -->
  <!-- = ClusterDomain element                                       = -->
  <!-- = This element encapsulates the cluster configuration         = -->
  <!-- = specification                                               = -->
  <!-- = Creating cluster domain of name db2HAdomain                 = -->
  <!-- = Creating an IP quorum device (IP 19.126.4.5)                = -->
  <!-- = The IP must be pingable at all times by each of the nodes in  = -->
  <!-- = the cluster domain                                          = -->
<!-- ================================================================= -->
 <ClusterDomain domainName="db2HAdomain">
  <Quorum quorumDeviceProtocol="network" quorumDeviceName="19.126.4.5"/>

<!-- ================================================================= -->
  <!-- = Physical network element                                    = -->
  <!-- = The physical network specifies the network type, protocol     = -->
  <!-- = IP address, subnet mask, and NIC name                       = -->
  <!-- = Define two logical groupings of NICs                        = -->
  <!-- = Define two logical groupings of NICs                        = -->
<!-- ================================================================= -->
  <PhysicalNetwork physicalNetworkName="db2_public_network_0"
physicalNetworkProtocol="ip">
   <Interface interfaceName="eth0" clusterNodeName="hasys01">
    <IPAddress baseAddress="19.126.52.139"
 subnetMask="255.255.255.0" networkName="db2_public_network_0"/>
   </Interface>
   <Interface interfaceName="eth0" clusterNodeName="hasys02">
    <IPAddress baseAddress="19.126.52.140"
 subnetMask="255.255.255.0" networkName="db2_public_network_0"/>
   </Interface>
  </PhysicalNetwork>

  <PhysicalNetwork physicalNetworkName="db2_private_network_0"
physicalNetworkProtocol="ip">
   <Interface interfaceName="eth1" clusterNodeName="hasys01">
    <IPAddress baseAddress="192.168.23.101"
networkName="db2_private_network_0"/>
   </Interface>
   <Interface interfaceName="eth1" clusterNodeName="hasys02">
     <IPAddress baseAddress="192.168.23.102"
networkName="db2_private_network_0"/>
   </Interface>
  </PhysicalNetwork>

<!-- ================================================================= -->
  <!-- = ClusterNodeName element                                     = -->
  <!-- = The set of nodes in the cluster domain                      = -->
  <!-- = Here the defined set of nodes in the domain is              = -->
  <!-- = hasys01, hasys02                                            = -->
<!-- ================================================================= -->
  <ClusterNode clusterNodeName="hasys01"/>
  <ClusterNode clusterNodeName="hasys02"/>
 </ClusterDomain>

<!-- ================================================================= -->
```

```
  <!-- = Failover policy element                                        = -->
  <!-- = The failover policy specifies the failover order of the       = -->
  <!-- = cluster nodes                                                  = -->
  <!-- = In the current sample the failover policy is to restart       = -->
  <!-- = instance in place (LocalRestart)                               = -->
<!-- ================================================================== -->
 <FailoverPolicy>
  <HADRFailover></HADRFailover>
 </FailoverPolicy>

<!-- ================================================================== -->
  <!-- = DB2 Partition element                                          = -->
  <!-- = The DB2 partition type specifies a DB2 Instance Name,          = -->
  <!-- = partition number                                               = -->
<!-- ================================================================== -->
 <DB2PartitionSet>
  <DB2Partition dbpartitionnum="0" instanceName="db2inst1">
  </DB2Partition>
 </DB2PartitionSet>

<!-- ================================================================== -->
  <!-- = HADRDBSet                                                      = -->
  <!-- = Set of HADR Databases for this instance                        = -->
  <!-- = Specify the databaseName, the name of the local instance on    = -->
  <!-- = this machine controlling the HADR database, the name of the    = -->
  <!-- = remote instance in this HADR pair, the name of the local       = -->
  <!-- = hostname and the remote hostname for the remote instance       = -->
<!-- ================================================================== -->
 <HADRDBSet>
  <HADRDB databaseName="HADRDB" localInstance="db2inst1"
   remoteInstance="db2inst1" localHost="hasys01" remoteHost="hasys02"/>
 </HADRDBSet>
</DB2Cluster>
```

DB2 高可用性实例配置实用程序 (db2haicu) 先决条件

在使用 DB2 高可用性实例配置实用程序 (db2haicu) 之前，必须执行一组任务。

一般信息

在数据库管理器实例所有者可以运行 **db2haicu** 之前，具有 root 用户权限的用户必须运行 **preprpnode** 命令。

preprpnode 是适用于 AIX 的 Reliable Scalable Cluster Technology (RSCT) 文件集与适用于 Linux 的 RSCT 程序包的一部分。**preprpnode** 负责为集群内通信初始化节点。设置集群时，需要运行 **preprpnode** 命令。有关 preprpnode 的更多信息，请参阅:

- preprpnode 命令 (AIX)
- preprpnode 命令 (Linux)

有关 RSCT 的更多信息，请参阅 RSCT Administration Guide - What is RSCT?

此外，具有 root 用户权限的用户必须禁用 iTCO_wdt 和 iTCO_vendor_support 模块。

- 在 SUSE 上，将以下各行添加至 /etc/modprobe.d/blacklist 文件:

  ```
  alias iTCO_wdt off
  alias iTCO_vendor_support off
  ```

- 在 RHEL 上，将以下各行添加至 /etc/modprobe.conf 文件:

  ```
  blacklist iTCO_wdt
  blacklist iTCO_vendor_support
  ```

可通过使用 **lsmod** 命令来验证这些模块是否处于禁用状态。

在运行 **db2haicu** 之前，数据库管理器实例所有者必须执行下列任务:

- 在将被添加到集群的所有机器上使服务文件同步。
- 为将用于创建集群域的数据库管理器实例运行 **db2profile** 脚本。
- 使用 **db2start** 命令启动数据库管理器。

DB2 高可用性灾难恢复 (HADR)

如果将使用 HADR 功能，那么执行下列任务:

- 确保按照各自主数据库和备用数据库角色来启动所有 DB2 高可用性灾难恢复 (HADR) 数据库，并且确保所有主/备用 HADR 数据库对处于对等状态。
- 将所有 HADR 数据库的 **hadr_peer_window** 配置为至少 120 秒的值。
- 禁用 DB2 故障监视器。

分区数据库环境

如果具有多个数据库分区需要配置以获取高可用性，请执行以下步骤:

- 在将被添加到集群域的所有机器上配置 **DB2_NUM_FAILOVER_NODES** 注册表变量。
- （可选）在运行 **db2haicu** 之前激活数据库。

使用 DB2 高可用性实例配置实用程序 (db2haicu) 来创建集群域

当首次为数据库管理器实例运行 DB2 高可用性实例配置实用程序 (db2haicu) 时，**db2haicu** 会创建被称为*集群域*的集群模型。

DB2 高可用性实例配置实用程序 (db2haicu) 自动检测到的数据库路径:

首次运行 DB2 高可用性实例配置实用程序 (db2haicu) 时，**db2haicu** 将搜索数据库系统以查找与集群配置相关的数据库配置信息。

单一数据库分区环境

在单一数据库分区环境中，**db2haicu** 将自动检测下列路径:

- 实例主目录路径
- 审计日志路径
- 审计归档日志路径
- 同步点管理器 (SPM) 日志路径
- DB2 诊断日志（**db2diag** 日志文件）路径
- 数据库相关路径:
  - 数据库日志路径
  - 数据库表空间容器路径
  - 数据库表空间目录路径
  - 本地数据库目录

  注: 如果任何与数据库相关的目录为符号链接，那么 **db2haicu** 实用程序会打印错误消息并退出。

多数据库分区环境

在多数据库分区环境中，**db2haicu** 只自动检测下列路径:

- 数据库日志路径
- 数据库表空间容器路径
- 数据库表空间目录路径
- 本地数据库目录

使用 DB2 高可用性实例配置实用程序 (db2haicu) 来维护集群域

当您使用 **db2haicu** 来修改集群环境的集群域模型时，数据库管理器将相关更改传播到数据库管理器实例和集群配置。

开始之前

在可以使用 **db2haicu** 来配置集群环境之前，必须创建和配置集群域。有关更多信息，请参阅第 113 页的『使用 DB2 高可用性实例配置实用程序 (db2haicu) 来创建集群域』

关于此任务

db2haicu 维护任务包括将诸如数据库或集群节点之类的集群元素添加到集群域，并且从集群域中除去元素。**db2haicu** 维护任务还包括修改集群域元素的详细信息，例如数据库管理器实例的故障转移策略。

过程

1. 运行 **db2haicu**

   当在维护方式下运行 **db2haicu** 时，**db2haicu** 向您展示可以对集群域执行的操作的列表:

   - 添加或除去集群节点（主机名标识的机器）
   - 添加或除去网络接口（网络接口卡）
   - 添加或除去数据库分区（仅限于分区数据库环境）
   - 添加或除去 DB2 高可用性灾难恢复 (HADR) 数据库
   - 添加或除去高可用性数据库
   - 添加或除去安装点
   - 添加或除去 IP 地址
   - 添加或除去非重要路径
   - 移动数据库分区和 HADR 数据库以进行预定维护
   - 更改当前实例的故障转移策略
   - 为集群域创建新的定额设备
   - 破坏集群域
2. 选择要执行的任务并回答 **db2haicu** 提出的后续问题。

结果

数据库管理器在集群域中使用该信息来与集群管理器进行协同工作。当使用 **db2haicu** 来配置数据库和集群元素时，DB2 高可用性 (HA) 功能部件提供的集成和自动化集群配置与管理中将包含那些元素。使用 **db2haicu** 来进行数据库管理器实例配置更改时，数据

库管理器将为您执行所需的集群管理器配置更改，因此您随后不必调用集群管理器。

下一步做什么

DB2 高可用性实例配置实用程序 (db2haicu) 没有单独的诊断日志。可以使用数据库管理器诊断日志、**db2diag** 日志文件和 **db2pd** 工具来调查与诊断 **db2haicu** 错误。 有关更多信息，请参阅: 『对 DB2 高可用性实例配置实用程序 (db2haicu) 进行故障诊断』

对 DB2 高可用性实例配置实用程序 (db2haicu) 进行故障诊断

DB2 高可用性实例配置实用程序 (db2haicu) 没有单独的诊断日志。可以使用数据库管理器诊断日志、**db2diag** 日志文件和 **db2pd** 工具来调查与诊断 **db2haicu** 错误。

DB2 高可用性实例配置实用程序 (db2haicu) 限制

使用 DB2 高可用性实例配置实用程序 (db2haicu) 时存在一些限制。

- 『软件和硬件』
- 『配置任务』
- 『使用说明』
- 第 116 页的『建议』

软件和硬件

- **db2haicu** 不支持 IP V6。
- **db2haicu** 不支持除 AIX 之外的任何平台上的逻辑卷管理器 (LVM)。

配置任务

您无法使用 **db2haicu** 来执行下列任务:

- 无法使用 **db2haicu** 来配置客户机自动重新路由。
- 从 DB2 for Linux, UNIX, and Windows V9.5 升级至更新版本时，不能使用 **db2haicu** 来迁移集群配置。要迁移集群配置，必须执行下列步骤:
  1. 删除现有集群域（如果存在集群域）
  2. 升级数据库服务器
  3. 使用 **db2haicu** 创建新的集群域

使用说明

db2haicu 实用程序在 DB2 pureScale环境中不受支持。请改用 db2cluster 实用程序来配置集群环境。

当规划集群配置和管理活动时，请考虑下列 **db2haicu** 使用说明:

- 即使 **db2haicu** 可执行某些通常需要 root 用户权限的管理任务，但 **db2haicu** 仍然要使用数据库管理器实例所有者的特权来运行。尽管 **db2haicu** 只有实例所有者特权，但由 root 用户执行的 **db2haicu** 初始化仍然使它能够执行所需的配置更改。
- 创建新的集群域时，**db2haicu** 不会验证您为新集群域指定的名称是否有效。例如，**db2haicu** 不会确认该名称长度是否有效，或是否包含有效的字符，或是否与现有集群域不同名。

- **db2haicu** 不会验证或确认用户指定或传递到集群管理器的信息。因为 **db2haicu** 无法知道所有集群管理器关于集群对象名的限制，例如，**db2haicu** 将文本传递到集群管理器，而没有确认它的字符或长度等是否有效。
- 如果在创建和配置新的集群域时发生错误且 **db2haicu** 失败，那么必须执行下列步骤:
  1. 通过运行使用 **-delete** 参数的 **db2haicu** 来除去创建了一部分的集群域的资源组
  2. 通过再次调用 **db2haicu** 来重新创建新的集群域。
- 当运行带 **-delete** 参数的 **db2haicu** 时，**db2haicu** 会在未确认这些资源组是否被锁定的情况下，立即删除与当前数据库管理器实例相关联的资源组。
- 要除去与一对 DB2 高可用性灾难恢复 (HADR) 主数据库和备用数据库的数据库管理器实例相关联的资源组，请执行下列步骤:
  1. 首先针对 HADR 备用数据库的数据库管理器实例运行带 **-delete** 参数的 **db2haicu**。
  2. 另外，还针对 HADR 主数据库的数据库管理器实例运行带 **-delete** 参数的 **db2haicu**。
- 要使用 **db2haicu** 从 HADR 资源组中除去某个虚拟 IP，您必须从在其中创建此虚拟 IP 的实例中将其除去。
- 如果您使用 **db2haicu** 尝试执行的集群操作超时，那么 **db2haicu** 将不会返回错误。除非在进行 **db2haicu** 调用后查看诊断日志；或除非后续集群操作失败，且在调查后续故障时，您才能确定原始的集群操作超时，否则当集群操作超时时，您将不知道操作已超时。
- 如果尝试将给定数据库实例的故障转移策略更改为主从配置，那么存在一种该配置操作将失败的情况，但对于此情况，**db2haicu** 不会返回错误。如果将当前脱机的机器指定为*活动*机器，那么 **db2haicu** 将不会使该机器成为活动机器，但 **db2haicu** 将不会返回错误，指示该更改失败了。
- 对于共享磁盘配置，**db2haicu** 不支持嵌套式安装配置，因为 DB2 未强制实施磁盘安装顺序。
- 将网络接口卡 (NIC) 添加到网络时，不能使用 **db2haicu** 将具有不同子网掩码的 NIC 添加到同一网络。如果要将具有不同子网掩码的 NIC 添加到同一网络，请使用下列 SA MP 命令:

```
mkequ <name> IBM.NetworkInterface:<eth0>:<node0>,...,<ethN>:<nodeN>
```

建议

以下是使用 **db2haicu** 时配置集群和数据库管理器实例的建议的列表。

- 当通过将条目添加到 **/etc/fstab** 来为集群添加新的安装点时，请使用 **noauto** 选项来防止安装点自动安装在集群中的多个机器上。例如:

```
dev/vpatha1      /db/svtpdb/NODE0010        ext3  noauto 0 0
```

受支持的集群管理软件

集群管理软件使 DB2 数据库操作能从集群中一个节点上的故障主数据库转移到集群中另一节点上的辅助数据库。

DB2 数据库支持以下集群管理软件:

- IBM PowerHA SystemMirror for AIX（先前称为 High Availability Cluster Multi-Processing for AIX 或 HACMP™）

有关如何通过 DB2 数据库产品配置 PowerHA SystemMirror 的详细信息，请参阅 http://www.redbooks.ibm.com/abstracts/sg247363.html?Open。

- Tivoli System Automation for Multiplatforms。

  有关如何通过 DB2 数据库产品配置 Tivoli System Automation 的详细信息，请参阅 http://www.redbooks.ibm.com/abstracts/sg247363.html?Open。

- Microsoft Cluster Server，用于 Windows 操作系统

  有关如何通过 DB2 数据库产品配置 Microsoft Cluster Server 的详细信息，请参阅 http://www.redbooks.ibm.com/abstracts/sg247363.html?Open。

- Sun Cluster 或 VERITAS Cluster Server，用于 Solaris 操作系统。

  有关 Sun Cluster 的信息，请参阅标题为“DB2 Universal Database™ and High Availability on Sun Cluster 3.X”的白皮书，可以从 IBM Software Library web 站点（http://www.ibm.com/software/sw-library/）获得该白皮书。有关 VERITAS Cluster Server 的信息，请参阅标题为“DB2 UDB and High Availability with VERITAS Cluster Server”的白皮书，可以从“IBM 支持与下载”Web 站点 (http://www.ibm.com/support/docview.wss?uid=swg21045033) 获得该白皮书。

- 惠普公司的 Multi-Computer/ServiceGuard

IBM PowerHA SystemMirror for AIX（先前称为 High Availability Cluster Multi-Processing for AIX 或 HACMP）

IBM PowerHA SystemMirror for AIX 是集群管理软件。PowerHA SystemMirror 集群中的节点会交换称为*脉动信号*或保持活动信息包的消息。如果某个节点停止发送这些消息，那么 PowerHA SystemMirror 会在集群中的其他节点间调用故障转移；并且在修复出现故障的节点之后，PowerHA SystemMirror 会将该节点重新集成到集群中。

有两种类型的事件：一种是在 PowerHA SystemMirror 的操作内期望的标准事件，另一种是与硬件和软件组件中的参数的监视相关联的用户定义的事件。

其中一个标准事件是 node_down 事件。这是当集群中的某个节点出现故障，并且 PowerHA SystemMirror 在集群中的其他节点间启动了故障转移时发生的事件。当计划应该执行什么操作来作为恢复过程的一部分时，PowerHA SystemMirror 允许两个故障转移选项：热（或空闲）备用和相互接管。

注：当使用 PowerHA SystemMirror 时，通过使用 **db2iauto** 实用程序确保在引导时不启动 DB2 实例，如下所示：

```
db2iauto -off InstName
```

其中，*InstName* 是实例的登录名。

集群配置

在*热备用*配置中，作为接管节点的 AIX 处理器节点不运行任何其他工作负载。在*相互接管*配置中，作为接管节点的 AIX 处理器节点要运行其他工作负载。

在分区数据库环境中，对于每个节点上的数据库分区，DB2 数据库通常以相互接管的方式来运行。目录分区是热备用配置的一部分时例外。

一个计划的注意事项是如何管理大型集群。管理一个小的集群比管理一个大的集群容易得多；但是，管理一个大的集群又比管理许多小的集群容易得多。在计划过程中，要考虑在集群环境中如何使用应用程序。例如，如果有一单个的、大型的、同类应用程序在 16 个节点上运行，那么将配置作为单个集群管理可能比作为 8 个两节点集群管理更容易些。如果相同的 16 个节点包含与不同网络、磁盘和节点相关的许多不同的应用程序，那么可能最好将这些节点分组到更小的集群中。请记住，一次只能将节点集成到一个 PowerHA SystemMirror 集群中；启动多个集群的一个配置比启动一个大集群快得多。只要某个节点及其备份在同一集群中，PowerHA SystemMirror 就可以同时支持单个和多个集群。

PowerHA SystemMirror 故障转移恢复允许资源组对物理节点的预定义（也称作*级联*）赋值。该故障转移恢复过程还允许资源组对物理节点的浮动（也称作*旋转*）赋值。IP 地址和外部磁盘卷组（或文件系统、或 NFS 文件系统）以及每个资源组内的应用程序服务器指定应用程序或应用程序组件，它们可由 PowerHA SystemMirror 通过故障转移和重新集成在物理节点间进行操纵。故障转移和重新集成行为由所创建的资源组类型和该资源组中的节点数来指定。

例如，考虑一个 DB2 数据库分区（逻辑节点）。如果其日志和表空间容器位于外部磁盘上，并且其他节点与那些磁盘链接，那么其他那些节点可能可以访问这些磁盘并重新启动该数据库分区（在接管节点上）。这是 PowerHA SystemMirror 自动执行的操作类型。PowerHA SystemMirror 也可用来复原 DB2 实例主用户目录所使用的 NFS 文件系统。

在分区数据库环境中计划使用 DB2 数据库进行恢复时，完整阅读 PowerHA SystemMirror 文档。应阅读 Concepts, Planning, Installation, and Administration Guides，然后为您的环境构建恢复体系结构。对于根据已知故障点来标识的要恢复的每个子系统，标识您需要的 PowerHA SystemMirror 集群，以及恢复节点（热备用或相互接管）。

如果计划在共享同一主目录的两台或更多计算机上使用 PowerHA SystemMirror，那么必须将此数据库管理器安装在同一安装路径中。使用指向相似安装路径的符号链接可能导致此场景中的问题。安装路径必须为同一物理路径。

强烈建议在外部磁盘配置中建立磁盘和适配器的镜像。对于为 PowerHA SystemMirror 配置的 DB2 物理节点，需要小心以确保卷组上的节点可与共享外部磁盘不同。在相互接管配置中，这种方案需要某些附加计划，以便配对的节点可相互访问对方的卷组而不造成冲突。在分区数据库环境中，这表示所有容器名在所有 SMS 或 DMS 表空间的所有数据库中都必须唯一。自动存储器表空间将为您管理此需求。

实现唯一性的一种方法是将数据库分区号包括在名称中。当创建 SMS 或 DMS 容器时，可指定容器字符串语法的节点表达式。当指定表达式时，节点号可以是容器名的一部分，或者，如果指定了附加自变量，那么那些自变量的结果可以是容器名的一部分。使用自变量 ″ $N″ (blank]$N) 来指示节点表达式。自变量必须在容器字符串的末尾，并且只能使用下列其中一种格式:

表 7. 用于创建容器的自变量. 假定该节点号为 5。

语法	示例	值
blank]$N	″ $N″	5
blank]$N+ number]	″ $N+1011″	1016
blank]$N% number]	″ $N%3″	2

表 7. 用于创建容器的自变量 (续). 假定该节点号为 5。

语法	示例	值
blank]$N+ number]% number]	″ $N+12%13″	4
blank]$N% number]+ number]	″ $N%3+20″	22

注:

1. % 是模数。
2. 在所有情况下，从左往右对运算符求值。

下面是如何使用此特殊自变量来创建容器的一些示例:

- 创建在两节点系统上使用的容器。

```
CREATE TABLESPACE TS1 MANAGED BY DATABASE USING
         (device '/dev/rcont $N' 20000)
```

  将使用下列容器:

```
/dev/rcont0   - 在节点 0 上
   /dev/rcont1   - 在节点 1 上
```

- 创建在有四个节点的系统上使用的容器。

```
CREATE TABLESPACE TS2 MANAGED BY DATABASE USING
         (file '/DB2/containers/TS2/container $N+100' 10000)
```

  将使用下列容器:

```
/DB2/containers/TS2/container100   - 在节点 0 上
   /DB2/containers/TS2/container101   - 在节点 1 上
   /DB2/containers/TS2/container102   - 在节点 2 上
   /DB2/containers/TS2/container103   - 在节点 3 上
```

- 创建在两节点系统上使用的容器。

```
CREATE TABLESPACE TS3 MANAGED BY SYSTEM USING
         ('/TS3/cont $N%2, '/TS3/cont $N%2+2')
```

  将使用下列容器:

```
/TS3/cont0   - 在节点 0 上
   /TS3/cont2   - 在节点 0 上
   /TS3/cont1   - 在节点 1 上
   /TS3/cont3   - 在节点 1 上
```

为 PowerHA SystemMirror配置 DB2 数据库分区

一旦完成配置，PowerHA SystemMirror 就启动实例中的每个数据库分区，一次启动一个物理节点。建议使用多个集群来启动多于四个节点的并行 DB2 配置。注意，在 64 节点的并行 DB2 配置中，并行启动 32 个两节点 PowerHA SystemMirror 集群比启动四个 16 节点集群要快。

脚本文件与 DB2 Enterprise Server Edition 封装在一起，以帮助配置热备用或相互接管节点中的 PowerHA SystemMirror 故障转移或恢复。对于单节点，该脚本文件称为 `rc.db2pe.ee` ，对于多节点，那么称为 `rc.db2pe.eee`。它们位于 `sqllib/samples/hacmp/es` 目录中。将正确的文件复制到 PowerHA SystemMirror 集群中每个系统上的 `/usr/bin`，并将其重命名为 `rc.db2pe`。

另外，可在相互接管配置中的故障转移阶段从 rc.db2pe 内定制 DB2 缓冲池大小。（当两个数据库分区在一个物理节点上运行时，可以配置缓冲池大小以确保正确的资源分配。）

PowerHA SystemMirror 事件监视和用户定义的事件

进程在给定节点上中断时启动故障转移操作是用户定义的事件的一个示例。作为集群设置的一部分，必须将事件手动配置为用户定义的事件。

有关高可用性的 IBM DB2 数据库环境的实现和设计的详细信息，请参阅 IBM 软件库 Web 站点 (http://www.ibm.com/software/sw-library/)。

相关信息:

PowerHA SystemMirror 信息中心

IBM Tivoli System Automation for Multiplatforms（Linux 和 AIX）

IBM Tivoli System Automation for Multiplatforms（Tivoli SA MP）是集群管理软件，便于将用户、应用程序和数据从一个数据库系统自动切换至集群中的另一个系统。Tivoli SA MP 自动控制 IT 资源（如进程、文件系统和 IP 地址）。

Tivoli SA MP 提供了一个框架来自动管理资源的可用性。资源的示例包括:

- 可以通过编写其启动、监视和停止脚本来进行控制的任何软件
- Tivoli SA MP 已获授权访问的任何网络接口卡 (NIC)。也就是说，Tivoli SA MP 通过在它有权访问的 NIC 之间浮动 IP 地址来管理用户要使用的任何 IP 地址的可用性。

例如，DB2 实例和“高可用性灾难恢复”功能都有启动、停止和监视命令。因此，可以编写 Tivoli SA MP 脚本来自动管理这些资源。紧密相关的资源（例如，同时一起在同一个节点上运行的资源）称为*资源组*。

DB2 资源

在单一分区的 DB2 环境中，单个 DB2 实例正在服务器上运行。此 DB2 实例具有对数据的本地访问权（数据的可执行映像和数据库由此实例拥有）。如果要使此 DB2 实例可供远程客户机访问，那么必须为此 DB2 实例指定一个未使用的 IP 地址。

DB2 实例、本地数据和 IP 地址都属于资源，它们必须由 Tivoli SA MP 自动管理。因为这些资源紧密相关（例如，它们同时一起在同一个节点上运行），所以它们称为资源组。

整个资源组被放在集群中的一个节点上。如果进行故障转移，那么整个资源组在另一个节点上启动。

组中的资源之间存在下列依赖性:

- 必须先启动本地磁盘，然后再启动 DB2 实例
- 必须先停止 DB2 实例，然后再停止本地磁盘
- 必须将 HA IP 地址与实例配合使用

磁盘存储器

DB2 数据库可以使用下列资源作为本地数据存储器:

- 原始磁盘（例如，/dev/sda1）
- 逻辑卷管理器（LVM）管理的逻辑卷
- 文件系统（例如，ext3 和 jfs）

DB2 数据可以全部存储在一个或多个原始磁盘上，全部存储在逻辑卷上，全部存储在文件系统上或同时存储在上述三种资源上。任何可执行文件都需要驻留在某种类型的文件系统上。

DB2 数据库对 HA IP 地址的要求

DB2 数据库对于 IP 地址没有任何特殊要求。不必为了使实例具有高可用性而定义高可用的 IP 地址。但是，一定要记住，受保护的 IP 地址（如果有）是数据的客户机访问点，因此所有客户机都必须知道此地址。实际上，建议将客户机在它们的 **CATALOG TCPIP NODE** 命令中使用的那个 IP 地址作为此 IP 地址。

Tivoli SA MP 资源组

IBM Tivoli System Automation for Multiplatforms 是一个产品，它通过自动控制基于 Linux 的集群中诸如进程、应用程序和 IP 地址之类的资源和其他资源来提供高可用性。要自动控制 IT 资源（如 IP 地址），必须对 Tivoli SA MP 定义此资源。此外，这些资源还必须包含在至少一个资源组中。如果总是要求在同一台机器上主管这些资源，那么应将它们放置在相同的资源组中。

需要将每个应用程序都定义为资源，以便通过 Tivoli SA MP 进行管理和自动控制。应用程序资源通常在一般资源类 IBM.Application 中定义。在此资源类中，有几个定义资源的属性，但其中至少三个属性是特定于应用程序的:

- StartCommand
- StopCommand
- MonitorCommand

这些命令可以是脚本或二进制可执行文件。

设置 DB2 环境的 Tivoli SA MP

有关可帮助您设置 Tivoli SA MP 以使用 DB2 环境的详细配置信息，请在 IBM 软件库 Web 站点 (http://www.ibm.com/software/sw-library/) 中搜索“Tivoli System Automation”。

Microsoft 故障转移集群支持（Windows）

Microsoft 故障转移集群支持 Windows 操作系统上的服务器的集群。它自动检测并响应服务器或应用程序故障，并且可以平衡服务器工作负载。

简介

Microsoft 故障转移集群是 Windows 服务器操作系统的功能。此软件支持集群中两个服务器的连接（在 DataCenter Server 中最多为 4 个服务器）以获取数据和应用程序的高

可用性和易于管理。故障转移集群还可自动检测服务器或应用程序故障并进行恢复。它可用来转移服务器工作负载以平衡机器利用率，以及在无需停机的情况下执行规划的维护工作。

下列 DB2 数据库产品支持 Microsoft 故障转移集群:

- DB2 Connect 服务器产品（DB2 Connect Enterprise Edition、DB2 Connect Application Server Edition、DB2 Connect Unlimited Edition for iSeries® 和 DB2 Connect Unlimited Edition for zSeries®）。
- DB2 Advanced Enterprise Server Edition
- DB2 Enterprise Server Edition
- DB2 Express Edition
- DB2 Workgroup Server Edition

DB2 故障转移集群组件

集群是两个或多个节点的配置，其中每个节点都是独立的计算机系统。集群对网络客户机而言就好像是单个服务器。

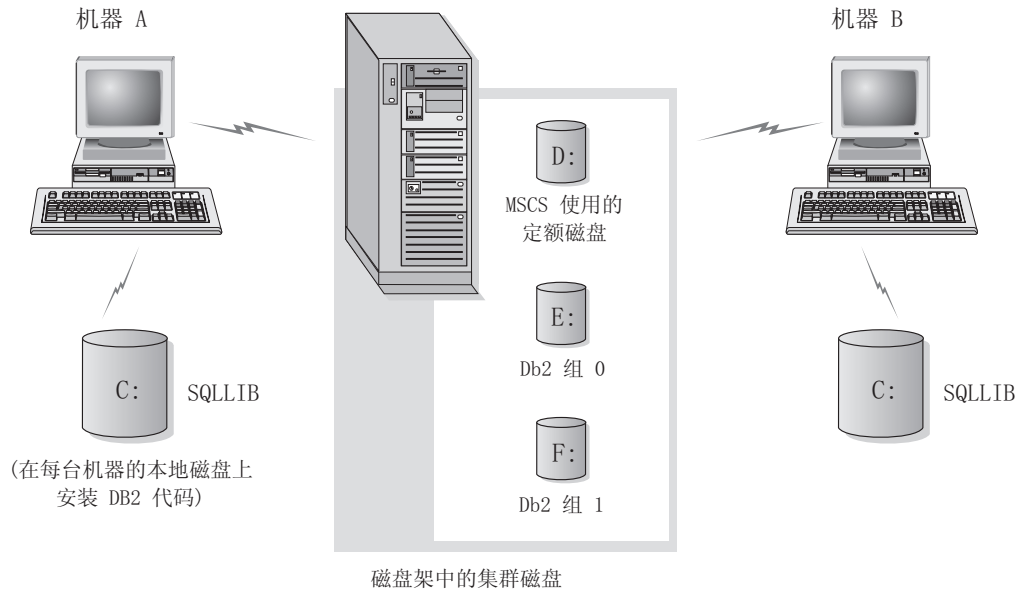


图 4. 故障转移集群配置的示例

进行故障转移集群的集群中的节点使用一个或多个共享存储器总线和一个或多个物理上独立的网络进行连接。仅将服务器与集群连接而未将客户机与集群连接的网络称为*专用*网络。支持客户机连接的网络称为*公共*网络。每个节点上有一个或多个本地磁盘。每个共享存储器总线与一个或多个磁盘连接。共享总线上的每个磁盘一次只能由集群的一个节点拥有。DB2 软件位于本地磁盘上。DB2 数据库文件（例如，表、索引和日志文件）位于共享磁盘上。因为 Microsoft 故障转移集群不支持在集群中使用原始分区，所以无法在 Microsoft 故障转移集群环境中将 DB2 配置为使用原始设备。

DB2 资源

在 Microsoft 故障转移集群环境中，资源是由集群软件管理的实体。例如，磁盘、IP 地址或类属服务可以作为资源来管理。DB2 通过创建它自己的资源类型（称为“DB2 服务

器”）来与 Microsoft 故障转移集群集成。每个“DB2 服务器”资源管理一个 DB2 实例，并且在分区数据库环境中运行时，每个“DB2 服务器”资源管理一个数据库分区。“DB2 服务器”资源的名称就是实例名，虽然在分区数据库环境中，“DB2 服务器”资源的名称由实例名和数据库分区（或节点）号组成。

联机前和联机后脚本

在将 DB2 资源联机前和联机后都可运行脚本。这些脚本分别被称为联机前和联机后脚本。联机前和联机后脚本是可以运行 DB2 和系统命令的 .BAT 文件。

在 DB2 的多个实例可以在同一机器上运行的情况下，可以使用联机前和联机后脚本来调整配置，以便可以成功启动实例。即使发生故障转移，也可以使用联机后脚本来执行手动数据库恢复。联机后脚本还可用来启动从属于 DB2 的任何应用程序或服务。

DB2 组

相关或从属资源被组织成资源组。组中的所有资源作为一个单元在集群节点之间移动。例如，在典型 DB2 单一分区集群环境中，将存在包含下列资源的 DB2 组:

1. DB2 资源。DB2 资源管理 DB2 实例（或节点）。
2. IP 地址资源。IP 地址资源允许客户机应用程序与 DB2 服务器连接。
3. “网络名”资源。“网络名”资源允许客户机应用程序通过使用名称而不使用 IP 地址与 DB2 服务器连接。“网络名”资源具有与 IP 地址资源的依赖性。“网络名”资源是可选的。（配置“网络名”资源可能影响故障转移性能。）
4. 一个或多个“物理磁盘”资源。每个“物理磁盘”资源管理集群中的一个共享磁盘。

注: DB2 资源的配置取决于同一组中的所有其他资源，因此仅当所有其他资源联机之后才能启动 DB2 服务器。

故障转移配置

可使用两种类型的配置:

- 主从
- 相互接管

在一个分区数据库环境中，集群不必全部具有相同类型的配置。您可将一些集群设置为使用主从配置，而将其他集群设置为使用相互接管配置。例如，如果 DB2 实例由五个工作站组成，那么可以将两台机器设置为使用相互接管配置，将另外两台机器设置为使用被动备用配置，而将剩余的一台机器配置为不支持故障转移。

主从配置

在主从配置中，Microsoft 故障转移集群中的一台机器提供专用的故障转移支持，而另一台机器则参与数据库系统。如果参与该数据库系统的机器发生故障，那么该机器上的数据库服务器将在故障转移机器上启动。如果在一个分区数据库环境中，您正在一台机器上运行多逻辑节点，而该机器发生故障，那么这些逻辑节点将在故障转移机器上启动。第 124 页的图 5 显示了一个主从配置的示例。

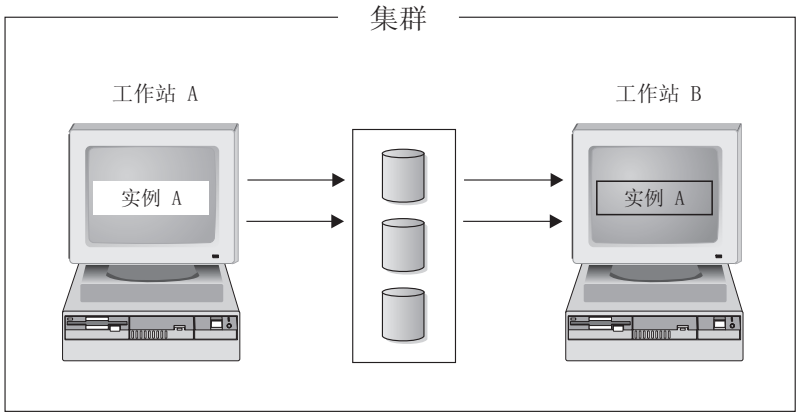


图 5. 主从配置

相互接管配置

在一个相互接管配置中，两个工作站都参与数据库系统（即，每台机器上至少有一个数据库服务器在运行）。如果 Microsoft 故障转移集群中的一个工作站发生故障，那么发生故障的机器上的数据库服务器将在另一台机器上启动以运行。在一个相互接管配置中，一台机器上的数据库服务器发生故障时不会影响另一台机器上的数据库服务器。在任何给定的时间点，任何数据库服务器可在任何机器上处于活动状态。图 6 显示了一个相互接管配置的示例。

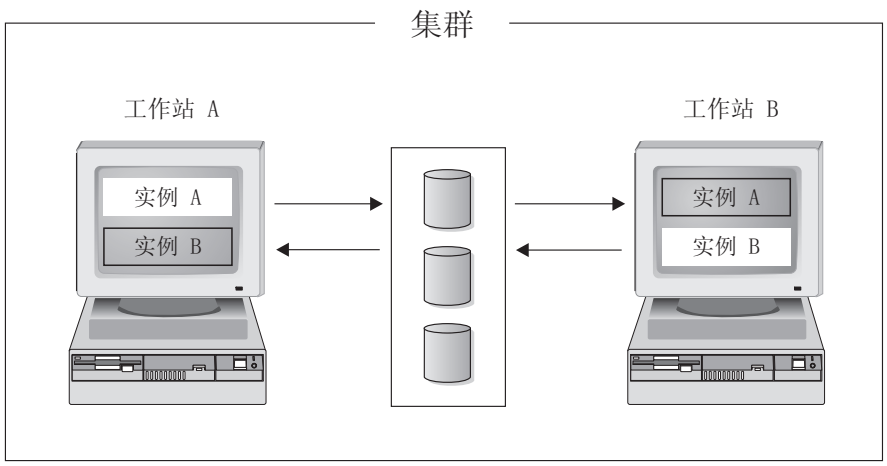


图 6. 相互接管配置

有关 Windows 操作系统上高可用性的 IBM DB2 数据库环境的实现和设计的详细信息，请访问 IBM Software Library Web 站点 (http://www.ibm.com/software/sw-library/)。

Windows Server 2008 故障转移集群支持

要将分区的 DB2 数据库系统配置为在 Windows Server 2008 故障转移集群上运行，请执行以下操作:

1. 遵循白皮书“Implementing IBM DB2 Universal Database V8.1 Enterprise Server Edition with Microsoft Cluster Server”中描述的相同过程，此白皮书可从 IBM Software Library Web 站点 (http://www.ibm.com/software/sw-library/) 获得。

2. 因为 Windows Server 2008 的故障转移集群功能部件的变化，可能需要以下附加设置:

   - 在 Windows Server 2008 故障转移集群中，Windows 集群服务在特殊“本地系统”帐户下运行，而在 Windows Server 2003 中，Windows 集群服务在管理员帐户下运行。这会影响 DB2 资源 (db2server.dll) 的操作，此资源在集群服务帐户的上下文下运行。

     如果 **DB2_EXTSECURITY** 注册表变量在 Windows 故障转移集群上设置为 YES，那么 DB2ADMNS 和 DB2USERS 组必须为域组。

     多分区实例在 Windows 故障转移集群上运行时，INSTPROF 路径必须设置为网络路径（例如，*NetName*\DB2MSCS-DB2\DB2PROFS）。如果您使用 **db2mscs** 命令来集群 DB2 数据库系统，那么此操作将自动完成。

     Windows Server 2008 故障转移集群形成时，将在 Active Directory 中创建表示新集群的计算机对象。例如，如果集群的名称为 MYCLUSTER，那么将在 Active Directory 中创建计算机对象 MYCLUSTER。如果用户集群多分区实例并且 **DB2_EXTSECURITY** 注册表变量设置为 YES（这是缺省设置），那么必须将此计算机对象添加至 DB2ADMNS 组。必须执行此操作以便 DB2 资源 DLL 可访问 *NetName*\DB2MSCS-DB2\DB2PROFS 路径。例如，如果 DB2 管理员组为 MYDOMAIN\DB2ADMNS，那么必须将计算机对象 MYCLUSTER 添加至此组。最后，将计算机对象添加到 DB2ADMNS 组之后，必须在集群中重新引导这两个节点。

   - 在 Windows Server 2008 故障转移集群中，“集群文件共享资源”不再受支持。已改为使用集群文件服务器。文件共享（常规文件共享）将以集群文件服务器资源为基础。Microsoft 要求集群中创建的集群文件服务器使用域名系统 (DNS) 来解析名称。运行多分区实例时，需要文件服务器资源来支持文件共享。db2mscs.cfg 文件中指定的 **NETNAME_NAME**、**NETNAME_VALUE** 和 **NETNAME_DEPENDENCY** 参数的值用于创建文件服务器和文件共享资源。*NetName* 基于 IP 地址，并且此 *NetName* 必须在 DNS 中。例如，如果 db2mscs.cfg 文件包含以下参数，那么会创建文件共享 \\MSCSV\DB2MSCS-DB2:

     ```
     ...
     NETNAME_NAME = MSCSN
     NETNAME_VALUE = MSCSV
     ...
     ```

     必须在 DNS 中注册名称 MSCSV。否则，为 DB2 集群创建的文件服务器或文件共享会在 DNS 解析失败时失败。

Solaris 操作系统集群支持

DB2 支持下列两个可用于 Solaris 操作系统的集群管理器: Sun Cluster 和 Veritas Cluster Server (VCS)。

注: 当使用 Sun Cluster 3.0 或 Veritas Cluster Server 时，确保在引导时通过使用 **db2iauto** 实用程序不启动 DB2 实例，如下所示:

```
db2iauto -off InstName
```

其中，*InstName* 是实例的登录名。

高可用性

提供数据服务的计算机系统包含许多不同的组件，每个组件都有一个关联的“平均无故障时间”(MTBF)。MTBF 是组件保持可用的平均时间。高质量硬盘驱动器的 MTBF 大约是一百万小时（大约 114 年）。虽然这看起来像是很长一段时间，但每 200 个磁盘中，就有一个有可能在 6 个月内发生故障。

虽然有很多方法可以提高数据服务的可用性，但最常用的方法还是 HA 集群。当用于高可用性时，集群由两台或更多机器、一组专用网络接口、一个或多个公用网络接口以及一些共享磁盘组成。这种特殊的配置允许将数据服务从一台机器移至另一机器。通过将数据服务移至集群中的另一机器，应该能够继续提供对其数据的访问。将数据服务从一台机器移至另一机器称为*故障转移*，图 7 对其进行了说明。

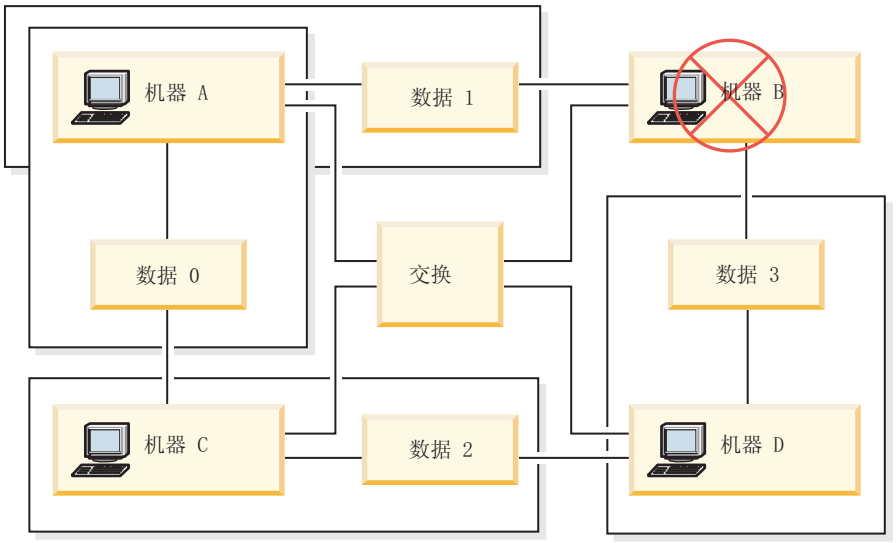


图 7. 故障转移. 当机器 B 故障时，它的数据服务被移到集群中的另一台机器上，以便仍可访问这些数据。

专用网络接口用来在集群中的机器之间发送*脉动信号*消息和控制消息。公用网络接口用来直接与 HA 集群的客户机通信。HA 集群中的磁盘与集群中的两台或更多机器相连，因此，当一台机器发生故障时，另一机器可以对它们进行访问。

在 HA 集群上运行的数据服务有一个或多个逻辑公用网络接口和一组磁盘与其关联。HA 数据服务的客户机只通过 TCP/IP 与该数据服务的逻辑网络接口相连。如果执行故障转移，那么该数据服务及其逻辑网络接口和磁盘组便移至另一机器。

HA 集群的其中一个好处是可以在没有支持人员辅助的情况下恢复数据服务，并且，可以随时这样做。另一个好处是冗余。集群中的所有部件都应该是冗余的，包括机器本身。集群应当能够经受住任何单个故障点。

虽然高度可用的数据服务在本质上可以有非常大的区别，但它们还是有一些公共的要求。高度可用数据服务的客户机期望该数据服务的网络地址和主机名保持不变，并期望能够以同一方式发出请求，而不考虑该数据服务具体是在哪一台机器上。

假定一个 Web 浏览器，它正在访问高可用的 Web 服务器。请求是使用 URL（统一资源定位符）发出的，该 URL 包含主机名和指向 Web 服务器上的文件的路径。浏览器期望该主机名和路径在 Web 服务器的故障转移之后保持不变。如果浏览器从 Web 服务器下载文件，而该服务器正在进行故障转移，那么该浏览器将需要重新发出请求。

数据服务的可用性是通过数据服务可供其用户使用的时间量测量的。最常用的可用性计量单位是“运行时间”的百分比；此百分比通常用数字“9”来表示:

```
99.99% => 服务（最多）停止 52.6 分钟/年
99.999% => 服务（最多）停止 5.26 分钟/年
99.9999% => 服务（最多）停止 31.5 秒/年
```

当设计和测试 HA 集群时:

1. 确保集群管理员熟悉系统并了解发生故障转移时应发生的情况。
2. 确保集群的每一部分都确实是冗余的，当它发生故障时，可以被快速替换。
3. 强制一个测试系统在受控环境中发生故障，并确保它每次都能正确地进行故障转移。
4. 跟踪每次故障转移的原因。虽然这应该不会经常发生，但找出任何导致集群不稳定的问题非常重要。例如，如果集群的一段一个月内导致了 5 次故障转移，那么找出原因并将它解决。
5. 当发生故障转移时，务必通知该集群的支持人员。
6. 不能使集群超负荷。确保其余的系统在发生故障转移之后，仍能够处理适量的工作负载。
7. 经常检查容易引起故障的组件（如磁盘），以便可以在发生问题之前进行替换。

容错

另一种提高数据服务可用性的方法是故障容错。*故障容错*机器内置了其所有冗余，应该能够经受住任何部件（包括 CPU 和内存）的单一故障。故障容错机器最常用在小型市场中，其实现成本比较高。机器位于不同地理位置的 HA 集群还有一个优点，即可以从仅影响那些位置的一部分的灾难中恢复。

由于 HA 集群可伸缩、易于使用以及实现成本相对较低，所以它是最常见的提高可用性的解决方案。

Sun Cluster 3.0（和更高版本）支持:

如果您打算在 Solaris 操作系统集群上运行 DB2 数据库解决方案，那么可以使用 Sun Cluster 3.0 来管理该集群。高可用性代理程序充当 DB2 数据库与 Sun Cluster 3.0 之间的介体。

在关于 Sun Cluster 3.0 支持的此主题中所做的声明也适用于 3.0 以上更高版本的 Sun Cluster。

图 8. DB2 数据库、Sun Cluster 3.0 和高可用性. DB2 数据库、Sun Cluster 3.0 与高可用性代理程序之间的关系。

故障转移

Sun Cluster 3.0 通过启用应用程序故障转移来提供高可用性。定期监视每个节点，集群软件自动将集群感知应用程序从失败的主节点重定位至指定的辅助节点。当故障转移发生时，客户机可能遇到短时间中断服务的情况，可能必须重新连接至服务器。但是，它们将不知道它们正从其中访问应用程序和数据的物理服务器。通过允许主节点故障时由集群中的其他节点自动主管工作负载，Sun Cluster 3.0 可以显著地减少停机时间并提高工作效率。

多主机磁盘

Sun Cluster 3.0 需要多主机磁盘存储器。这表示磁盘一次可与多个节点连接。在 Sun Cluster 3.0 环境中，多主机存储器能使磁盘机变得高可用。位于多主机存储器上的磁盘机允许单个节点故障，因为通过备用服务器节点，仍然有一条物理路径指向数据。通过主节点可以全局访问多主机磁盘。如果客户机请求正通过一个节点来访问数据，且该节点故障，那么将这些请求转换至具有与相同磁盘的直接连接的另一个节点。卷管理器为多主机磁盘的数据冗余提供镜像或 RAID 5 配置。通常，Sun Cluster 3.0 支持 Solstice DiskSuite 和 VERITAS Volume Manager 作为卷管理器。将多主机磁盘与磁盘镜像和组合分割区组合在一起防止节点故障和各个磁盘故障。

全局设备

使用全局设备，可以提供从任何节点对集群范围内的集群中的任何设备的高可用访问，而不必考虑设备的物理位置。所有磁盘都包括在带有指定设备标识 (DID) 的全局名称空间中，并且被配置为全局设备。因此，从所有集群节点上看，磁盘本身是可视的。

文件系统和全局文件系统

集群或全局文件系统是内核（在一个节点上）与底层文件系统卷管理器（在具有与一个或多个磁盘的物理连接的节点上）之间的代理。集群文件系统依赖于具有与一个或多个节点的物理连接的全局设备。它们与底层文件系统和卷管理器无关。通常，可以通过使用 Solstice DiskSuite 或 VERITAS Volume Manager 在 UFS 上构建集群文件系统。仅当磁盘上的文件系统整体上作为集群文件系统装上时，数据才变得对所有节点都可用。

设备组

所有多主机磁盘都必须受控于 Sun Cluster 框架。由 Solstice DiskSuite 或 VERITAS Volume Manager 管理的磁盘组首先是在多主机磁盘上创建的。然后，将它们注册为 Sun Cluster 磁盘设备组。磁盘设备组是一种类型的全局设备。多主机设备组是高可用的。如果当前管理设备组的节点失败，那么可以通过备用通道访问磁盘。管理设备组的节点的故障不影响对设备组的访问，但执行恢复和一致性校验所需的时间除外。在此时间段，所有请求停止（对应用程序是透明的），直到系统使设备组可用为止。

资源组管理器 (RGM)

RGM 提供高可用性的机制并在每个集群节点上作为守护程序运行。它根据预先配置策略在选择的节点上自动启动和停止资源。即使在发生节点故障的情况下，RGM 允许资源高可用，或通过在受影响的节点上停止并在另一个节点上启动它来重新引导。RGM 还会自动启动和停止特定于资源的监视器，以便检测资源故障并将失败的资源重定位至另一个节点。

数据服务

术语“数据服务”用来描述已配置为在集群而不是单个服务器上运行的第三方应用程序。数据服务包括应用软件和启动、停止以及监视该应用程序的 Sun Cluster 3.0 软件。Sun Cluster 3.0 提供用来控制和监视集群内的应用程序的数据服务方法。这些方法在“资源组管理器”(RGM) 的控制下运行，它使用这些方法来启动、停止和监视集群节点上的应用程序。这些方法与集群框架软件和多主机磁盘一起，使应用程序成为高可用的数据服务。作为高可用的数据服务，它们可以在集群内任何单个故障后防止重要的应用程序中断，而不考虑故障是在节点上、接口组件上还是在应用程序本身中发生的。RGM 还管理集群中的资源，包括网络资源（逻辑主机名和共享地址）和应用程序实例。

资源类型、资源和资源组

资源类型是由下列部分组成:

1. 要在集群上运行的软件应用程序。
2. RGM 用作回调方法以管理作为集群资源的应用程序的控制程序。
3. 组成集群的静态配置的部分的一组属性。

RGM 使用资源类型属性来管理特定类型的资源。

资源继承其资源类型的属性和值。这是在集群上运行的基础应用程序的实例。每个实例在集群内都需要有唯一的名称。每个资源都必须在资源组中配置。RGM 将同一节点上的某个组的所有资源同时联机和脱机。当 RGM 将资源组联机或脱机时，它对该组中的各个资源调用回调方法。

资源组当前在其上联机的节点被其主节点或主体调用。资源组由它的每一个主体管理。每个资源组都有相关联的 Nodelist 属性，由集群管理员设置，以指定资源组的所有潜在主体或主方。

VERITAS Cluster Server 支持:

如果您打算在 Solaris 操作系统集群上运行 DB2 数据库解决方案，那么可以使用 VERITAS Cluster Server 来管理该集群。

VERITAS Cluster Server 可以管理异种环境中的大量应用程序；并且在存储区域网络（SAN）和传统客户机/服务器环境中，它最多支持 32 个节点集群。

硬件要求

以下内容是当前受 VERITAS Cluster Server 支持的硬件的列表:

- 对于服务器节点:
  - Sun Microsystems 提供的运行 Solaris 2.6 或更新版本的任何 SPARC/Solaris 服务器，RAM 不得低于 128 MB。
- 对于磁盘存储器:
  - EMC Symmetrix、IBM Enterprise Storage Server®、HDS 7700 和 9xxx、Sun T3、Sun A5000、Sun A1000、Sun D1000 以及 VCS 2.0 或更新版本支持的任何其他磁盘存储器；VERITAS 代表可以确认哪些磁盘子系统是受支持的，您也可以参阅 VCS 文档。
  - 典型环境将需要镜像专用磁盘（在每个集群节点中）以用于 DB2 二进制文件和 DB2 数据的节点之间的共享磁盘。
- 对于网络互连:
  - 对于公用网络连接，支持基于 IP 寻址的任何网络连接。

– 对于脉动信号连接（集群内部），冗余脉动信号连接是必需的；这种要求可通过每个服务器使用两个附加以太网控制器或者每个服务器使用一个附加以太网控制器且每个集群使用一个共享 GABdisk 来满足

软件要求

下列 VERITAS 软件组件是限定配置:

- VERITAS Volume Manager 3.2 或更新版本、VERITAS File System 3.4 或更新版本以及 VERITAS Cluster Server 2.0 或更新版本。
- DB Edition for DB2 Solaris 版 1.0 或更新版本。

当 VERITAS Cluster Server 不需要卷管理器时，强烈建议使用 VERITAS Volume Manager 来轻松安装、配置和管理。

故障转移

VERITAS Cluster Server 是一个可用性集群解决方案，它通过启用应用程序故障转移来管理应用程序服务（如 DB2 数据库）的可用性。会定期监视每个单独的集群节点的状态及其相关联的软件服务通常。当发生中断应用程序服务（在此情况下是 DB2 数据库服务）的故障时，VERITAS Cluster Server 和/或 VCS HA-DB2 代理程序将检测到该故障并自动执行一些步骤来复原该服务。用来复原该服务的步骤包括在同一节点上重新启动 DB2 数据库系统，或者将 DB2 数据库系统移至集群中的另一个节点并在该节点上将其重新启动。如果需要将应用程序迁移到新节点，那么 VERITAS Cluster Server 将与该应用程序相关联的一切（即，网络 IP 地址和基础存储器的所有权）移至新节点，以使用户不会意识到实际上服务在另一个节点上运行。他们将仍然通过使用相同的 IP 地址来访问该服务，但是这些地址现在将指向另一个集群节点。

当 VERITAS Cluster Server 发生故障转移时，用户可能看到也可能看不到服务中断。这将取决于客户机与应用程序服务的连接的类型（有状态或无状态）。在带有有状态连接的应用程序环境（如 DB2 数据库）中，用户可能会发现服务短时间中断，并且在故障转移完成之后可能需要重新连接。在带有无状态连接的应用程序环境（如 NFS）中，用户可能发现服务的短时间延迟，但通常将不会发现中断并且不需要再次登录。

通过支持应用程序作为可以在集群节点之间自动迁移的服务，VERITAS Cluster Server 不仅可以减少未计划的停机时间，而且还可以缩短与计划停机时间（维护和升级）相关联的运行中断持续时间。还可以手动启动故障转移。如果硬件或操作系统升级必须在特定节点上执行，那么 DB2 数据库系统可以迁移至集群中的另一个节点，并且可以执行更新，然后可以将 DB2 数据库系统迁移回原始节点。

建议在这些类型的集群环境中使用的应用程序应允许崩溃。允许崩溃的应用程序在仍然保持落实数据的完整性的同时可以从意外崩溃中恢复。允许崩溃的应用程序有时被称为*集群友好*应用程序。DB2 数据库系统是允许崩溃的应用程序。

共享存储器

与“VCS HA-DB2 代理程序”配合使用时，Veritas Cluster Server 需要共享存储器。共享存储器是具有与集群中的多个节点的物理连接的存储器。位于共享存储器上的磁盘机允许节点故障，因为通过一个或多个备用集群节点，指向磁盘机的物理路径仍然存在。

在 VERITAS Cluster Server 的控制下，集群节点可以通过称为“磁盘组”的逻辑构造来访问共享存储器。磁盘组表示逻辑定义的存储设备的集合，这些存储设备的所有权是可在集群中的节点之间自动迁移。在给定时间，只能将磁盘组导入至单个节点。例如，如果“磁盘组 A”导入至“节点 1”并且“节点 1”发生故障，那么可以从故障节点导出“磁盘组 A”，并将它导入至集群中的另一节点。VERITAS Cluster Server 可以同时控制单个集群中的多个磁盘组。

除了允许磁盘组定义之外，卷管理器还可以通过使用镜像或 RAID 5 在共享存储器上提供冗余数据配置。VERITAS Cluster Server 支持 VERITAS Volume Manager 和 Solstice DiskSuite 作为逻辑卷管理器。将共享存储器与磁盘镜像和分割组合区组合在一起以防止节点故障和各个磁盘或控制器故障。

VERITAS Cluster Server 全局原子广播 (GAB) 和低等待时间传输 (LLT)

集群配置中需要节点间的通信机制，以便节点可以根据硬件和软件状态交换信息，记录集群成员信息并使所有集群节点上的此信息同步。通过低等待时间传输 (LLT) 运行的“全局原子广播”(GAB) 工具提供由 VERITAS Cluster Server 使用的高速、低等待时间机制来完成此任务。GAB 被作为内核模块装入到每个集群节点上，并提供原子广播机制，以确保所有节点同时获取状态更新信息。

通过利用内核间通信功能，LLT 对需要在集群节点间交换和同步的所有信息提供高速和低等待时间传输。GAB 在 LLT 顶部运行。VERITAS Cluster Server 不将 IP 用作脉动信号机制，但提供两个其他可靠选项。带有 LLT 的 GAB 可配置为充当脉动信号机制，或者可将 GABdisk 配置为基于磁盘的脉动信号。脉动信号必须通过冗余连接运行。这些连接可以是集群节点间两个专用以太网连接或一个专用以太网连接和一个 GABdisk 连接。使用两个 GABdisks 的配置不受支持，因为节点间集群状态的交换需要专用以太网连接。

有关 GAB 或 LLT 或有关如何在 VERITAS Cluster Server 配置中配置它们的更多信息，请参阅 VERITAS Cluster Server 2.0 User’s Guide for Solaris。

捆绑代理程序和企业代理程序

代理程序是用来管理特定资源或应用程序的可用性的程序。当启动代理程序时，它从 VCS 获取必要的配置信息，然后定期监视资源或应用程序并用状态来更新 VCS。通常，使用代理程序来使资源联机、使资源脱机、或监视资源，同时提供四种类型的服务：启动、停止、监视和清除。启动和停止用来使资源联机或脱机，监视用来测试特定资源或应用程序以了解它的状态，而在恢复过程中使用清除。

将各种捆绑代理程序包括为 VERITAS Cluster Server 的一部分，并在安装 VERITAS Cluster Server 时安装这些代理程序。捆绑代理程序是这样的 VCS 进程，它们管理通常在集群配置中发现的预定义资源类型（即，IP、安装、处理和共享），并且它们有助于极大地简化集群安装和配置。超过 20 个捆绑代理程序与 VERITAS Cluster Server 捆绑在一起。

企业代理程序想要将重点放在特定应用程序（例如，DB2 数据库应用程序）上。“VCS HA-DB2 代理程序”可视作“企业代理程序”，它通过 VCS Agent 框架与 VCS 交互。

VCS 资源、资源类型和资源组

资源类型是用来定义 VCS 集群中将要监视的资源的对象定义。资源类型包括资源类型名和一组与该资源相关联的属性，这些属性从高可用性方面来看是非常重要的。资源继承其资源类型的属性和值，且资源名称在集群范围内必须是唯一的。

有两种类型的资源：持久和标准（非持久）。持久资源是如网络接口控制器 (NIC) 之类的资源，它们受监视但未被 VCS 联机或脱机。标准资源是其联机和脱机状态受 VCS 控制的资源。

受监视的最低级别对象是资源，有各种资源类型（即，共享和安装）。每个资源必须配置到资源组中，且 VCS 将使特定资源组中的所有资源同时联机和脱机。要使资源组联机或脱机，VCS 将调用启动或停止方法以用于组中每个资源。有两种类型的资源组：故障转移和并行。高可用的 DB2 数据库配置，不论它是否为分区数据库环境，都将使用故障转移资源组。

“主体”或“主方”节点是可以潜在主管资源的节点。名为 systemlist 的资源组属性用来指定集群中哪些节点可以是特定资源组中的主体。在双节点集群中，通常两个节点都包括在 systemlist 中，但是在主管几个高可用的应用程序的较大多节点集群中，可能需要确保某些应用程序服务（由它们的资源在最低级别定义）永远不可能故障转移至某些节点。

可在资源组之间定义依赖性，在评估各种资源故障和管理恢复时，VERITAS Cluster Server 从属于此资源组依赖性层次结构。例如，如果不能使资源组 ClientApp1 联机（除非资源组 DB2 已成功启动），资源组 ClientApp1 将被视作从属于资源组 DB2。

使分区数据库环境中的时钟同步

应该使所有数据库分区服务器的系统时钟保持相对同步，以确保数据库操作顺利进行以及正向可恢复性不受限制。

数据库分区服务器之间的时差加上事务的任何潜在操作和通信延迟，应小于对 *max_time_diff*（节点之间的最大时差）数据库管理器配置参数指定的值。

为确保日志记录时间戳记反映分区数据库环境中的事务顺序，DB2 使用每台机器上的系统时钟以及文件 SQLOGCTL.LFH 中存储的虚拟时间戳记作为日志记录时间戳记的基准。但是，如果将系统时钟设置得提前，就会自动将日志时钟设置得提前。虽然可以将系统时钟往后调，但是日志时钟却不能这样设置，它会保持*相同的*超前时间，直至系统时钟与此时间匹配为止。于是，这两个时钟便同步了。这意味着一个数据库节点上的短期系统时钟错误可能会对数据库日志时间戳记产生长期的影响。

例如，假定数据库分区服务器 A 上的系统时钟被错误地设置为 2005 年 11 月 7 日，而当前年份是 2003 年，并假定在该数据库分区服务器上的数据库分区中落实了更新事务之后更正了此错误。如果继续使用该数据库，并且随着时间的推移定期对其进行更新，那么 2003 年 11 月 7 日至 2005 年 11 月 7 日之间的任何时间点实际上是无法通过前滚恢复到达的。当数据库分区服务器 A 上的 COMMIT 完成时，数据库日志中的时间戳记被设置为 2005，而日志时钟会停留在 2005 年 11 月 7 日，直到系统时钟与此时间相匹配为止。如果尝试前滚到这个时间范围内的某个时间点，那么操作将在指定停止点后的第一个时间戳记（即 2003 年 11 月 7 日）处停止。

虽然 DB2 无法控制对系统时钟的更新，但是 *max_time_diff* 数据库管理器配置参数减少了发生此类问题的机会：

- 此参数的可配置值的范围是 1 分钟至 24 小时。
- 当对非目录分区发出第一个连接请求时，数据库分区服务器会将它的时间发送至该数据库的目录分区。该目录分区就会检查请求连接的数据库分区上的时间与它自己的时间是否在 *max_time_diff* 参数指定的范围之内。如果超出此范围，那么拒绝该连接。
- 涉及到数据库中两个以上数据库分区服务器的更新事务，必须先验证参与数据库分区服务器上的时钟是否同步，然后才可落实该更新。如果两个或更多个数据库分区服务器的时差超出 *max_time_diff* 允许的限制，那么会回滚该事务，以防止将不正确的时间传播至其他数据库分区服务器。

客户机/服务器时间戳记转换

时间戳记转换可帮助您在数据库中维护活动的准确记录。它允许您按当地时间查看采用 GMT 时区格式记录的活动，即使数据库服务器处于另一时区的远程位置也是如此。

时间戳记对于进行审计非常重要。在分区数据库环境中，保持所有数据分区的时间戳记的完整性非常重要。

本节说明客户机/服务器环境中时间戳记的生成:

- 如果对前滚操作指定了本地时间，那么返回的所有消息也以本地时间表示。

  注: 在服务器上和（在分区数据库环境中）目录数据库分区上所有时间都会转换。
- 在服务器上时间戳记字符串转换为 GMT，所以时间表示服务器的时区而不是客户机的时区。如果客户机与服务器在不同的时区，那么应该使用服务器的本地时间。
- 如果时间戳记字符串接近由夏令时导致的时间更改，那么了解停止时间是在时间更改之前还是之后是很重要的，这样才能正确地将它指定。

第 5 章 管理和维护高可用性解决方案

一旦创建、配置和开始运行 DB2 数据库高可用性解决方案后，您必须执行一些后续活动。您必须监视、维护和修复数据库解决方案，使它保持对客户机应用程序可用。

过程

当数据库系统运行时，需要监视和响应下列各种情况:

1. 管理日志文件。

   日志文件越来越大，需要归档；有些日志文件需要复制和移动，以便可用于复原操作。
2. 执行维护活动:
   - 安装软件
   - 升级硬件
   - 重组数据库表
   - 数据库性能调整
   - 数据库备份
3. 同步主数据库和辅助或备用数据库，以便故障转移平稳进行。
4. 标识和响应意外的硬件或软件故障。

日志文件管理

DB2 数据库管理器使用编号方案来命名日志文件。此命名策略会影响日志文件复用和日志序列。此外，没有客户机应用程序连接的 DB2 数据库在下一个客户机应用程序连接至该数据库服务器时会使用新的日志文件。

DB2 Data Server 数据库日志记录行为的这两个方面，对于您在日志文件管理上进行选择会有影响。

管理数据库日志时，请考虑以下事项:

- 归档日志的编号方案以 S0000000.LOG 开始，直到 S9999999.LOG，符合日志文件的最大潜在大小 10000000。如果发生以下情况，数据库管理器将复位到 S0000000.LOG:
  - 数据库配置文件更改为启用前滚恢复
  - 数据库配置文件更改为*禁用*前滚恢复
  - 已使用了 S9999999.LOG。

  在复原数据库之后（执行或未执行前滚恢复），DB2 数据库管理器复用日志文件名。数据库管理器会确保在前滚恢复期间不会应用错误的日志。在复原操作后，如果 DB2 数据库管理器复用日志文件名，那么将新的日志文件归档到独立的目录，以便可以归档多个同名的日志文件。日志文件的位置记录在恢复历史记录文件中，以便在前滚恢复期间应用这些位置记录。必须确保前滚恢复可以找到正确的日志。

  前滚操作成功完成后，使用的最后一个日志会截断，而日志记录会从下一个按顺序的日志开始。日志路径目录中，序号大于用于前滚恢复的最后一个日志的任何日志

© Copyright IBM Corp. 2001, 2013

都将复用。截断的日志中，跟在截断点后的任何条目都将用 0 覆盖。确保在调用 ROLLFORWARD 实用程序前建立了日志的副本。（可以调用用户出口程序将日志复制到另一位置。）

- 如果某个数据库尚未激活（通过 **ACTIVATE DATABASE** 命令），那么 DB2 数据库管理器会在所有应用程序已从该数据库断开连接后截断当前日志文件。下次有应用程序与该数据库连接时，DB2 数据库管理器开始将日志记录到一个新日志文件中。如果系统上产生了很多小的日志文件，则可能需要考虑使用 **ACTIVATE DATABASE** 命令。使用该命令不仅能节省应用程序连接时初始化数据库所用的开销，还可节省分配大日志文件、截断它然后再分配新的大日志文件所用的开销。
- 因为日志文件名是重复使用的（请参阅图 9），所以归档日志可能会与某个数据库的两个或多个不同的*日志序列*相关。例如，如果您要恢复“备份 2”，可以使用两种可能的日志序列。如果在完整的数据库恢复期间前滚至某个时间点，然后在到达日志末尾前停止，那么您已创建了一个新的日志序列。两个日志序列不能合在一起。如果联机备份映像跨越了第一个日志序列，那么必须使用此日志序列来完成前滚恢复。

  如果在恢复之后创建了新的日志序列，那么旧日志序列中的所有表空间备份映像都是无效的。这通常是在复原时识别的，但是如果在数据库复原操作之后立即执行表空间复原操作，那么 RESTORE 实用程序无法识别旧日志序列上的表空间备份映像。在实际上前滚了数据库前，将使用的日志序列都是未知的。如果表空间在旧日志序列上，它一定会由表空间前滚操作“捕获”。使用无效备份映像的复原操作可能会成功完成，但该表空间的表空间前滚操作将失败，而表空间将仍处于复原暂挂状态。

  例如，假设一个表空间级的备份操作“备份 3”在顶级日志序列的 S0000013.LOG 和 S0000014.LOG 之间完成（请参阅图 9）。如果要使用数据库级别的备份映像“备份 2”进行复原和前滚，您将需要前滚 S0000012.LOG。然后，可以继续前滚到顶端日志序列或（较新的）底端日志序列。如果前滚底端日志序列，将不能使用表空间级别的备份映像“备份 3”来执行表空间复原和前滚恢复。

  要使用表空间级别的备份映像“备份 3”完成至日志末尾的表空间前滚操作，需要复原数据库级别的备份映像“备份 2”，然后使用顶端日志序列前滚。一旦复原了表空间级的备份映像“备份 3”，就可以启动至日志末尾的前滚操作。

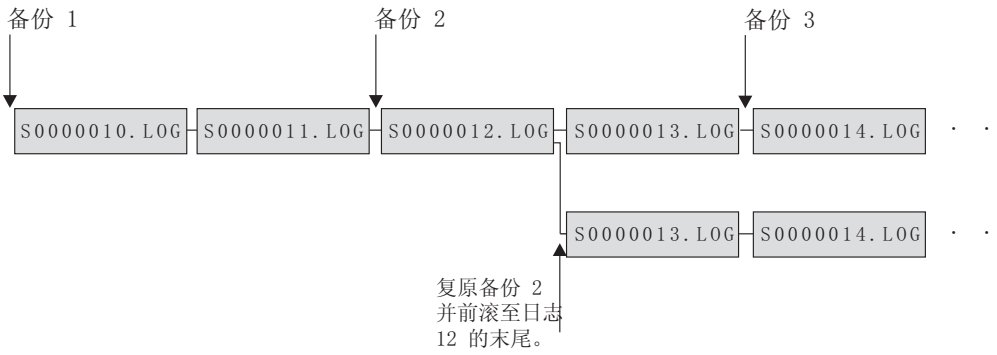


图 9. 复用日志文件名

随需应变日志归档

IBM DB2 服务器支持随时对可恢复数据库关闭活动日志（如果已启用活动日志，那么支持归档活动日志）。因此您可以到某个已知点为止的一组完整的日志文件，然后使用这些日志文件来更新一个备用数据库。

可通过调用 **ARCHIVE LOG** 命令或 db2ArchiveLog API 来启动随需应变日志归档。

使用 db2tapemgr 来进行日志归档

可以使用 **db2tapemgr** 实用程序将已归档的日志文件存储到磁带设备。**db2tapemgr** 实用程序将日志文件从磁盘复制到指定的磁带设备，并使用已复制的日志文件的新位置更新恢复历史记录文件。

配置

将数据库配置参数 **logarchmeth1** 设置为要复制到磁带的日志文件在磁盘上的位置。**db2tapemgr** 实用程序读取此 **logarchmeth1** 值以找到要复制的日志文件。在分区数据库环境中，必须在包含要复制的日志文件的每个数据库分区上设置 **logarchmeth1** 配置参数。

db2tapemgr 实用程序不使用 **logarchmeth2** 数据库配置参数。

STORE 和 DOUBLE STORE 参数

发出指定了 **STORE** 或 **DOUBLE STORE** 参数的 **db2tapemgr** 命令，以将归档的日志从磁盘传送到磁带。

- **STORE** 参数将某个范围的日志文件或所有日志文件从日志归档目录存储到指定的磁带设备，并从磁盘中删除那些文件。
- **DOUBLE STORE** 参数扫描历史记录文件以查看先前是否已将那些日志存储到磁带。
  - 如果先前从未存储过某个日志，**db2tapemgr** 就会将该日志文件存储到磁带，但不从磁盘中删除该文件。
  - 如果先前已存储过某个日志，**db2tapemgr** 就会将该日志文件存储到磁带并从磁盘中删除该文件。

  如果要在磁带和磁盘上都保存归档日志的副本，或者要将相同的日志存储在两盘不同的磁带上，请使用 **DOUBLE STORE**。

当发出指定了 **STORE** 或 **DOUBLE STORE** 参数的 **db2tapemgr** 命令时，**db2tapemgr** 实用程序首先扫描历史记录文件以查找 **logarchmeth1** 配置参数设置为磁盘的条目。如果它未找到原本应该在磁盘上的文件，将发出警告。如果 **db2tapemgr** 实用程序找不到要存储的日志文件，它将停止操作并发出消息以通知您没有要执行的操作。

RETRIEVE 参数

发出指定了 **RETRIEVE** 参数的 **db2tapemgr** 命令将文件从磁带传送到磁盘。

- 使用 **RETRIEVE ALL LOGS** 或 **LOGS** *n* **TO** *n* 参数来检索所有符合指定条件的归档日志，并将它们复制到磁盘。
- 使用 **RETRIEVE FOR ROLLFORWARD TO POINT-IN-TIME** 参数来检索执行前滚操作所需的所有归档日志，并将它们复制到磁盘。
- 使用 **RETRIEVE HISTORY FILE** 参数来从磁带检索历史记录文件并将其复制到磁盘。

行为

- 如果 **db2tapemgr** 实用程序在磁盘上找到日志文件，它就会读取磁带头以确保可以将日志文件写入磁带。它还将更新那些当前在磁带上的文件的历史记录。如果更新失败，操作将停止，并且将显示错误消息。
- 如果磁带是可写的，那么 **db2tapemgr** 实用程序就会将日志复制到磁带。复制文件后，将从磁盘中删除日志文件。最后，**db2tapemgr** 实用程序将历史记录文件复制到磁带并从磁盘中删除该文件。
- **db2tapemgr** 实用程序不会将日志文件追加到磁带。如果存储操作未写满整盘磁带，未使用的空间就会浪费掉。
- 对于任何给定磁带，**db2tapemgr** 实用程序只能在该磁带上存储一次日志文件。作此限制的目的是避免与写磁带介质相关的固有问题，例如导致磁带拉长。
- 在分区数据库环境中，**db2tapemgr** 实用程序每次仅对一个数据库分区执行。必须对每个数据库分区运行适当的命令，并使用 **db2tapemgr** 命令的 **ON DBPARTITIONNUM** 参数来指定数据库分区号。还必须确保每个数据库分区都有权访问磁带设备。
- **db2tapemgr** 实用程序在 DB2 pureScale环境中不受支持。

示例

以下示例显示如何使用 **db2tapemgr** 命令，以将数据库分区号 0 中 sample 数据库的主归档日志路径中的所有日志文件存储到磁带设备，并从归档日志路径中将它们除去:

```
db2tapemgr db sample on dbpartitionnum 0 store on /dev/rmt0.1 all logs
```

以下示例显示如何将主归档日志路径中的前 10 个日志文件存储到磁带设备，并从归档日志路径中将它们除去:

```
db2tapemgr db sample on dbpartitionnum store on /dev/rmt0.1 10 logs
```

以下示例显示如何将主归档日志路径中的前 10 个日志文件存储到磁带设备、将这些日志文件存储到第二盘磁带并从归档日志路径中将它们除去:

```
db2tapemgr db sample on dbpartitionnum double store on /dev/rmt0.1 10 logs
db2tapemgr db sample on dbpartitionnum double store on /dev/rmt1.1 10 logs
```

以下示例显示如何从磁带检索所有日志文件并将它们存储到目录中:

```
db2tapemgr db sample on dbpartitionnum retrieve all logs from /dev/rmt1.1
        to /home/dbuser/archived_logs
```

使用用户出口程序使日志文件归档和检索自动进行

通过创建 DB2 数据库管理器调用来执行归档或检索操作的用户出口程序，可以自动执行日志文件归档和检索。

当 DB2 数据库管理器调用用户出口程序时，会执行下列操作:

- 数据库管理器将控制权交给用户出口程序;
- 数据库管理器将参数传递给用户出口程序；并且
- 在完成时，用户出口程序再将返回码发送回数据库管理器。

配置

在调用用户出口程序进行日志文件的归档或检索前，确保 **logarchmeth1** 数据库配置参数已设置为 USEREXIT。它还使数据库可进行前滚恢复。

用户出口程序的要求

- 用户出口程序的可执行文件的名称必须为 db2uext2。
- 用户出口程序必须将日志文件从有效日志路径复制到归档日志路径，而不移动这些文件。不要从活动日志路径中除去日志文件。如果从活动日志路径中除去日志文件，那么在出现故障时，DB2 数据库可能无法成功恢复。

  在恢复期间，DB2 数据库要求日志文件位于活动日志路径中。DB2 数据库服务器将从有效日志路径中除去恢复操作不再需要的归档日志文件。
- 用户出口程序必须处理错误情况。由于 DB2 数据库管理器只能处理有限的返回情况，所以用户出口程序必须处理错误。

  请参阅第 141 页的『用户出口错误处理』。
- 每个 DB2 数据库管理器实例只能调用一个用户出口程序。由于数据库管理器实例只能调用一个用户出口程序，所以必须将您的用户出口程序设计成有一个部分包含它必须执行的每个操作。

样本用户出口程序

对所有受支持的平台都提供了样本用户出口程序。您可以修改这些程序以适应您的特定要求。样本程序中有很好的注释信息，它有助于您进行更有效的使用。

您应该知道用户出口程序必须从活动日志路径*复制*日志文件到归档日志路径中。不要从活动日志路径中除去日志文件。（这可能会在数据库恢复期间导致问题。）DB2 从活动日志路径中除去恢复操作不再需要的归档日志文件。

以下是 DB2 数据服务器附带的样本用户出口程序的描述。

- **UNIX 操作系统**

  用于 UNIX 操作系统的DB2 数据服务器的用户出口样本程序位于 sqllib/samples/c 子目录中。虽然提供的样本是用 C 语言编码的，您的用户出口程序仍可以不同的编程语言编写。

  您的用户出口程序必须是可执行文件，名称为 db2uext2。

  对于 UNIX 操作系统，存在以下四个样本用户出口程序:

– db2uext2.ctsm

此样本使用 Tivoli Storage Manager 来归档和检索数据库日志文件。

– db2uext2.ctape

该样本使用磁带介质来归档和检索数据库日志文件。

– db2uext2.cdisk

该样本使用操作系统的 COPY 命令以及磁盘介质来归档和检索数据库日志文件。

– db2uxt2.cxbsa

此样本使用 X/Open group 发布的 XBSA Draft 0.8。可将它用于归档和检索数据库日志文件。此样本只在 AIX 上受支持。

- **Windows 操作系统**

DB2 数据服务器 Windows 版操作系统中的用户出口样本程序位于 sqllib\samples\c 子目录中。虽然提供的样本是用 C 语言编码的，您的用户出口程序仍可以不同的编程语言编写。

您的用户出口程序必须是可执行文件，名称为 db2uext2。

对于 Windows 操作系统，有两个样本用户出口程序:

– db2uext2.ctsm

此样本使用 Tivoli Storage Manager 来归档和检索数据库日志文件。

– db2uext2.cdisk

该样本使用操作系统的 COPY 命令以及磁盘介质来归档和检索数据库日志文件。

用户出口程序的调用格式

DB2 数据库管理器调用用户出口程序时，它将一组参数（数据类型为 CHAR）发送到该程序。

命令语法

```
db2uext2 -OS<os> -RL<db2rel> -RQ<request> -DB<dbname>
   -NN<nodenum> -LP<logpath> -LN<logname> -AP<tsmpasswd>
-SP<startpage> -LS<logsize>
```

os 指定运行实例的平台。有效值包括: AIX、Solaris、HP-UX、SCO、Linux 和 NT。

db2rel 指定 DB2 发行版级别。例如，SQL07020。

request
指定请求类型。有效的值是: ARCHIVE 和 RETRIEVE。

dbname
指定数据库名称。

nodenum
指定本地节点号，例如 5。

logpath
指定日志文件的标准路径。该路径必须包含尾部路径分隔符。例如 /u/database/log/path/ 或 d:\logpath\。

logname

指定要归档或检索的日志文件的名称，例如 S0000123.LOG。

tsmpasswd

指定 TSM 密码。（如果先前已指定了数据库配置参数 *tsm_password* 的值，该值将发送到用户出口程序。）

startpage

指定日志扩展数据块开始的那个设备的 4 KB 偏移页的数量。

logsize

指定日志扩展数据块的大小，以 4 KB 页为单位。只有将原始设备用于日志记录时，此参数才有效。

用户出口错误处理

如果创建了一个用户出口程序来自动执行日志文件归档和检索，那么该用户出口程序会将返回码传递给调用它的 DB2 数据库管理器。

DB2 数据库管理器只能处理数目有限的特定错误代码列表。但是，用户出口程序可能会遇到许多不同类型的错误情况，例如，操作系统错误。用户出口程序必须将它遇到的错误情况映射至数据库管理器可以处理的错误代码。

表 8 显示了可由用户出口程序返回的代码，并描述数据库管理器如何解释这些代码。如果返回码未在该表中列出，那么将其值当作 32 处理。

表 8. 用户出口程序返回码

返回码	说明
0	成功。
4	遇到临时资源错误。[a]
8	需要操作员介入。[a]
12	硬件错误。[b]
16	用户出口程序或该程序使用的软件功能出错。[b]
20	传送给用户出口程序一个或多个参数出错。验证用户出口程序是否正确地处理指定的参数。[b]
24	找不到用户出口程序。[b]
28	输入/输出 (I/O) 故障或操作系统导致的错误。[b]
32	用户出口程序由用户终止。[b]
255	由用户出口程序无法为可执行文件装入库文件而导致的错误。[c]

表 8. 用户出口程序返回码 (续)

返回码	说明
[a] 对于归档与检索请求，返回码 4 或 8 导致在五分钟后重试。如果用户出口程序继续对向同一日志文件发出的检索请求返回 4 或 8，那么 DB2 将继续重试，直到成功为止。（这适用于前滚操作或对 **db2ReadLog** API（由复制实用程序使用）的调用。） [b] 用户出口请求将暂挂 5 分钟。在此期间，将忽略所有的请求，包括导致错误状态的请求。在这个时间为 5 分钟的暂挂后，再处理下一个请求。如果处理此请求时没有错误，将继续处理新的用户出口请求，且 DB2 会重新发出失败过的或先前被暂挂的归档请求。如果在重试期间生成一个大于 8 的返回码，那么会将请求再暂挂 5 分钟。此 5 分钟的暂挂将继续，直到该校正该问题或直到停止并重新启动该数据库。一旦从该数据库断开了与所有应用程序的连接，DB2 会对先前可能未成功归档的任何日志文件发出归档请求。如果用户出口程序对日志文件归档失败，磁盘可能会填满了日志文件，从而可能使性能降低。一旦磁盘变满，数据库管理器将不再接受应用程序对更新数据库的请求。如果调用了用户出口程序来检索日志文件，那么前滚恢复会暂挂但不停止，除非指定了 ROLLFORWARD STOP 选项。如果未指定 STOP 选项，可以更正问题并继续恢复。 [c] 如果用户出口程序返回错误代码 255，可能是程序无法装入可执行文件所需的库文件。要进行验证，可手动调用用户出口程序。会显示更多信息。 **注**：在归档和检索操作期间，对 0 和 4 之外的所有返回码都发出警报消息。该警报消息包含来自用户出口程序的返回码和提供给用户出口程序的输入参数的副本。	

分配和除去日志文件

崩溃恢复需要的日志文件称为活动日志。除非启用了无限日志记录，否则不会除去活动日志路径中的日志文件（崩溃恢复可能需要这些文件）。

如果启用了无限日志记录且需要使空间可用于更多活动日志文件，那么数据库管理器会归档活动日志文件，并将其重命名以创建新的活动日志文件。如果在使用了无限日志记录时需要进行崩溃恢复，那么可能需要从归档日志路径中检索日志文件，以完成崩溃恢复。

当 **logarchmeth1** 数据库配置参数未设置为 OFF 时，仅当崩溃恢复不再需要已满的日志文件之后，才应该考虑除去该日志文件（启用了无限日志记录的情况除外，在此情况下，可能会改为将日志文件移动到归档日志路径）。

当 **logarchmeth1** 或 **logarchmeth2** 设置为 OFF、LOGRETAIN 或 USEREXIT 以外的值时，可启用归档日志文件压缩以减少归档日志文件所需的磁盘空间量。

分配新日志文件和除去旧日志文件的过程取决于 **logarchmeth1** 和 **logarchmeth2** 数据库配置参数的设置:

logarchmeth1 和 logarchmeth2 都设置为 OFF

使用循环日志记录。对于循环日志记录，不支持前滚恢复，但支持崩溃恢复。

在循环日志记录期间，不会生成新日志文件（辅助日志除外）且不删除旧日志文件。日志文件以循环的方式来处理。即，当最后一个日志文件已满时，数据库管理器开始写入第一个日志文件。

如果所有日志文件都是活动的且循环日志记录进程不能回绕至第一个日志文件，那么日志已满情况就可能发生。当所有主日志文件处于活动状态且已满

时，会创建辅助日志文件。当取消激活数据库，或活动日志文件需要辅助日志文件所使用的空间时，将删除辅助日志文件。

logarchmeth1 或 logarchmeth2 设置为 LOGRETAIN

使用归档日志。数据库是可恢复的数据库。启用前滚恢复和崩溃恢复。数据库管理器不会管理日志文件。归档日志文件之后，必须从当前日志路径中删除它们，以便新的日志文件可以复用磁盘空间。要确定哪些日志文件是归档的日志，请检查数据库配置参数 **loghead** 的值。此参数指示活动的最低编号的日志。那些序号小于 **loghead** 值的日志是不活动的，并可将其归档和除去。

logarchmeth1 或 logarchmeth2 设置为 OFF 或 LOGRETAIN 以外的值

使用归档日志。数据库是可恢复的数据库。启用前滚恢复和崩溃恢复。当日志文件已满时，数据库管理器会自动对它进行归档。

不会删除日志文件。相反，当需要新的日志文件而没有可用的日志文件时，那么重命名归档日志文件并再次使用该文件。一旦归档日志文件已关闭并被复制到归档日志目录中，就不会被删除或重命名。数据库管理器等待至需要新日志文件，才重命名最旧的归档日志。当不再需要在恢复期间移至数据库目录的日志文件时会在恢复过程中将其除去。

如果在归档日志文件时发生错误，那么归档会暂挂一段时间，此时间由 **archretrydelay** 数据库配置参数指定。还可以使用 **numarchretry** 数据库配置参数来指定数据库管理器尝试将日志文件归档到主要或辅助归档目录的次数，然后它再尝试将日志文件归档到故障转移目录（由 **failarchpath** 数据库配置参数指定）。只有在设置 **failarchpath** 数据库配置参数之后，才使用 **Numarchretry**。如果 **numarchretry** 设置为 0，那么数据库管理器将继续从主要或辅助日志路径尝试归档。

除去旧日志文件的最简单方法是重新启动数据库。一旦重新启动了数据库，就只能在数据库目录中找到新日志文件和数据库管理器归档失败的日志文件。

当数据库重新启动时，数据库日志目录中日志的最小数目将等于主日志的数目，这些主日志可以通过使用 **logprimary** 数据库配置参数来配置。可在日志目录中找到多于主日志数目的日志。如果关闭数据库时日志目录中空日志数大于重新启动数据库时 **logprimary** 配置参数的值，那么会发生这种情况。如果在关闭数据库之后、重新启动数据库之前更改了 **logprimary** 配置参数的值，或者如果分配了辅助日志且从未使用，那么将发生这种情况。

当重新启动数据库时，如果空日志数小于 **logprimary** 配置参数指定的主日志的数目，那么分配附加日志文件以弥补所差的日志数目。如果数据库目录中有多于主日志的空日志可用，那么在重新启动数据库，可在数据库目录中找到所有可用的空日志。数据库关闭之后，已创建的辅助日志文件在重新启动数据库时将保留在活动日志路径中。

使用备份映像包括日志文件

执行联机备份操作时，可以指定在备份映像中包括复原和恢复数据库所需的日志文件。

这意味着，如果需要将备份映像交付至灾难恢复位置，那么无需单独发送日志文件或自行将其打包。此外，不必决定需要哪些日志文件来保证联机备份的一致性。这为成功恢复提供某种保护，以防止删除所需的日志文件。

要使用此功能，请指定 **BACKUP DATABASE** 命令的 **INCLUDE LOGS** 选项。当指定此选项时，BACKUP 实用程序将截断当前有效日志文件并将必要的日志扩展数据块集合复制到备份映像中。

要从备份映像复原日志文件，可使用 **RESTORE DATABASE** 命令的 **LOGTARGET** 选项，并指定存在于 DB2 服务器上的标准路径。然后，复原数据库实用程序将日志文件从映像写至目标路径。如果目标路径中已存在名称相同的日志文件，复原操作将失败并返回一个错误。如果未指定 **LOGTARGET** 选项，那么不会从备份映像中复原任何日志文件。

如果指定 **LOGTARGET** 选项，且备份映像不包含任何日志文件，那么在尝试复原任何表空间数据之前，将返回一个错误。如果指定无效路径或只读路径，复原操作也将失败。在指定 **LOGTARGET** 选项条件下，复原数据库或表空间的过程中，如果不能抽取一个或多个日志文件，那么复原操作将失败并返回错误。

还可以选择只复原保存在备份映像中的日志文件。要执行此操作，可使用 **RESTORE DATABASE** 命令的 **LOGTARGET** 选项指定 **LOGS** 选项。以此方式复原日志文件时，如果复原操作遇到任何问题，那么复原操作将失败并返回错误。

自动增量复原操作期间，将仅从备份映像中检索复原操作的目标映像中包括的日志。不会从那些备份映像中抽取增量复原操作期间所引用的中间映像中包括的任何日志。手动增量复原期间，当复原包括日志文件的备份映像时，如果指定日志目标目录，那么会复原此备份映像中的日志文件。

如果前滚一个数据库，而该数据库是使用包含日志文件的联机备份映像复原的，就会发生错误 SQL1268N，此错误表示前滚恢复已由于检索日志时接收到错误而停止。如果您尝试将备份映像复原到的目标系统无权访问源系统归档其事务日志时使用的设施，就会发生此错误。

如果备份数据库时指定了 **BACKUP DATABASE** 命令的 **INCLUDE LOGS** 选项，然后执行使用该备份映像的复原操作和前滚操作，那么虽然备份映像包含日志，DB2 在前滚数据库时也仍然会搜索其他事务日志。标准前滚行为会一直搜索其他事务日志，直到找不到更多日志为止。有可能有多个时间戳记相同的日志文件。因此，DB2 在找到第一个与数据库前滚目标时间点匹配的时间戳记时不会立即停止，因为其他日志文件也可能拥有该时间戳记。DB2 将继续查找事务日志，直到找到大于指定时间点的时间戳记为止。

当找不到任何其他日志时，前滚操作将成功地结束。但是，如果搜索其他事务日志文件时出错，就会返回错误 SQL1268N。在初始复原期间，可能会由于某些数据库配置参数被复位或覆盖而发生错误 SQL1268N。在这些数据库配置参数中，有三个是 TSM 参数，即 **tsm_nodename**、**tsm_owner** 和 **tsm_password**。它们全都被被重置为 NULL。要前滚至日志末尾，在执行前滚操作前，需要将这些数据库配置参数重置为与源系统相对应。此外，可以在发出 **ROLLFORWARD DATABASE** 命令时指定 **NORETRIEVE** 选项。这将防止 DB2 数据库系统尝试获取可能已在别处丢失的事务日志。

注:

1. 脱机备份不支持此功能。
2. 当联机备份映像中包括日志时，在 V8.2 之前的 DB2 数据库发行版中无法复原生成的映像。

防止日志文件意外丢失

在需要删除数据库或执行时间点前滚恢复的情况下，可能会丢失以后恢复操作所需的日志文件。在这些情况下，对当前数据库日志路径目录中的所有日志进行复制是很重要的。

请考虑以下方案:

- 如果计划在复原操作之前删除数据库，那么需要在发出 **DROP DATABASE** 命令之前将日志文件保存在有效日志路径中。因为删除数据库前其中一些日志文件可能尚未归档，所以，复原数据库后，前滚恢复可能需要这些日志文件。通常，不需要在发出 **RESTORE** 命令之前删除数据库。但是，如果该数据库已损坏到会导致 **RESTORE** 命令失败的程度，那么可能必须删除该数据库（或者通过指定 **DROP DATABASE** 命令的 **AT DBPARTITIONNUM** 参数来在一个数据库分区中删除该数据库）。您也可能决定在执行复原操作前删除数据库以便从头开始。
- 如果将数据库前滚至特定时间点，那么指定时间戳记之后的日志数据会被覆盖。在完成时间点前滚操作并重新连接到数据库之后，如果确定实际上需要将该数据库前滚至稍后的时间点，将无法实现此操作，这是因为日志已被覆盖。或许原始日志文件集已归档，但是 DB2 可能调用用户出口程序来自动归档新生成的日志文件。视编写的用户出口程序的不同，这可能导致归档日志目录中的原始日志文件集被覆盖。即使原始日志文件集和新日志文件集都存在于归档日志目录中（相同文件的不同版本），也必须确定应对以后的恢复操作使用的那组日志。

尽量降低维护对可用性的影响

您将需要对 DB2 数据库解决方案执行维护，例如，软件或硬件升级、数据库性能调整、数据库备份、统计信息收集和商业目的的监视。

为了尽量降低执行该维护对解决方案可用性的影响，需要执行以下工作：谨慎安排脱机维护，以及使用可降低联机维护对可用性的影响的 DB2 功能部件和功能。

开始之前

在可以使用下列步骤来尽量降低维护对 DB2 数据库解决方案的可用性的影响之前，必须:

- 配置自动维护；以及
- 安装高可用性灾难恢复 (HADR) 功能部件。

过程

1. 允许自动维护为您进行维护。

   DB2 数据库可以使许多数据库维护活动自动进行。一旦配置了自动维护，将不需要您采取任何其他步骤就能执行该维护。
2. 使用 DB2 高可用性灾难恢复 (HADR) 滚动升级来尽量降低其他维护活动的影响。

如果要升级软件或硬件，或者如果要修改某些数据库管理器配置参数，可以使用 HADR 功能部件在对可用性中断最小的情况下完成这些更改。由 HADR 实现的这种无缝更改被称为滚动升级。

即使是在 HADR 环境中，有些维护活动也需要在执行维护之前关闭数据库。在某些情况下，关闭 HADR 数据库的过程与关闭标准数据库的过程会稍微不同：如果 HADR 数据库是由与之连接的客户机应用程序启动的，那么必须使用 **DEACTIVATE DATABASE** 命令。

停止 DB2 高可用性灾难恢复 (HADR)

如果要使用 DB2 高可用性灾难恢复 (HADR) 功能，那么可能需要在主数据库或备用数据库上停止 HADR 操作以执行维护。仅应在您要执行维护的数据库上停止 HADR 操作。要完全停止使用 HADR，请在这两个数据库上停止 HADR。

关于此任务

警告：如果要停止指定的数据库，但仍然想保持其 HADR 主数据库角色或备用数据库角色，那么不要发出 STOP HADR 命令。如果发出 **STOP HADR** 命令，那么该数据库将变成标准数据库，并且可能需要重新初始化才能继续作为 HADR 数据库运行。请改为发出 **DEACTIVATE DATABASE** 命令。

如果针对标准数据库发出 **STOP HADR** 命令，那么将返回错误。

过程

要在主数据库或备用数据库上停止 HADR 操作，请执行以下操作：

- 从 CLP 中，对要停止 HADR 操作的数据库发出 **STOP HADR** 命令。

  在以下示例中，停止数据库 SOCKS 上的 HADR 操作：

  ```
  STOP HADR ON DATABASE SOCKS
  ```

  如果针对不活动的主数据库发出此命令，数据库切换至标准数据库并保持脱机。

  如果针对不活动的备用数据库发出此命令，数据库切换至标准数据库、置于前滚暂挂状态，并保持脱机。

  如果在活动的主数据库上发出此命令，会停止将日志装入备用数据库，并且在主数据库上关闭所有 HADR 引擎分派单元（EDU）。数据库切换至标准数据库并保持联机。可以继续进行事务处理。可以发出带有 AS PRIMARY 选项的 START HADR 命令将数据库角色重新切换为主数据库。

  如果在活动的备用数据库上发出此命令，那么返回错误消息，指示必须取消激活备用数据库，然后才能尝试将其转换至标准数据库。
- 从应用程序中调用 **db2HADRStop** 应用程序编程接口 (API)。
- 从 IBM Data Studio 中，对 **STOP HADR** 命令打开任务助手。

在高可用性灾难恢复 (HADR) 环境中激活和取消激活数据库

如果标准数据库是由客户机连接启动的，那么最后一个客户机断开连接时，该数据库将关闭。如果 HADR 主数据库是由客户机连接启动的，那么等同于使用 **ACTIVATE DATABASE** 命令来启动数据库。

要关闭由客户机连接启动的 HADR 主数据库，需要显式发出 **DEACTIVATE DATABASE** 命令。

在处于前滚暂挂状态的标准数据库上，**ACTIVATE DATABASE** 和 **DEACTIVATE DATABASE** 命令不适用。只能继续前滚、停止前滚或者使用 **START HADR** 命令来启动该数据库才能将该数据库作为 HADR 备用数据库启动。一旦已将某个数据库作为 HADR 备用数据库来启动，就可使用 **ACTIVATE DATABASE** 和 **DEACTIVATE DATABASE** 命令来启动和停止该数据库。

使用下列方法来激活主数据库:

- 客户机连接
- **ACTIVATE DATABASE** 命令
- IBM Data Studio 中用于 **ACTIVATE DATABASE** 命令的任务助手
- 指定了 AS PRIMARY 选项的 **START HADR** 命令

使用下列方法来主数据库取消激活:

- **DEACTIVATE DATABASE** 命令

  注: 如果使用 **DEACTIVATE DATABASE** 命令或 sqle_deactivate_db API 取消激活处于断开连接的对等状态的 HADR 主数据库，那么该数据库将处于不一致状态。在重新启动时该数据库将需要崩溃恢复，并且将无法对其进行脱机备份，直到其重新启动为止。
- IBM Data Studio 中用于 **DEACTIVATE DATABASE** 命令的任务助手
- 带 **FORCE** 参数的 **db2stop** 命令

使用下列方法来激活备用数据库:

- **ACTIVATE DATABASE** 命令
- IBM Data Studio 中用于 **ACTIVATE DATABASE** 命令的任务助手
- 带 AS STANDBY 选项的 **START HADR** 命令

使用下列方法将备用数据库取消激活:

- **DEACTIVATE DATABASE** 命令
- IBM Data Studio 中用于 **DEACTIVATE DATABASE** 命令的任务助手
- 带 **FORCE** 参数的 **db2stop** 命令

关闭 HADR 对的建议顺序

n

警告: 尽管可对主数据库和/或备用数据库使用 **STOP HADR** 命令来停止 HADR，但应谨慎使用。如果要停止指定的数据库，但仍然想保留其 HADR 主数据库角色或备用数据库角色，请不要发出 **STOP HADR** 命令。如果发出 **STOP HADR** 命令，那么该数据库将变为标准数据库并且可能需要重新初始化才能继续作为 HADR 数据库运行。请改为发出 **DEACTIVATE DATABASE** 命令。

如果只想关闭 HADR 操作，那么下面是关闭 HADR 对的建议方法:

1. 取消激活主数据库
2. 对主数据库停止 DB2

3. 取消激活备用数据库
4. 对备用数据库停止 DB2

DB2 高可用性灾难恢复 (HADR) 环境中的表空间重新平衡注意事项

可以使用 ALTER TABLESPACE REBALANCE 或 ALTER TABLESPACE USING STOGROUP 语句来在主数据库上启动重新平衡操作。将在备用数据库上重放该语句，并且将启动相应的重新平衡操作。

在重新平衡操作期间，可以指定带 REBALANCE SUSPEND 子句的 ALTER TABLESPACE 语句来暂挂主数据库上的重新平衡操作。要恢复已暂挂的重新平衡操作，请指定带 REBALANCE RESUME 子句的 ALTER TABLESPACE 语句。

备用数据库在重放 ALTER TABLESPACE REBALANCE SUSPEND 语句时仍然处于活动状态。因为在主数据库上暂挂了重新平衡，所以当备用数据库作为新的主数据库进行接管时，将暂挂新的主数据库上的重新平衡操作，并且将隐式恢复新的备用数据库上的重新平衡操作。

使用分割镜像作为克隆数据库或备用数据库来复原数据库时，将在启动数据库时自动恢复表空间的任何已暂挂的重新平衡操作。

在 DB2 高可用性灾难恢复 (HADR) 环境中执行滚动更新和升级

当升级软件/硬件、更新 DB2 数据库系统或更改数据库配置参数时，在高可用性灾难恢复 (HADR) 环境中使用此过程。

此过程保持数据库服务在整个升级过程中可用，只在处理从一个数据库切换至另一数据库时有短暂的服务中断。对于多备用数据库设置，可以在更新或升级过程中提供连续的 HA 和 DR 保护。

开始之前

复查 HADR 的系统需求。请参阅第 53 页的『高可用性灾难恢复 (HADR) 的系统要求』。

在启动滚动升级之前，HADR 数据库对应该处于对等状态。

注: 应该在测试环境中实现所有 DB2 数据库系统修订包和升级，然后再应用到生产系统中。

关于此任务

此过程不能用于从 DB2 数据库系统的较低版本升级到较高版本；例如，不能使用此过程从 V8 升级到 V9 数据库系统。只能使用此过程来对数据库系统执行滚动更新（从一个修改级别滚动更新为另一个修改级别），例如，通过应用修订包。在滚动更新期间，在一小段时间内，备用数据库的修改级别（例如，修订包级别）可高于主数据库修改级别以测试新级别。但是，不应长时间保持此配置，以便降低所使用功能部件在这些级别之间可能不兼容的风险。如果主数据库的数据库系统的修改级别比备用数据库的数据库系统的修改级别高，那么主数据库与备用数据库不会彼此连接。

如果更新 DB2 HADR 配置参数，此过程就不起作用。对 HADR 配置参数的更新应该单独进行。由于 HADR 要求主数据库与备用数据库上的参数相同，所以这可能要求同

时将主数据库和备用数据库取消激活并进行更新。

过程

要在 HADR 环境中执行滚动升级:

1. 升级备用数据库所在的系统:
   a. 使用 **DEACTIVATE DATABASE** 命令关闭备用数据库。
   b. 如有必要，请关闭备用数据库上的实例。
   c. 更改下列一项或多项: 软件、硬件或 DB2 配置参数。

      注: 在执行滚动升级时，不能更改任何 HADR 配置参数。
   d. 如有必要，请重新启动备用数据库上的实例。
   e. 使用 **db2pd** 命令重新启动备用数据库。
   f. 确保备用数据库进入对等状态。使用 **GET SNAPSHOT** 命令来检查此状态。
2. 切换主数据库和备用数据库的角色:
   a. 在备用数据库上发出 **TAKEOVER HADR** 命令。
   b. 将客户机导向新的主数据库。可以使用客户机自动重新路由来实现此操作。

      注: 因为备用数据库作为主数据库接管，所以现在升级新的主数据库。如果正在应用 DB2 数据库系统修订包，那么 **TAKEOVER HADR** 命令会将原始主数据库的角色更改为备用数据库。但是，此命令不允许新的备用数据库连接至最近更新的主数据库。因为新的备用数据库使用较低版本的 DB2 数据库系统，所以它不可能理解由更新的主数据库生成的新日志记录，因而将关闭此备用数据库。为了使新备用数据库与新主数据库重新连接（即，针对要改正的 HADR 对），还必须对新备用数据库进行更新。
3. 使用与步骤 1 中相同的过程来升级原始主数据库（现在是备用数据库）。完成此操作之后，两个数据库均已升级且以 HADR 对等状态相互连接。HADR 系统提供完整的数据库服务和充分的高可用性保护。
4. 可选: 要返回至原始配置，如步骤 2 中所述，切换主数据库和备用数据库的角色。

   要在滚动升级期间启用 HADR 的“在备用数据库上读取”功能，请推迟执行可选步骤 4 并执行下列步骤。内部 DB2 程序包的绑定在首次连接时出现，并且只能在主数据库上成功完成。这些步骤是必要步骤，以确保备用数据库上的内部 DB2 程序包在执行读操作之前的一致性。
5. 按照以下步骤在备用数据库上启用 HADR 的“在备用数据库上读取”功能:
   a. 在备用数据库上将 **DB2_HADR_ROS** 注册表变量设置为 ON。
   b. 使用 **DEACTIVATE DATABASE** 命令关闭备用数据库。
   c. 重新启动备用数据库上的实例。
   d. 使用 **ACTIVATE DATABASE** 命令重新启动备用数据库。
   e. 使用 **GET SNAPSHOT** 命令来检查备用数据库是否进入对等状态。
6. 按照以下步骤来切换主数据库和备用数据库的角色:
   a. 在备用数据库上发出 **TAKEOVER HADR** 命令。
   b. 将客户机导向新的主数据库。
7. 重复步骤 5 中的相同过程以在新备用数据库上启用 HADR 的“在备用数据库上读取”功能。

8. 可选: 要返回至原始配置，如步骤 2 中所述，切换主数据库和备用数据库的角色。

在自动化的高可用性灾难恢复 (HADR) 环境中滚动升级

使用集成的高可用性 (HA) 功能来自动进行 HADR 时，将软件（操作系统或 DB2 数据库系统）和硬件升级或者更改数据库配置参数时，需要执行其他步骤。使用此过程来在自动化的 HADR 环境中执行滚动升级。

开始之前

必须准备好下列先决条件，才能执行过程部分中所述的各步骤:

- 两个 DB2 实例（在此示例中，在每个节点上名为 stevera）。
- 两个节点（grom04 和 grom03）。grom04 机器最初主管 HADR 主数据库。
- 实例最初以 DB2 V9.8 GA 代码运行。
- 使用 HADR 故障转移的集成 HA 控件配置实例。集群域名为 test。

注: 应该在测试环境中实现所有 DB2 数据库系统修订包和升级，然后再应用到生产系统中。

在启动滚动升级之前，HADR 数据库对应该处于对等状态。

限制

此过程不能用于从 DB2 数据库系统的较低版本迁移到较高版本；例如，不能使用此过程从 V8 迁移到 V9 数据库系统。只能使用此过程来将数据库系统从一个修改级别更新为另一个修改级别，例如，通过应用修订包。

如果更新 DB2 HADR 配置参数，此过程就不起作用。对 HADR 配置参数的更新应该单独进行。由于 HADR 要求主数据库与备用数据库上的参数相同，所以这可能要求同时将主数据库和备用数据库取消激活并进行更新。

过程

1. 显示初始系统状态:

```
root@grom03:# lsrpdomain
Name   OpState   RSCTActiveVersion   MixedVersions   TSPort   GSPort
-----  --------  ------------------  -------------   -------  -------
test   Online    2.4.7.1             No              12347    12348

root@grom03:# lssam
Online IBM.ResourceGroup:db2_stevera_grom03_0-rg Nominal=Online
        '- Online IBM.Application:db2_stevera_grom03_0-rs
                '- Online IBM.Application:db2_stevera_grom03_0-rs:grom03
Online IBM.ResourceGroup:db2_stevera_grom04_0-rg Nominal=Online
        '- Online IBM.Application:db2_stevera_grom04_0-rs
                '- Online IBM.Application:db2_stevera_grom04_0-rs:grom04
Online IBM.ResourceGroup:db2_stevera_stevera_SVTDB-rg Nominal=Online
        |- Online IBM.Application:db2_stevera_stevera_SVTDB-rs
                |- Offline IBM.Application:db2_stevera_stevera_SVTDB-rs:grom03
                '- Online IBM.Application:db2_stevera_stevera_SVTDB-rs:grom04
        '- Online IBM.ServiceIP:db2ip_9_26_124_22-rs
                |- Offline IBM.ServiceIP:db2ip_9_26_124_22-rs:grom03
                '- Online IBM.ServiceIP:db2ip_9_26_124_22-rs:grom04

root@grom03:# lsrpnode
```

```
Name     OpState   RSCTVersion
-----    -------   ------------
grom03   Online    2.4.7.1
grom04   Online    2.4.7.1
```

根据此示例，您可以知道必须在 grom03 上将备用数据库实例升级。为此，请停止在 grom03 上主管的所有资源组。

2. 停止备用节点上的所有资源组并确认更改:

```
root@grom03:# chrg -o Offline db2_stevera_grom03_0-rg

root@grom03:# lssam g db2_stevera_grom03_0-rg
Offline IBM.ResourceGroup:db2_stevera_grom03_0-rg Nominal=Offline
        '- Offline IBM.Application:db2_stevera_grom03_0-rs
                '- Offline IBM.Application:db2_stevera_grom03_0-rs:grom03
```

3. 停止集群节点（备用节点）并确认更改:

```
root@grom03:# stoprpnode grom03

root@grom03:# lsrpdomain
Name   OpState   RSCTActiveVersion   MixedVersions   TSPort   GSPort
----   -------   -----------------   -------------   ------   ------
test   Offline   2.4.7.1             No              12347    12348
```

4. 在备用节点上安装 DB2 修订包:

   可选: 在 AIX 上，可能需要对有问题的修订包安装 RSCT 必备软件。

```
root@grom03:# ./installFixPack -b /opt/ibm/db2/V9.8
DBI1017I  installFixPack is updating the DB2 product(s) installed in
      location /opt/ibm/db2/V9.8.
```

   DB2 修订包安装即会开始。

5. 启动节点并使资源组联机:

   在安装成功完成后。

```
root@grom03:# startrpdomain test

root@grom03:# lsrpdomain
Name   OpState   RSCTActiveVersion   MixedVersions   TSPort   GSPort
-----  --------  ------------------  --------------  -------  -------
test   Online    2.4.7.1             Yes             12347    12348

root@grom03:# chrg -o Online db2_stevera_grom03_0-rg
```

6. 验证是否已应用修订包且 HADR 是否再次处于对等状态:

```
stevera@grom03% db2level

stevera@grom03% db2pd hadr db SVTDB
```

7. 执行 TAKEOVER:

   要升级另一个节点（在此情况中为 grom04），请执行 TAKEOVER，以便节点 grom03 主管 HADR 主数据库。

```
root@grom03:# su - stevera

stevera@grom03% db2 takeover hadr on db SVTDB
DB20000I  The TAKEOVER HADR ON DATABASE command completed successfully.

root@grom03:# lssam
Online IBM.ResourceGroup:db2_stevera_grom03_0-rg Nominal=Online
        '- Online IBM.Application:db2_stevera_grom03_0-rs
                '- Online IBM.Application:db2_stevera_grom03_0-rs:grom03
```

```
Online IBM.ResourceGroup:db2_stevera_grom04_0-rg Nominal=Online
        '- Online IBM.Application:db2_stevera_grom04_0-rs
                '- Online IBM.Application:db2_stevera_grom04_0-rs:grom04
Online IBM.ResourceGroup:db2_stevera_stevera_SVTDB-rg Nominal=Online
        |- Online IBM.Application:db2_stevera_stevera_SVTDB-rs
                |- Online IBM.Application:db2_stevera_stevera_SVTDB-rs:grom03
                '- Offline IBM.Application:db2_stevera_stevera_SVTDB-rs:grom04
        '- Online IBM.ServiceIP:db2ip_9_26_124_22-rs
                |- Online IBM.ServiceIP:db2ip_9_26_124_22-rs:grom03
                '- Offline IBM.ServiceIP:db2ip_9_26_124_22-rs:grom04
```

8. 在节点 grom04 上执行升级:

```
root@grom03:# ssh root@grom04

root@grom04:# chrg -o Offline db2_stevera_grom04_0-rg

root@grom04:# lssam g db2_stevera_grom04_0-rg
Offline IBM.ResourceGroup:db2_stevera_grom04_0-rg Nominal=Offline
        '- Offline IBM.Application:db2_stevera_grom04_0-rs
                '- Offline IBM.Application:db2_stevera_grom04_0-rs:grom04

root@grom04:# stoprpnode grom04
```

可选: 在 AIX 上, 可能需要对有问题的修订包安装 RSCT 必备软件。

```
root@grom04:# ./installFixPack -b /opt/ibm/db2/V9.8
DBI1017I  installFixPack is updating the DB2 product(s) installed in
      location /opt/ibm/db2/V9.8
```

DB2 修订包安装即会开始。

在安装成功完成后。

```
root@grom04:# lsrpdomain
Name   OpState   RSCTActiveVersion   MixedVersions   TSPort   GSPort
-----  --------  ------------------  --------------  -------  -------
test   Offline   2.4.7.1             Yes             12347    12348

root@grom04:# startrpdomain test

root@grom04:# lsrpdomain
Name   OpState   RSCTActiveVersion   MixedVersions   TSPort   GSPort
-----  --------  ------------------  --------------  -------  -------
test   Online    2.4.7.1             Yes             12347    12348

root@grom04:# chrg -o Online db2_stevera_grom04_0-rg
```

9. 验证是否已应用修订包（通过运行 **db2level** 来验证），以及 HADR 是否处于对等状态（通过运行 db2pd hadr db svtdb 来验证）:

```
root@grom04:# su - stevera

stevera@grom04% db2pd -hadr -db svtdb

Database Partition 0 -- Database SVTDB -- Active -- Up 0 days 00:00:05

HADR Information:
Role     State    SyncMode    HeartBeatsMissed    LogGapRunAvg (bytes)
-----    ------   ---------   ----------------    --------------------
Standby  Peer     Sync        0                   0

ConnectStatus   ConnectTime                            Timeout
--------------  ------------                           --------
Connected       Tue May 5 13:20:58 2009 (1241544058)   120

PeerWindowEnd                          PeerWindow
-------------                          -----------
```

```
Tue May  5 13:25:58 2009 (1241544358)  300

LocalHost                               LocalService
----------                              -------------
grom04                                  55555

RemoteHost                              RemoteService      RemoteInstance
----------                              -------------      --------------
grom03                                  55555              stevera

PrimaryFile    PrimaryPg    PrimaryLSN
------------   ----------   -----------
S0000001.LOG   1            0x0000000003389487

StandByFile    StandByPg    StandByLSN          StandByRcvBufUsed
------------   ----------   -----------         -----------------
S0000001.LOG   1            0x0000000003389487  0%
```

```
root@grom04:# lssam
Online IBM.ResourceGroup:db2_stevera_grom03_0-rg Nominal=Online
        '- Online IBM.Application:db2_stevera_grom03_0-rs
                '- Online IBM.Application:db2_stevera_grom03_0-rs:grom03
Online IBM.ResourceGroup:db2_stevera_grom04_0-rg Nominal=Online
        '- Online IBM.Application:db2_stevera_grom04_0-rs
                '- Online IBM.Application:db2_stevera_grom04_0-rs:grom04
Online IBM.ResourceGroup:db2_stevera_stevera_SVTDB-rg Nominal=Online
        |- Online IBM.Application:db2_stevera_stevera_SVTDB-rs
                |- Online IBM.Application:db2_stevera_stevera_SVTDB-rs:grom03
                '- Offline IBM.Application:db2_stevera_stevera_SVTDB-rs:grom04
        '- Online IBM.ServiceIP:db2ip_9_26_124_22-rs
                |- Online IBM.ServiceIP:db2ip_9_26_124_22-rs:grom03
                '- Offline IBM.ServiceIP:db2ip_9_26_124_22-rs:grom04
```

10. 迁移 TSA 域:

```
root@grom04:# export CT_MANAGEMENT_SCOPE=2

root@grom04:# runact -c IBM.PeerDomain CompleteMigration Options=0
Resource Class Action Response for CompleteMigration

root@grom04:# samctrl -m

Ready to Migrate! Are you Sure? [Y|N]:.

Y
```

11. 对于集群组件，确保 **MixedVersions** 不再设置为 Yes:

```
root@grom04:# lsrpdomain
Name   OpState   RSCTActiveVersion   MixedVersions   TSPort   GSPort
-----  --------  ------------------  --------------  -------  -------
test   Online    2.5.1.2             No              12347    12348
```

12. 确保现行版本号 (AVN) 与 HA 管理器的已安装版本号 (IVN) 相匹配:

```
root@grom04:# lssrc ls IBM.RecoveryRM |grep VN
   Our IVN         : 2.2.0.7
   Our AVN         : 2.2.0.7
```

13. 可选: 在 grom04 机器上以实例所有者 stevera 的身份执行接管以让 grom04 成为 HADR 主数据库（就如原来的主数据库）。

使用分割镜像来克隆数据库

在DB2 pureScale 环境外部的某个环境中，使用以下过程来创建克隆数据库。尽管您可以写入克隆数据库，但通常此类数据库用于只读活动，例如，运行报告。

关于此任务

如果配置主数据库以进行日志归档，那么克隆的数据库将共享相同的日志归档配置。如果克隆的数据库可以访问归档日志位置，那么这可能会导致克隆的数据库将日志文件归档到与主数据库相同的位置，并可影响这两个数据库的可恢复性。当克隆的数据库最初使用与主数据库不同的日志链时，主数据库最终会使用与克隆的数据库相同的日志链值。在运行 **db2inidb** 命令之前，应该先将克隆的数据库的日志归档目标更改为与主数据库不同的目标，以避免可恢复性问题。

不能备份克隆的数据库，不能在原始系统上复原备份映像，也不能通过原始系统上生成的日志文件前滚。克隆的数据库仅提供在暂挂 I/O 时数据库的瞬时副本；在对克隆执行 **db2inidb** 命令之后，将回滚所有其他存在的未落实工作。

过程

要克隆数据库:

1. 使用以下命令来连接至主数据库:

   ```
   db2 connect to db_name
   ```

2. 使用以下命令来暂挂主数据库上的 I/O 写操作:

   ```
   db2 set write suspend for database
   ```

   注: 当数据库处于暂挂状态时，您不应该运行其他实用程序或工具。您只应生成数据库副本。可选择在发出 **SET WRITE SUSPEND** 之前清空所有缓冲池，以将恢复时间减到最少。可以使用 FLUSH BUFFERPOOLS ALL 语句来清空所有缓冲池。

3. 使用相应的操作系统级别和储存器级别命令从主数据库创建一个或多个分割镜像。

   注:

   - 您务必复制整个数据库目录（其中包括卷目录）。还必须复制日志目录和存在于数据库目录之外的任何容器目录。要收集此信息，请参阅 DBPATHS 管理视图，该视图显示了需要分割的数据库的所有文件和目录。
   - 如果您对 **SET WRITE** 命令指定了 EXCLUDE LOGS，请勿将日志文件包括在副本中。

4. 使用以下命令来恢复主数据库上的 I/O 写操作:

   ```
   db2 set write resume for database
   ```

5. 在辅助系统上对镜像数据库进行编目。

   注: 缺省情况下，镜像数据库与主数据库不能存在于同一个系统上。它必须位于具有相同目录结构的辅助系统上并使用与主数据库相同的实例名。如果镜像数据库与主数据库必须存在于同一个系统上，那么可使用 **db2relocatedb** 实用程序或 **db2inidb** 命令的 **RELOCATE USING** 选项来实现这一点。

6. 使用以下命令来启动辅助系统上的数据库实例:

   ```
   db2start
   ```

7. 在辅助系统上初始化镜像数据库:

   ```
   db2inidb database_alias as snapshot
   ```

   如果需要，请指定 **db2inidb** 命令的 RELOCATE USING 选项来重定位克隆数据库:

   ```
   db2inidb database_alias as snapshot relocate using relocatedbcfg.txt
   ```

   其中 relocatedbcfg.txt 文件包含重定位数据库所需的信息。

注:

- 此命令将回滚分割发生时未完成的事务，启动新的日志链序列以使主数据库的所有日志都不能在克隆数据库上重放。
- 如果为主数据库配置了日志归档功能，那么克隆的数据库将共享相同的日志归档配置。这意味着当克隆的数据库可以访问主数据库所使用的相同位置时，克隆的数据库会尝试将日志文件归档到该位置。虽然克隆数据库最初使用与主数据库不同的日志链，但是主数据库最终会使用与克隆的数据库相同的日志链值。这可能会导致主数据库将日志文件归档到克隆数据库所归档的日志文件的顶部，或者反过来。这可能会影响这两个数据库的可恢复性。应该将克隆的数据库的日志归档目标更改为与主数据库的日志归档目标不同，以避免这些问题。

在 DB2 pureScale 环境中使用分割镜像来克隆数据库

在 DB2 pureScale环境中使用以下过程来创建克隆数据库。尽管您可以写入克隆数据库，但通常此类数据库用于只读活动，例如，运行报告。

关于此任务

如果配置主数据库以进行日志归档，那么克隆的数据库将共享相同的日志归档配置。如果克隆的数据库可以访问归档日志位置，那么这可能会导致克隆的数据库将日志文件归档到与主数据库相同的位置，并可影响这两个数据库的可恢复性。当克隆的数据库最初使用与主数据库不同的日志链时，主数据库最终会使用与克隆的数据库相同的日志链值。在运行 **db2inidb** 命令之前，应该先将克隆的数据库的日志归档目标更改为与主数据库不同的目标，以避免可恢复性问题。

不能备份克隆的数据库，不能在原始系统上复原备份映像，也不能通过原始系统上生成的日志文件前滚。克隆的数据库仅提供在暂挂 I/O 时数据库的瞬时副本；在对克隆执行 **db2inidb** 命令之后，将回滚所有其他存在的未落实工作。

过程

要克隆数据库:

1. 使用以下命令来连接至主数据库:

   ```
   db2 connect to <db_namd>
   ```

2. 通过抽取并导入主集群的设置，在辅助集群上配置 General Parallel File System (GPFS)。在主集群上，运行以下 GPFS 命令:

   ```
   mmfsctl filesystem syncFSconfig -n remotenodefile
   ```

   其中 *remotenodefile* 是辅助集群中主机的列表。

3. 使用以下命令来列示集群管理器域:

   ```
   db2cluster -cm -list -domain
   ```

4. 使用以下命令来停止集群中每台主机上的集群管理器:

   ```
   db2cluster -cm -stop -host host -force
   ```

   注: 您关闭的最后一台主机必须是您将从中发出此命令的主机。

5. 使用以下命令来停止辅助系统上的 GPFS 集群:

   ```
   db2cluster -cfs -stop -all
   ```

6. 使用以下命令来暂挂主数据库上的 I/O 写操作:

```
db2 set write suspend for database
```

注: 当数据库处于暂挂状态时，您不应该运行其他实用程序或工具。您只应生成数据库副本。可选择在发出 **SET WRITE SUSPEND** 之前清空所有缓冲池，以将恢复时间减到最少。可以使用 FLUSH BUFFERPOOLS ALL 语句来清空所有缓冲池。

7. 使用以下命令来确定必须暂挂和复制的文件系统:

   ```
   db2cluster -cfs -list -filesystem
   ```

8. 使用以下命令来暂挂每个包含数据或日志数据的 GPFS 文件系统:

   ```
   /usr/lpp/mmfs/bin/mmfsctl filesystem suspend-write
   ```

   其中 *filesystem* 表示包含数据或日志数据的文件系统。

   注: 当 GPFS 文件系统处于暂挂状态时，将仅阻塞写操作。

9. 使用相应的操作系统级别和储存器级别命令从主数据库创建一个或多个分割镜像。

   注:

   - 您务必复制整个数据库目录（其中包括卷目录）。还必须复制日志目录和存在于数据库目录之外的任何容器目录。要收集此信息，请参阅 DBPATHS 管理视图，该视图显示了需要分割的数据库的所有文件和目录。
   - 如果您对 **SET WRITE** 命令指定了 EXCLUDE LOGS，请勿将日志文件包括在副本中。

10. 对每个已暂挂的文件系统使用以下命令来恢复已暂挂的 GPFS 文件系统:

    ```
    /usr/lpp/mmfs/bin/mmfsctl filesystem resume
    ```

    其中 *filesystem* 表示包含数据或日志数据的暂挂文件系统。

11. 恢复主数据库上的 I/O 写操作:

    ```
    db2 set write resume for database
    ```

12. 使用以下命令来启动辅助系统上的 GPFS 集群:

    ```
    db2cluster -cfs -start -all
    ```

13. 使用以下命令来启动集群管理器

    ```
    db2cluster -cm -start -domain domain
    ```

14. 在辅助系统上对镜像数据库进行编目:

    注: 缺省情况下，镜像数据库与主数据库不能存在于同一个系统上。它必须位于具有相同目录结构的辅助系统上并使用与主数据库相同的实例名。如果镜像数据库与主数据库必须存在于同一个系统上，那么可使用 **db2relocatedb** 实用程序或 **db2inidb** 命令的 **RELOCATE USING** 选项来实现这一点。

15. 使用以下命令来启动辅助系统上的数据库实例:

    ```
    db2start
    ```

16. 使用以下命令来初始化辅助系统上的镜像数据库:

    ```
    db2inidb database_alias as snapshot
    ```

    如果需要，请指定 **db2inidb** 命令的 **RELOCATE USING** 选项来重定位克隆数据库:

    ```
    db2inidb database_alias as snapshot relocate using relocatedbcfg.txt
    ```

    其中 relocatedbcfg.txt 文件包含重定位数据库所需的信息。

注:

- 此命令将回滚分割发生时未完成的事务，启动新的日志链序列以使主数据库的所有日志都不能在克隆数据库上重放。
- 如果为主数据库配置了日志归档功能，那么克隆数据库将共享相同的日志归档配置。如果克隆的数据库可以访问日志归档目标，那么备用数据库会在执行前滚时自动从该目标检索日志文件。然而，一旦备用数据库脱离前滚暂挂状态，那么克隆数据库会尝试将日志文件归档到主数据库所使用的相同位置。虽然备用数据库最初使用与主数据库不同的日志链，但是主数据库最终会使用与克隆的数据库相同的日志链值。这可能会导致主数据库将日志文件归档到克隆的数据库所归档的日志文件的顶部，或者反过来。这可能会影响这两个数据库的可恢复性。应该将克隆的数据库的日志归档目标更改为与主数据库的日志归档目标不同，以避免这些问题。

方案: 更改系统时钟

调整或更改系统时钟时，不需要停止 DB2 数据库管理器。DB2 for Linux, UNIX, and Windows 成功地处理了全世界一年两次的夏令时更改，且没有出现任何问题。

同时，全面支持使用 NTP 以同步系统间的时钟的配置。

关于此任务

更改系统时间时，必须注意一些最佳实践。

限制

在绝大多数方案中，更改系统时钟时，绝对不会存在任何影响。

多数时间轮换发生时，您必须注意两种情况。

- 如果执行时间点恢复，那么需要注意任何重要时间轮换。
- 函数定义包括函数创建的时间和日期（采用时间戳记的形式）。函数调用期间，DB2 for Linux, UNIX, and Windows 将尝试解析函数定义。在函数解析过程中，会检查在创建时记录在函数定义中的时间戳记值。如果将系统时钟调回到创建函数之前的时间，那么 DB2 for Linux, UNIX, and Windows不会解析对这些函数的引用。

过程

用于避免这两种情况的最佳实践:

1. 如果您向前移动时间，那么转至步骤 3。
2. 如果您向后移动 *X* 分钟时间:
   a. 选择执行更改的时间（如果过去 *X* 分钟未创建新函数，并且未来 *X* 分钟内不进行任何更新事务）。
   b. 如果找不到步骤 a 中概述的时间，那么仍可在 DB2 for Linux, UNIX, and Windows 联机的情况下将系统时钟后移 *X* 分钟。但您必须接受以下结论:
      - 您可能无法使用时间点恢复来恢复至 *X* 分钟内的某点。即，您可能无法恢复在 X 分钟内执行的一小部分更新事务。
      - 在更改后的 *X* 分钟，在更改前 X 分钟内创建的函数可能无法解析。
3. 更改系统时钟。

结果

通过遵循所概述的最佳实践，您将能够避免更改系统时钟时的任何潜在时间点恢复或函数解析问题。

同步主数据库和备用数据库

有一种高可用性策略是配备一个主数据库，一个辅助或备用数据库，前者出现故障时由后者接管操作。

如果备用数据库必须接管故障主数据库的数据库操作，那么它必须包含完全相同的数据，了解所有未完成事务并且以与主数据库服务器不发生故障时完全相同的方式继续数据库处理。更新备用数据库，使之成为主数据库副本的进行过程称为同步。

开始之前

在同步主数据库和备用数据库之前，必须:

- 创建并配置主数据库和备用数据库。
- 配置主数据库和备用数据库之间的通信。
-

  选择同步策略（例如，日志装入、日志镜像、暂挂 I/O 和磁盘镜像或 HADR。）

  有几种保持主数据库服务器和备用数据库服务器同步的策略:
  - 将日志从主数据库装入备用数据库并在备用数据库上前滚;
  - 同时将数据库日志写入主数据库和备用数据库，称为日志镜像;
  - 使用具有磁盘镜像的暂挂 I/O 支持来定期获取主数据库的副本、分割镜像并将副本初始化为新的备用数据库服务器；以及
  - 使用 DB2 高可用性灾难恢复 (HADR) 功能等可用性功能来保持主数据库和备用数据库同步。

过程

1. 如果使用日志来同步主数据库和辅助或备用数据库，请配置 DB2 数据库以为您执行所需要日志管理。 例如，如果希望 DB2 数据库为日志生成镜像，那么可将 **mirrorlogpath** 配置参数设置为日志另一副本的保存位置。
2. 如果使用 DB2 数据库暂挂 I/O 功能来分割主数据库的磁盘镜像，那么您必须执行以下操作:
   a. 初始化主数据库的磁盘镜像。
   b. 当需要分割主数据库的镜像时，请按照“将分割镜像用作备用数据库”主题中的指示信息进行操作。
3. 如果使用 HADR 功能来管理主数据库和备用数据库的同步，请为 HADR 配置 DB2 数据库，并允许 DB2 数据库为您同步主数据库和备用数据库。

创建表空间时解决日志重放错误

创建表空间时解决日志重放错误

如果在主数据库上创建表空间，并且对备用数据库执行的日志重放由于容器不可用而失败，那么主数据库不会接收到指示日志重放失败的错误消息。

要检查日志重放错误，在创建新的表空间时必须监视 **db2diag** 日志文件以及备用数据库上的管理通知日志文件。

如果执行了接管操作，那么您创建的新表空间在新的主数据库上不可用。要从此情况恢复，应通过备份映像在新的主数据库上复原表空间。

在以下示例中，表空间 **MY_TABLESPACE** 是在数据库 **MY_TABLESPACE** 用作新的主数据库之前在该数据库上复原的:

1. **db2 connect to my_database**
2. **db2 list tablespaces show detail**

   注: 运行 **db2 list tablespaces show detail** 命令以显示所有表空间的状态以及获取步骤 5 所需的表空间标识号。
3. db2 stop hadr on database my_database
4. db2 ″restore database my_database tablespace (my_tablespace) online redirect″
5. db2 ″set tablespace containers for my_tablespace_ID_# ignore rollforward container operations using (path '/my_new_container_path/')″
6. db2 ″restore database my_database continue″
7. db2 rollforward database my_database to end of logs and stop tablespace ″(my_tablespace)″
8. db2 start hadr on database my_database as primary

DB2 高可用性灾难恢复 (HADR) 复制的操作

DB2 高可用性灾难恢复 (HADR) 使用数据库日志将数据从主数据库复制到备用数据库。在备用数据库上重放日志时，某些活动可能会导致备用数据库落后于主数据库。

某些活动要进行大量记录，它们生成的大量日志文件可能会导致存储问题。虽然使用日志将数据复制到备用数据库是可用性策略的核心，但记录本身可能会对解决方案的可用性产生负面影响。合理设计维护策略，配置系统以尽可能降低日志记录的负面影响，并允许日志记录保护您的事务数据。

在高可用性灾难恢复 (HADR) 中，将以下操作从主数据库复制到备用数据库:

- 数据定义语言（DDL）
- 数据操作语言（DML）
- 缓冲池操作
- 表空间操作
- 联机重组
- 脱机重组
- 存储过程和用户定义的函数（UDF）的元数据（但不是相关对象或库文件）

联机重组过程中，详细记录所有操作。结果，HADR 可以复制操作，而不会使备用数据库比它在进行更多典型数据库更新时更加远远地落在后面。但是，由于生成大量日志记录，所以此行为可能对系统产生较大影响。

如果未如联机重组那样大量地记录脱机重组，通常按几百或几千个受影响的行来记录操作。这意味着备用数据库将落后，因为它等待每个日志记录，然后立刻重放许多更新。如果脱机重组是非集群的，那么在整个重组操作之后生成单一日志记录。此方式在最大程度上影响备用数据库与主数据库保持同步的功能。备用数据库从主数据库接收日志记录之后，将执行整个重组过程。

HADR 不复制存储过程、UDF 对象和库文件。必须在主数据库和备用数据库中相同路径上创建文件。如果备用数据库无法找到引用的对象或库文件，那么备用数据库上的存储过程或 UDF 调用将失败。

DB2 高可用性灾难恢复 (HADR) 不复制的操作

DB2 高可用性灾难恢复 (HADR) 使用数据库日志将数据从主数据库复制到备用数据库。主数据库允许不进行日志记录的操作，但不会将此类操作复制到备用数据库。

如果要在备用数据库中反映未日志记录的操作（例如，对历史记录文件的更新），那么必须执行额外的步骤来实现此目的。

以下是一些情况示例，在这些情况下，不会将主数据库上的操作复制到备用数据库:

- 在指定了 NOT LOGGED INITIALLY 选项的情况下创建的表不会被复制。在 HADR 备用数据库作为主数据库接管后尝试访问这些表会导致错误。
- 将复制所有已进行日志记录的 LOB 列。将不会复制未进行日志记录的 LOB 列。但是，在备用数据库上会为它们分配空间，将二进制的零作为该列的值。
- 不复制使用 **UPDATE DATABASE CONFIGURATION** 和 **UPDATE DATABASE MANAGER CONFIGURATION** 命令对数据库配置所作的更新。
- 不复制数据库配置参数和数据库管理器配置参数。
- 对于用户定义的函数 (UDF) 来说，不复制对数据库外部的对象（例如相关的对象和库文件）所作的更改。您需要通过其他方法在备用数据库上对它们进行设置。
- 不会自动地将恢复历史记录文件 (db2rhist.asc) 以及对其所作的更改从主数据库复制到备用数据库。

  通过发出带 **REPLACE HISTORY FILE** 参数的 **RESTORE DATABASE** 命令，可以将历史记录文件的原始副本（从主数据库的备份映像中获取）放到备用数据库上:

  ```
  RESTORE DB KELLY REPLACE HISTORY FILE
  ```

  初始化 HADR 并接着对主数据库执行备份活动后，备用数据库上的历史记录文件就已过期。但是，每个备份映像中都存储了历史记录文件的一个副本。通过使用以下命令从备份映像中抽取历史记录文件，可以更新备用数据库上的历史记录文件:

  ```
  RESTORE DB KELLY HISTORY FILE
  ```

  请不要使用正规操作系统命令将数据库目录中的历史记录文件从主数据库复制到备用数据库。进行复制时，如果主数据库正在更新历史记录文件，那些文件就会损坏。

  如果执行接管操作并且备用数据库有最新的历史记录文件，那么对新的主数据库执行的备份和复原操作将在历史记录文件中生成新记录，并且与原始主数据库上生成的记录完全混合。如果历史记录文件过期或者缺少条目，那么可能无法进行自动增量复原；而是，您将需要执行手动增量复原操作。

DB2 高可用性灾难恢复 (HADR) 备用数据库状态

无论何时，高可用性灾难恢复 (HADR) 备用数据库都处于五个状态之一：本地同步复制、远程同步复制暂挂、远程同步复制、对等和断开对等。这些状态由日志装入状态定义。无论该状态为何，日志重放都会并发进行，重放所有可用的日志。

主数据库日志位置、备用数据库日志接收位置和备用数据库日志重放位置均由 HADR 的标准监视接口报告：MON_GET_HADR 表函数和指定了 **-hadr** 参数的 **db2pd** 命令。备用数据库的状态在 HADR_STATE 字段中报告。如果主数据库已连接到一个备用数据库，那么监视接口会将备用数据库的状态作为其 HADR_STATE 来报告；否则，它会报告 DISCONNECTED。

第 162 页的图 10 显示不同备用数据库状态的变化顺序。

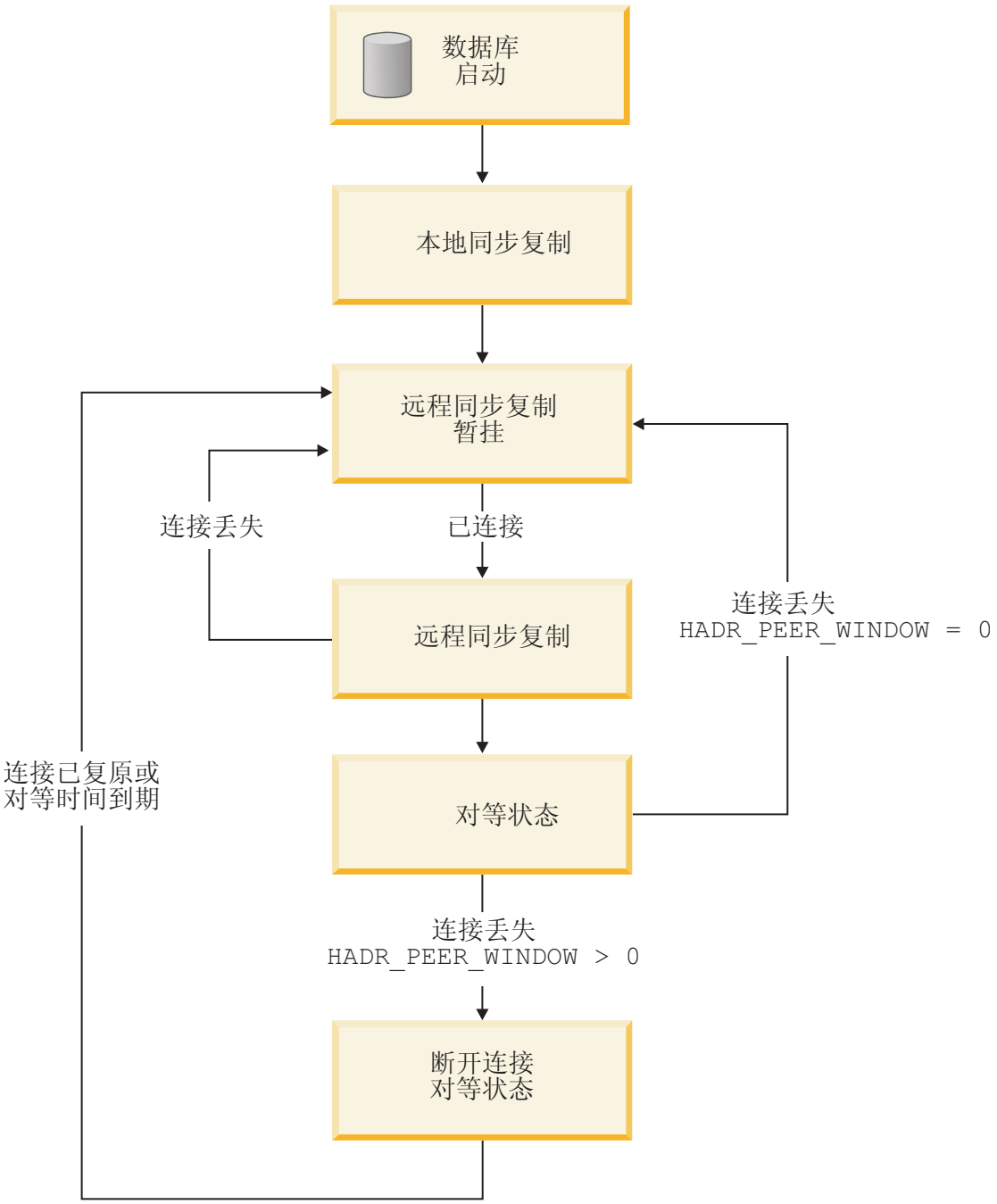


图 10. 备用数据库的状态

本地同步复制状态

使用 HADR 功能，当启动备用数据库时，它进入本地同步复制状态，并读取其本地日志路径中的日志文件以确定哪些日志在本地可用。在此状态下，即使已配置日志归档方法，也不会从归档检索日志。并且，在此状态下，不需要连接到主数据库；但是，如果连接不存在，备用数据库会尝试与主数据库连接。当到达本地日志文件末尾时，备用数据库进入远程同步复制暂挂状态。

远程同步复制暂挂状态

进入远程同步复制暂挂状态时，如果尚未建立与主数据库的连接，那么备用数据库将等待连接。 建立连接之后，备用数据库会获取主数据库的当前日志链信息。这使备用数据库能够从归档检索日志文件并验证日志文件是否有效（如果配置了日志归档）。

在远程同步复制和对等状态下，如果备用数据库失去与主数据库的连接，那么它将回到远程同步复制暂挂状态。重新建立连接时，备用数据库会尝试从归档检索日志。因此，如果您配置了共享归档设备，那么备用数据库也许能够找到比使用独立归档设备时更多的日志。此行为倾向于从归档检索而不是通过 HADR 连接从主数据库提供日志文件，以便使对主数据库的影响降至最低。

远程同步复制状态

在远程同步复制状态下，主数据库从其日志路径中或者通过日志归档方法读取日志数据，并将日志数据发送到备用数据库。备用数据库接收到主数据库机器磁盘上的所有日志数据后，主数据库和备用数据库进入对等状态。如果您正在使用 SUPERASYNC 同步方式，那么主数据库和备用数据库永远不会进入对等状态。它们会永远留在远程同步复制状态，这可以防止在对等状态下发生主数据库日志写入阻塞。

如果在远程同步复制状态期间主数据库和备用数据库之间的连接断开，那么备用数据库将进入远程同步复制暂挂状态。

对等状态

在对等状态下，每当主数据库将其日志页清空到磁盘时，日志数据从主数据库的日志写缓冲区直接递送到备用数据库。HADR 同步方式指定了主数据库是否等待备用数据库发送已收到日志数据的确认消息。日志页始终写至备用数据库上的本地日志文件。此行为可针对崩溃情况提供保护，并允许在发生接管后将文件归档到新的主数据库上（如果已将该文件归档到旧的主数据库）。在将接收到的日志页写至本地磁盘后，可在备用数据库上重放这些日志页。如果禁用了日志假脱机（缺省设置），那么重放将仅从日志接收缓冲区读取日志。

如果日志重放较慢，那么接收缓冲区可能会被填满，这时备用数据库会停止接收新日志。如果发生这种情况，那么主数据库日志写入将被阻塞。如果启用日志假脱机，那么将释放部分日志缓冲区（即使其尚未重放）。稍后，日志重放进程会从磁盘读回日志数据。如果假脱机设备填满或达到配置的假脱机限制，那么备用数据库仍会停止接收，并且主数据库仍会被阻塞。

如果在对等状态期间主数据库和备用数据库之间的连接断开，并且 **hadr_peer_window** 数据库配置参数设置为 0（缺省设置），那么备用数据库将进入远程同步复制暂挂状态。但是，如果在对等状态期间主数据库和备用数据库之间的连接断开，并且 **hadr_peer_window** 参数设置为非零值（意味着您配置了*对等时间*），那么备用数据库将进入断开连接的对等状态。

断开连接的对等状态

如果配置了对等时间并且在对等状态期间主数据库和备用数据库之间的连接断开，那么在配置的时间段（称为*对等时间*）内，或直到备用数据库重新连接（以这两种情况

中先发生者为准），主数据库仍像主数据库与备用数据库处于对等状态那样继续运行。主数据库和备用数据库之间的连接断开，但它们仍像处于对等状态那样运行的状态，称为*断开连接的对等*。

配置对等时间的优点是可以降低多重或级联失败期间丢失事务的风险。如果没有对等时间，当主数据库和备用数据库之间的连接断开时，主数据库将立即脱离对等状态并继续进行事务处理。这些事务将不会被复制到备用数据库。如果主数据库在与备用数据库的连接断开后的短时间内发生故障，那么在故障转移中发生事务丢失的风险很高。启用对等时间时，在主数据库断开与处于对等状态下的备用数据库的连接后，在指定的一段时间内，主数据库会阻塞事务处理，以便防止发生级联故障。并且，备用数据库可以在对等时间内进行接管，没有数据丢失的风险。

配置对等时间的缺点是主数据库上的事务处理将要花较长的时间，甚至当主数据库在对等时间中等待复原与备用数据库的连接或等待对等时间到期时发生超时。并且，间歇性的网络故障可能对主数据库事务处理造成严重影响。

可以使用 MON_GET_HADR 表函数或带 **-hadr** 参数的 **db2pd** 命令确定对等时间大小，即 **hadr_peer_window** 数据库配置参数的值。

手动将日志文件从主数据库复制到备用数据库

一种用于同步主数据库和备用数据库的方法是将主数据库日志文件手动复制到备用数据库日志路径或溢出日志路径（如果已配置）。当主数据库与备用数据库之间存在较大的日志间隔时（例如，因为备用数据库长时间停机），这种方法尤其有帮助。这种方法可以缩短备用数据库从归档检索日志时的延迟，还可以降低对主数据库的影响（主数据库将必须提供这些日志文件，而主数据库可能必须从归档检索这些日志文件）。有一点很重要，那就是必须在激活备用数据库之前完成此步骤。激活备用数据库后，它将继续搜索本地日志文件，重试从归档检索，并使主数据库忙于日志递送（如上所述）。在备用数据库已激活后将日志文件复制到该备用数据库会妨碍其正常操作。

确定 HADR 备用数据库状态

DB2 高可用性灾难恢复 (HADR) 备用数据库所处的状态决定了它可以执行的操作。建议通过两种方法来确定备用数据库的状态: **db2pd** 命令和 MON_GET_HADR 表函数。

过程

要确定主-备用 HADR 数据库对中 HADR 备用数据库的状态，请执行以下操作:

- 从主数据库或备用数据库发出指定了 **-hadr** 参数的 **db2pd** 命令。
  - 如果从主数据库发出查询，那么该命令将对 HADR 设置中的每个备用数据库返回一组数据。
  - 如果从备用数据库发出该命令，那么该命令仅返回一组数据，因为备用数据库不知道任何其他备用数据库，即使 HADR 设置处于多备用数据库方式。
- 在主数据库或备用数据库上使用 MON_GET_HADR 表函数发出查询:

  ```
  db2 "select STANDBY_ID, HADR_STATE, from table (mon_get_hadr(NULL))"
  ```

  将返回以下信息:

```
STANDBY_ID HADR_STATE
---------- ----------------------
         1 PEER
         2 REMOTE_CATCHUP
         3 REMOTE_CATCHUP

3 record(s) selected.
```

– 如果对主数据库发出查询，那么表函数将对 HADR 设置中的每个备用数据库返回一行信息。
– 如果对备用数据库发出查询，那么该表函数将仅返回一行信息，因为备用数据库不知道任何其他备用数据库，即使 HADR 设置处于多备用数据库方式。

在 HADR 备用数据库上从表空间错误中恢复

如果在日志重放期间 HADR 备用数据库在特定表空间上遇到错误，那么备用数据库将继续重放有关其他表空间的日志，但是会停止重放有关受影响的表空间的日志。

关于此任务

受影响的表空间的表空间状态将更改为复原暂挂、前滚暂挂或脱机。因为在备用数据库接管主角色时此表空间中的数据将不可用，所以您需要在此数据库上恢复此表空间。

过程

1. 找出该错误的根本原因并进行改正。 可能的原因包括:
   - 空间不足
   - 未安装文件系统
   - 找不到装入副本
2. 修复受影响的表空间。为此，通过复原主数据库的备份映像来完全重新初始化备用数据库。

HADR 角色切换和停顿的表空间

在高可用性灾难恢复 (HADR) 环境中，在角色切换期间不会保留表空间停顿。

在主数据库上停顿表空间时，不会生成日志记录，所以对备用数据库没有影响。如果在释放停顿前备用数据库必须作为主数据库接管，那么该表空间在新的主数据库上将完全可用。您应该知道，如果继续完成要求表空间在原始主数据库上停顿的作业，那么在新的主数据库上，该作业不再受停顿保护

如果存在角色切换（即，如果旧主数据库现在是新备用数据库），那么对新主数据库上的表空间的更改将在新备用数据库上重放。但是，如果主角色通过故障恢复过程归还给旧主数据库，那么停顿状态仍将对该表空间生效。

HADR 延迟重放

HADR 延迟重放有助于防止由于错误事务而导致的数据丢失。要实现 HADR 延迟重放，请对 HADR 备用数据库设置 **hadr_replay_delay** 数据库配置参数。

有意延迟重放时，会延迟重放备用数据库的日志，从而使该备用数据库保持在早于主数据库的时间点的某个时间点。如果对主数据库执行了错误事务，那么您需要直到经

过已配置的时间延迟之后才能执行操作，以防止对备用数据库重放错误事务。要恢复已丢失的数据，您可以将此数据复制回主数据库，也可使备用数据库接管作为新的主数据库。

延迟重放的工作方式是将对主数据库生成的日志流中的时间戳记与备用数据库的当前时间进行比较。因此，使主数据库与备用数据库的时钟保持同步非常重要。按照以下等式对备用数据库重放事务落实:

（备用数据库上的当前时间 - hadr_replay_delay 配置参数的值）>= 已落实的日志记录的时间戳记

应将 **hadr_replay_delay** 数据库配置参数设置为足够大的值，以便有时间检测主数据库的错误事务并作出反应。

您可以在“单个备用数据库”方式或“多个备用数据库”方式下使用此功能。在“多个备用数据库”方式下，通常有一个或多个备用数据库与主数据库保持同步以实现高可用性或者灾难恢复目的，为一个备用数据库配置了延迟重放以防止发生错误事务。如果您在“单个备用数据库”方式下使用此功能，那么您不应启用 IBM Tivoli System Automation for Multiplatforms，这是因为接管将失败。

对于延迟重放有若干项重要限制:

- 您只能对备用数据库设置 **hadr_replay_delay** 配置参数。
- 对于已启用延迟重放的备用数据库执行 **TAKEOVER** 命令将失败。您必须首先将 **hadr_replay_delay** 配置参数设置为 0，然后取消激活并重新激活备用数据库以使新值生效，然后发出 **TAKEOVER** 命令。
- 延迟重放功能仅在 SUPERASYNC 方式下受支持。因为已延迟进行日志重放，因此许多未重放的日志数据可能会在备用数据库中累积，从而填满接收缓冲区和池（如果已配置）。在其他同步方式下，这将导致主数据库阻塞。

  此功能的目标是防止应用程序发生错误。如果您想要使用此功能并确保在主数据库发生故障的情况下不丢失任何数据，那么可考虑对主要的备用数据库使用具有更同步的设置的多个备用数据库设置。

建议

延迟重放和灾难恢复

如果您要将备用数据库用于进行灾难恢复以及防止发生错误事务，那么可考虑使用较短的延迟。

延迟重放和 HADR 的“在备用数据库上读取”功能

如果您要将备用数据库用于“在备用数据库上读取”，那么可考虑使用较短延迟，以便阅读器会话可以查看更多最新数据。此外，因为“在备用数据库上读取”是在“未落实的读”隔离级别运行，所以它可以查看已应用、但是尚未落实的更改，从技术上而言仍然是延迟重放这些更改。当您将备用数据库前滚到您需要的 PIT 时，可以在错误事务恢复过程中回滚这些未落实的事务，然后将它们停止。

延迟重放和日志假脱机

如果您启用延迟重放，那么建议您还要通过设置 **hadr_spool_limit** 数据库配置参数来启用日志假脱机。由于是有意延迟，因此，重放位置可能远远落后于备用数据库上的日志接收位置。在不进行假脱机的情况下，日志接收超过重放位置的量只能为接收缓冲区的数量。在进行假脱机的情况下，备用数据库可以接

收超过重放位置的更多日志，从而更能防止在主数据库发生故障的情况下丢失数据。请注意，在任何一种情况下，由于必须采用 SUPERASYNC 方式，因此，主数据库将不会被延迟重放阻塞。

使用 HADR 延迟重放来恢复数据

通过使用 HADR 延迟重放功能，您可以恢复由于主数据库上的错误事务而丢失的数据，其方法是在重放该事务之前对备用数据库停止 HADR。

开始之前

必须已经对备用数据库启用延迟重放。

如果备用数据库上的日志重放（由 STANDBY_REPLAY_LOG_TIME 指示）已经经过了备用数据库上的错误事务的落实时间，那么您将无法使用以下过程来恢复数据。您可以使用附带 **-hadr** 参数的 **db2pd** 命令或者使用 MON_GET_HADR 表函数来确定 STANDBY_REPLAY_LOG_TIME。

限制：无法接管对其设置 **hadr_replay_delay** 配置参数的备用数据库并将其作为主数据库；您必须先对该备用数据库禁用延迟重放。

过程

要从错误事务进行恢复，请对为其启用了延迟重放的备用数据库执行下列步骤：

1. 验证计时：
   a. 请确保备用数据库尚未重放此事务。STANDBY_REPLAY_LOG_TIME 值不得已到达错误事务落实时间。
   b. 请确保备用数据库已接收到相关日志。STANDBY_LOG_TIME 值（它指示接收到的日志）必须已到达错误事务落实时间之前的某个 PIT，但是接近错误事务落实时间。这将是步骤 3 中所使用的前滚 PIT。如果备用数据库尚未接收到足够的日志文件，那么您可以一直等到提供了更多日志为止，但是存在延迟时间到达错误事务时间的风险。例如，如果延迟时间为 1 小时，那么应当在错误事务时间之后不超过 50 分钟就停止 HADR（允许保留 10 分钟的安全余量），即使日志装入尚未到达您需要的 PIT 也是如此。

      或者，如果共享日志归档可用，并且已将日志归档，那么不需要等待。如果日志尚未归档，那么可以使用 **ARCHIVE LOG** 命令将日志归档。否则，用户可以将完整的日志文件从主数据库手动复制到延迟的备用数据库（首选溢出日志路径，否则使用日志路径）。对于这些备用方法，请首先取消激活备用数据库，以避免妨碍备用数据库日志装入和重放。

   可以通过发出 db2pd -db *dbname* -hadr，或者通过对备用数据库启用“在备用数据库上读取”功能，然后发出以下查询（它使用 MON_GET_HADR 表函数）来确定这些时间：

   ```
   DB2 "select HADR_ROLE, STANDBY_ID, STANDBY_LOG_TIME, STANDBY_REPLAY_LOG_TIME,
   varchar(PRIMARY_MEMBER_HOST,20) as PRIMARY_MEMBER_HOST,
   varchar(STANDBY_MEMBER_HOST,20) as STANDBY_MEMBER_HOST
   from table (mon_get_hadr(NULL))"
   ```

2. 对备用数据库停止 HADR：

   ```
   DB2 STOP HADR ON DATABASE dbname
   ```

3. 将备用数据库前滚到您需要的 PIT，然后将其停止：

DB2 ROLLFORWARD DB *dbname* to *time-stamp* and STOP

4. 请使用下列其中一种方法:
   - 复原主数据库上丢失的数据:
     a. 从备用数据库中复制受影响的数据，然后将其发送回主数据库。

        如果错误事务已废弃某个表，那么您可以将此表从备用数据库中导出，然后将其导入到主数据库中。如果错误事务已删除某个表中的行，那么您可以从备用数据库中导出此表，然后对主数据库使用导入替换操作。
     b. 因为延迟重放的备用数据库的日志流已与主数据库的日志流分开，所以要重新初始化此备用数据库。因为任何其他备用数据库继续遵循主数据库，并且也会将对主数据库执行的任何数据修复复制到这些备用数据库中，所以不需要对这些备用数据库执行任何操作。
     c. 使用对主数据库生成的备份映像来复原数据库。随时都可以生成此映像。
     d. 除去备用数据库日志路径中的所有日志文件。此步骤很重要。步骤 3 中的 **ROLLFORWARD**... STOP 命令会使数据库日志流与主数据库分开。如果单独保留文件，那么新复原的数据库将遵循该日志流，并且也会与主数据库分开。此外，您可以在复原之前删除数据库以进行干净启动，但是您也将失去当前配置（其中包括 HADR 配置）。
     e. 对此数据库发出附带 AS STANDBY 选项的 **START HADR** 命令。此数据库应当会激活，并且连接至主数据库。
   - 使具有完整数据的备用数据库成为主数据库:
     a. 关闭旧的主数据库以避免发生分裂情况
     b. 对于延迟重放的数据库，将 **hadr_replay_delay** 配置参数设置为 0。如果需要，请重新配置诸如 **hadr_target_list** 等其他参数。然后，对此数据库运行附带 AS PRIMARY BY FORCE 选项的 **START HADR** 命令，以将其转换为新的主数据库。因为并不保证已配置的主要备用数据库（有可能是旧的主数据库）将能够进行连接，所以使用 BY FORCE 选项。
     c. 将客户机重定向至新的主数据库。
     d. 会将其他备用数据库自动重定向至新的主数据库。但是，如果在旧的主数据库与新的主数据库分开的时间（也就是步骤 3 中使用的 PIT）之后，备用数据库从旧的主数据库接收到日志，那么此备用数据库将被新的主数据库拒绝。如果发生这种情况，那么按照重新初始化旧的主数据库的同一过程重新初始化此备用数据库。
     e. 因为旧的主数据库的日志流已与新的主数据库的日志流分开，所以要重新初始化此旧的主数据库。
     f. 使用对新的主数据库生成的备份映像或者在步骤 3 中所使用的 PIT 之前对旧的主数据库生成的备份映像来复原数据库。
     g. 除去日志路径中的所有日志文件。如果您不执行此操作，那么新复原的数据库将遵循旧的主数据库的日志流，并与新的主数据库分开。此外，您可以在复原之前删除数据库以进行干净启动，但是您也会失去当前配置（其中包括 HADR 配置）。
     h. 对此数据库发出附带 AS STANDBY 选项的 **START HADR** 命令。此数据库应当会激活，并且连接至主数据库。

DB2 高可用性灾难恢复 (HADR) 管理

DB2 高可用性灾难恢复 (HADR) 管理包括配置和维护 HADR 系统状态。

对 HADR 的管理包括下列任务:

- 编目 HADR 数据库。
- 第 30 页的『初始化高可用性灾难恢复 (HADR)』
- 检查或改变与 HADR 相关的数据库配置参数。
- 第 207 页的『切换高可用性灾难恢复 (HADR) 中的数据库角色』
- 第 205 页的『执行 HADR 故障转移操作』
- 第 200 页的『高可用性灾难恢复 (HADR) 监视』
- 第 146 页的『停止 DB2 高可用性灾难恢复 (HADR)』

可以使用下列方法来管理 HADR:

- 命令行处理器
- DB2 管理 API
- IBM Data Studio V3.1 或更高版本中用于管理 HADR 的任务助手。

相关信息:

使用任务助手管理数据库

DB2 高可用性灾难恢复 (HADR) 命令

DB2 高可用性灾难恢复 (HADR) 功能为 DB2 高可用性数据库解决方案提供复杂的日志记录、故障转移和恢复功能。

无论 HADR 提供的功能有多复杂，您只需要直接命令 HADR 执行下列几个操作：启动 HADR，停止 HADR 以及让备用数据库作为主数据库接管操作。

可以使用三个高可用性灾难恢复 (HADR) 命令来管理 HADR:

- **START HADR**
- **STOP HADR**
- **TAKEOVER HADR**

要调用这些命令，请使用命令行处理器或管理 API。

发出带 AS PRIMARY 或 AS STANDBY 选项的 **START HADR** 命令会将数据库角色更改为指定角色（如果数据库还不是该角色）。如果数据库尚未处于激活状态，此命令还将激活该数据库。

STOP HADR 命令将 HADR 数据库（主数据库或备用数据库）更改为标准数据库。任何与 HADR 相关的数据库配置参数都保持不变，因此，您可以方便地重新激活该数据库以使其成为 HADR 数据库。

TAKEOVER HADR 命令（只能对备用数据库发出此命令）将备用数据库更改为主数据库。如果未指定 BY FORCE 选项，那么主数据库和备用数据库会切换角色。如果指定 BY FORCE 选项，那么备用数据库会单方面地切换为主数据库。在此情况下，备用数据库将尝试停止旧的主数据库上的事务处理。但是，不保证事务处理将会停止。只有在故障

转移情况下，才应使用 BY FORCE 选项以强制执行接管操作。在发出带 BY FORCE 选项的 **TAKEOVER HADR** 命令前，您应该尽可能地确保当前主数据库确实已发生故障，也可以自己将其关闭。

切换 HADR 数据库角色

可以动态地并且重复地在主角色与标准角色之间切换数据库。当数据库处于联机或脱机状态时，您可以发出带 AS PRIMARY 选项的 **START HADR** 命令和 **STOP HADR** 命令。

可以静态地在备用角色与标准角色之间切换数据库。仅当数据库处于前滚暂挂状态时，您才可以重复地执行此操作。当标准数据库处于脱机状态和前滚暂挂状态时，可发出带 AS STANDBY 选项的 **START HADR** 命令将标准数据库更改为备用数据库。当备用数据库处于脱机状态时，可使用 **STOP HADR** 命令将该数据库更改为标准数据库。发出 **STOP HADR** 命令后，数据库将保持前滚暂挂状态。发出带 AS STANDBY 选项的后续 **START HADR** 命令将使该数据库重新成为备用数据库。对备用数据库停止 HADR 后，如果发出带 STOP 选项的 **ROLLFORWARD DATABASE** 命令，那么不能使其重新成为备用数据库。由于该数据库已不再处于前滚暂挂状态，所以，您可以将其用作标准数据库。这称为获取备用数据库快照。在将现有备用数据库更改为标准数据库后，您应该考虑为实现高可用性而创建新的备用数据库。

要切换主数据库和备用数据库的角色，请在不使用 BY FORCE 选项的情况下执行接管操作。

要单方面地将备用数据库更改为主数据库（而不将主数据库更改为备用数据库），请执行强制接管操作。以后，可能能够重新集成旧的主数据库以使其成为新的备用数据库。

HADR 角色是持久的。一旦确定了 HADR 角色，数据库就保持具有该角色，即使重复地停止并接着重新启动 DB2 实例，或者取消激活或激活 DB2 数据库亦如此。

启动备用数据库的操作是异步的

如果发出带 AS STANDBY 选项的 **START HADR** 命令，那么相关引擎可分派单元 (EDU) 一成功启动该命令就会返回。此命令不会等待备用数据库连接至主数据库。相反，在主数据库连接到备用数据库之前，主数据库不会被视为已启动（对主数据库发出带 BY FORCE 选项的 **START HADR** 命令时例外）。如果备用数据库遇到错误（例如，连接被主数据库拒绝），那么带 AS STANDBY 选项的 **START HADR** 命令可能已成功返回。因此，HADR 无法将错误指示返回给任何用户提示。HADR 备用数据库会将一条消息写入 DB2 诊断日志并自行关闭。您应该监视 HADR 备用数据库的状态，以确保它成功地与 HADR 主数据库连接。

重放错误（备用数据库在重放日志记录时遇到的错误）也会导致备用数据库关闭。例如，如果没有足够的内存来创建缓冲池，或者在创建表空间时找不到路径，就会发生这些错误。您应该持续监视备用数据库的状态。

不要从某一使用已启用客户机重新路由的数据库别名的客户机运行 HADR 命令

如果已设置客户机自动重新路由，数据库服务器就有预定义的备用服务器，因此客户机应用程序可以在使用原始数据库服务器或使用备用服务器之间进行切换，并将工作中断降至最少。在这样的环境中，当客户机通过 TCP 连接至数据库时，实际连接可以

是连接至原始数据库，也可以是连接至替代数据库。实现了 HADR 命令以通过正规客户机连接逻辑来标识目标数据库。因此，如果为目标数据库定义了替代数据库，就难以确定命令实际上作用于哪个数据库。虽然 SQL 客户机不需要知道它所连接的数据库，但 HADR 命令必须应用于特定的数据库。为了适应这种限制，应该在服务器本地发出 HADR 命令以绕过客户机重新路由（客户机重新路由只影响 TCP/IP 连接）。

HADR 命令必须在具备有效许可证的服务器上运行

START HADR、**STOP HADR** 和 **TAKEOVER HADR** 命令要求已在执行该命令的服务器上安装有效 HADR 许可证。如果没有许可证，这些命令就会失败并返回特定于命令的错误代码（分别为 SQL1767N、SQL1769N 或 SQL1770N）以及原因码 98。要解决此问题，请使用 **db2licm** 来安装有效的 HADR 许可证，或者安装服务器分发包含有效 HADR 许可证的服务器版本。

HADR 多备用数据库

高可用性灾难恢复 (HADR) 功能支持多个备用数据库。使用多个备用数据库，您可以将数据放在两个以上站点中，这样通过一种技术就可以提供改进的数据保护。

以“多个备用数据库”方式部署 HADR 功能时，设置中最多可有 3 个备用数据库。您指定其中一个数据库作为*主体 HADR 备用数据库*；并将任何其他备用数据库作为*辅助 HADR 备用数据库*。两种类型的 HADR 备用数据库都通过直接 TCP/IP 连接与 HADR 主数据库同步，这两种类型都支持在备用数据库上读取，并且可配置这两种类型以执行延时日志重放。此外，可对任何备用数据库发出强制或非强制接管命令。主体备用数据库与辅助备用数据库之间存在一些重要差异，但是:

- 只有主体备用数据库支持 IBM Tivoli System Automation for Multiplatforms (SA MP) 自动化故障转移。必须对其中一个辅助备用数据库手动发出接管命令以使它们中的一个成为主数据库。
- 主体备用数据库支持所有 HADR 同步方式，但辅助备用数据库只能以 SUPERASYNC 方式运行。

使用多 HADR 备用数据库设置有许多优点。可使用 HADR 同时实现高可用性目标和灾难恢复目标，而不是使用 HADR 功能实现高可用性目标并使用另一种技术来实现灾难恢复目标。可以将主体备用数据库与主数据库部署在同一个位置。如果主数据库停运，那么主体备用数据库可在恢复时间目标内接管主数据库角色。还可将辅助备用数据库部署到较远位置，以防止会同时影响主数据库和主体备用数据库的大范围灾难。距离和主数据域与辅助备用数据库之间可能的网络延迟对主数据库的活动没有影响，因为辅助备用数据库使用的是 SUPERASYNC 方式。如果灾难影响到主数据库和主体备用数据库，那么可以对任一个辅助备用数据库发出接管命令。可以使用 **hadr_target_list** 数据库配置参数使另一个辅助备用数据库成为新的主体备用数据库。但是，即使该辅助数据库不具有可用的备用数据库，它也可以作为主数据库进行接管。例如，如果主数据库和主体备用数据库发生停运，那么一个辅助数据库即使没有对应的备用数据库，它也可以作为主数据库进行接管。但是，如果在该数据库成为新的主数据库之后将其停止，那么除非其主体备用数据库已启动，否则它将不能再作为 HADR 主数据库启动。

多备用数据库方式的限制

如果您计划以“多个备用数据库”方式部署 HADR 功能，那么有许多限制应注意。

限制如下所示:

- 最多可有 3 个备用数据库: 一个主体备用数据库以及最多两个辅助备用数据库。
- 只有主体备用数据库支持所有 HADR 同步方式，所有辅助备用数据库都将在 SUPERASYNC 方式下运行。
- IBM Tivoli System Automation for Multiplatforms (SA MP) 支持仅适用于主 HADR 数据库和其主体备用数据库之间。
- 必须对多备用设置中的所有数据库设置 **hadr_target_list** 数据库配置参数。每个备用数据库必须在其 **hadr_target_list** 设置中包括主数据库。

以多备用数据库方式初始化 HADR

以多备用数据库方式初始化 HADR 系统类似于以单备用数据库方式执行此操作。这两种方式的主要差别在于，您必须通过在设置中对所有数据库设置 **hadr_target_list** 数据库配置参数来启用多备用数据库方式。

关于此任务

本任务涵盖了如何在多备用数据库方式下初始化 HADR。如果要将单一备用数据库设置转换为多备用数据库设置，请参阅第 174 页的『在预先存在的 HADR 设置中启用多备用数据库方式』。

多备用数据库方式需要在所有参与的数据库上设置 **hadr_target_list** 配置参数。此参数列示在该数据库成为主数据库的情况下的备用数据库。即使在备用数据库上此参数也是必需的。相互包含是必需的（即，如果 A 的目标列表中有 B，那么 B 的目标列表中也必须有 A）。这确保在从任何备用数据库进行接管后，新主数据库总是可以保留旧主数据库作为其备用数据库。您在目标列表中指定的第一个备用数据库被指定为*主体 HADR 备用数据库*。其他备用数据库为*辅助 HADR 备用数据库*。目标列表不必始终包含所有参与者。并且，如果有多个备用数据库，那么不需要对称或对等；即使将数据库 B 指定为数据库 A 的主体备用数据库，也不必将数据库 A 指定为数据库 B 的主体备用数据库。对于在数据库 A 的目标列表中指定的每个备用数据库，其目标列表中也必须指定数据库 A。处理每个数据库的目标列表是一个重要步骤。

作为一种特殊情况，可以为多备用数据库方式仅配置一个备用数据库。例如，在多备用数据库方式下，您可以将两个数据库配置为主数据库和备用数据库。此行为与单备用设置的行为不同，这是因为多备用行为（例如，自动化配置）将生效，并且可以动态添加或除去备用数据库目标。

提示: 可在每个数据库的单一更新中执行步骤 2 到步骤 4。

过程

要在多备用数据库方式下初始化 HADR，请执行下列步骤:

1. 通过使用已复原备份或分割镜像创建备用数据库或数据库。有关如何执行此操作的指示信息，请参阅第 45 页的『初始化备用数据库』或第 30 页的『初始化高可用性灾难恢复 (HADR)』的步骤 2。
   - 在主数据库上，发出以下命令:

```
BACKUP DB dbname
```

- 在备用数据库上，发出以下命令:

```
RESTORE DB dbname
```

2. 在每个数据库上设置 **hadr_local_host**、**hadr_local_svc**、**hadr_local_svc** 和 **hadr_sync_mode** 配置参数:

```
"UPDATE DB CFG FOR dbname USING
    HADR_LOCAL_HOST   hostname
    HADR_LOCAL_SVC    servicename
    HADR_SYNCMODE     syncmode"
```

3. 在所有备用数据库和主数据库上设置 **hadr_target_list** 配置参数。

```
DB2 UPDATE DB CFG FOR dbname USING
     HADR_TARGET_LIST  principalhostname:principalservicename|
auxhostname1: auxservicename1|
auxhostname2:auxservicename2
```

4. 在所有数据库上，设置 **hadr_remote_host**、**hadr_remote_svc** 和 **hadr_remote_inst** 配置参数。

   此步骤不是必需的，因为在多备用数据库方式下，如果未设置这些值，那么会自动设置它们；如果这些值设置不正确，那么会对它们进行自动重设。但是，显式地将它们设置为正确值可使正确值立即可用。这些值对于 IBM Tivoli System Automation for Multiplatforms (SA MP) 软件很有帮助，该软件可能需要 **hadr_remote_inst** 值才能构造资源名称。

   - 在主数据库上，通过发出以下命令将这些参数设置为主体备用数据库上的相应值:

```
DB2 "UPDATE DB CFG FOR  dbname USING
     HADR_REMOTE_HOST   principalhostname
     HADR_REMOTE_SVC    principalservicename
     HADR_REMOTE_INST   principalinstname"
```

   - 在每个备用数据库上，通过发出以下命令将这些参数设置为主数据库上的相应值:

```
DB2 "UPDATE DB CFG FOR  dbname USING
     HADR_REMOTE_HOST   primaryhostname
     HADR_REMOTE_SVC    primaryservicename
     HADR_REMOTE_INST   primaryinstname"
```

5. 连接至每个备用实例。
6. 在备用实例上，发出带 **AS STANDBY** 参数的 **START HADR** 命令:

```
START HADR ON DB dbname AS STANDBY
```

7. 连接至主实例。
8. 在主实例上，发出带 **AS PRIMARY** 参数的 **START HADR** 命令:

```
START HADR ON DB dbname AS PRIMARY
```

结果

备用数据库以本地同步复制状态启动，在此状态下，会读取并重放本地可用日志文件。在重放所有日志文件之后，这些数据库将进入远程同步复制暂挂状态。主数据库启动后，备用数据库进入远程同步复制状态，在此状态下，它们从主数据库接收日志页并重放这些日志页。备用数据库重放主数据库机器的磁盘上的所有日志文件后，所发生的情况取决于同步方式的类型。处于 SUPERASYNC 方式下的主体备用数据库和所

有辅助备用数据库将仍处于远程同步复制方式。具有 SYNC、NEARSYNC 或 ASYNC 方式的主体备用数据库将进入对等方式。

在预先存在的 HADR 设置中启用多备用数据库方式

以多备用数据库方式初始化 HADR 系统类似于以单备用数据库方式执行此操作。这两种方式的主要差别在于，您必须通过在设置中对所有数据库设置 **hadr_target_list** 数据库配置参数来启用多备用数据库方式。

开始之前

- 确定所有参与的数据库的（要用于 **hadr_local_host** 设置的）主机名或主机 IP 地址，以及（要用于 **hadr_local_svc** 设置的）服务名称或端口号。
- 确定每个数据库的目标列表。
- 确定在数据库成为主数据库的情况下每个数据库的主要备用数据库的同步方式和对等时间。
- 确定 **hadr_timeout** 配置参数的设置；对于所有数据库，此参数必须具有相同的设置。
- 确定主数据库与每个备用数据库之间是否有足够的网络带宽。需要时请进行升级。
- 确定主网络接口是否可以支持其他备用网络接口的出局数据流。需要时请进行升级。

关于此任务

多备用数据库方式需要在所有参与的数据库上设置 **hadr_target_list** 配置参数。此参数列示在该数据库成为主数据库的情况下的备用数据库。即使在备用数据库上此参数也是必需的。相互包含是必需的（即，如果 A 的目标列表中有 B，那么 B 的目标列表中也必须有 A）。这确保在从任何备用数据库进行接管后，新主数据库总是可以保留旧主数据库作为其备用数据库。您在目标列表中指定的第一个备用数据库被指定为*主体 HADR 备用数据库*。其他备用数据库为*辅助 HADR 备用数据库*。目标列表不必始终包含所有参与者。并且，如果有多个备用数据库，那么不需要对称或对等；即使将数据库 B 指定为数据库 A 的主体备用数据库，也不必将数据库 A 指定为数据库 B 的主体备用数据库。对于在数据库 A 的目标列表中指定的每个备用数据库，其目标列表中也必须指定数据库 A。处理每个数据库的目标列表是一个重要步骤。

作为一种特殊情况，可以为多备用数据库方式仅配置一个备用数据库。例如，在多备用数据库方式下，您可以将两个数据库配置为主数据库和备用数据库。此行为与单备用设置的行为不同，这是因为多备用行为（例如，自动化配置）将生效，并且可以动态添加或除去备用数据库目标。

在此任务中，您首先将仅创建和配置新的备用数据库。通过将原始配置一直保留到最终步骤，您可以尽可能使“主数据库/备用数据库”对保持起作用。如果您太早更改原始备用数据库的配置，并且无意中取消激活然后重新激活了此备用数据库，那么您可以中断旧的 HADR 对以使新配置生效。

过程

要以多备用数据库方式启用 HADR，请完成下列步骤:

1. 使用已复原的备份或分割镜像来创建任何其他备用数据库。有关如何执行此操作的指示信息，请参阅第 45 页的『初始化备用数据库』或第 30 页的『初始化高可用性灾难恢复 (HADR)』的步骤 2。

- 在主数据库上:

  ```
  DB2 BACKUP DB dbname
  ```

- 在备用数据库上:

  ```
  DB2 RESTORE DB dbname
  ```

2. 按如下所示配置每个新的备用数据库:
   a. 将 **hadr_local_host** 和 **hadr_local_svc** 设置为 HADR 连接所使用的 TCP 地址。
   b. 将 **hadr_remote_host**、**hadr_remote_svc** 和 **hadr_remote_inst** 配置参数设置为指向主数据库。
   c. 对于所有数据库，将 **hadr_timeout** 配置设置为同一设置。
   d. 按照先前计划设置 **hadr_target_list** 配置参数。
   e. 如果主要的备用数据库成为主数据库，请对此备用数据库设置 **hadr_syncmode** 和 **hadr_peer_window** 配置参数。
   f. 根据您所需要的设置，设置任何其他特定于 HADR 的参数，例如，**hadr_spool_limit** 或 **hadr_replay_delay**。
3. 连接至每个新的备用数据库实例，并发出附带 **AS STANDBY** 选项的 **START HADR** 命令。

   ```
   START HADR ON DB dbname AS STANDBY
   ```

4. 遵循步骤 2 中的相同指示信息来重新配置原始备用数据库。
5. 按如下所示重新配置主数据库:
   a. 将 **hadr_local_host** 和 **hadr_local_svc** 设置为 HADR 连接所使用的 TCP 地址。如果您要使用新的网络接口卡 (NIC) 以支持更高的网络带宽，那么您可能需要进行更新以接受更多备用数据库。
   b. 将 **hadr_remote_host**、**hadr_remote_svc** 和 **hadr_remote_inst** 配置参数设置为指向主要的备用数据库。
   c. 将 **hadr_timeout** 配置设置为与所有备用数据库具有相同设置。
   d. 按照先前计划设置 **hadr_target_list** 配置参数。
   e. 设置 **hadr_syncmode** 和 **hadr_peer_window** 配置参数，主要的备用数据库将使用这两个参数。
   f. 根据您所需要的设置，设置任何其他特定于 HADR 的参数，例如，**hadr_spool_limit** 或 **hadr_replay_delay**。
6. 取消激活原始备用数据库，然后将其重新激活以使新配置生效。
7. 对主数据库停止 HADR，然后重新启动主数据库以使新配置生效。

结果

所有备用数据库都应在几秒钟之内连接至主数据库。您可以使用附带 `-hadr` 选项的 **db2pd** 命令或者使用 MON_GET_HADR 表函数来监视这些备用数据库的状态。

修改多备用数据库设置

HADR 多备用数据库设置启动并运行后，您可能想要进行其他更改，例如，添加或除去辅助备用数据库或更改指定的主体备用数据库。您可进行这些种类的修改而不会导致主数据库停运。

添加辅助备用数据库

有一些原因会导致您可能想要添加辅助备用数据库:

- 为了部署附加备用数据库以处理只读工作负载
- 为了部署附加备用数据库以处理延迟重放
- 为了部署附加备用数据库以用于灾难恢复
- 要添加先前活动的 HADR 部署中包含的备用数据库，但它已成为*孤立数据库*，因为新的主数据库未在其 **hadr_target_list** 配置参数中指定该备用数据库

仅当 HADR 部署为多备用数据库方式时，才能添加辅助备用数据库。即，必须已在至少一个备用数据库上设置 **hadr_target_list** 配置参数。

要向 HADR 部署添加辅助备用数据库，请使用来自该备用数据库的主机和端口信息更新主数据库的目标列表。此信息与备用数据库上的 **hadr_local_host** 和 **hadr_local_svc** 参数相对应。还必须将主数据库的主机和端口信息添加至新备用数据库的目标列表。

提示: 尽管并非必需，但最好还是将新备用数据库的主机和端口信息添加至部署中其他备用数据库的目标列表。还应在新备用数据库的目标列表中指定这些备用数据库的主机和端口信息。如果未进行这些附加更新并且某个其他备用数据库作为新主数据库接管，那么系统会拒绝新备用数据库作为备用数据库目标并关闭该备用数据库。

除去辅助备用数据库

唯一可通过动态方式除去的备用数据库是辅助备用数据库。如果从多备用部署中以动态方式除去辅助备用数据库，那么不影响主数据库和主体备用数据库上的常规 HADR 操作。要除去辅助备用数据库，请对该备用数据库发出 **STOP HADR** 命令；然后，可从主数据库和任何其他备用数据库的目标列表中除去该备用数据库。

更改主体备用数据库

只有先在主数据库上停止 HADR，才能更改主体备用数据库；这不会导致停运，因为您不必取消激活主数据库。

要更改主体备用数据库，您必须在主数据库上停止 HADR。然后，更新主数据库的目标列表以先列出新的主体备用数据库。如果新的主体备用数据库还不是备用数据库，那么将主数据库的地址添加到其目标列表，配置其他 HADR 参数，然后激活该备用数据库。如果它已经是备用数据库，那么无需执行任何操作。

提示: 尽管并非必需，但最好还是将新主体备用数据库的主机和端口信息添加至部署中其他备用数据库的目标列表。还应在新主体备用数据库的目标列表中指定这些备用数据库的主机和端口信息。如果未进行这些附加更新并且任一备用数据库作为新主数据库接管，那么系统会拒绝另一个备用数据库作为备用数据库目标并关闭该备用数据库。

多个 HADR 备用数据库的数据库配置

有许多有关多 HADR 备用数据库设置中的数据库配置的注意事项。

HADR 参数的自动重新配置

在 HADR 启动后重新配置

在“多个备用数据库”方式下，如果未正确设置用于标识备用数据库的主数据库和主数据库的主体备用数据库的配置参数，那么会在 HADR 启动时自动重新配置这些配置参数。此行为适用于下列配置参数:

- **hadr_remote_host**
- **hadr_remote_inst**
- **hadr_remote_svc**

提示: 即使此自动重新配置发生，您也应始终尝试设置正确的初始值，因为直到在备用数据库与其主数据库之间建立连接，重新配置才有可能生效。在某些 HADR 部署中，可能需要这些初始值。例如，如果正在使用 IBM Tivoli System Automation for Multiplatforms 软件，那么需要 **hadr_remote_inst** 配置参数的值才能构造资源名称。

注: 如果 **DB2_HADR_NO_IP_CHECK** 注册表变量设置为 ON，那么不会自动更新 **hadr_remote_host** 和 **hadr_remote_svc**。

重新配置以 **hadr_target_list** 配置参数的值正确为基础；如果目标列表条目中的任何内容不正确，那么您必须手动更正。

在主数据库上，重新配置按以下方式进行:

- 如果 **hadr_remote_host** 和 **hadr_remote_svc** 配置参数的值与 *host*:*port* 对（**hadr_target_list** 配置参数的第一个条目）（即，主体备用数据库）不匹配，那么系统会使用目标列表中的值更新 **hadr_remote_host** 和 **hadr_remote_svc** 配置参数。
- 如果 **hadr_remote_inst** 配置参数的值与主体备用数据库的实例名称不匹配，那么主体备用数据库连接至主数据库后，系统会将正确的实例名称复制到主数据库的 **hadr_remote_inst** 配置参数。

在备用数据库上，重新配置按以下方式进行:

- 备用数据库启动时，它会尝试连接至其 **hadr_remote_host**、**hadr_remote_inst** 和 **hadr_remote_svc** 配置参数中指定的数据库。
- 如果备用数据库无法连接至主数据库，那么它会等待主数据库与其连接。
- 主数据库使用其 **hadr_target_list** 参数中所列的地址连接至其备用数据库。主数据库连接至备用数据库之后，系统会使用主数据库的正确值更新备用数据库的 **hadr_remote_host**、**hadr_remote_inst** 和 **hadr_remote_svc** 配置参数。

在接管期间和之后重新配置

在非强制接管中，新主数据库上的 **hadr_remote_host**、**hadr_remote_inst** 和 **hadr_remote_svc** 配置参数的值会自动更新为其主体备用数据库，该新主数据库的 **hadr_target_list** 中所列的备用数据库上的这些参数会自动更新为指向该新主数据库。不会更新未列在新主数据库的 **hadr_target_list** 中的任何数据库。这些数据库继续尝试连接到旧主数据库并遭到拒绝，因为旧主数据库现在已成为备用数据库。旧主数据库一定会出现在新主数据库的目标列表中，因为系统要求在目标列表中进行相互包含。

在强制接管中，新主数据库及其备用数据库（不包括旧主数据库）进行自动更新的方式与在非强制接管中相同。但是，直到旧主数据库关闭并作为备用数据库重新启动以进行重新集成之后，它才会自动更新。

在接管期间未联机的任何数据库会在启动后自动重新配置。在启动时自动重新配置可能不会立即生效，因为它依赖于新主数据库定期与备用数据库联系。在启动时，备用数据库可能会尝试连接到旧主数据库并遵循该旧主数据库的日志流，从而使其与新主数据库的日志流分开，并使该备用数据库无法与新主数据库配对。因此，在接管之前，您必须关闭旧主数据库以避免发生此类分裂情况。

备用数据库缺乏对同步方式和对等时间的控制

在“多个备用数据库”方式下，只有当前主数据库的 **hadr_syncmode** 和 **hadr_peer_window** 配置参数设置相关。备用数据库的这些参数的设置由主数据库（对于主体备用数据库）定义，或者由其角色（作为辅助备用数据库）定义。

同步方式

在“多个备用数据库”方式下，**hadr_syncmode** 配置参数的设置在主数据库和备用数据库上不必相同。在备用数据库上指定的任意 **hadr_syncmode** 配置参数设置将被视为其*已配置的同步方式*；此配置仅在备用数据库成为主数据库时具有相关性。系统会对备用数据库指定*有效同步方式*。对于任何辅助备用数据库，有效同步方式始终为 SUPERASYNC。对于主体备用数据库，有效同步方式为主数据库的 **hadr_syncmode** 配置参数设置。对于备用数据库，监视接口会将有效同步方式显示为同步方式。

对等时间

在“多个备用数据库”方式下，**hadr_peer_window** 配置参数的设置在主数据库和备用数据库上不必相同。实际上，系统会忽略辅助备用数据库上 **hadr_peer_window** 配置参数的任何设置，因为对等时间功能与 SUPERASYNC 方式不兼容。主体备用数据库使用主数据库的对等时间设置，它仅在备用数据库的 **hadr_syncmode** 配置参数值为 SYNC 或 NEARSYNC时适用（就像“单个备用数据库”方式一样）。

以 HADR 多备用数据库方式进行滚动升级

就像使用 HADR 单备用数据库方式一样，您可以使用滚动升级。关键差别在于，使用多备用数据库方式时，通过保持一个主数据库和一个备用数据库处于活动状态，可以在使用此过程的同时维持 HADR 保护。

始终有一个主数据库提供数据库服务，并且该主数据库始终具有至少一个提供 HA 和 DR 保护的备用数据库。

使用多备用数据库方式时，在对主数据库执行更新或升级之前，您应该先对所有备用数据库执行更新或升级操作。如果正在升级修订包级别，那么这尤其重要，因为 HADR 不允许主数据库处于高于备用数据库的修订包级别。

该过程基本上与单备用数据库方式相同，只是应该每次仅对一个数据库执行升级并从辅助备用数据库开始。例如，考虑以下 HADR 设置:

- host1 是主数据库
- host2 是主体备用数据库
- host3 是辅助备用数据库

对于此设置，按照以下顺序执行滚动升级或更新:

1. 取消激活 host3，进行所需更改，激活 host3 并在 host3（作为备用数据库）上启动 HADR。
2. 在 host3 执行完日志重放之后，取消激活 host2，进行所需更改，激活 host2 并在 host2（作为备用数据库）上启动 HADR。
3. 在 host2 执行完日志重放并与 host1 处于对等状态之后，对 host2 发出接管命令。
4. 取消激活 host1，进行所需更改，激活 host1 并在 host1（作为备用数据库）上启动 HADR。
5. 在 host1 与 host 2 处于对等状态之后，对 host1 发出接管命令，以便使其重新成为主数据库并使 host2 重新成为主体备用数据库。

多备用数据库方式下的高可用性灾难恢复 (HADR) 监视

HADR 多备用数据库方式支持与单备用数据库方式下相同的监视接口；但是，您应该只使用 **db2pd** 命令和 MON_GET_HADR 表函数，因为其他监视接口不提供所有备用数据库的完整视图。

监视接口返回的信息取决于发出该监视接口时的对象。监视备用数据库只会返回有关该备用数据库和主数据库的信息；不会提供有关任何其他备用数据库的信息。如果使用 **db2pd** 命令或 MON_GET_HADR 表函数来监视主数据库，那么会返回有关所有备用数据库的信息。即使未连接某些备用数据库，只要在主数据库的 **hadr_target_list** 配置参数中配置了该备用数据库，就会显示该数据库。诸如 **GET SNAPSHOT FOR DATABASE** 命令等其他接口仅报告主数据库和主体备用数据库。

db2pd 命令和 MON_GET_HADR 表函数会返回基本相同的信息，但 **db2pd** 命令不需要在备用数据库上启用读取（用于从备用数据库进行报告）。并且，在接管期间 **db2pd** 是首选的，因为其中某段时间可能主数据库和备用数据库都不允许客户机连接。

db2pd 命令

在以下示例中，DBA 对一个具有三个备用数据库的主数据库发出 **db2pd** 命令。返回了三组数据，每组分别表示一个“主数据库/备用数据库”日志装入通道。HADR_ROLE 字段表示对其发出 **db2pd** 命令的数据库的角色，因此，在所有数据组中该字段均显示为 PRIMARY。两个辅助备用数据库（hostS2 和 hostS3）的 HADR_STATE 为 REMOTE_CATCHUP，因为它们自动以 SUPERASYNC 方式运行（这也反映在 **db2pd** 输出中），无论它们所配置的 **hadr_syncmode** 设置为何。STANDBY_ID 用于区分备用数据库。它由系统生成，并且“标识至备用数据库”映射在各个查询之间可能更改；但是标识“1”总是被指定给主体备用数据库。

注：在输出中可能会省略与当前状态无关的字段。例如，在以下输出中，未包含有关“仅重放”窗口（例如，开始时间和事务计数）的信息，因为“仅重放”窗口不活动。

```
db2pd -db hadr_db -hadr

  Database Member 0 -- Database hadr_db -- Active -- Up 0 days 00:23:17 --
Date 06/08/2011 13:57:23

                            HADR_ROLE = PRIMARY
                          REPLAY_TYPE = PHYSICAL
                        HADR_SYNCMODE = SYNC
                           STANDBY_ID = 1
                        LOG_STREAM_ID = 0
                           HADR_STATE = PEER
                  PRIMARY_MEMBER_HOST = hostP.ibm.com
                     PRIMARY_INSTANCE = db2inst1
```

```
                        PRIMARY_MEMBER = 0
                     STANDBY_MEMBER_HOST = hostS1.ibm.com
                        STANDBY_INSTANCE = db2inst2
                          STANDBY_MEMBER = 0
                     HADR_CONNECT_STATUS = CONNECTED
                HADR_CONNECT_STATUS_TIME = 06/08/2011 13:38:10.199479 (1307565490)
             HEARTBEAT_INTERVAL(seconds) = 30
                   HADR_TIMEOUT(seconds) = 120
          TIME_SINCE_LAST_RECV(seconds) = 3
               PEER_WAIT_LIMIT(seconds) = 0
            LOG_HADR_WAIT_CUR(seconds) = 0.000
    LOG_HADR_WAIT_RECENT_AVG(seconds) = 0.006298
   LOG_HADR_WAIT_ACCUMULATED(seconds) = 0.516
                   LOG_HADR_WAIT_COUNT = 82
SOCK_SEND_BUF_REQUESTED,ACTUAL(bytes) = 0, 50772
SOCK_RECV_BUF_REQUESTED,ACTUAL(bytes) = 0, 87616
              PRIMARY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
              STANDBY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
                   HADR_LOG_GAP(bytes) = 0
     STANDBY_REPLAY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
       STANDBY_RECV_REPLAY_GAP(bytes) = 0
                     PRIMARY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
                     STANDBY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
                STANDBY_REPLAY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
          STANDBY_RECV_BUF_SIZE(pages) = 16
              STANDBY_RECV_BUF_PERCENT = 0
            STANDBY_SPOOL_LIMIT(pages) = 0
                  PEER_WINDOW(seconds) = 0
              READS_ON_STANDBY_ENABLED = Y
    STANDBY_REPLAY_ONLY_WINDOW_ACTIVE = N
                            HADR_ROLE = PRIMARY
                        REPLAY_TYPE = PHYSICAL
                       HADR_SYNCMODE = SUPERASYNC
                          STANDBY_ID = 2
                       LOG_STREAM_ID = 0
                          HADR_STATE = REMOTE_CATCHUP
                     PRIMARY_MEMBER_HOST = hostP.ibm.com
                        PRIMARY_INSTANCE = db2inst1
                          PRIMARY_MEMBER = 0
                     STANDBY_MEMBER_HOST = hostS2.ibm.com
                        STANDBY_INSTANCE = db2ins3t
                          STANDBY_MEMBER = 0
                     HADR_CONNECT_STATUS = CONNECTED
                HADR_CONNECT_STATUS_TIME = 06/08/2011 13:35:51.724447 (1307565351)
             HEARTBEAT_INTERVAL(seconds) = 30
                   HADR_TIMEOUT(seconds) = 120
          TIME_SINCE_LAST_RECV(seconds) = 16
               PEER_WAIT_LIMIT(seconds) = 0
            LOG_HADR_WAIT_CUR(seconds) = 0.000
    LOG_HADR_WAIT_RECENT_AVG(seconds) = 0.006298
   LOG_HADR_WAIT_ACCUMULATED(seconds) = 0.516
                   LOG_HADR_WAIT_COUNT = 82
SOCK_SEND_BUF_REQUESTED,ACTUAL(bytes) = 0, 16384
SOCK_RECV_BUF_REQUESTED,ACTUAL(bytes) = 0, 87380
              PRIMARY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
              STANDBY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
                   HADR_LOG_GAP(bytes) = 0
     STANDBY_REPLAY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
       STANDBY_RECV_REPLAY_GAP(bytes) = 0
                     PRIMARY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
                     STANDBY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
                STANDBY_REPLAY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
          STANDBY_RECV_BUF_SIZE(pages) = 16
              STANDBY_RECV_BUF_PERCENT = 0
            STANDBY_SPOOL_LIMIT(pages) = 0
                  PEER_WINDOW(seconds) = 0
              READS_ON_STANDBY_ENABLED = Y

                            HADR_ROLE = PRIMARY
                        REPLAY_TYPE = PHYSICAL
                       HADR_SYNCMODE = SUPERASYNC
```

```
                            STANDBY_ID = 3
                         LOG_STREAM_ID = 0
                            HADR_STATE = REMOTE_CATCHUP
                    PRIMARY_MEMBER_HOST = hostP.ibm.com
                       PRIMARY_INSTANCE = db2inst1
                         PRIMARY_MEMBER = 0
                    STANDBY_MEMBER_HOST = hostS3.ibm.com
                       STANDBY_INSTANCE = db2inst3
                         STANDBY_MEMBER = 0
                    HADR_CONNECT_STATUS = CONNECTED
               HADR_CONNECT_STATUS_TIME = 06/08/2011 13:46:51.561873 (1307566011)
             HEARTBEAT_INTERVAL(seconds) = 30
                   HADR_TIMEOUT(seconds) = 120
          TIME_SINCE_LAST_RECV(seconds) = 6
               PEER_WAIT_LIMIT(seconds) = 0
             LOG_HADR_WAIT_CUR(seconds) = 0.000
     LOG_HADR_WAIT_RECENT_AVG(seconds) = 0.006298
    LOG_HADR_WAIT_ACCUMULATED(seconds) = 0.516
                    LOG_HADR_WAIT_COUNT = 82
  SOCK_SEND_BUF_REQUESTED,ACTUAL(bytes) = 0, 16384
  SOCK_RECV_BUF_REQUESTED,ACTUAL(bytes) = 0, 87380
              PRIMARY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
              STANDBY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
                    HADR_LOG_GAP(bytes) = 0
      STANDBY_REPLAY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
        STANDBY_RECV_REPLAY_GAP(bytes) = 0
                      PRIMARY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
                      STANDBY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
               STANDBY_REPLAY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
          STANDBY_RECV_BUF_SIZE(pages) = 16
              STANDBY_RECV_BUF_PERCENT = 0
            STANDBY_SPOOL_LIMIT(pages) = 0
                  PEER_WINDOW(seconds) = 0
              READS_ON_STANDBY_ENABLED = N
```

MON_GET_HADR 表函数

在以下示例中，DBA 对一个具有三个备用数据库的主数据库调用 **MON_GET_HADR** 表函数。返回了三行内容。每行表示一个“主数据库/备用数据库”日志装入通道。HADR_ROLE 列表示针对其发出查询的数据库的角色。因此，在所有行上，该列均为 PRIMARY。两个辅助备用数据库（hostS2 和 hostS3）的 HADR_STATE 为 REMOTE_CATCHUP，因为它们自动以 SUPERASYNC 方式运行，无论它们所配置的 **hadr_syncmode** 设置为何。

```
db2 "select HADR_ROLE, STANDBY_ID, HADR_STATE, varchar(PRIMARY_MEMBER_HOST,20)
as PRIMARY_MEMBER_HOST, varchar(STANDBY_MEMBER_HOST,20) as
STANDBY_MEMBER_HOST from table (mon_get_hadr(NULL))"

  HADR_ROLE  STANDBY_ID HADR_STATE      PRIMARY_MEMBER_HOST  STANDBY_MEMBER_HOST
  ---------  ---------- -------------  -------------------  --------------------
  PRIMARY               1 PEER                 hostP.ibm.com        hostS1.ibm.com
  PRIMARY               2 REMOTE_CATCHUP       hostP.ibm.com        hostS2.ibm.com
  PRIMARY               3 REMOTE_CATCHUP       hostP.ibm.com        hostS3.ibm.com

 3 record(s) selected.
```

以 HADR 多备用数据库方式进行接管

当 HADR 备用数据库接管作为“多个备用数据库”环境中的主数据库时，与单备用数据库方式具有许多重要差别。

对于 HADR，有以下两种类型的接管：*角色切换*和*故障转移*。仅当主数据库可用并且切换主数据库与备用数据库的角色时，才能执行角色切换（有时称为正常接管或者非强制接管）。当主数据库不可用时，可以执行故障转移或者强制接管。这种接管通常用

于主数据库故障情况，以使备用数据库成为新的主数据库。在强制接管情况下，旧的主数据库仍然保持主数据库角色。在多备用数据库方式下，这两种类型的接管均受支持，并且任何备用数据库都可以接管作为主数据库。但是，要记住的至关重要的事项是，如果某个备用数据库未包括在新的主数据库的目标列表中，那么会认为此备用数据库是孤立的，无法连接至新的主数据库。

在接管时，DB2 会自动对配置进行许多更改，以便新的主数据库的目标列表中所列示的备用数据库可以连接至新的主数据库。在新的主数据库和所列示的备用数据库中，会按以下方式更新 **hadr_remote_host**、**hadr_remote_svc** 和 **hadr_remote_inst** 配置参数:

- **在新的主数据库中**: 它们表示主要的备用数据库（也就是新的主数据库的目标列表中列示的第一个数据库）。
- **在备用数据库中**: 它们表示新的主数据库。当重新集成旧的主数据库使其成为备用数据库时，**START HADR** AS STANDBY 命令首先会将其转换为备用数据库。因此，如果它列示在新的主数据库的目标列表中，那么还可以将其自动重定向至新的主数据库。

  注: 采用此方式时，不会自动更新孤立的备用数据库。如果您希望它们作为备用数据库加入，那么需要确保它们位于新的主数据库的目标列表中，并且它们将新的主数据库包括在它们的目标列表中。

角色切换

正如单备用数据库方式一样，多备用数据库方式下的角色切换将保证旧的主数据库与新的主数据库之间不会丢失数据。会将新的主数据库的 **hadr_target_list** 配置参数中所配置的其他备用数据库自动重定向至新的主数据库并且继续接收日志。

故障转移

正如单备用数据库方式一样，如果在多备用数据库方式下故障转移导致任何数据丢失（这意味着新的主数据库只具备旧的主数据库的部分数据），那么旧的主数据库与新的主数据库的日志流将分开，并且需要重新初始化旧的主数据库。对于其他备用数据库，如果备用数据库在分开时间之后从旧的主数据库接收到日志，那么需要重新初始化此备用数据库。否则，它可以连接至新的主数据库并继续进行日志装入和重放。因此，您检查所有备用数据库的日志位置并选择具有大多数数据的备用数据库作为故障转移目标非常重要。您可以使用 **db2pd** 命令或者使用 MON_GET_HADR 表函数来查询此信息。

注: 成功地自动重新配置备用数据库的 **hadr_remote_host**、**hadr_remote_svc** 和 **hadr_remote_inst** 配置参数以指向新的主数据库，并不意味着接受将此备用数据库与新的主数据库配对。它只允许备用数据库与主数据库建立 TCP 连接。进行连接时，如果 DB2 确定这两个数据库具有分开的日志流，那么配对请求将被拒绝并且连接会被关闭。

方案: 部署 HADR 多备用数据库设置

此方案说明一家名为 ExampleBANK 的银行的 HADR 设置的规划、配置和部署。此设置具有三个备用数据库: 一个主体备用数据库和两个辅助备用数据库。

背景

因为银行业务是 24x7 业务，所以高可用性对 ExampleBANK 的技术策略很关键。此外，ExampleBANK 经历了一场飓风袭击城市 A（其总部所在地）的风险事件，因此该

银行还需要灾难恢复策略。高可用性灾难恢复 (HADR) 提供了一个解决方案，可帮助该银行借助一项技术同时达成这两个目标：HADR 多备用数据库。

ExampleBANK 将以下要求视为其 HADR 解决方案的基本要求：

一个进取型的恢复时间目标

由于银行提供 24 小时联机服务，因此 ExampleBANK 希望使应用程序无法连接到期数据库的时间减少到最短。

一个进取型的恢复点目标

0ExampleBANK 无法容忍数据损失，因此 RPO 应尽可能地接近 0。

接近于零的计划停机时间

ExampleBANK 的数据库应在尽可能多的时间内可用，即使在计划的活动（例如，升级和维护）期间也是如此。

通过地理分布来提供数据保护

作为其合规标准的一部分，ExampleBANK 希望具有在远程位置恢复运营的能力。

易于部署和管理

ExampleBANK 负担沉重的 IT 部门希望采用一种配置相对简单且具有自动化功能的解决方案。

如以下方案所展示，在多备用数据库方式下使用 HADR 功能帮助 ExampleBANK 满足所有这些要求。

计划多备用设置

ExampleBANK 希望其 HADR 设置具有高可用性和灾难恢复保护，因此该银行决定使用备用数据库的最大数目：三个。为实现此 RTO，该银行必须具有一个与主数据库紧密同步并与主数据库搭配使用的备用数据库（使用 SYNC 或 NEARSYNC 方式的一个备用数据库）。最有意义的做法是将该备用数据库作为主体备用数据库，因为只有该类备用数据库支持所有的同步方式。主数据库和主体备用数据库均位于 ExampleBANK 设于城市 A 的总部并通过局域网连接。

此外，为保护该银行的数据以免因灾难丢失，ExampleBANK DBA 选择在该银行设于城市 B 的区域办事处设置两个备用数据库。该区域办事处通过广域网与城市 A 的总部连接。两个城市之间的距离将不会影响主数据库，因为这两个备用数据库为辅助备用数据库，它们自动在 SUPERASYNC 方式下运行。DBA 可以通过将这两个备用数据库中的一个设置为使用“在备用数据库上读取”功能，另一个使用“延迟重放”功能，从而为这些额外数据库的成本提供理由。另外，这些备用数据库还可通过滚动更新或维护方案帮助维持数据库可用性，防止失去 HADR 保护。

配置多备用设置

ExampleBANK DBA 生成了打算作为主数据库的 HADR_DB 的备份：

```
DB2 BACKUP DB hadr_db TO backup_dir
```

然后，该 DBA 通过发出以下命令将该备份复原到打算作为备用数据库的每台主机：

```
DB2 RESTORE DB hadr_db FROM backup_dir
```

提示：有关创建备用数据库的选项的更多信息，请参阅第 45 页的『初始化备用数据库』。

对于初始设置，ExampleBANK DBA 决定大多数缺省配置设置已足够。但是，在常规 HADR 设置中，必须显式设置以下数据库配置参数:

- **hadr_local_host**
- **hadr_local_svc**
- **hadr_remote_host**
- **hadr_remote_inst**
- **hadr_remote_svc**

为了获取这些配置参数的正确值，DBA 将确定包含在 HADR 设置中的四个数据库的主机名、端口号和实例名:

表 9. 数据库的主机名、端口号和实例名

预期的角色	主机名	端口号	实例名
主数据库	host1	10	dbinst1
主体备用数据库	host2	40	dbinst2
辅助备用数据库	host3	41	dbinst3
辅助备用数据库	host4.ibm.com	42	dbinst4

在主数据库上，**hadr_remote_host**、**hadr_remote_inst** 和 **hadr_remote_svc** 配置参数的设置分别与主体备用数据库的主机名、实例名和端口号相对应。在备用数据库上，这些配置参数的值对应于主数据库的主机名、端口号和实例名。此外，DBA 将使用该主机名和端口值来设置所有数据库上的 **hadr_target_list** 配置参数。另外，尽管并非必需，DBA 还是将有关设置中所有备用数据库的信息添加到每个其他数据库的目标列表。关于此主题的更多信息，请参阅第 33 页的『高可用性灾难恢复 (HADR) 的数据库配置』。

如上所述，该银行希望在主数据库与主体备用数据库之间尽可能紧密地保持同步，因此 DBA 将主数据库上的 **hadr_syncmode** 参数设置为 SYNC。尽管主体备用数据库在连接到主数据库后，会自动将其有效同步方式设置为 SYNC，DBA 还是将主体备用数据库上的 **hadr_syncmode** 参数设置为 SYNC。这是因为，如果主体备用数据库与主数据库交换角色，那么新的主数据库/主体备用数据库对的同步方式将仍为 SYNC。

DBA 决定将 host2（与辅助备用数据库位于不同城市）指定为辅助备用数据库的主体备用数据库。如果其中一个辅助备用数据库成为主数据库，那么新主数据库与位于远程位置的 host2 之间的好的同步方式是 SUPERASYNC。因此，尽管辅助备用数据库在连接到主数据库后会自动将其有效同步方式设置为 SUPERASYNC，DBA 还是将两个辅助备用数据库上的 **hadr_syncmode** 参数设置为 SUPERASYNC。关于此主题的更多信息，请参阅第 50 页的『高可用性灾难恢复 (HADR) 同步方式』。

最后，DBA 已了解有关新的 HADR 延迟重放功能的知识，该功能可用于有意地通过延迟重放日志来使某个备用数据库保持在早于主数据库的时间点的某个时间点。DBA 决定启用此功能将有助于 ExampleBANK 针对主数据库上的错误事务提供数据保护。DBA 选择将辅助备用数据库 host4 用于此功能，并记下必须先禁用此功能，然后 host4 作为主数据库才能进行接管。关于此主题的更多信息，请参阅第 165 页的『HADR 延迟重放』。

DBA 发出以下命令以更新每个数据库上的配置参数:

- 在 host1（主数据库）上:

```
DB2 "UPDATE DB CFG FOR hadr_db USING
     HADR_TARGET_LIST  host2:40|host3:41|host4:42
     HADR_REMOTE_HOST  host2
     HADR_REMOTE_SVC   40
     HADR_LOCAL_HOST   host1
     HADR_LOCAL_SVC    10
     HADR_SYNCMODE     sync
     HADR_REMOTE_INST  db2inst2"
```

- 在 host2（主体备用数据库）上:

```
DB2 "UPDATE DB CFG FOR hadr_db USING
     HADR_TARGET_LIST  host1:10|host3:41|host4:42
     HADR_REMOTE_HOST  host1
     HADR_REMOTE_SVC   10
     HADR_LOCAL_HOST   host2
     HADR_LOCAL_SVC    40
     HADR_SYNCMODE     sync
     HADR_REMOTE_INST  db2inst1"
```

- 在 host3（辅助备用数据库）上:

```
DB2 "UPDATE DB CFG FOR hadr_db USING
     HADR_TARGET_LIST  host2:40|host1:10|host4:42
     HADR_REMOTE_HOST  host1
     HADR_REMOTE_SVC   10
     HADR_LOCAL_HOST   host3
     HADR_LOCAL_SVC    41
     HADR_SYNCMODE     superasync
     HADR_REMOTE_INST  db2inst1"
```

- 在 host4（辅助备用数据库）上:

```
DB2 "UPDATE DB CFG FOR hadr_db USING
     HADR_TARGET_LIST  host2.:40|host1:10|host3:41
     HADR_REMOTE_HOST  host2
     HADR_REMOTE_SVC   10
     HADR_LOCAL_HOST   host4
     HADR_LOCAL_SVC    42
     HADR_SYNCMODE     superasync
     HADR_REMOTE_INST  db2inst1
     HADR_REPLAY_DELAY 86400"
```

最后，出于以下原因，ExampleBANK DBA 希望启用 HADR 的“在备用数据库上读取”功能:

- 能够对备用数据库上的某些 HADR 配置参数进行联机更改
- 对备用数据库调用 MON_GET_HADR 表函数
- 转移主数据库的一些只读工作负载

DBA 通过分别对 host2、host3 和 host4 发出以下命令来更新这些备用数据库上的注册表变量:

```
DB2SET DB2_HADR_ROS=ON
DB2SET DB2_STANDBY_ISO=UR
```

启动 HADR 数据库

DBA 首先通过分别对 host2、host3 和 host4 发出以下命令来启动这些备用数据库:

```
DB2 START HADR ON DB hadr_db AS STANDBY
```

接着，DBA 对主数据库 host1 启动 HADR:

```
DB2 START HADR ON DB hadr_db AS PRIMARY
```

为了验证 HADR 已建立并正在运行，DBA 通过从 host1 上的主数据库发出 **db2pd** 命令来查询数据库的状态，这将返回有关所有备用数据库的信息:

```
db2pd -db hadr_db -hadr

  Database Member 0 -- Database hadr_db -- Active -- Up 0 days 00:23:17 -- Date 06/08/2011 13:57:23

                             HADR_ROLE = PRIMARY
                           REPLAY_TYPE = PHYSICAL
                         HADR_SYNCMODE = SYNC
                            STANDBY_ID = 1
                         LOG_STREAM_ID = 0
                            HADR_STATE = PEER
                    PRIMARY_MEMBER_HOST = host1
                       PRIMARY_INSTANCE = db2inst1
                         PRIMARY_MEMBER = 0
                    STANDBY_MEMBER_HOST = host2
                       STANDBY_INSTANCE = db2inst2
                         STANDBY_MEMBER = 0
                    HADR_CONNECT_STATUS = CONNECTED
               HADR_CONNECT_STATUS_TIME = 06/08/2011 13:38:10.199479 (1307565490)
            HEARTBEAT_INTERVAL(seconds) = 30
                  HADR_TIMEOUT(seconds) = 120
          TIME_SINCE_LAST_RECV(seconds) = 3
               PEER_WAIT_LIMIT(seconds) = 0
             LOG_HADR_WAIT_CUR(seconds) = 0.000
      LOG_HADR_WAIT_RECENT_AVG(seconds) = 0.006298
     LOG_HADR_WAIT_ACCUMULATED(seconds) = 0.516
                    LOG_HADR_WAIT_COUNT = 82
  SOCK_SEND_BUF_REQUESTED,ACTUAL(bytes) = 0, 50772
  SOCK_RECV_BUF_REQUESTED,ACTUAL(bytes) = 0, 87616
              PRIMARY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
              STANDBY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
                    HADR_LOG_GAP(bytes) = 0
       STANDBY_REPLAY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
         STANDBY_RECV_REPLAY_GAP(bytes) = 0
                    PRIMARY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
                    STANDBY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
                STANDBY_REPLAY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
           STANDBY_RECV_BUF_SIZE(pages) = 16
               STANDBY_RECV_BUF_PERCENT = 0
             STANDBY_SPOOL_LIMIT(pages) = 0
                  PEER_WINDOW(seconds) = 0
               READS_ON_STANDBY_ENABLED = Y
      STANDBY_REPLAY_ONLY_WINDOW_ACTIVE = N
                             HADR_ROLE = PRIMARY
                           REPLAY_TYPE = PHYSICAL
                         HADR_SYNCMODE = SUPERASYNC
                            STANDBY_ID = 2
                         LOG_STREAM_ID = 0
                            HADR_STATE = REMOTE_CATCHUP
                    PRIMARY_MEMBER_HOST = host1
                       PRIMARY_INSTANCE = db2inst1
                         PRIMARY_MEMBER = 0
                    STANDBY_MEMBER_HOST = host3
                       STANDBY_INSTANCE = db2inst3
                         STANDBY_MEMBER = 0
                    HADR_CONNECT_STATUS = CONNECTED
               HADR_CONNECT_STATUS_TIME = 06/08/2011 13:35:51.724447 (1307565351)
            HEARTBEAT_INTERVAL(seconds) = 30
                  HADR_TIMEOUT(seconds) = 120
          TIME_SINCE_LAST_RECV(seconds) = 16
               PEER_WAIT_LIMIT(seconds) = 0
             LOG_HADR_WAIT_CUR(seconds) = 0.000
      LOG_HADR_WAIT_RECENT_AVG(seconds) = 0.006298
     LOG_HADR_WAIT_ACCUMULATED(seconds) = 0.516
                    LOG_HADR_WAIT_COUNT = 82
  SOCK_SEND_BUF_REQUESTED,ACTUAL(bytes) = 0, 16384
  SOCK_RECV_BUF_REQUESTED,ACTUAL(bytes) = 0, 87380
              PRIMARY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
```

```
          STANDBY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
                HADR_LOG_GAP(bytes) = 0
   STANDBY_REPLAY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
     STANDBY_RECV_REPLAY_GAP(bytes) = 0
                  PRIMARY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
                  STANDBY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
           STANDBY_REPLAY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
       STANDBY_RECV_BUF_SIZE(pages) = 16
           STANDBY_RECV_BUF_PERCENT = 0
         STANDBY_SPOOL_LIMIT(pages) = 0
              PEER_WINDOW(seconds) = 0
          READS_ON_STANDBY_ENABLED = Y
  STANDBY_REPLAY_ONLY_WINDOW_ACTIVE = N
                        HADR_ROLE = PRIMARY
                    REPLAY_TYPE = PHYSICAL
                  HADR_SYNCMODE = SUPERASYNC
                     STANDBY_ID = 3
                  LOG_STREAM_ID = 0
                     HADR_STATE = REMOTE_CATCHUP
               PRIMARY_MEMBER_HOST = host1
                  PRIMARY_INSTANCE = db2inst1
                    PRIMARY_MEMBER = 0
               STANDBY_MEMBER_HOST = host4
                  STANDBY_INSTANCE = db2inst4
                    STANDBY_MEMBER = 0
               HADR_CONNECT_STATUS = CONNECTED
          HADR_CONNECT_STATUS_TIME = 06/08/2011 13:46:51.561873 (1307566011)
        HEARTBEAT_INTERVAL(seconds) = 30
             HADR_TIMEOUT(seconds) = 120
      TIME_SINCE_LAST_RECV(seconds) = 6
           PEER_WAIT_LIMIT(seconds) = 0
        LOG_HADR_WAIT_CUR(seconds) = 0.000
  LOG_HADR_WAIT_RECENT_AVG(seconds) = 0.006298
 LOG_HADR_WAIT_ACCUMULATED(seconds) = 0.516
               LOG_HADR_WAIT_COUNT = 82
SOCK_SEND_BUF_REQUESTED,ACTUAL(bytes) = 0, 16384
SOCK_RECV_BUF_REQUESTED,ACTUAL(bytes) = 0, 87380
          PRIMARY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
          STANDBY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
                HADR_LOG_GAP(bytes) = 0
   STANDBY_REPLAY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
     STANDBY_RECV_REPLAY_GAP(bytes) = 0
                  PRIMARY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
                  STANDBY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
           STANDBY_REPLAY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
       STANDBY_RECV_BUF_SIZE(pages) = 16
           STANDBY_RECV_BUF_PERCENT = 0
         STANDBY_SPOOL_LIMIT(pages) = 0
              PEER_WINDOW(seconds) = 0
          READS_ON_STANDBY_ENABLED = Y
  STANDBY_REPLAY_ONLY_WINDOW_ACTIVE = N
```

示例: 以 HADR 多备用数据库方式进行接管

该组 HADR 多备用数据库方式下的接管（同时包括强制和非强制）示例基于三个备用数据库设置。这些示例用于展示多备份自动重新配置在接管情况下的工作方式。

- 第 188 页的『主体备用数据库以优雅的方式接管（角色交换）』
- 第 189 页的『辅助备用数据库以强制方式进行接管（故障转移）』
- 第 190 页的『在 SA MP 环境中，辅助备用数据库以强制方式进行接管（故障转移）』

每个示例的初始设置如下所示:

- 一个主数据库 (host1)
- 一个主体备用数据库 (host2)

- 两个辅助备用数据库（host3 和 host4）

所有数据库均称为 hadr_db。主数据库和主体备用数据库的同步方式设为 SYNC，其他备用数据库的同步方式设为 SUPERASYNC。

表 10 中显示了各个数据库的配置。

表 10. 各 HADR 数据库的配置值

配置参数	Host1	Host2	Host3	Host4
hadr_target_list	host2:40|host3:41| host4:42	host1:10|host3:41| host4:42	host2:40|host1:10| host4:42	host2:40|host1:10| host3:41
hadr_remote_host	host2	host1	host1	host1
hadr_remote_svc	40	10	10	10
hadr_remote_inst	dbinst2	dbinst1	dbinst1	dbinst1
hadr_local_host	host1	host2	host3	host4
hadr_local_svc	10	40	41	42
配置的 **hadr_syncmode**（指的是显式设置的同步方式，当数据库成为主数据库时将使用该方式）	SYNC	SYNC	SUPERASYNC	SUPERASYNC
有效的 **hadr_syncmode**（指的是数据库当前为备用数据库时所使用的同步方式）	不适用	SYNC	SUPERASYNC	SUPERASYNC

主体备用数据库以优雅的方式接管（角色交换）

DBA 通过在 host2 上发出以下命令在主体备用数据库上进行接管:

```
DB2 TAKEOVER HADR ON DB hadr_db
```

在成功完成接管后，host2 将成为新的主数据库，而 host1（host2 的 **hadr_target_list** 中的第一个条目，如 表 10 中所示）将成为其主体备用数据库。它们的同步方式为 SYNC 方式，因为 host2 的 **hadr_syncmode** 配置为 SYNC。辅助备用数据库目标 host3 和 host4 的 **hadr_remote_host** 和 **hadr_remote_svc** 指向旧主数据库 host1，但是将被自动重定向到新主数据库 host2。在此重定向操作中，host3 和 host4 将更新（持续不断地）它们的 **hadr_remote_host**、**hadr_remote_svc** 和 **hadr_remote_inst** 配置参数。它们将作为辅助备用数据库重新连接到 host2， 并且 host2 将会命令它们使用有效同步方式 SUPERASYNC（无论它们本地配置的 **hadr_syncmode** 设置为何）。它们不会持续不断地更新自己的 **hadr_syncmode** 设置。表 11 中显示了各个数据库的配置。

表 11. 在角色交换后各 HADR 数据库的配置值. 第 4 列和第 5 列的第 3 行到第 5 行以粗体显示，表示它们已自动重新配置

配置参数	Host1	Host2	Host3	Host4
hadr_target_list	host2:40|host3:41| host4:42	host1:10|host3:41| host4:42	host2:40|host1:10| host4:42	host2:40|host1:10| host3:41
hadr_remote_host	host2	host1	**host2**	**host2**
hadr_remote_svc	40	10	**40**	**40**

表 11. 在角色交换后各 HADR 数据库的配置值 (续). 第 4 列和第 5 列的第 3 行到第 5 行以粗体显示，表示它们已自动重新配置

配置参数	Host1	Host2	Host3	Host4
hadr_remote_inst	dbinst2	dbinst1	**dbinst2**	**dbinst2**
hadr_local_host	host1	host2	host3	host4
hadr_local_svc	10	40	41	42
配置的 **hadr_syncmode**	SYNC	SYNC	SUPERASYNC	SUPERASYNC
有效的 **hadr_syncmode**	SYNC	不适用	SUPERASYNC	SUPERASYNC

注: 由于以下原因，许多值不会更新

- 因为 host2 已将其 **hadr_remote_host** 和 **hadr_remote_svc** 配置参数指向其主体备用数据库 host1，所以不会更新 host2 上的这些值。
- 因为 host1 已将其 **hadr_remote_host** 和 **hadr_remote_svc** 配置参数指向新的主数据库，所以不会更新 host1 上的这些值。
- 因为 host1 的操作同步方式为 SYNC，并且 host3 和 host4 的操作同步方式为 SUPERASYNC，所以未更改有效同步方式。

辅助备用数据库以强制方式进行接管（故障转移）

城市 A 发生的一次大面积断电导致主数据库 (host1) 变得不可用。通常，处于 SYNC 方式的主体数据库 (host2) 是进行接管并成为新主数据库的最佳候选者，但是断电意味着 host2 也暂时不可用。DBA 查询两个辅助备用数据库以确定哪一个具有最多的日志数据:

```
db2pd -hadr -db hadr_db | grep 'PRIMARY_LOG_FILE,PAGE,POS|STANDBY_LOG_FILE,PAGE,POS'
```

DBA 确定 host3 是最新的（尽管它在日志重放上仍然略有落后）并将该主机选为新主数据库:

```
DB2 TAKEOVER HADR ON DB hadr_db BY FORCE
```

在接管成功完成后，host3 将成为新的主数据库。同时，host2 将变得重新可用。host3 通知 host2 和 host4，它现在已成为主数据库。在 host3 上，**hadr_remote_host**、**hadr_remote_svc** 和 **hadr_remote_inst** 的值已重新配置为指向 host2，该数据库是主体备用数据库，因为它是 host3 上的 **hadr_target_list** 中的第一个条目。在 host2 上，同步方式已重新配置为 SUPERASYNC，因为这是 host3 上的 **hadr_syncmode** 设置；此外，**hadr_remote_host**、**hadr_remote_svc** 和 **hadr_remote_inst** 已更新（持续不断地）。host4 会自动重定向至新的主数据库 host3。在此重定向操作中，host4 将更新（持续不断地）它的 **hadr_remote_host**、**hadr_remote_svc** 和 **hadr_remote_inst** 配置参数。直到 host1 变得重新可用为止，不会对其进行自动重新配置。表 12 中显示了各个数据库的配置。

表 12. 在故障转移后各 HADR 数据库的配置值. 第 3 列到第 5 列的第 3 行到第 5 行以粗体显示，表示它们已自动重新配置

配置参数	Host1 （不可用）	Host2	Host3	Host4
hadr_target_list	host2:40|host3:41| host4:42	host1:10|host3:41| host4:42	host2:40|host1:10| host4:42	host2:40|host1:10| host3:41
hadr_remote_host	host2	**host3**	**host2**	**host3**

表 12. 在故障转移后各 HADR 数据库的配置值 (续). 第 3 列到第 5 列的第 3 行到第 5 行以粗体显示，表示它们已自动重新配置

配置参数	Host1（不可用）	Host2	Host3	Host4
hadr_remote_svc	40	**41**	**40**	**41**
hadr_remote_inst	dbinst2	**dbinst3**	**dbinst2**	**dbinst3**
hadr_local_host	host1	host2	host3	host4
hadr_local_svc	10	40	41	42
配置的 **hadr_syncmode**	SYNC	SYNC	SUPERASYNC	SUPERASYNC
有效的 **hadr_syncmode**	不适用	**SUPERASYNC**	不适用	SUPERASYNC

在一段短时间后，host1 变得可用。DBA 尝试将 host1 作为备用数据库启用，但因为 host1 具有的日志多于已传播到 host3 的日志，所以在与新主数据库初始握手期间已拒绝 host1。DBA 生成了新主数据库的备份，将该备份复原到 host1 并在该主机上启动 HADR:

```
DB2 BACKUP DB hadr_db

DB2 RESTORE DB hadr_db

DB2 START HADR ON DB hadr_db AS STANDBY
```

如 表 13 中所示，host1 已重新配置。

表 13. 重新集成的备用数据库的配置值. 第 2 列中的各行以粗体显示，表示它们已自动重新配置

配置参数	Host1	Host2	Host3	Host4
hadr_target_list	host2:40|host3:41| host4:42	host1:10|host3:41| host4:42	host2:40|host1:10| host4:42	host2:40|host1:10| host3:41
hadr_remote_host	**host3**	host3	host2	host3
hadr_remote_svc	**41**	41	40	41
hadr_remote_inst	**dbinst3**	dbinst3	dbinst2	dbinst3
hadr_local_host	host1	host2	host3	host4
hadr_local_svc	10	40	41	42
配置的 **hadr_syncmode**	SYNC	SYNC	SUPERASYNC	SUPERASYNC
有效的 **hadr_syncmode**	**SUPERASYNC**	SUPERASYNC	不适用	SUPERASYNC

如果 DBA 希望使 host1 再次成为主数据库，那么只需要进行故障回退（该操作将复原 第 188 页的表 10 中所示的原始配置）。

在 SA MP 环境中，辅助备用数据库以强制方式进行接管（故障转移）

此示例与上一个示例类似，但已使用 IBM Tivoli System Automation for Multiplatforms (SA MP) 将 HADR 部署为自动化故障转移。

城市 A 发生的断电导致主体备用数据库 (host2) 变得不可用。接着，主数据库 (host1) 停止。通常，集群管理器 SA MP 会自动故障转移到主体备用数据库 (host2)，但断电意味着其中一个辅助备用数据库需要成为接管目标。因为不能自动故障转移到辅助备用数据库，所以 DBA 必须手动执行。但是，在执行该操作之前，DBA 需要确保已禁用

TSA，以便在 host1 或 host2 变得可用时，不可能发生分裂情况，在该情况下，多个数据库同时作为主数据库独立地运行。为实现此目的，DBA 对 host1 和 host2 发出以下命令（每当它们变得可用时）：

```
db2haicu -disable
```

此外，DBA 需要保持 host1 脱机，以防止出现当客户机连接到旧主数据库时该数据库重新启动的情况。

DBA 查询两个辅助备用数据库以确定哪一个具有最多的日志数据:

```
db2pd -hadr -db hadr_db | grep 'STANDBY_LOG_FILE,PAGE,POS'
```

DBA 确定 host3 是最新的并将该主机选为新主数据库。

然后，DBA 将对 host3 发出强制接管命令:

```
DB2 TAKEOVER HADR ON DB hadr_db BY FORCE
```

在接管成功完成后，host3 将成为新的主数据库。同时，host2 将变得重新可用。host3 通知 host2 和 host4，它现在已成为主数据库。在 host3 上，**hadr_remote_host**、**hadr_remote_svc** 和 **hadr_remote_inst** 的值已重新配置为指向 host2，该数据库是主体备用数据库，因为它是 host3 上的 **hadr_target_list** 中的第一个条目。在 host2 上，同步方式已重新配置为 SUPERASYNC，因为这是 host3 上的 **hadr_syncmode** 设置；此外，**hadr_remote_host**、**hadr_remote_svc** 和 **hadr_remote_inst** 已更新（持续不断地）。host4 会自动重定向至新的主数据库 host3。在此重定向操作中，host4 将更新（持续不断地）它的 **hadr_remote_host**、**hadr_remote_svc** 和 **hadr_remote_inst** 配置参数。不会对 host1 进行自动重新配置。表 14 中显示了各个数据库的配置。

表 14. 在故障转移后各 HADR 数据库的配置值. 第 3 列到第 5 列的第 3 行到第 5 行以粗体显示，表示它们已自动重新配置

配置参数	Host1（不可用）	Host2	Host3	Host4
hadr_target_list	host2:40|host3:41| host4:42	host1:10|host3:41| host4:42	host2:40|host1:10| host4:42	host2:40|host1:10| host3:41
hadr_remote_host	host2	**host3**	**host2**	**host3**
hadr_remote_svc	40	**41**	**40**	**41**
hadr_remote_inst	dbinst2	**dbinst3**	**dbinst2**	**dbinst3**
hadr_local_host	host1	host2	host3	host4
hadr_local_svc	10	40	41	42
配置的 **hadr_syncmode**	SYNC	SYNC	SUPERASYNC	SUPERASYNC
有效的 **hadr_syncmode**	不适用	**SUPERASYNC**	不适用	SUPERASYNC

HADR“在备用数据库上读取”功能

在高可用性和灾难恢复 (HADR) 解决方案中，可以使用“在备用数据库上读取”功能在备用数据库上运行只读操作。在备用数据库上运行的读操作不会影响备用数据库的主要角色（重放主数据库递送来的日志）。

“在备用数据库上读取”功能可降低 HADR 安装的总拥有成本。备用数据库的这一扩展角色可让您以新的方式利用备用数据库，例如，运行部分工作负载，否则必须在主数据库上运行这些工作负载。这又可以释放主数据库以执行其他工作负载。

读取客户机和写入客户机继续连接至主数据库；然而，读取客户机还可以连接至已启用读取的备用数据库（或*活动备用数据库*），只要该客户机未处于本地同步复制状态或仅重放时间中。活动备用数据库的主要角色仍然是重放主数据库递送来的日志。因此，备用数据库上的数据应与主数据库上的数据几乎完全相同。如果发生故障转移，那么与备用数据库的任何用户连接将会终止，而备用数据库将作为新的主数据库接管工作。

在备用数据库上，支持所有读取查询类型，包括可滚动和不可滚动游标。所有四种 HADR 同步方式（SYNC、NEARSYNC、ASYNC 和 SUPERASYNC）和所有 HADR 状态（本地同步复制除外）都支持读取能力。

启用“在备用数据库上读取”

可以使用 **DB2_HADR_ROS** 注册表变量对高可用性和灾难恢复 (HADR) 备用数据库启用“在备用数据库上读取”功能。

开始之前

建议将数据库配置参数 **logindexbuild** 设置为 ON。这可避免任何无效的索引，从而防止查询存取方案产生的性能影响。

同时建议您在启用“在备用数据库上读取”时使用虚拟 IP。客户机重新路由不能区分可写入数据库（主数据库和标准数据库）和只读数据库（备用数据库）。配置主数据库与备用数据库之间的客户机重新路由可将应用程序路由至它们不打算运行于的数据库。

过程

1. 将 **DB2_HADR_ROS** 注册表变量设置为 ON。
2. 为 HADR 设置和初始化主数据库和备用数据库。 请参阅第 30 页的『初始化高可用性灾难恢复 (HADR)』。

结果

现在，备用数据库被视为*活动备用数据库*，这意味着它将接受只读工作负载。

下一步做什么

现在，您可以根据需要利用备用数据库，例如，配置部分只读工作负载在其上运行。

要使应用程序能够保持对备用数据库的访问权，请遵循 『Continuous access to Read on Standby databases using Virtual IP addresses』 白皮书中所描述的步骤。

活动备用数据库上的数据并行性

在 HADR 主数据库上进行的更改不一定会在 HADR 活动备用数据库上反映。直到主数据库清空或发送其日志到磁盘之后，主数据库上未落实的更改才会复制到备用数据库。

只能保证在落实日志之后将它们清空到磁盘 - 因而发送到备用数据库。日志缓冲区已满等不确定条件也会触发日志清空。因此，主数据库上未落实的更改很可能会在主数据库的日志缓冲区中保留很长时间。因为记录器会避免清空部分页面，此情况尤其会影响主数据库上细小的未落实更改。

如果在备用数据库上运行的工作负载要求数据与主数据库上的数据几乎完全相同，那么应考虑提高落实事务的频率。

活动备用数据库上的隔离级别

在活动备用数据库（启用了读取的 HADR 备用数据库）上，唯一受支持的隔离级别是未落实的读 (UR)。如果应用程序、语句或子语句请求的隔离级别高于 UR，那么将会返回错误（SQL1773N 原因码 1）。

如果您需要 UR 以外的隔离级别，请考虑将 HADR 主数据库而不是备用数据库用于此应用程序。如果您只是想避免接收到此消息，请将 **DB2_STANDBY_ISO** 注册表变量设置为 UR。当 **DB2_STANDBY_ISO** 设置为 UR 时，会以静默方式将隔离级别强制为 UR。此设置优先于所有其他隔离设置，如语句隔离和程序包隔离。

活动备用数据库上的仅重放时间

当 HADR 活动备用数据库重放 DDL 日志记录或维护操作时，备用数据库将进入*仅重放时间*。当备用数据库处于仅重放时间中时，与备用数据库的现有连接将会终止，与备用数据库的新连接将被阻止（SQL1776N 原因码 4）。

完成所有活动 DDL 或维护操作的重放后，备用数据库上就会允许新连接。

在仅重放窗口中，只有在备用数据库上保持活动的用户连接才可以执行 **DEACTIVATE DATABASE** 或 **TAKEOVER** 命令。如果在仅重放时间的开头强制关闭应用程序，那么将会返回错误 (SQL1224N)。根据连接至活动备用数据库的阅读器数，在备用数据库上重放 DDL 日志记录或维护操作之前可能会有短暂延迟。

在 HADR 主数据库上运行时，有许多 DDL 语句和维护操作会触发备用数据库上的仅重放时间。下面的列表并不全面。

DDL 语句

- CREATE、ALTER 或 DROP TABLE（用于 DGTT 的 DROP TABLE 除外）
- CREATE GLOBAL TEMP TABLE
- TRUNCATE TABLE
- RENAME TABLE
- RENAME TABLESPACE
- CREATE、DROP 或 ALTER INDEX
- CREATE 或 DROP VIEW
- CREATE、ALTER 或 DROP TABLESPACE
- CREATE、ALTER 或 DROP BUFFER POOL
- CREATE、ALTER 或 DROP FUNCTION
- CREATE、ALTER 或 DROP PROCEDURE
- CREATE 或 DROP TRIGGER
- CREATE、ALTER 或 DROP TYPE

- CREATE、ALTER 或 DROP ALIAS
- CREATE 或 DROP SCHEMA
- CREATE、ALTER 或 DROP METHOD
- CREATE、ALTER 或 DROP MODULE
- CREATE、ALTER 或 DROP NICKNAME
- CREATE、ALTER 或 DROP SEQUENCE
- CREATE、ALTER 或 DROP WRAPPER
- CREATE、ALTER 或 DROP FUNCTION MAPPING
- CREATE 或 DROP INDEX EXTENSION
- CREATE 或 DROP INDEX FOR TEXT
- CREATE 或 DROP EVENT MONITOR
- CREATE、ALTER 或 DROP SECURITY LABEL
- CREATE、ALTER 或 DROP SECURITY LABEL COMPONENT
- CREATE、ALTER 或 DROP SECURITY POLICY
- CREATE 或 DROP TRANSFORM
- CREATE、ALTER 或 DROP TYPE MAPPING
- CREATE、ALTER 或 DROP USER MAPPING
- CREATE 或 DROP VARIABLE
- CREATE、ALTER 或 DROP WORKLOAD
- GRANT USAGE ON WORKLOAD
- REVOKE USAGE ON WORKLOAD
- CREATE、ALTER 或 DROP SERVICE CLASS
- CREATE、ALTER 或 DROP WORK CLASS SET
- CREATE、ALTER 或 DROP WORK ACTION SET
- CREATE、ALTER 或 DROP THRESHOLD
- CREATE、ALTER 或 DROP HISTOGRAM TEMPLATE
- AUDIT
- CREATE、ALTER 或 DROP AUDIT POLICY
- CREATE 或 DROP ROLE
- CREATE、ALTER 或 DROP TRUSTED CONTEXT
- REFRESH TABLE
- SET INTEGRITY

维护操作

- 典型重组或脱机重组
- 适当重组或联机重组
- 索引重组（全部索引，个别索引）
- MDC 和 ITC 回收重组
- 装入
- 绑定或重新绑定

- db2rbind
- Runstats
- 表移动
- 自动统计信息
- 自动重组
- 实时统计信息

其他操作或动作

- 为具有 COMPRESS YES 属性的表自动创建字典
- 在已拆离的表分区上执行异步索引清除
- 隐式重新绑定
- 隐式索引重建
- 手动更新统计信息
- 延迟 MDC 转出
- 在 MDC 转出后执行异步索引清除
- 在插入到 MDC 或 ITC 表时复用已删除的 MDC 或 ITC 块
- 对于插入、更新和删除任务，更新目录表 SYSJOBS 和 SYSTASKS 的异步后台进程

监视仅重放时间

要监视活动备用数据库上的仅重放窗口，请使用带 **-hadr** 选项的 **db2pd** 命令。在以下示例中，三个相关元素如下:

- **ReplayOnlyWindowStatus**，指出在备用数据库上，DDL 或维护操作重放是否在进行中。该值一般为“Inactive”，但仅重放时间处于活动状态时，该值为“Active”。
- **ReplayWindowStartTime**，指出当前仅重放时间（如果有）变成活动的时间。
- **MaintenanceTxCount** 或 **DDLTxCount**，指出在当前仅重放时间（如果有）中，到目前为止已执行的现有未落实 DLL 或维护事务总数。

```
db2pd -db hadrdb -hadr
Database Partition 0 -- Database HADRDB -- Active -- Up 0 days 00:00:06

HADR Information:
Role    State  SyncMode HeartBeatsMissed   LogGapRunAvg (bytes)
Standby Peer  Nearsync 0                  0

ConnectStatus ConnectTime                            Timeout
Connected     Sat Jun 15 03:09:35 2008           120

ReplayOnlyWindowStatus       ReplayOnlyWindowStartTime          MaintenanceTxCount
Active                       Sun Jun 16 08:09:35 2008           5

LocalHost  LocalService
skua       52601

RemoteHost  RemoteService  RemoteInstance
gull        52600          vinci

PrimaryFile  PrimaryPg  PrimaryLSN
S0000000.LOG 1          0x000000000137126F

StandByFile  StandByPg  StandByLSN
S0000000.LOG 0          0x000000000137092E
```

对尽量降低仅重放时间的影响的建议

因为在 HADR 备用数据库上，重放操作优先于阅读器，所以频繁的只读窗口可能会扰乱已连接至或尝试连接至备用数据库的阅读器。要避免或尽量降低此影响，请考虑下列建议:

- 在预定维护窗口期间运行 DDL 和维护操作，最好在非高峰时间运行。
- 统一运行 DDL 操作，而不是在多个组中运行。
- 仅对所需的表而不是所有表运行 **REORG** 或 **RUNSTATS**。
- 在主数据库上运行 DDL 或维护操作之前，使用带 **ALL** 选项的 **FORCE APPLICATION** 命令终止活动备用数据库上的应用程序。监视仅重放时间以确定它何时不活动，然后在备用数据库上重新部署应用程序。

在活动备用数据库上临时终止读取应用程序

虽然可以使用 HADR 活动备用数据库来运行只读工作负载，但是它的主要角色是重放日志记录，以便在它必须接管主数据库的角色时，能够与 HADR 主数据库保持同步。

如果只读工作负载导致备用数据库在其日志重放中落后，那么您可能要临时终止与备用数据库的所有连接，以使它可以保持同步。

关于此任务

使用以下过程来使阅读器临时无法访问活动备用数据库。

过程

1. 发出 **FORCE APPLICATION** 命令。这会终止与备用数据库的现有连接。
2. 更改虚拟 IP 配置。这可防止与备用数据库建立新连接。

下一步做什么

备用数据库通过日志重放与主数据库保持同步后，需要将虚拟 IP 配置还原为其原始设置，以允许与活动备用数据库的连接继续。

“在备用数据库上读取”限制

高可用性和灾难恢复 (HADR)“在备用数据库上读取”功能可让您在 HADR 活动备用数据库上运行只读工作负载。除了只读限制之外，此功能还存在应了解的下列限制。

- 不允许在备用数据库上执行写入操作。在此上下文中，写入操作是修改目录、表和索引等永久数据库对象的操作。在备用数据库上执行写入操作会返回错误（SQL1773N 原因码 5）。特别是，不能执行会导致在备用数据库上生成日志记录的任何操作。
- 在重放 DDL 日志记录或维护操作期间（*仅重放时间*），用户连接无法访问备用数据库。有关更多信息，请参阅第 193 页的『活动备用数据库上的仅重放时间』。
- 当备用数据库处于本地同步复制状态时，用户连接无法访问该数据库。尝试连接此状态的客户机将收到错误（SQL1776N 原因码 1）。
- 在备用数据库上只支持未落实的读（UR）隔离级别。请求更高隔离级别的应用程序、语句或子语句将收到错误（SQL1773N 原因码 1）。有关更多信息，请参阅第 193 页的『活动备用数据库上的隔离级别』。

- 不会将实例级别审计配置复制到备用数据库。必须使用 db2audit 工具确保实例级别审计设置在主数据库和备用数据库上是相同的。
- 在备用数据库上不支持已声明临时表 (DGTT)。在备用数据库上尝试创建或访问它们将收到错误（SQL1773N 原因码 4）。
- 创建临时表 (CGTT) 只能创建在主数据库上，且它们的定义会被复制到备用数据库。但是，在备用数据库上不支持访问 CGTT，尝试创建或访问它们将收到错误（SQL1773N 原因码 4）。
- 在主数据库上创建“创建临时表”(CGTT) 将触发备用数据库上的仅重放窗口。
- 在备用数据库上不能访问最初未进行日志记录（NLI）表。在备用数据库上尝试读取 NLI 表的应用程序将收到错误（SQL1477N）。
- 备用数据库上的查询只能使用 SMS 系统临时表空间。在备用数据库上执行使用 DMS 系统临时表空间的查询可能会导致错误（SQL1773N 原因码 5）。
- 要成功查询 XML 和大对象 (LOB) 数据，这些数据必须是直接插入的，否则会返回错误（SQL1773N 原因码 3）。
- 不能查询下列数据: 长字段 (LF)、基于这些数据类型的其中一种的单值类型以及结构化类型列。尝试查询这些数据类型将收到错误（SQL1773N 原因码 3）。
- 在备用数据库上不支持说明工具（**db2exfmt** 和 **db2expln**）和 **db2batch** 工具（SQL1773N 原因码 5）。如果要分析只读工作负载的性能，那么首先应在主数据库上运行这些工具，在主数据库上对工作负载进行必要的优化，然后将优化后的工作负载移至备用数据库。
- 在备用数据库上不支持程序包的显式绑定及重新绑定和隐式重新绑定。尝试运行引用失效对象的静态程序包及这些程序包的隐式重新绑定将导致错误（分别为 SQL1773N 原因码 5 和 6）。应该转为在主数据库上绑定程序包，并在将更改复制到备用数据库后，在备用数据库上运行程序包。
- 在备用数据库上不支持自调整内存管理器（STMM）。如果要调整备用数据库（以适合运行只读工作负载或以在接管后执行良好），那么必须手动调整。
- 主数据库上的工作负载管理器（WLM）DDL 语句将在备用数据库上重放，但它们在备用数据库上不会生效；但是，存在于数据库备份（用于建立备用数据库）中的任何定义在启用了读取的备用数据库上将是活动的。
- 在备用数据库上不支持创建和改变序列。同样，不能使用 NEXT VALUE 表达式来生成序列中的下一个值。
- 在备用数据库上不支持无效对象的运行时重新验证。
- 不能将备用数据库配置为 Federation Server。
- 在备用数据库上不支持备份和归档操作。
- 在备用数据库上不支持停顿操作。

在高可用性解决方案中检测和响应系统中断

实现高可用性解决方案并不能防止硬件或软件故障。但是，具备冗余系统和故障转移机制可以使您的解决方案能够检测和响应故障并重新路由工作负载，从而使用户应用程序仍然可以工作。

过程

当出现故障时，数据库解决方案必须执行以下操作:

1. 检测故障。

   故障转移软件可以使用脉动信号监视来确定系统组件的可用性。脉动信号监视器侦听来自所有系统组件的正常通信。如果脉动信号监视器无法侦听到来自某个组件的通信，它将向系统发出信号，指示该组件出现故障。

2. 响应故障: 故障转移。

   a. 标识辅助组件、使之联机并初始化以接管故障组件的操作。

   b. 将工作负载重新路由至辅助组件。

   c. 从系统除去故障组件。

3. 恢复故障。

   如果主数据库服务器出现故障，首先要做的是将客户机重定向至备用服务器或将故障转移至备用数据库，使客户机应用程序在尽可能少中断的情况下工作。一旦故障转移成功，您必须修复故障数据库服务器上的所有问题，使它能够重新集成到解决方案中。修复故障数据库服务器可能只需将它重新启动。

4. 恢复正常操作。

   一旦修复故障数据库系统后，必须将它重新集成到数据库解决方案中。主数据库出现故障后，备用数据库接管其工作而成为新的主数据库，这时可以切换两者的数据库角色，从而重新集成原先的主数据库。也可以强制修复的数据库服务器重新作为主数据库服务器接管。

下一步做什么

DB2 数据库可以为您执行下列步骤中的一些。例如:

- DB2 高可用性灾难恢复 (HADR) 脉动信号监视元素 **hadr_heartbeat** 可以检测主数据库何时出现故障。
- DB2 客户机重新路由可以将工作负载从故障数据库服务器转移到辅助数据库服务器上。
- DB2 故障监视器可以重新启动意外终止的数据库实例。

管理通知日志

管理通知日志（*instance_name*.nfy）是可从中获取有关大量数据库管理和维护活动的信息的存储库。数据库管理员可以使用这些信息来诊断问题、调整数据库或仅监视数据库。

DB2 数据库管理器会将以下类型的信息写入 UNIX 和 Linux 操作系统平台上的管理通知日志（在 Windows 操作系统平台上，事件日志用来记录管理通知事件）:

- DB2 实用程序（例如 **REORG** 和 **BACKUP**）的状态
- 客户机应用程序错误
- 服务类更改
- 许可证发放活动
- 文件路径

- 存储问题
- 监视活动
- 建立索引活动
- 表空间问题

还会使用标准化消息格式将管理通知日志消息记录到 **db2diag** 日志文件中。

通知消息提供了其他信息以补充提供的 SQLCODE。

管理通知日志文件可以下面两种不用的形式存在:

单个管理通知日志文件

一个活动的管理通知日志文件，名为 *instance_name*.nfy，其大小无限地增大。这是缺省形式，每当 **diagsize** 数据库管理器配置参数的值为 0（此参数的缺省值为 0）时，都存在这种形式。

旋转管理通知日志文件

单个活动日志文件（名为 *instance_name*.*N*.nfy，其中 *N* 是文件名下标，即从 0 开始不断增大的数字），尽管可以在 **diagpath** 配置参数定义的位置中找到一系列管理通知日志文件，但是每个这样的日志文件都在增大直到达到受限的大小，此时，该日志文件会被关闭，并且会创建并打开新的日志文件来进行记录，它带有增量的文件名下标（*instance_name*.*N+1*.nfy）。每当 **diagsize** 数据库管理器配置参数具有非零值时，都存在这种形式。

注: 在 Windows 操作系统平台上，不提供单个管理通知日志文件和旋转管理通知日志文件。

通过相应地设置 **diagsize** 数据库管理器配置参数，可以选择系统上存在这两种形式中的哪一种。

配置

通过设置下列数据库管理器配置参数，可以在大小、位置和事件类型以及所记录详细信息级别这些方面对管理通知日志文件进行配置:

diagsize

diagsize 的值决定将采用哪种形式的管理通知日志文件。如果该值为 0，那么将采用单个管理通知日志文件。如果该值不为 0，那么将采用旋转管理通知日志文件，此非零值还指定所有旋转诊断日志文件以及所有旋转管理通知日志文件的总体大小。必须重新启动实例才能使 **diagsize** 参数的新值生效。有关完整详细信息，请参阅“diagsize - 诊断日志文件大小配置参数”主题。

diagpath

诊断信息可指定写入到 **diagpath** 配置参数所定义位置处的管理通知日志文件中。有关完整详细信息，请参阅“diagpath - 诊断数据目录路径配置参数”主题。

notifylevel

可以使用 **notifylevel** 配置参数来指定写入管理通知日志文件的事件类型和详细信息级别。有关完整详细信息，请参阅“notifylevel - 通知级别配置参数”主题。

注: 如果 **diagsize** 配置参数设置为非零值，那么该值指定所有旋转管理通知日志文件和诊断数据目录内包含的所有旋转诊断日志文件的组合的总大小。例如，如果具有 4 个

数据库分区的系统将 **diagsize** 设置为 1 GB，那么通知日志和诊断日志组合之后所获得的最大总大小可以达到 4 GB（4 x 1 GB）。

检测意外中断

在对组件故障作出响应之前，必须先检测到出现故障的组件。DB2 数据服务器有几个用于监视数据库运行状况或检测数据库是否失败的工具。

可以配置这些工具，以便在检测到故障时通知您或执行预定义的操作。

过程

可以使用下列工具来检测 DB2 数据库解决方案中的某个部件何时出现了故障:

DB2 故障监视器工具

DB2 故障监视器工具使 DB2 数据库实例正常运行。当 DB2 故障监视器连接的 DB2 数据库实例意外终止时，DB2 故障监视器将重新启动该实例。如果数据库解决方案是在集群中实现的，那么应该配置集群管理软件，以重新启动出现故障的数据库实例，而不是重新启动 DB2 故障监视器。

集群环境中的脉动信号监视

集群管理软件在集群中的节点之间使用脉动信号消息来监视节点的运行状况。当节点停止响应或发送任何消息时，集群管理器可检测到该节点出现故障。

监视 DB2 高可用性灾难恢复 (HADR) 数据库

HADR 功能具有它自己的脉动信号监视器。主数据库和备用数据库都期望定期接收到其他数据库发出的脉动信号消息。

高可用性灾难恢复 (HADR) 监视

监视是设置和维护 HADR 设置的一个重要组成部分。DB2 监视界面会显示环境的配置和运行状况的详细图片。

可以使用许多方法来监视 HADR 数据库的状态。以下是监视 HADR 的两种首选方法:

- db2pd 命令
- MON_GET_HADR 表函数

还可以使用下列方法，但是从 V10.1 开始，不推荐使用这些方法，在将来的发行版中可能会除去这些方法:

- **GET SNAPSHOT FOR DATABASE** 命令
- db2GetSnapshot API
- SNAPHADR 管理视图
- SNAP_GET_HADR 表函数
- 其他快照管理视图和表函数

db2pd 命令

此命令从 DB2 内存集中检索信息。可从主数据库或备用数据库发出此命令。如果正使用多备用数据库方式，并且从备用数据库发出此命令，那么它不会返回任何有关其他备用数据库的信息。如果从主数据库发出此命令，那么它会返回有关所有备用数据库的信息。

要查看数据库 HADRDB 的高可用性灾难恢复信息，请发出以下命令:

```
db2pd -db HADRDB -hadr
```

如果您从主机发出该命令，您将收到类似于如下示例输出的内容:

```
  Database Member 0 -- Database HADRDB -- Active -- Up 0 days 00:23:17 --
Date 06/08/2011 13:57:23

                            HADR_ROLE = PRIMARY
                          REPLAY_TYPE = PHYSICAL
                        HADR_SYNCMODE = SYNC
                           STANDBY_ID = 1
                        LOG_STREAM_ID = 0
                           HADR_STATE = PEER
                   PRIMARY_MEMBER_HOST = hostP.ibm.com
                      PRIMARY_INSTANCE = db2inst
                        PRIMARY_MEMBER = 0
                   STANDBY_MEMBER_HOST = hostS1.ibm.com
                      STANDBY_INSTANCE = db2inst
                        STANDBY_MEMBER = 0
                   HADR_CONNECT_STATUS = CONNECTED
              HADR_CONNECT_STATUS_TIME = 06/08/2011 13:38:10.199479 (1307565490)
            HEARTBEAT_INTERVAL(seconds) = 25
                  HADR_TIMEOUT(seconds) = 100
          TIME_SINCE_LAST_RECV(seconds) = 3
               PEER_WAIT_LIMIT(seconds) = 0
             LOG_HADR_WAIT_CUR(seconds) = 0.000
      LOG_HADR_WAIT_RECENT_AVG(seconds) = 0.006298
     LOG_HADR_WAIT_ACCUMULATED(seconds) = 0.516
                    LOG_HADR_WAIT_COUNT = 82
  SOCK_SEND_BUF_REQUESTED,ACTUAL(bytes) = 0, 50772
  SOCK_RECV_BUF_REQUESTED,ACTUAL(bytes) = 0, 87616
              PRIMARY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
              STANDBY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
                    HADR_LOG_GAP(bytes) = 0
       STANDBY_REPLAY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
         STANDBY_RECV_REPLAY_GAP(bytes) = 0
                     PRIMARY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
                     STANDBY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
                STANDBY_REPLAY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
           STANDBY_RECV_BUF_SIZE(pages) = 16
               STANDBY_RECV_BUF_PERCENT = 0
             STANDBY_SPOOL_LIMIT(pages) = 0
                  PEER_WINDOW(seconds) = 0
              READS_ON_STANDBY_ENABLED = Y
      STANDBY_REPLAY_ONLY_WINDOW_ACTIVE = N
```

MON_GET_HADR 表函数

如果对主数据库发出此查询，那么它会返回有关所有备用数据库的信息。如果要对备用数据库发出 MON_GET_HADR 函数，那么应注意以下几点:

- 必须启用对备用数据库的读取。
- 即使 HADR 设置处于多备用数据库方式，该表函数也不会返回有关任何其他备用数据库的任何信息。

例如，您可以对主数据库发出以下查询:

```
db2 "select HADR_ROLE, STANDBY_ID, HADR_STATE,
            varchar(PRIMARY_MEMBER_HOST,20) as PRIMARY_MEMBER_HOST,
            varchar(STANDBY_MEMBER_HOST,20) as STANDBY_MEMBER_HOST
     from table (mon_get_hadr(NULL))"
```

样本输出如下所示:

```
HADR_ROLE  STANDBY_ID  HADR_STATE  PRIMARY_MEMBER_HOST  STANDBY_MEMBER_HOST
---------  ----------  ----------  -------------------  -------------------
PRIMARY             1  PEER        hostP.ibm.com        hostS1.ibm.com

1 record(s) selected.
```

GET SNAPSHOT FOR DATABASE 命令

此命令收集状态信息并对输出进行格式化。返回的信息是发出此命令时的数据库管理器工作状态快照。在输出中，HADR 信息显示在标题 HADR status 下面。

db2GetSnapshot API

此 API 收集数据库管理器监视器信息并将其写至用户分配的数据缓冲区。返回的信息是调用此 API 时的数据库管理器工作状态快照。

SNAPHADR 管理视图和 SNAP_GET_HADR 表函数

此管理视图及此表函数从数据库快照返回有关 HADR（特别是 HADR 逻辑数据组）的信息。

其他快照管理视图和表函数

可以使用下列快照管理视图和表函数（它们并不特定于 HADR，而是会返回一组更广泛的信息）来查询部分 HADR 信息:

- ADMIN_GET_STORAGE_PATHS
- MON_GET_TRANSACTION_LOG
- SNAPDB
- SNAPDB_MEMORY_POOL
- SNAPDETAILLOG
- SNAP_GET_DB
- SNAP_GET_DB_MEMORY_POOL

响应意外中断

如果数据库管理软件或集群管理软件检测到某个数据库服务器出现故障，那么数据库解决方案必须尽快而且尽可能平稳地响应该故障。

数据库解决方案必须通过重新路由工作负载（如果可能）并将故障转移到辅助或备用数据库（如果有），尝试对用户应用程序屏蔽该故障。

过程

如果数据库或集群管理软件检测到数据库服务器出现故障，那么您或您的数据库或集群管理软件必须执行以下操作:

1. 确定辅助数据库服务器、使之联机并初始化，以接管故障数据库服务器的操作。

   如果是使用 DB2 高可用性灾难恢复 (HADR) 来管理主数据库服务器和备用数据库服务器，HADR 将使备用数据库与主数据库保持同步；而且 HADR 将管理备用数据库对主数据库的接管。
2. 将用户应用程序工作负载重新路由至辅助数据库服务器。

   DB2 客户机重新路由可以将客户机应用程序从故障数据库服务器自动重新路由至先前为此目的标识和配置的辅助数据库服务器。
3. 从系统除去故障数据库服务器以将它修复。

   一旦将用户应用程序重新路至辅助或备用数据库，故障数据库服务器将不能处理任何客户机应用程序请求，直到重新启动或修复。例如，如果主数据库上出现故障的原因是数据库实例意外终止，那么 DB2 故障监视设施将自动将它重新启动。

客户机自动重新路由示例

DB2 数据服务器客户机重新路由可将客户机应用程序从故障数据库服务器自动重新路由至先前为此目的标识并配置的辅助数据库服务器。您可以方便地创建客户机应用程序来测试和演示此 DB2 数据服务器功能。

以下是客户机应用程序的自动客户机路由示例（仅使用伪码显示）：

```
int checkpoint = 0;

        check_sqlca(unsigned char *str, struct sqlca *sqlca)
{
   if (sqlca->sqlcode == -30081)
   {
                    // as communication is lost, terminate the application
                       right away
                    exit(1);
   }
         else

                    // print out the error
      printf(...);

 if (sqlca->sqlcode == -30108)
 {
            // connection is re-established, re-execute the failed
               transaction
                          if (checkpoint == 0)
         {
                                  goto checkpt0;
         }
            else if (checkpoint == 1)
         {
                                goto checkpt1;
         }
                          else if (checkpoint == 2)
         {
                                  goto checkpt2;
         }
         ....
                          exit;
 }
 }
}

main( )
{
              connect to mydb;
              check_sqlca("connect failed", &sqlca);

        checkpt0:
              EXEC SQL set current schema XXX;
              check_sqlca("set current schema XXX failed", &sqlca);

   EXEC SQL create table t1...;
              check_sqlca("create table t1 failed", &sqlca);

              EXEC SQL commit;
              check_sqlca("commit failed", &sqlca);

              if (sqlca.sqlcode == 0)
   {
                    checkpoint = 1;
   }

        checkpt1:
```

```
                EXEC SQL set current schema YYY;
                check_sqlca("set current schema YYY failed", &sqlca);

        EXEC SQL create table t2...;
                check_sqlca("create table t2 failed", &sqlca);

                EXEC SQL commit;
                check_sqlca("commit failed", &sqlca);

                if (sqlca.sqlcode == 0)
        {
                      checkpoint = 2;
        }
    ...
    }
```

在客户机上，对名为“mydb”的数据库编目，此数据库引用节点“hornet”，其中“hornet”也在节点目录（主机名为“hornet”，端口号为 456）予以编目。

涉及非 HADR 数据库的示例

在服务器“hornet”（主机名为 hornet，并且带有端口号）上，创建数据库“mydb”。而且，还在备用服务器处创建数据库“mydb”（主机名为“montero”，端口号为 456）。您还需要在服务器“hornet”上更新数据库“mydb”的备用服务器，如下所示:

```
db2 update alternate server for database mydb using hostname montero port 456
```

在上述样本应用程序中，未设置客户机自动重新路由功能，如果 `create table t1` 语句中存在通信错误，那么该应用程序将终止。设置客户机自动重新路由功能之后，DB2 数据库管理器将尝试再次建立与主机“hornet”（端口为 456）的连接。如果仍然无法工作，那么 DB2 数据库管理器将尝试备用服务器位置（主机名为“montero”，端口为 456）。假定连接至备用服务器位置时没有出现通信错误，那么应用程序可以继续运行后面的语句（并重新运行失败事务）。

涉及 HADR 数据库的示例

在服务器“hornet”（主机名为 hornet，并且带有端口号），创建主数据库“mydb”。另外还在端口为 456 的主机“montero”上创建备用数据库。有关如何为主数据库和备用数据库设置 HADR 的信息可于 *数据恢复及高可用性指南与参考* 中找到。您还需要为数据库“mydb”更新备用服务器，如下所示:

```
db2 update alternate server for database mydb using hostname montero port 456
```

在上述样本应用程序中，未设置客户机自动重新路由功能，如果 `create table t1` 语句中存在通信错误，那么该应用程序将终止。设置客户机自动重新路由功能之后，DB2 数据库系统将尝试再次建立与主机“hornet”（端口为 456）的连接。如果仍然无法工作，那么 DB2 数据库系统将尝试备用服务器位置（主机名为“montero”，端口为 456）。假定连接至备用服务器位置时没有出现通信错误，那么应用程序可以继续运行后面的语句（并重新运行失败事务）。

涉及 SSL 的示例

当同时将 SSL 用于连接时，还可以使用客户机重新路由功能。该设置与对 HADR 数据库的先前示例显示的设置类似。

在客户机上，对引用节点“hornet_ssl”的数据库“mydb”的数据库别名“mydb_ssl”编目。在节点目录（主机名为“hornet”，SSL 端口号为 45678 且安全性参数设置为 SSL）中对“hornet_ssl”编目。

在备用服务器（主机名为“montero”，SSL 端口号为 45678 且安全性参数设置为 SSL）上，也对某个数据库别名编目。还需要在服务器“hornet”上更新别名“mydb_ssl”的备用服务器，如下所示:

```
db2 update alternate server for database mydb_ssl using hostname montero port 45678
```

在上述样本应用程序中，将连接语句更改为 `connect to mydb_ssl`。在未设置客户机自动重新路由功能的情况下，如果 `create table t1` 语句中存在通信错误，那么该应用程序将终止。在设置了客户机自动重新路由功能的情况下，DB2 数据库管理器将尝试使用 SSL 再次建立与主机“hornet”（端口为 45678）的连接。如果仍然无法工作，那么 DB2 数据库管理器将使用 SSL 尝试备用服务器位置（端口 45678 处的主机“montero”）。假定连接至备用服务器位置时没有出现通信错误，那么应用程序可以继续运行后面的语句（并重新运行失败事务）。

执行 HADR 故障转移操作

因为当前主数据库不可用，如果要当前备用数据库成为新的主数据库，那么可以执行故障转移。

关于此任务

警告:

此过程可能导致数据丢失。执行此紧急过程之前，复查以下信息:

- 确保主数据库不再处理数据库事务。如果主数据库仍在运行，但无法与备用数据库通信，那么执行强制接管操作（发出具有 **BY FORCE** 选项的 **TAKEOVER HADR** 命令）会产生两个主数据库。当有两个主数据库时，每个数据库将拥有不同的数据，并且不能再自动同步两个数据库。
  - 如有可能，请取消激活主数据库或停止其实例。（如果主系统已挂起、崩溃或不可访问，那么可能无法执行此操作。）执行接管操作之后，如果稍后重新启动了发生故障的数据库，它将不会自动地承担主数据库的角色。
- 事务损失的可能性和程度取决于特定配置和情况:
  - 如果主数据库在对等状态或断开对等状态下发生故障，且同步方式为同步(SYNC)，那么备用数据库将不会丢失在主数据库发生故障之前已报告且已落实到应用程序的事务。
  - 如果主数据库在对等状态或断开对等状态下发生故障，且同步方式为接近同步(NEARSYNC)，那么当主数据库和备用数据库同时发生故障时，备用数据库只会丢失由主数据库落实的事务。
  - 如果主数据库在对等状态或断开对等状态下发生故障，且同步方式为异步(ASYNC)，那么在执行接管操作前备用数据库未接收事务的所有日志记录的情况下，此备用数据库可能丢失由主数据库落实的事务。如果备用数据库在能够将所有已接收到的日志写入磁盘之前崩溃，那么备用数据库也可能丢失由主数据库落实的事务。

    注: 因为在 ASYNC 方式下不允许使用对等时间，所以在该方式下，主数据库永远不会进入断开对等状态。

– 如果主数据库在远程同步复制状态下发生故障，且同步方式为超级异步 (SUPERASYNC)，那么在执行接管操作之前备用数据库未接收到事务的所有日志记录时，备用数据库可能丢失由主数据库落实的事务。如果备用数据库在能够将所有已接收到的日志写入磁盘之前崩溃，那么备用数据库也可能丢失由主数据库落实的事务。

注: 在 SUPERASYNC 方式下，数据库永远不会处于对等状态或断开对等状态。仅当同步方式为 SUPERASYNC 时，才允许在远程同步复制状态下执行故障转移（强制接管）。

– 如果主数据库在远程同步复制暂挂状态下发生故障，将丢失备用数据库尚未接收和处理的事务。

注: 数据库快照中显示的任何日志间隔将表明主数据库和备用数据库最后一次互相通信的间隔。从那时起，主数据库可能已处理了大量事务。

- 确保任何连接至新的主数据库（或者通过客户机重新路由功能来重新路由至新的主数据库）的应用程序已准备好处理下列情况:
  – 在故障转移期间丢失了数据。新的主数据库没有所有已在旧的主数据库上落实的事务。即使在 **hadr_syncmode** 配置参数已设置为 SYNC 时，也可能发生这种错误。由于 HADR 备用数据库按顺序应用日志，所以您可以假定以下情况: 如果在新的主数据库上落实 SQL 会话中的某个事务，那么已在这个新的主数据库上落实了同一会话中的所有先前事务。只能通过详细地分析日志流来确定跨多个会话的事务落实顺序。
  – 可能会发生以下情况: 对原始主数据库发出了一个事务，在原始主数据库上已落实该事务并将其复制至新的主数据库（原始备用数据库），但由于原始主数据库在向客户机报告已落实事务之前崩溃，所以未将该事务报告为已落实。您编写的任何应用程序都应该能够处理以下情况: 对原始主数据库发出了事务，但未报告已在原始主数据库上落实这些事务，将在新的主数据库（原始备用数据库）上落实这些事务。
  – 某些操作（例如对数据库配置和外部 UDF 对象所作的更改）不会被复制。
- 只能对备用数据库发出 **TAKEOVER HADR** 命令。
- HADR 无法与 DB2故障监视器 (db2fm) 配合使用，此故障监视器可用于自动重新启动发生故障的数据库。如果启用故障监视器，应该知道在可能发生故障的主数据库上可能执行的故障监视器操作。
- 仅当主数据库和备用数据库处于对等状态，或备用数据库处于远程同步复制暂挂状态时，接管操作才会发生。如果备用数据库处于任何其他状态，将返回错误。

  注: 通过将处于本地同步复制状态的备用数据库转换为标准数据库，可以使其正常使用。要执行此操作，通过发出 **DEACTIVATE DATABASE** 命令来关闭数据库，然后发出 **STOP HADR** 命令。一旦停止了 HADR，就必须在前述的备用数据库上完成前滚操作，才能使用它。当数据库从备用数据库转换为标准数据库之后，该数据库不能重新加入 HADR 数据库对。要在两台服务器上重新启动 HADR，请执行初始化 HADR 的过程。
  如果配置了对等时间，请在该时间到期前关闭主数据库，以避免在任何相关故障转移中丢失潜在事务。

在故障转移方案中，可以通过命令行处理器 (CLP) 或 db2HADRTakeover 应用程序编程接口 (API) 来执行接管操作。

过程

以下过程说明如何使用 CLP 在主数据库或备用数据库上初始化故障转移:

1. 完全禁用发生故障的主数据库。当数据库遇到内部错误时，正常关闭命令可能无法完全关闭该数据库。您可能需要使用操作系统命令来除去资源，例如进程、共享内存或网络连接。
2. 在备用数据库上，发出带有 **BY FORCE** 选项的 **TAKEOVER HADR** 命令。 在以下示例中，故障转移发生在数据库 LEAFS 上:

   ```
   TAKEOVER HADR ON DB LEAFS BY FORCE
   ```

   需要使用 **BY FORCE** 选项将主数据库脱机。

   如果未完全禁用主数据库，备用数据库仍会与主数据库连接，并且将消息发送到主数据库以请求其关闭。无论备用数据库是否接收到主数据库已关闭的确认信息，它仍将切换为主数据库的角色。

切换高可用性灾难恢复 (HADR) 中的数据库角色

在高可用性灾难恢复 (HADR) 期间，使用 **TAKEOVER HADR** 命令来切换主数据库和备用数据库的角色。

关于此任务

- 只能对备用数据库发出 **TAKEOVER HADR** 命令。如果发出此命令时主数据库尚未与备用数据库连接，接管操作就会失败。
- 仅当数据库处于对等状态时，才能使用 **TAKEOVER HADR** 命令切换主数据库和备用数据库的角色。如果备用数据库处于其他任何状态，就会返回错误消息。

过程

要切换 HADR 数据库角色，请执行以下操作:

- 使用 CLP 对备用数据库启动接管操作，对备用数据库发出不带 **BY FORCE** 选项的 **TAKEOVER HADR** 命令。

  在以下示例中，对备用数据库 LEAFS 执行接管操作:

  ```
  TAKEOVER HADR ON DB LEAFS
  ```

  在接管操作完成后，紧跟着发生日志满错误的可能性会略高。为了降低此类错误的可能性，在每次接管完成时都会自动启动异步缓冲池清空操作。随着异步缓冲池清空操作的进展，日志满错误的可能性也就随之降低。此外，如果配置提供了足够的活动日志空间量，就更不大可能会发生日志满错误。如果确实发生了日志满错误，当前事务将中止并回滚。

  注: 发出不带 **BY FORCE** 选项的 **TAKEOVER HADR** 命令将强制断开当前已连接至 HADR 主数据库的所有应用程序。此操作设计成与客户机自动重新路由功能配合工作，以便于在角色切换后将客户机重新路由到新的 HADR 主数据库。但是，如果强制断开应用程序与主数据库的连接对环境有破坏性，您可以实现自己的过程来在执行角色切换前关闭此类应用程序，接着在角色切换完成后重新启动这些应用程序并使用新的 HADR 主数据库作为它们的目标。
- 从应用程序调用 db2HADRTakeover 应用程序编程接口 (API)。
- 在 IBM Data Studio 中对 **TAKEOVER HADR** 命令打开任务助手。

在执行接管操作之后重新集成数据库

如果因为主数据库故障而在高可用性灾难恢复 (HADR) 环境中执行了接管操作，那么可以将发生故障的数据库恢复联机，并将此数据库用作备用数据库或将其返回至作为主数据库时的状态。

过程

要将发生故障的主数据库作为新的备用数据库重新集成到 HADR 数据库对中:

1. 修复原始主数据库所在的系统。这可能涉及修复发生故障的硬件或重新引导已崩溃的操作系统。
2. 将发生故障的主数据库作为备用数据库重新启动。 在以下示例中，将数据库 LEAFS 作为备用数据库启动:

   ```
   START HADR ON DB LEAFS AS STANDBY
   ```

   注: 如果数据库的两个副本的日志流不兼容，那么重新集成操作将失败。特别是，HADR 要求在原始主数据库上应用的操作不包括: 在原始备用数据库作为新的主数据库接管前，从未反映在该数据库上的已进行日志记录的操作。如果发生此种情况，通过复原新主数据库的备份映像，或者通过初始化分割镜像，可以将原始主数据库作为备用数据库重新启动。

   此命令成功返回并不意味着重新集成操作已成功; 这仅仅表示已启动数据库。重新集成操作仍在进行中。以后，如果重新集成操作失败，该数据库将自行关闭。应该使用 **GET SNAPSHOT FOR DATABASE** 命令或 **db2pd** 工具来监视备用数据库状态，以确保备用数据库保持处于联机状态并进行正常的状态过渡。必要时，可以检查管理通知日志文件和 **db2diag** 日志文件以了解数据库状态。

下一步做什么

原始主数据库作为备用数据库重新加入 HADR 数据库对之后，可以选择执行故障恢复操作来切换数据库角色，以再次将原始主数据库启用为主数据库。要执行此故障恢复操作，在备用数据库上发出以下命令:

```
TAKEOVER HADR ON DB LEAFS
```

注:

1. 如果 HADR 数据库未处于对等状态或者在数据库对之间未建立连接，那么此命令将失败。
2. 强制关闭在主数据库上打开的会话，并回滚未完成的事务。
3. 当切换主数据库和备用数据库的角色时，无法指定 **TAKEOVER HADR** 命令的 **BY FORCE** 选项。

第 6 章 DB2 集群服务的故障管理

本节包含 DB2 集群服务的概述及有关它如何处理特定类型的主机、集群高速缓存设施和成员故障的详细信息。

DB2 集群服务是提供自动脉动信号故障检测并在检测到故障后自动启动所需恢复操作的软件。它还提供了集群文件系统，此系统允许 DB2 pureScale实例中的每个主机访问公共文件系统。DB2 集群服务包括来自 IBM Tivoli System Automation for Multiplatforms (Tivoli SA MP) 软件、IBM Reliable Scalable Clustering Technology (RSCT) 软件及 IBM General Parallel File System (GPFS) 软件的技术。此技术作为 DB2 pureScale Feature 的集成部件打包。

自动集群高速缓存设施故障转移

如果主集群高速缓存设施 (CF) 失败，那么 DB2 集群服务自动尝试将其重新启动并将主角色故障转移至辅助集群高速缓存设施（假定建议的双集群高速缓存设施设置）。

因为集群高速缓存设施在 DB2 pureScale实例中充当的整合角色，失败的集群高速缓存设施将尽快重新启动并重新整合。故障的影响取决于以下因素:

- DB2 pureScale实例中有多少个集群高速缓存设施

  如果 DB2 pureScale实例中只有一个集群高速缓存设施，那么此实例将关闭。如果集群高速缓存设施因为软件故障失败，那么会自动启动组重新启动。如果集群高速缓存设施因为硬件故障而失败，那么用户需要修正此问题，之后将自动启动组重新启动。

  如果 DB2 pureScale实例中有两个集群高速缓存设施（这是建议设置），那么 DB2 集群服务会尝试将主角色故障转移至辅助集群高速缓存设施。如果主集群高速缓存设施因为软件故障而失败，那么它将作为辅助集群高速缓存设施自动重新启动并重新整合。如果主集群高速缓存设施因为硬件故障而失败，那么用户需要修正此问题，之后它将作为辅助集群高速缓存设施自动重新启动并重新整合。
- 主集群高速缓存设施失败时的辅助集群高速缓存设施的状态如何

  如果辅助集群高速缓存设施处于 PEER 状态，那么 DB2 集群服务会向其故障转移主角色。

  如果辅助集群高速缓存设施未处于 PEER 状态，那么实例将关闭。然后，DB2 集群服务会使用先前的辅助集群高速缓存设施（现在处于主角色）启动组重新启动。

自动重新启动

在 DB2 pureScale 环境中，DB2 集群服务自动检测软件和硬件故障，并根据所发生的故障类型启动成员重新启动或组重新启动。

自动重新启动帮助确保主机、成员或集群高速缓存设施故障对数据库的影响降至最低。成员故障（和后续重新启动）对于应用程序是透明的，仅暂时影响在失败成员上运行的未落实事务。尽管多个主机或成员发生故障，但应用程序可继续访问数据库。

© Copyright IBM Corp. 2001, 2013

在 DB2 集群服务不能在其原始主机上重新启动失败成员的情况下，失败成员将在称为*轻量级重新启动*的进程中的另一主机上重新启动。某些极严重故障（例如，DB2 pureScale实例中的所有集群高速缓存设施丢失）会导致数据库停机。在这类情况下，DB2 集群服务会启动集群高速缓存设施和成员的组重新启动。

成员重新启动和崩溃恢复

*成员重新启动*是在失败成员上重新启动数据库服务器进程并在需要成员的每个数据库上执行成员崩溃恢复的过程。

如果主机上的软件或硬件故障导致成员失败，那么 DB2 集群服务会检测此故障并自动重新启动成员。成员重新启动可以是*本地重新启动*（意味着它在其*原始主机*上重新启动）或*轻量级重新启动*（意味着它在另一主机上重新启动）：

本地重新启动

如果成员因为软件故障失败但成员的原始主机仍处于活动状态，那么 DB2 集群服务会尝试本地重新启动。本地成员重新启动使用缩减内存模型以确保未处理的数据快速恢复到一致状态。一旦恢复了未处理的数据，那么数据库的常规内存模型就会初始化以进行完整事务处理。

轻量级重新启动

如果成员的原始主机处于不活动状态或者本地重新启动尝试失败，那么成员会在另一成员的原始主机上作为*访客成员*以最小量的资源自动重新启动。以轻量级重新启动方式运行的成员不会处理新事务，因为其唯一用途就是执行成员崩溃恢复。

通常可使用独立并行成员重新启动恢复来恢复多个成员故障，因此，通常不需要组重新启动。此数据库仍然可供其他可用成员访问。只有在失败成员上处于未处理状态的数据在成员重新启动持续时间内不可用。

成员崩溃恢复

成员崩溃恢复负责在成员重新启动期间回滚未完成的事务并作为成员重新启动的一部分完成已落实的事务。 这将确保此成员修改的数据库数据进入一致状态。如果成员崩溃恢复结束时存在不确定事务，那么此成员可解决这些事务。

成员崩溃恢复将在可用集群高速缓存设施仍然可用并且成员上的数据库已激活时执行，确定的是，此成员的日志流因为异常终止而导致在磁盘上处于不一致状态。在大多数情况下，成员崩溃恢复将通过成员重新启动和自动恢复代理程序自动调用。自动恢复代理程序在实例启动时启动，它会在检测到成员上的数据库处于不一致状态时采取措施。

如果数据库的成员已在其原始主机上重新启动，那么此成员崩溃恢复完成后将能够接受来自其他应用程序的入局连接请求。

组重新启动和崩溃恢复

*组重新启动*是通过对所有成员和集群高速缓存设施重新启动数据库服务器进程并执行组崩溃恢复以使数据库恢复联机状态来重新启动整个 DB2 pureScale实例的过程。

DB2 pureScale 实例中没有可用主集群高速缓存设施时，会发生组重新启动。DB2 集群服务会自动检测并处理此事件。主集群高速缓存设施和成员变为可用时，组重新启动将自动启动。发生组重新启动时，所有成员都无法访问数据库。

有一些情况可能导致需要组重新启动:

- 此实例仅与一个集群高速缓存设施一起运行，并且集群高速缓存设施失败时。
- 主集群高速缓存设施在辅助集群高速缓存设施变为 PEER 状态前失败。
- 主和辅助集群高速缓存设施都失败时。

组崩溃恢复

组崩溃恢复负责通过重做未写至磁盘的所有工作并回滚故障时未落实的所有事务以使数据库保持一致。组崩溃恢复与 DB2 pureScale 环境外部的 DB2 崩溃恢复类似，但它使用来自数据库上活动的所有成员的合并日志流。因为组崩溃恢复作为组重新启动的一部分自动发生，所以需要组崩溃恢复并且存在生效集群管理器时，用户通常不必采取任何措施。

轻量级重新启动

失败成员无法在其*原始主机*上重新启动，DB2 集群服务在 DB2 pureScale实例中的其他某个可用主机上重新启动该成员。此过程称为*轻量级重新启动*，允许执行成员崩溃恢复而不会严重影响实例的余下部分。

已在另一主机上以轻量级重新启动方式重新启动的成员称为*访客成员*，而在其原始主机上运行的成员称为*常驻成员*。访客成员使用的资源比常驻成员少，并且它不能接受实例连接或来自外部应用程序的数据库连接 (SQL1032N)。

以轻量级重新启动方式启动成员的唯一用途是执行成员崩溃恢复。访客成员是使用预先分配的缩减内存模型重新启动的，以使对其主机用于轻量级重新启动的常驻成员的影响降至最低。以轻量级重新启动方式运行的访客成员在所有必需数据库上完成成员崩溃恢复，然后等待故障恢复至其原始主机。直到故障恢复至其原始主机，它才能处理新事务。请注意，查询处于轻量级重新启动方式的成员的状态时，它看起来已启动并且正在运行（但处于 WAITING_FOR_FAILBACK 状态）。失败原始主机恢复联机时，处于轻量级重新启动方式的成员将由 DB2 集群服务自动故障恢复至其原始主机，并且再次成为可处理新事务的完全活动的成员。

轻量级重新启动是按如下方式工作的自动过程: 对于成员的原始主机仍处于活动状态的软件故障，DB2 集群服务尝试在其原始主机上重新启动失败的成员。如果失败成员的第一次重新启动尝试在其原始主机上未成功，那么该成员会在另一原始主机上作为访客成员以轻度方式重新启动。如果发生下列其中一种情况，那么会立即对成员启动轻量级重新启动:

- 网络故障（原始主机断开与 DB2 pureScale 实例余下部分的连接）
- 原始主机意外变为不活动状态（因为断电或硬件、LPAR、VM 或操作系统重置）
- 成员异常终止（在其原始主机因为维护而停止时）

轻量级重新启动的运行速度很快，因为每个主机上都有一组 DB2 空闲进程，每个主机会为访客成员的重新启动预分配资源。DB2 将激活这些进程而不是创建新进程来以轻度方式执行成员重新启动。因为轻量级重新启动期间未创建新进程，并且访客成员不需要与常驻成员竞争资源，所以成员恢复处理速度很快。

禁用自动成员故障恢复

在某些情况下，您可能想要延迟自动故障恢复，这是通过 **db2cluster** 命令实现的。

关于此任务

缺省情况下，成员的原始主机变为可用并且所有警报被清除后，DB2 集群服务会立即对任何以轻量级重新启动方式运行的成员执行故障恢复。如果要调查主机故障的原因，禁用自动成员故障恢复直到更适当的时间会很有用。

过程

要禁用自动故障恢复，请执行以下操作:

1. 发出 **db2cluster** 命令:

   ```
   db2cluster -cm -set -option autofailback -value off
   ```

2. 重新启动 DB2 pureScale 实例。

结果

任何处于轻量级重新启动方式的成员以及任何未来轻量级重新启动成员将保留在其访客主机上，即使所有成员警报已被清除并且成员的常驻主机处于活动状态也是如此。

示例

遇到导致成员处于轻量级重新启动方式的重复主机故障后，DBA 决定最好让所有失败主机脱离其常驻成员直到能够确定该故障的原因。DBA 发出以下命令:

```
db2cluster -cm -set -option autofailback -value off
```

返回了以下消息:

```
db2cluster
命令成功。AUTOFAILBACK 选项已设置为"OFF"。下次重新启动 DB2 数据库管理器时将禁用自动故障恢复。
```

注: 例如，如果因为某个原因已禁用自动故障恢复，那么命令应成功完成，但消息将不提及重新启动该实例。

DBA 重新启动该实例。调查主机故障原因后，DBA 发出以下命令以启用自动成员故障恢复:

```
db2cluster -cm -set -option autofailback on
```

返回了以下消息:

```
db2cluster
命令成功。AUTOFAILBACK 选项已设置为"ON"。下次重新启动 DB2 数据库管理器时将启用自动故障恢复。
```

重新启动该实例后，DBA 使用以下命令来确保已启用自动成员故障恢复:

```
db2cluster -cm -list -option -autofailback
```

返回了以下消息:

```
AUTOFAILBACK 选项已设置为"ON"。
```

轻量级重新启动的内存注意事项

启动 DB2 时系统会保留有限内存量以用于恢复，从而使成员以轻量级重新启动方式顺利恢复。已预定义此保留轻量级重新启动内存，这样可以改进恢复性能，因为内存已保留并且立即可供在恢复期间使用。

可为用于在给定主机上进行轻量级重新启动恢复而保留的内存量由 *rstrt_light_mem* 数据库管理器配置参数限制。*rstrt_light_mem* 的缺省值是 AUTOMATIC，这意味着 DB2 自

动计算要预先分配并为轻量级重新启动恢复用途保留的内存量的固定上限，并在 DB2 启动时设置此值。DB2 根据 *instance_memory* 和 *numdb* 配置参数的设置及主机上成员的数目来计算该值。自动计算的值范围在实例内存限制的 1% 到 10%，并且包括在实例内存的总量中。但是，因为保留的轻量级重新启动内存量会影响常驻成员的性能，所以用户可调整轻量级重新启动内存配置以适合他们的特定工作负载。

显示保留的轻量级重新启动内存

要显示有关在 DB2 主机上分配的总内存量的信息，请将 **db2pd** 命令与 **-totalmem** 选项配合使用。此信息包括在正访问的当前 DB2 主机上预先分配的保存轻量级重新启动内存量。要检索集群中所有主机的信息，请以并行方式在不同主机上运行 db2pd。在以下示例中，db2pd 在包含成员 20 的主机 B 上运行。

```
db2pd -totalmem

                         Controller     Memory        Current        HWM           Cached
                         Automatic      Limit         Usage          Usage         Memory
Member 20                Yes           25750080 KB   9031201 KB     9391744 KB    480064 KB
 Restart Light Memory   Yes         2575008 KB     64182 KB       69265 KB       5250 KB

Total current usage: 9095383 KB
Total cached memory:  485314 KB
```

恢复隐藏缓冲池

对于成员崩溃恢复，系统会对缓冲池使用缩减内存模型。因为缓冲池通常会消耗数据库共享内存集中的最多内存，所以大缓冲池的分配非常耗时。缩减内存模型会改进恢复性能，因为分配的是小型恢复隐藏缓冲池而不是成本极高的大型用户定义缓冲池。与现有隐藏缓冲池一样，现在有 4 个恢复隐藏缓冲池，大小分别为 4K、8K、16K 和 32K。但是，隐藏缓冲池的大小始终为 16 页，恢复隐藏缓冲池的最小大小为 250 页并且可以更大，这取决于轻量级重新启动内存集大小和缓冲池大小计算。

在以下示例中，已为数据库 TESTDB 创建两个用户缓冲池（大小为 100 页的 BP1 和大小为 200 页的 BP2）。成员 0 处于轻量级重新启动方式，成员 1 未处于轻量级重新启动方式。此示例包括以下 db2pd 命令的输出的一部分。成员 1 显示用户创建的缓冲池以及隐藏缓冲池，成员 0 仅显示 4 个恢复隐藏缓冲池。

```
db2pd -allmembers -db testdb -bufferpools

Database Member 1--Database TESTDB--Active--Up 0 days 00:00:14--Date 08/12/2010 18:55:19

Bufferpools:
First Active Pool ID      1
Max Bufferpool ID         3
Max Bufferpool ID on Disk 3
Num Bufferpools           7

Address            Id   Name               PageSz     PA-NumPgs  BA-NumPgs  BlkSize
0x00002AAADE443140 1    IBMDEFAULTBP       4096       1000       0          0
0x00002AAADE45B080 2    BP1                4096       100        0          0
0x00002AAADE45F060 3    BP2                4096       200        0          0
0x00002AAADDB83CC0 4096 IBMSYSTEMBP4K      4096       16         0          0
0x00002AAADDB13CC0 4097 IBMSYSTEMBP8K      8192       16         0          0
0x00002AAADDB03CC0 4098 IBMSYSTEMBP16K     16384      16         0          0
0x00002AAADE453140 4099 IBMSYSTEMBP32K     32768      16         0          0

...NumTbsp    PgsToRemov CurrentSz  PostAlter  SuspndTSCt Automatic
   3          0          1000       1000       0          False
   0          0          100        100        0          False
   0          0          200        200        0          False
   0          0          16         16         0          False
   0          0          16         16         0          False
   0          0          16         16         0          False
   0          0          16         16         0          False
```

```
Database Member 0 -- Database TESTDB -- Active -- Up 0 days 00:00:13

Bufferpools:
First Active Pool ID      4096
Max Bufferpool ID         0
Max Bufferpool ID on Disk 3
Num Bufferpools           4

Address            Id   Name               PageSz     PA-NumPgs  BA-NumPgs  BlkSize
0x00002AAAD9F946E0 4096 IBMSYSTEMBP4K      4096       9954       0          0
0x00002AAADA743140 4097 IBMSYSTEMBP8K      8192       250        0          0
0x00002AAADA733140 4098 IBMSYSTEMBP16K     16384      250        0          0
0x00002AAADA74B080 4099 IBMSYSTEMBP32K     32768      250        0          0

...NumTbsp    PgsToRemov CurrentSz  PostAlter  SuspndTSCt Automatic
   3          0          9954       9954       0          False
   0          0          250        250        0          False
   0          0          250        250        0          False
   0          0          250        250        0          False
```

轻量级重新启动期间的内存消耗

理想状态是，重新启动允许失败的成员在该成员原始主机以外的主机上快速恢复而不影响该主机的常驻成员。为实现此目的，首先使用为轻量级重新启动保留的恢复内存来执行数据库恢复操作。但是，如果数据库恢复需要的内存资源超过为轻量级重新启动内存分配的资源，那么轻量级重新启动会提出额外的内存请求以获取可用实例内存。这些紧急内存请求尝试缩减常驻成员使用的当前内存。如果用于完成轻量级重新启动的内存资源仍然不足，那么 DB2 会向操作系统请求额外内存。如果发生此情况，那么所有其他非紧急内存请求会失败，直到已为恢复操作释放足够内存。在常驻成员上运行的应用程序发生内存不足故障，但常驻成员应该仍保持启动状态。完成恢复并且数据库处于一致状态后，访客成员使用的超过原始保留恢复内存的所有额外内存会被释放。

这些针对轻量级重新启动的内存资源的额外请求是临时的，但它们会对常驻成员的工作负载产生负面影响。如果您发现保留恢复内存不足，请考虑增加 *rstrt_light_mem* 数据库管理器配置参数的大小。此参数可配置但不支持联机配置，所以任何更改都需要全局 **db2stop** 和 **db2start**，或者如果您想要逐个成员更新 *rstrt_light_mem*（即，您不希望同时停止所有成员），那么必须停止然后启动每个成员以及每个成员的主机上的实例，如下所示:

```
db2 update dbm cfg using RSTRT_LIGHT_MEM 5

db2stop member 10
db2stop instance on hostA.torolab.ibm.com
db2start instance on hostA.torolab.ibm.com
db2start member 10

db2stop member 20
db2stop instance on hostB.torolab.ibm.com
db2start instance on hostB.torolab.ibm.com
db2start member 20
```

显示轻量级重新启动内存消耗情况

有两种方法可用来显示有关主机上由常驻成员和访客成员使用的总内存量的信息:

1. 在每个主机上将 **db2pd** 命令与 **-totalmem** 选项配合使用。执行以下示例:
   - 主机 A 上的成员 0
   - 主机 B 上的成员 1
   - 主机 C 上的成员 2

成员 0 以轻量级重新启动方式故障转移至主机 B。此用户在包含常驻成员 1 的主机 B 上运行 **db2pd** 命令，而访客成员 0 以轻量级重新启动方式运行。然后用户在主机 C 上运行 **db2pd** 命令以显示成员 2 的内存。成员 0 在屏幕输出中标记为“guest”，内存使用情况以千字节为单位显示。db2pd 不需要数据库连接。

主机 B:

```
$ db2pd -totalmem
Total Memory Statistics in KB

                    Controller     Memory        Current        HWM           Cached
                    Automatic      Limit         Usage          Usage         Memory
=============================================================================+=====
Member 0 (guest)         Yes       20677572      242496      244032       15168
Member 1                 Yes       20677572      186624      697088       46080
Restart Light Memory     Yes         839720      459392      459392       19200

Total current usage:  888512
Total cached memory:  80448
```

主机 C:

```
$ db2pd -totalmem
Total Memory Statistics in KB

                    Controller     Memory        Current        HWM           Cached
                    Automatic      Limit         Usage          Usage         Memory
=============================================================================+=====
Member 2                 Yes       20153284     4728832     4728832     1055104
Restart Light Memory     Yes         839720      689088      689088       28800

Total current usage:  5417920
Total cached memory:  1083904
```

2. 使用 SQL 界面来显示所有成员（包括处于轻量级重新启动方式的成员）的内存使用情况。它只需要从一个主机运行。但是，此方法需要数据库连接，所以它不能从处于轻量级重新启动方式的成员运行，因为轻量级重新启动成员不接受连接。此屏幕不会将成员标记为“guest”，并且内存使用情况会以字节为单位显示。

```
$ db2 'SELECT * FROM TABLE (SYSPROC.ADMIN_GET_MEM_USAGE()) AS T'

DBPARTITIONNUM MAX_PARTITION_MEM    CURRENT_PARTITION_MEM PEAK_PARTITION_MEM
-------------- -------------------- --------------------- --------------------
             1          21173833728            4605345792           4605345792
             0          21173833728             248840192            249888768
             2          20636962816            4651352064           4651352064

  3 record(s) selected.
```

在轻量级重新启动中监视成员

通过在任何活动成员上运行 **LIST UTILITIES** 命令，可监视以轻量级重新启动方式运行的成员的恢复进度。**LIST UTILITIES** 命令可返回所有成员（包括处于轻量级重新启动方式的成员）的全局恢复状态。

```
LIST UTILITIES SHOW DETAIL

ID                   = 1
类型                           = MEMBER CRASH RECOVERY
Database Name                  = SAMPLE
分区号                           = 0
描述                         = 成员崩溃恢复（轻度方式）
开始时间                         = 11/22/2007 15:20:05.646020
State                          = Executing
调用类型                       = 自动
```

```
    进度监视:
    估计完成百分比 = 0
    阶段号 [当前]        = 1
       描述                = 前进
       总工作量                = 4193976 字节
       Completed Work          = 0 bytes
       开始时间                = 11/22/2007 15:20:05.646121
    阶段号                     = 2
       描述                = 后退
       总工作量                = 4193976 字节
       Completed Work          = 0 bytes
       Start Time              = Not Started
```

可使用若干方法来获取 DB2 pureScale 环境（包括 DB2_GET_INSTANCE_INFO 表函数、DB2_MEMBER 管理视图、**LIST INSTANCE** 命令和 **db2instance** 命令）的时间点视图。使用下列任一方法允许您确定任何成员是否处于轻量级重新启动方式、它们正在哪个主机上恢复以及它们当前所处的轻量级重新启动恢复状态。

在此示例中，DB2_MEMBER 管理视图会显示成员 2 在主机 so3 上以轻度方式重新启动。

```
SELECT ID,
       varchar(HOME_HOST,10) AS HOME_HOST,
       varchar(CURRENT_HOST,10) AS CURRENT_HOST,
       varchar(STATE,21) AS STATE,
       ALERT
FROM SYSIBMADM.DB2_MEMBER

ID     HOME_HOST CURRENT_HOST STATE      ALERT
------ --------- ------------ ---------- -----
     1 so1       so1          STARTED    NO
     2 so2       so3          RESTARTING NO
     3 so3       so3          STARTED    NO

     3 record(s) selected.
```

场景: 轻量级重新启动

以下场景描述成员以轻度方式重新启动期间执行的步骤。它会讨论一种最常见的情况，其中一个主其中一个主机故障导致主机的常驻成员作为访客成员在仍处于活动状态的另一主机上自动重新启动。此场景还会讨论访客成员如何故障恢复至其原始主机。

初始设置

DB2 pureScale 实例中有 6 个主机（HostA、HostB、HostC、HostD、HostE 和 HostF）：

- 成员 10 正在 HostA（其原始主机）上运行
- 成员 20 正在 HostB（其原始主机）上运行
- 成员 30 正在 HostC（其原始主机）上运行
- 成员 40 正在 HostD（其原始主机）上运行
- 集群高速缓存设施 128 (CF 128) 正在 HostE 上运行
- 集群高速缓存设施 129 (CF 129) 正在 HostF 上运行

在每个主机上，该实例都有一组 DB2 空闲进程，这些进程具有为轻量级重新启动恢复用途保留的预分配内存。DB2 集群服务会监视集群中的所有资源。

可使用 **LIST INSTANCE** 命令来显示主机、成员和 CF 的状态信息。此时，**LIST INSTANCE** 命令返回:

```
LIST INSTANCE

MEMBER_ID TYPE    STATE   HOME_HOST CURRENT_HOST ALERT
--------- ------- ------- --------- ------------ -----
       10 MEMBER  STARTED hostA     hostA        NO
       20 MEMBER  STARTED hostB     hostB        NO
       30 MEMBER  STARTED hostC     hostC        NO
       40 MEMBER  STARTED hostD     hostD        NO
      128 CF      PRIMARY hostE     -            NO
      129 CF      PEER    hostF     -            NO


HOSTNAME STATE    INSTANCE_STOPPED ALERT
-------- ------ ---------------- -----
hostA    ACTIVE NO               NO
hostB    ACTIVE NO               NO
hostC    ACTIVE NO               NO
hostD    ACTIVE NO               NO
hostE    ACTIVE NO               NO
hostF    ACTIVE NO               NO
```

主机故障

HostA 服务器上发生了电源故障。DB2 集群服务无法在 HostA 上重新启动成员 10，所以它以轻度方式在下一个可用主机（即 HostB）上重新启动成员。

此时，**LIST INSTANCE** 命令显示成员 10 的状态现在为 RESTARTING，其当前主机现在是 HostB，HostA 的状态为 INACTIVE（请注意，未设置 INSTANCE_STOPPED 字段，因为未在 HostA 上手动停止此实例）并且存在警报:

```
LIST INSTANCE

MEMBER_ID TYPE    STATE      HOME_HOST CURRENT_HOST ALERT
--------- ------- ---------- --------- ------------ -----
       10 MEMBER  RESTARTING hostA     hostB        NO
       20 MEMBER  STARTED    hostB     hostB        NO
       30 MEMBER  STARTED    hostC     hostC        NO
       40 MEMBER  STARTED    hostD     hostD        NO
      128 CF      PRIMARY    hostE     -            NO
      129 CF      PEER       hostF     -            NO


HOSTNAME STATE    INSTANCE_STOPPED ALERT
-------- -------- ---------------- -----
hostA    INACTIVE NO               YES
hostB    ACTIVE   NO               NO
hostC    ACTIVE   NO               NO
hostD    ACTIVE   NO               NO
hostE    ACTIVE NO               NO
hostF    ACTIVE   NO               NO
```

等待故障恢复

启动进程模板后，会在需要成员崩溃恢复的每个数据库上执行成员崩溃恢复。要检查成员崩溃恢复的进度，请将 **LIST UTILITIES** 命令与 **SHOW DETAIL** 选项配合使用，如监视处于轻量级重新启动方式的成员中所述。成员崩溃恢复完成后，成员 10 会等待故障恢复至 HostA 并且无法处理任何新事务直到故障恢复完成。成员 10 等待故障恢复时可能有一些不确定事务需要解决。

此时，**LIST INSTANCE** 命令显示成员 10 的状态现在为 WAITING_FOR_FAILBACK:

```
LIST INSTANCE

MEMBER_ID TYPE    STATE                HOME_HOST CURRENT_HOST ALERT
--------- ------- -------------------- --------- ------------ -----
       10 MEMBER  WAITING_FOR_FAILBACK hostA     hostB        NO
       20 MEMBER  STARTED              hostB     hostB        NO
       30 MEMBER  STARTED              hostC     hostC        NO
       40 MEMBER  STARTED              hostD     hostD        NO
      128 CF      PRIMARY              hostE     -            NO
      129 CF      PEER                 hostF     -            NO


HOSTNAME STATE    INSTANCE_STOPPED ALERT
-------- -------- ---------------- -----
hostA    INACTIVE NO               YES
hostB    ACTIVE   NO               NO
hostC    ACTIVE   NO               NO
hostD    ACTIVE   NO               NO
hostE    ACTIVE NO               NO
hostF    ACTIVE   NO               NO
```

主机问题已解决

HostA 的电源已恢复，所以 HostA 在 DB2 pureScale 实例中再次变为活动状态。

此时，**LIST INSTANCE** 命令显示 HostA 现在处于活动状态并且警报已被清除:

```
LIST INSTANCE

MEMBER_ID TYPE    STATE                HOME_HOST CURRENT_HOST ALERT
--------- ------- -------------------- --------- ------------ -----
       10 MEMBER  WAITING_FOR_FAILBACK hostA     hostB        NO
       20 MEMBER  STARTED              hostB     hostB        NO
       30 MEMBER  STARTED              hostC     hostC        NO
       40 MEMBER  STARTED              hostD     hostD        NO
      128 CF      PRIMARY              hostE     -            NO
      129 CF      PEER                 hostF     -            NO


HOSTNAME STATE    INSTANCE_STOPPED ALERT
-------- ------ ---------------- -----
hostA    ACTIVE NO               NO
hostB    ACTIVE NO               NO
hostC    ACTIVE NO               NO
hostD    ACTIVE NO               NO
hostE    ACTIVE NO               NO
hostF    ACTIVE NO               NO
```

故障恢复至原始主机

DB2 集群服务检测 HostA 是否处于活动状态并将成员 10 自动故障恢复至该主机。

此时，**LIST INSTANCE** 命令显示成员 10 的状态现在为 RESTARTING 并且其当前主机再次变为 HostA:

```
LIST INSTANCE SHOW DETAIL

MEMBER_ID TYPE    STATE      HOME_HOST CURRENT_HOST ALERT
--------- ------- ---------- --------- ------------ -----
       10 MEMBER  RESTARTING hostA     hostA        NO
       20 MEMBER  STARTED    hostB     hostB        NO
       30 MEMBER  STARTED    hostC     hostC        NO
       40 MEMBER  STARTED    hostD     hostD        NO
      128 CF      PRIMARY    hostE     -            NO
      129 CF      PEER       hostF     -            NO
```

```
HOSTNAME STATE    INSTANCE_STOPPED ALERT
-------- ------ ---------------- -----
hostA    ACTIVE NO               NO
hostB    ACTIVE NO               NO
hostC    ACTIVE NO               NO
hostD    ACTIVE NO               NO
hostE    ACTIVE NO               NO
```

在原始主机上重新启动

成员 10 在 HostA 上成功完成成员重新启动时，它的状态会更改为 STARTED，并且它现在可处理新事务并接受用户连接。此时，**LIST INSTANCE** 命令返回的有关 DB2 pureScale 实例的信息与初始步骤中的信息相同。

故障状态时手动干预

在大部分故障状态中，用户不需要采取措施，因为 DB2 集群服务会自动启动所需的重新启动或崩溃恢复类型。但是，有时您必须手动执行重新启动或崩溃恢复，或者执行完成组崩溃恢复所需的步骤。

启动组崩溃恢复

组崩溃恢复几乎一直自动执行，不需要任何用户操作。只有在集群管理器尝试执行组崩溃恢复重复失败或不存在（可能因为集群管理被禁用）或者 **autorestart** 配置参数设为 OFF 时，才应启动组崩溃恢复。

关于此任务

在给定时间，只有一个成员能在数据库上执行组崩溃恢复。如果两个成员同时对同一数据库启动组崩溃恢复，那么第二个成员的命令将等待组崩溃恢复在第一个成员上完成。

如果崩溃恢复操作完成时组崩溃恢复成员上有不确定事务，那么执行组崩溃恢复的成员仍处于活动状态并且能够接受新工作；但是，与不确定事务相关联的所有数据不可访问，因为不确定事务访问的行必须处于受保护状态（即，它们被锁定）直到该不确定事务最终解决。组崩溃恢复成员上存在不确定事务的事实意味着，如果成员在不确定事务解决前失败，那么后续成员崩溃恢复只有在数据库下一次在该成员上激活时才能进行。

如果成员 A 上发生了组崩溃恢复，并且发现在组崩溃恢复期间 成员 B 在一个或多个不确定事务中与其相关联，那么 成员 B 在组崩溃恢复后会保持不一致状态。这暗示 成员 B 只能在组崩溃恢复完成后启动，成员崩溃恢复必须在成员 B 上进行。成员崩溃恢复完成后，成员 B 可用于接受新连接。应该注意的是，此后续成员崩溃恢复会非常迅速地完成，因为不需要任何重做或撤销工作（只需要崩溃恢复操作将不确定事务装入到成员的事务表中），并且保护不确定事务影响的数据时所需的所有锁定都已为集群高速缓存设施中的成员保留。此外，在大多数情况下，用户不会见到此后续 RESTART 操作，因为启用 AUTORESTART 后，第一次与给定成员上的数据库连接时会自动执行成员崩溃恢复。

过程

要手动启动组崩溃恢复，请从某个成员发出 **RESTART DATABASE** 命令。

结果

组崩溃恢复完成时，运行此命令的成员将能够接受新连接并且所有数据将可用，仍被未解决的不确定事务锁定的数据除外。**RESTART DATABASE** 命令完成后，如果用户具有对数据库的 CONNECT 特权，那么与该数据库的连接将保留。此数据库不会在运行了组崩溃恢复的成员以外的任何成员上激活。

启动成员崩溃恢复

成员崩溃恢复几乎一直自动执行，不需要任何用户操作。只有在集群管理器尝试执行成员崩溃恢复重复失败或不存在（可能因为集群管理被禁用）或者 **autorestart** 配置参数设置为 OFF 时，才应手动启动崩溃恢复。

关于此任务

如果 DB2 pureScale 实例有多个成员需要成员崩溃恢复，那么集群管理器会并行发出多个成员崩溃恢复。这有助于避免因为一个失败的成员依赖于另一失败的成员持有的资源导致的挂起情况。

过程

要手动启动成员崩溃恢复，请发出 **RESTART DATABASE** 命令。
发出 **RESTART DATABASE** 命令时，DB2 for Linux, UNIX, and Windows 会确定是需要组崩溃恢复还是成员崩溃恢复。

结果

一旦成员崩溃恢复完成，那么给定成员上的数据库将激活并可用于从其他应用程序接收连接。

在表空间已损坏的情况下恢复

如果表空间容器已损坏并且需要在数据库上进行组崩溃恢复，那么组崩溃恢复操作将失败。在带有自动恢复的 DB2 pureScale 环境中，可按如下所示解决此类场景的问题。

开始之前

以下场景说明如何解决 DB2 pureScale 环境中的已损坏表空间的问题以便自动组崩溃恢复可继续。

关于此任务

假定自动恢复处于活动状态并且因为表空间已损坏而导致恢复数据库失败。重复失败后，DB2 成员会移至另一主机并尝试轻量级重新启动。经历更多自动恢复数据库失败后，集群管理器生成警报。失败的成员未启动并且处于不活动状态。

过程

1. 通过将 **autorestart** 配置参数设为 OFF 来禁用自动恢复。 **autorestart** 配置参数并非动态参数，其新值将在下一次成员启动时生效。 以下命令会将数据库 WSDB 的 **autorestart** 配置参数更改为 OFF:

   ```
   UPDATE DATABASE CONFIGURATION FOR WSDB USING AUTORESTART OFF
   ```

2. 使用 **db2cluster** 命令来清除所有成员的所有未解决警报。 而且，DB2 集群服务将自动发出 **db2start** 命令以在成员的原始主机上将它们重新启动。重新启动的成员将使已更改 **autorestart** 值生效并且不启动自动恢复。

   ```
   db2cluster -clear -alert
   ```

3. 发出 **RESTART DATABASE** 命令以解决表空间已损坏的问题。 已损坏表空间标记为脱机，并且将置于废弃暂挂状态。

   ```
   RESTART DATABASE WSDB DROP PENDING TABLESPACES (tbsp1, tbsp2)
   ```

4. 通过将 **autorestart** 配置参数设为 ON 来重新启用自动恢复。 例如，以下命令会将数据库 WSDB 的 **autorestart** 配置参数更改为 ON:

   ```
   UPDATE DATABASE CONFIGURATION FOR WSDB USING AUTORESTART ON
   ```

   autorestart 参数并非动态参数，其新值将在下一次成员启动时生效。

第 2 部分 数据恢复

恢复指的是在发生问题（例如介质或存储器故障、断电或者应用程序故障）后重建数据库或表空间。如果已经备份了数据库或各个表空间，那么可以在它们由于某种原因被损坏或毁坏时对其进行重建。

有四种类型的恢复操作:

- 崩溃恢复保护数据库在事务（也称为工作单元）意外中断后不会处于不一致或不可用的状态。
- 灾难恢复，包括在发生火灾、地震、恶意破坏或其他灾难性事件时复原数据库的过程。
- 版本恢复指的是使用备份操作期间创建的映像来复原数据库的先前版本。
- 前滚恢复可以用来重新应用创建备份后落实的事务所作的更改。

断电后，DB2 数据库管理器会自动启动崩溃恢复以尝试恢复数据库。可以使用版本恢复或前滚恢复功能来恢复损坏的数据库。

© Copyright IBM Corp. 2001, 2013

第 7 章 开发备份和恢复策略

数据库可能会因为硬件和/或软件故障而不可用。您可能会不时遇到存储问题、断电或应用程序故障以及各种需要采取不同恢复措施的故障情况。

如果已准备了一个进行过彻底演习的恢复策略，它可保护您的数据免被丢失。

在开发恢复策略时，您应该回答的一些问题包括:

- 数据库是可恢复的吗？
- 恢复数据库可能花费多长时间？
- 在备份操作之间将花费多长时间？
- 可以为备份副本和归档日志分配多少存储空间？
- 表空间级的备份是否足够？或是否需要进行完整的数据库备份？
- 是否应该通过手动或高可用性灾难恢复 (HADR) 来配置备用系统？

数据库恢复策略应确保在数据库恢复操作需要时，所有信息都可用。它应包括一个进行数据库备份的固定时间表，在分区数据库环境中则包括缩放系统时（添加或删除数据库分区服务器或节点时）的备份。完整的策略还应包括恢复命令脚本、应用程序、用户定义的函数 (UDF)、操作系统库中的存储过程代码以及装入副本的过程。

在以下几节中还讨论了不同的恢复方法，您将能发现最适用于您的业务环境的恢复方法。

数据库*备份*的概念与其他任何数据备份的概念一样：即，复制一份数据，然后将它存储在另一介质上，以防原始介质发生故障或毁坏。最简单的备份情况需要关闭数据库（以确保不发生更多的事务），然后简单地对其进行备份。以后，如果数据库被损坏或毁坏，就可以重新创建。

数据库的重新创建称为*恢复*。*版本恢复*指的是使用备份操作期间创建的映像来复原数据库的先前版本。*前滚恢复*是指复原了数据库或表空间备份映像后，重新应用记录在数据库日志文件中的事务。

*崩溃恢复*是指在完成并落实所有更改（这些更改是一个或多个工作单元（事务）的一部分）之前如果发生故障，会自动恢复数据库。这是通过回滚未完成的事务，并完成在发生崩溃时仍在内存中的已落实事务来实现的。

恢复日志文件和恢复历史记录文件是在创建数据库时自动创建的 (第 226 页的图 11)。在需要恢复丢失或损坏的数据时，这些日志文件是很重要的。

每个数据库都包括*恢复日志*，它们用来从应用程序或系统错误中恢复。与数据库备份一起，它们用来将数据库的一致性恢复到出错时的时间点。

*恢复历史记录文件*包含当数据库的所有或部分必须恢复到给定时间点时，可用来确定恢复选项的备份信息的摘要。恢复历史记录文件用来跟踪其他操作中与恢复相关的事件，如备份和复原操作。此文件位于数据库目录中。

© Copyright IBM Corp. 2001, 2013

表空间更改历史记录文件（它也位于数据库目录中）包含可用来确定哪些日志文件对特定表空间的恢复是必需的信息。

不能直接修改恢复历史记录文件或表空间更改历史记录文件；但是可以使用 **PRUNE HISTORY** 命令来删除文件中的条目。还可以使用 **rec_his_retentn** 数据库配置参数来指定这些历史记录文件将保留的天数。

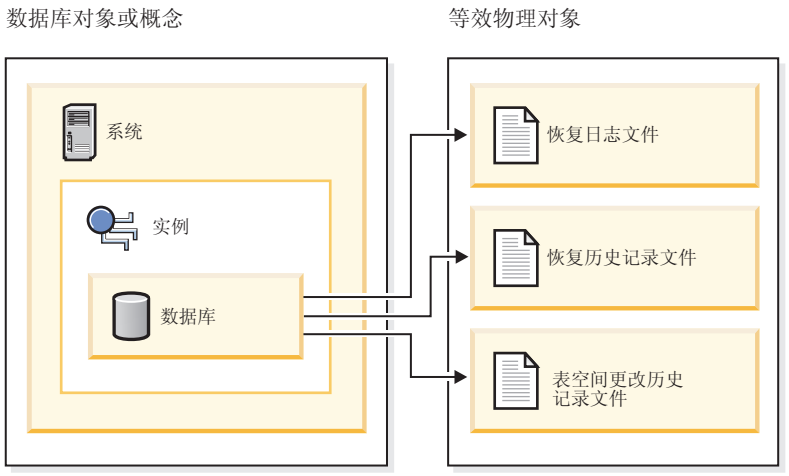


图 11. 数据库恢复文件

那些很容易重新创建的数据可存储在不可恢复数据库中。这些数据包括：用于只读应用程序的外部源中的数据以及不常进行更新的表；由于对它们进行的日志记录量较小，如果在复原操作之后还要进行复杂的日志文件文件管理工作和前滚操作，就不太合理了。如果 **logarchmeth1** 和 **logarchmeth2** 数据库配置参数设置为 OFF，那么数据库为*不可恢复*。这表明将只保存崩溃恢复所必需的日志。这些日志称为*活动日志*，它们包含当前事务数据。使用*脱机*备份的版本恢复是解决不可恢复数据库的恢复问题的主要手段。（脱机备份表示当备份操作正在进行时，其他应用程序无法使用该数据库。）这样的数据库只能进行脱机复原。它被复原为生成备份映像时的状态且不支持前滚恢复。

那些不容易重新创建的数据应存储在可恢复的数据库中。这些数据包括在装入后它的源已被破坏的数据、手动输入到表中的数据以及在装入数据库后由应用程序或用户修改的数据。*可恢复数据库*的 **logarchmeth1** 或 **logarchmeth2** 数据库配置参数设置为 OFF 以外的值。活动日志仍可用于崩溃恢复，但您还有*已归档日志*，它包含已落实的事务数据。这样的数据库只能进行脱机复原。它被复原为创建备份映像时的状态。但对于前滚恢复，可通过使用活动日志和已归档日志将数据库前滚（即，越过创建备份映像的时间）至特定的时间点，或者滚动至活动日志末尾。

可恢复数据库备份操作可以脱机执行，也可以*联机*执行（联机表示在备份操作期间其他应用程序可与该数据库连接）。仅当数据库可恢复时，才支持联机表空间复原和前滚操作。如果数据库是不可恢复的，必须脱机执行数据库复原和前滚操作。联机备份操作期间，前滚恢复确保捕获了*所有*表更改，且在复原该备份时重新应用这些更改。

如果有一个可恢复数据库，那么可备份、复原并将各个表空间前滚，而不必对整个数据库操作。当联机备份表空间时它仍然可用，同时发生的更新记录在日志中。当对表空间执行联机复原或前滚操作时，在该操作完成之前，该表空间本身不可用，但不阻止用户访问其他表空间中的表。

自动备份操作

因为确定是否及何时运行维护活动（例如，备份操作）可能很费时间，所以可以使用自动维护。通过自动维护指定维护目标（包括自动维护可以运行的时间）。然后，DB2 使用这些目标来确定是否需要执行维护活动，然后在下一可用维护窗口期间（用户定义的、运行自动维护活动的时间段）仅运行所需的维护活动。

注：配置自动维护时，仍可执行手动备份操作。如果需要自动备份操作，DB2 将只执行这些自动备份操作。

确定备份频率

因为备份数据库时需要耗用时间和系统资源，所以恢复计划应该允许定期地安排备份操作。恢复计划可能既包括完整数据库备份操作也包括增量备份操作。并且，进行备份的频率和类型也会影响数据库恢复时间。

即使已将日志归档，也应定期执行完整数据库备份，以允许前滚恢复。要恢复数据库，可以使用包含所有表空间备份映像的完整数据库备份映像，也可以使用所选表空间映像来重建数据库。表空间备份映像对于从被隔离的磁盘故障或应用程序错误进行恢复时很有用。在分区数据库环境中，只需要复原发生故障的数据库分区上的表空间。并不需要复原所有表空间或所有数据库分区。

虽然可以根据表空间映像重建数据库而使数据库恢复不再需要完整数据库备份，但是不定期地进行数据库的完全备份仍然是一个较好的习惯。

还应考虑不要覆盖备份映像和日志，应保存至少两个完整的数据库备份映像及其相关的日志，作为额外的预防措施。

在恢复和前滚活动数据库时，如果应用已归档的日志所需的时间是主要关注的问题，请考虑进行更频繁的数据库备份所需的成本。进行更频繁的备份可减少前滚时需要应用的已归档日志数。

联机和脱机备份注意事项

可以在数据库处于联机或脱机状态时启动备份操作。如果它是联机的，在运行备份操作的同时，其他应用程序或进程可以与该数据库连接并读取和修改数据。如果正在脱机运行备份操作，那么其他应用程序不能连接至数据库。

要缩短数据库处于不可用状态的时间，应考虑使用联机备份操作。仅当启用前滚恢复时，才支持联机备份操作。如果已启用前滚恢复，并且有一组完整的恢复日志，那么可以在需要时复原数据库。仅当您的日志涵盖了运行备份操作期间的时间段时，才能使用联机备份映像进行恢复。

脱机备份操作比联机备份操作更快，因为不存在数据文件的争用。

有选择性的表空间备份注意事项

可以使用备份实用程序来仅备份所选表空间。如果使用 DMS 表空间，那么可以将不同类型的数据存储在它们各自的表空间中，以减少备份操作所需的时间。可以将表数据保存在一个表空间中，将长整数字段和 LOB 数据保存在另一个表空间中，再将索引保

存在一个表空间中。如果将数据分开保存在不同表空间中并且磁盘发生故障，那么磁盘故障可能只影响其中一个表空间。复原或前滚其中一个表空间将比复原包含所有数据的单个表空间花费更少的时间。

还可以通过在不同时间对不同表空间进行备份来节省时间，只要对它们的更改不相同。因此，如果长字段或 LOB 数据不像其他数据那样作频繁地更改，那么可以不那么频繁地备份这些表空间。如果长字段和 LOB 数据不是恢复所必需的，那么可以考虑不备份包含该数据的表空间。如果可从单独的源复制 LOB 数据，那么当创建或改变一个表以包括 LOB 列时选择 NOT LOGGED 选项。

如果将长字段数据、LOB 数据和索引保存在单独的表空间中，但不将它们一起进行备份，请考虑以下情况：如果您备份未包含所有表数据的表空间，那么不能对该表空间执行时间点前滚恢复。包含一个表的任何类型数据的所有表空间都必须同时前滚至同一个时间点。

表重组注意事项

如果重组一个表，应该在该操作完成后备份受影响的表空间。如果不得不复原这些表空间，将不必通过数据重组来前滚。

表空间修改状态注意事项

还可以通过检查表空间的修改状态，以在了解更多信息的情况上作出是否备份该表空间的决策。**db2pd -tablespaces trackmodstate** 命令和 **tbsp_trackmode_state** 监视元素将显示该表空间在上次或下次备份时的状态。您可以使用此信息来确定是否修改了该表空间或者是否需要备份该表空间。

数据库恢复时间注意事项

恢复数据库所需的时间由两个部分组成:

- 完成备份复原所需的时间。
- 前滚操作期间应用日志所需的时间（如果对数据库启用了正向恢复）。

制订恢复计划时，请考虑这些恢复成本和它们对业务运营的影响。测试整体恢复计划有助于确定恢复数据库所需的时间是否适合给定的业务要求。在每次测试之后，可能要提高创建备份的频率。如果前滚恢复是策略的一部分，这个增大的备份频率会减少备份之间归档的日志数，因此也减少了复原操作之后前滚数据库所需的时间。

恢复的存储器注意事项

当确定要使用的恢复方法时，请考虑必需的存储空间。备份和归档日志文件压缩可帮助减少数据库环境中的存储开销。

版本恢复方法需要空间来容纳数据库的备份副本和复原的数据库。前滚恢复方法需要空间来容纳数据库或表空间的备份副本、复原的数据库和归档的数据库日志。

如果一个表包含长型字段或大对象 (LOB) 列，那么应考虑将此数据置于单独的表空间中。这会影响对存储空间的规划，并影响恢复计划。如果使用单独的表空间来存储长型字段和 LOB 数据，并知道备份长型字段和 LOB 数据所需的时间，那么可以决定使

用仅以较低的频率保存此表空间备份的恢复计划。在创建或改变一个表以包括 LOB 列时，也可选择不记录对那些列的更改。此操作将减少必需的日志空间和对应的归档日志文件空间。

要防止介质故障破坏数据库并因此导致无法复原它，应该在不同的设备上保存数据库备份、数据库日志和数据库本身。鉴于此原因，极力建议您使用 *newlogpath* 配置参数，这样一旦创建了该数据库，就将数据库日志置于单独的设备中。

数据库日志可能会占用大量的存储器。如果计划使用前滚恢复方法，那么必须确定管理和压缩归档日志的方法。选项如下所示:

- 使用 LOGARCHMETH1 或 LOGARCHMETH2 配置参数指定归档日志文件方法。
- 使用 LOGARCHCOMPR1 和 LOGARCHCOMPR2 配置参数启用归档日志文件压缩。
- 当某些日志不再出现在活动的日志集中后，以手动方式将这些日志复制到存储设备或非数据库日志路径目录的一个目录中。
- 使用用户出口程序将这些日志复制到您环境中的另一个存储设备中。

备份压缩

除了通过在活动数据库中进行行压缩可以获取的存储器节省以外，还可以使用备份压缩来减小数据库备份的大小。

鉴于行压缩逐表进行，对备份使用压缩时，会压缩备份映像中的*所有*数据，包括目录表、索引对象、LOB 对象、辅助数据库文件和数据库元数据。

可以将备份压缩用于使用行压缩的表。但是，请注意，备份压缩需要额外的 CPU 资源和额外时间。单独使用表压缩可能足以达到减少备份存储器的要求。如果要使用行压缩，那么仅当存储器优化的优先级高于执行备份所需的额外时间时，才应考虑使用备份压缩。

提示: 仅当下列条件适用时，才应考虑在不包含压缩数据的表空间上使用备份压缩:

- 数据和索引对象独立于 LOB 及长字段数据
- 分别对大部分数据表和索引使用行和索引压缩

要对备份使用压缩，请在 **BACKUP DATABASE** 命令上使用 COMPRESS 选项。

归档日志文件压缩

对于 DB2 V10.1，您可以压缩归档日志文件。该压缩功能、数据和索引压缩以及备份压缩功能，减少了数据库环境所需的磁盘空间量。

归档日志文件是前滚可恢复数据库的第三大主要空间使用者。归档日志文件包含大量数据，并且这些归档可能会快递增长。如果修改的数据已在压缩表中，那么可通过在日志记录中包括压缩记录映像来减少日志记录。即使在这些环境中，对归档日志文件进行压缩也可以进一步增加节省的存储空间。

要对归档日志文件使用压缩，可以使用 **UPDATE DB CFG** 命令来将 **logarchcompr1** 和 **logarchcompr2** 配置参数设置为 ON。

限制

- 在以下条件下归档日志文件压缩不生效。

- 对应的归档日志文件方法未设置为 DISK、TSM 或 VENDOR。当对应的归档日志文件方法设置为上述方法时，系统会将日志文件从活动日志路径或镜像日志路径中物理除去。
- 当启用了归档日志文件压缩，但对应的日志归档方法设置为 OFF、LOGRETAIN或 USEREXIT 时，归档日志文件压缩将无效。对 **logarchmeth1** 和 **logarchmeth2** 或 **logarchcompr1** 和 **logarchcompr2** 数据库配置参数的更新（此更新导致返回 SQL1663W 警告消息）。

  注: 如果激活了数据库，那么设置或更改归档日志文件压缩数据库配置参数时不会返回 SQL1663W。而是返回 SQL1363W ，这是一个具有更高优先级的消息。如果数据库未激活，那么会返回 SQL1663W 警告消息。

- 使用 **db2adutl** 手动归档和检索。
  - **db2adutl** 实用程序在 UPLOAD 或 EXTRACT 操作期间不执行压缩或解压缩。**db2adutl** 完全支持将压缩日志文件移入移出归档位置。
  - 如果已使用 **db2adutl** 将日志上载到 Tivoli Storage Manager，并且要对归档日志文件进行压缩，那么必须在将日志归档到磁盘位置（在 **db2adutl** 上载它们之前）时启用归档日志文件压缩。如果手动使用 **db2adutl** 检索压缩日志，那么在第一次访问时会将它们解压缩。
- 当将原始设备用于数据库日志记录时，不支持归档日志文件压缩。
  - 当 **logpath** 或 **newlogpath** 数据库配置参数指向原始设备时，不支持归档日志文件压缩。当 **logpath** 或 **newlogpath** 数据库配置参数指向原始设备时，导致启用归档日志文件压缩的任何数据库配置更新会失败，SQL1665N。
- 使用 **logarchcompr1** 和 **logarchcompr2** 数据库配置参数启用归档日志文件压缩时，不会影响到已存储在备份映像中的日志。

将相关数据保存在一起

您应当将相关数据分组在一起，以帮助进行数据恢复。

在设计数据库的过程中，您将了解各个表之间存在的关系。可以在下列级别上表示这些关系:

- 应用程序级别（当事务更新多个表时）
- 数据库级别（当各个表之间存在引用完整性时，或者当一个表上的触发器影响另一个表时）。

当制定恢复计划时，应该考虑这些关系。您要将相关的数据集一起备份。可以在表空间级别或数据库级别上建立这样的集合。通过将相关的数据集保存在一起，可以恢复至所有数据都一致的时间点。如果您希望能够对表空间执行时间点前滚恢复，这一点尤其重要。

不同操作系统和硬件平台之间的备份和复原操作

DB2 数据库系统支持在不同操作系统和硬件平台之间执行一些备份和复原操作。

可以将支持 DB2 备份和复原操作的平台分为以下三种系列:

- 大尾数法 Linux 和 UNIX
- 小尾数法 Linux 和 UNIX

- Windows

一种平台系列的数据库备份只能在相同平台系列的任何系统上复原。对于 Windows 操作系统，可复原在 DB2 V9.7 数据库系统上的 DB2 V9.5 中创建的数据库。对于 Linux 和 UNIX 操作系统，只要备份和复原平台的字节存储次序（大尾数法或小尾数法）相同，就可复原在下级版本上创建的备份。

下表显示了 DB2 支持的每个 Linux 和 UNIX 平台，并指示了这些平台是采用大尾数法还是小尾数法:

表 15. DB2 支持的 Linux 和 UNIX 操作系统的字节存储次序

平台	字节存储次序
AIX	大尾数法
HP on IA64	大尾数法
Solaris x64	小尾数法
Solaris SPARC	大尾数法
Linux on zSeries	大尾数法
Linux on pSeries®	大尾数法
Linux on IA-64	小尾数法
Linux on AMD64 和 Intel EM64T	小尾数法
32 位 Linux on x86	小尾数法

目标系统必须与源系统具有相同（或比源系统更新的）DB2 数据库产品。不能将在一个数据库产品版本中创建的备份复原到运行较低版本的数据库产品的系统。例如，可复原 DB2 V9.7 数据库系统上的 DB2 V9.5，但不能复原 DB2 UDB V9.5 数据库系统上的 DB2 V9.7 备份。

注: 可以将数据库从在 32 位级别创建的备份映像复原为 64 位级别，但反之则不然。DB2 备份实用程序和复原实用程序应该用来备份和复原数据库。建议不要将文件集从一台机器移至另一台机器，因为这可能破坏数据库的完整性。

在不允许某些备份和复原组合的情况下，可以使用其他方法在 DB2 数据库之间移动表:

- **db2move** 命令
- 使用 **EXPORT** 实用程序，接着使用 **IMPORT** 或 **LOAD** 实用程序

注: 如果备份中的值超出了将在其上复原数据库的环境所允许的范围，那么数据库配置参数将设置为其缺省值。64 位数据库复原到 32 位实例中时，内存可调参数可能会发生此情况。

DB2 pureScale环境中的日志流合并和日志文件管理

在 DB2 pureScale环境中，每个成员都会在共享磁盘上维护自己的一组事务日志文件（即*日志流*），每组事务日志文件位于单独的日志路径中。成员的日志文件包含该成员发生的所有数据更改的历史记录。

多个应用程序（其中每个应用程序都同时访问不同的成员）可能在运行期间生成相依赖事务。例如，如果两个事务都更改同一行，那么这两个事务之间会发生依赖关系。要有效解释日志记录，DB2 数据服务器必须检查所有日志流中的记录并对这些记录进行

排序，以使它们反映运行时发生的更新顺序。此排序称为*日志流合并*操作。DB2 pureScale 环境中的若干操作类型需要执行日志流合并；这些操作包括组崩溃恢复、数据库前滚操作以及表空间前滚操作（还有其他操作）。

DB2 pureScale环境中的日志记录配置参数

表 16 显示了与日志记录相关的哪些数据库配置参数是全局范围的，以及哪些参数是可以动态更新的。

表 16. 与日志记录相关的数据库配置参数

参数	全局?	可动态更新?
archretrydelay	是	是
blk_log_dsk_ful	否	是
failarchpath	是	是
logarchcompr1	是	是
logarchcompr2	是	是
logarchmeth1	是	是
logarchmeth2	是	是
logarchopt1	是	是
logarchopt2	是	是
logbufsz	否	是
logfilsiz	是	否
logprimary	是	否
logsecond	是	是
max_log	否	是
mirrorlogpath [1]	是	否
newlogpath [1]	是	否
num_log_span	否	是
numarchretry	是	是
overflowlogpath	是	是
softmax	是	否
vendoropt	是	是

[1] 连接至数据库或激活数据库的第一个成员将处理此日志路径参数的更改。DB2 数据库管理器会验证路径是否存在，以及它对该路径是否具有读和写访问权。它还会为日志文件创建特定于成员的子目录。如果其中任何一个操作失败，那么 DB2 数据库管理器会拒绝指定的路径，并使用旧路径让数据库联机。如果数据库管理器接受所指定的路径，那么会将新值传播给每个成员。如果某个成员在尝试切换至新路径时失败，那么后续尝试激活数据库或连接至它都将失败，并返回 SQL5099N。所有成员都必须使用相同的日志路径。

在 DB2 pureScale环境中检索日志流合并操作的日志

将在路径中创建子目录以存储已检索到的日志文件。此子目录具有以下格式：*log_path*/LOGSTREAM*xxxx*，其中 *log_path* 表示日志路径、溢出日志路径或镜像日志路径，并且 *xxxx* 是一个 4 位数字的日志流标识。（日志流标识不必等于相关联的成员标识。）在此子目录中，如果某个成员要求检索日志，那么 DB2 数据库管理器会为从每个成员检索的日志创建另一个子目录层次。例如，如果在具有 3 个成员的系统上将溢出日志路径

指定为 /home/dbuser/overflow/，并且成员 0 上的应用程序必须检索其他成员所拥有的日志，那么成员 0 的路径为 /home/dbuser/overflow/NODE0000/LOGSTREAM0000，而此路径中的子目录会包含已检索的其他成员所拥有的日志，如下列所示:

```
Member 0 retrieves its own logs here:
   /home/dbuser/overflow/NODE0000/LOGSTREAM0000/LOGSTREAM0000
Member 0 retrieves logs that belong to member 1 here:
   /home/dbuser/overflow/NODE0000/LOGSTREAM0000/LOGSTREAM0001
Member 0 retrieves logs that belong to member 2 here:
   /home/dbuser/overflow/NODE0000/LOGSTREAM0000/LOGSTREAM0002
```

注: 请勿将日志文件手动插入到这些检索子目录中。如果要手动检索日志文件，请改为使用溢出日志路径。

当读取其他成员所拥有的归档日志文件时，成员可能需要将日志文件检索到该成员自己的日志路径或溢出日志路径中。在此情况下，日志流合并操作会根据需要为每个日志流创建 **db2logmgr** 引擎可分派单元 (EDU)。

如前所述，有三个路径可用于存储其他成员所拥有的日志文件，如以下列表所示:

1. 如果设置 **overflowlogpath** 数据库配置参数，那么会使用溢出日志路径。

   提示: 可以使用 **ROLLFORWARD DATABASE** 和 **RECOVER DATABASE** 命令选项来指定备用溢出日志路径；这些选项的值会覆盖数据库配置，以执行单个恢复操作。
2. 主日志路径
3. 如果设置 **mirrorlogpath** 数据库配置参数，那么会使用镜像日志路径。

如果 DB2 数据库管理器无法在第一个路径中存储日志文件，那么它会尝试使用列表中的下一个路径。如果这些路径都不可用，那么调用日志流合并操作的实用程序会返回特定于该实用程序的错误。

DB2 pureScale环境中 **GET DATABASE CONFIGURATION** 命令的输出将标识后面跟有成员名称的每个日志路径。例如，如果将镜像日志路径设置为 /home/dbuser/mirrorpath/，那么对于成员 2，输出会显示 /home/dbuser/mirrorpath/NODE0000/LOGSTREAM0002。

如果必须手动检索其他成员所拥有的日志文件，请确保数据库管理器可以通过使用自动创建的相同目录结构，来访问日志文件。例如，要使成员 2 的日志在成员 1 的溢出日志路径中可用，请将日志放在 /home/dbuser/overflow/NODE0000/LOGSTREAM0001/LOGSTREAM0002 目录中。

不再需要已检索的日志文件时，会将其自动删除。会保留日志流合并操作期间创建的子目录，以供将来使用。

检测日志流合并操作期间丢失的日志

如果意外删除、移动或归档并丢失了恢复操作所需的日志文件，那么您可以将数据库前滚恢复至丢失日志文件之前的最后一个一致点。

如果在日志流合并操作期间，DB2 数据库管理器确定其中一个日志流中丢失日志文件，那么会返回错误。ROLLFORWARD 实用程序将返回 SQL1273N；db2ReadLog API 将返回 SQL2657N。

图 12 显示了一个示例，说明两个成员如何将日志记录写入其活动日志流中的日志文件。每个日志文件由一个框表示。

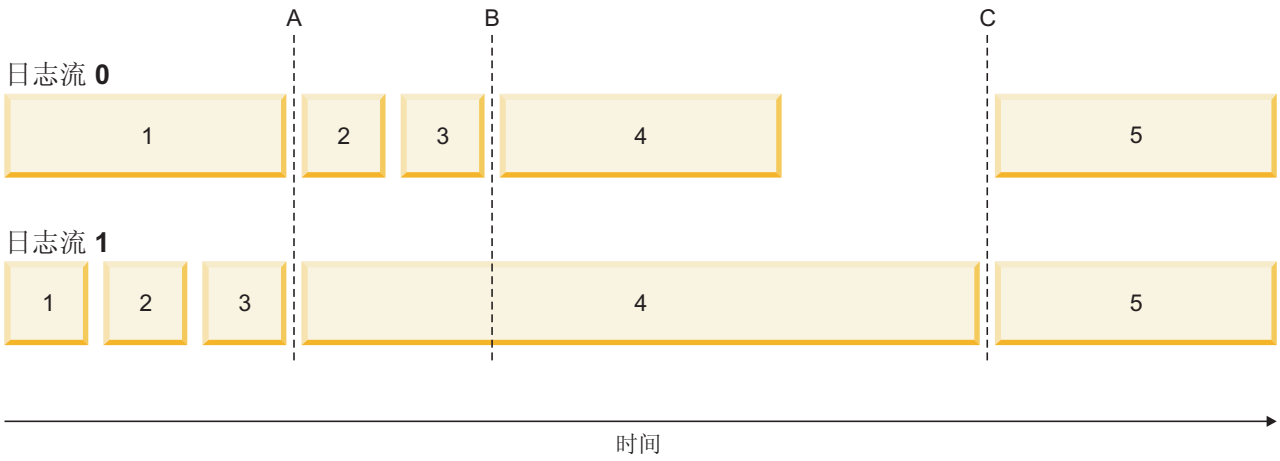


图 12. DB2 pureScale环境中的日志文件

考虑仅丢失了日志流 1 中的日志文件 4 的情况，前滚至时间 A 的操作将成功，而前滚至时间 B、时间 C 或 END OF LOGS 的操作将失败。由于日志文件 4 不可用，所以 ROLLFORWARD 命令将返回 SQL1273N。此外，因为在开始日志流 1 中的日志文件 4 的相同时间段写入了日志流 0 的文件 2 和 3 中的日志记录，所以在日志流 1 中的日志文件 4 可用之前，前滚操作无法处理日志文件 2 和 3。其结果就是前滚操作在时间 A 停止，并且在流 1 中的日志 4 可用之前，在时间 A 后无法执行任何后续前滚操作。

考虑前滚操作期间仅丢失了日志流 0 中的日志文件 4 的另一种情况。如果发出带 **END OF LOGS** 选项（或时间 B 后的任何时间）的 **ROLLFORWARD** 命令，那么前滚操作将在时间 B 停止，并且将返回 SQL1273N，这是因为丢失了流 0 中的日志文件 4。前滚操作可以重放时间 B 之前日志流 0 中文件 2 和 3 中的日志记录以及流 1 中文件 4 中的某些日志。前滚操作必须在时间 B 停止，即使流 1 中的其他日志可用也是如此，这是因为日志合并过程要求所有流中的所有日志都可用。

如果您可以找到丢失的日志文件，请使其可用并重新发出 **ROLLFORWARD DATABASE** 命令。如果找不到丢失的日志文件，请发出 **ROLLFORWARD DATABASE...STOP** 命令，以在丢失的日志文件之前的最后一个一致点处完成前滚操作。

虽然检测丢失日志的功能确保不会因丢失日志文件而导致数据库毁坏，但在丢失了日志文件的情况下，将会阻止重放某些事务，并且如果找不到丢失的日志文件，那么会因此而丢失数据。

必要资源

日志流合并操作需要其他 EDU。在数据库激活期间，将在每个成员上创建一个 **db2lfr** EDU。启动需要执行日志流合并的日志读操作时，将为每个日志流创建一个 **db2shred** EDU 和一个 **db2lfr** EDU。虽然每个 **db2lfr-db2shred** 组都会分配它自己的一组日志页和日志记录缓冲区，但这并不会消耗大量的其他内存或系统资源；将为日志流合并中涉及的每个成员分配大约 400 KB。

在日志流合并操作期间，成员会将其他成员所拥有的日志文件检索到该成员的溢出日志路径、主日志路径或镜像日志路径。在 DB2 pureScale环境中，请先确保检索路径具有足够的可用磁盘空间，然后再启动前滚操作。这允许该操作在 DB2 pureScale环境中根据需要从归档中检索更多的文件，而不会影响性能。请使用以下经验算式来计算检索所有成员的活动日志文件将需要多少空间：(**logprimary** + **logsecond**) * 成员数。

示例

- 更新 **newlogpath** 全局数据库配置参数:

```
db2 update db cfg for db mydb using newlogpath /home/dbuser/logdir
```

- 更新单个成员上的 **max_log** 每个成员的数据库配置参数:

```
db2 update db cfg for db mydb member 1 using max_log 5
```

- 更新主日志路径:

```
db2 connect to mydb
db2 update db cfg for mydb using newlogpath /home/dbuser/newlogpath
db2 get db cfg for mydb
...
Changed path to log files   (NEWLOGPATH) = /home/dbuser/newlogpath/NODE0000/LOGSTREAM0000/
Path to log files                        = /home/dbuser/dbuser/NODE0000/LOGSTREAM0000/
...
```

更改未生效，因为成员仍然是活动的。

```
db2 terminate
db2 deactivate db mydb
db2 connect to mydb
db2 get db cfg for mydb
...
Changed path to log files   (NEWLOGPATH) =
Path to log files                        = /home/dbuser/newlogpath/NODE0000/LOGSTREAM0000/
...
```

每个成员都使用 /home/dbuser/newlogpath/NODE0000/LOGSTREAM*xxxx* 日志路径，其中 *xxxx* 是使用该路径的日志流的日志流标识。

- 复原备份映像时设置新的主日志路径:

```
db2 restore db mydb newlogpath '/home/dbuser/newlogpath' without prompting
```

DB2 pureScale环境中的日志序号

DB2 数据库使用日志序号（64 位标识）来确定用于生成日志记录的操作的顺序。

LSN 是一个持续增加的值。每个成员都会写入它自己的日志文件集（*日志流*），并且单一日志流中的 LSN 是唯一号码。

因为 LSN 是在每个成员上独立生成，并且存在多个日志流，所以在不同日志流之间可能存在重复的 LSN 值。日志记录标识 (LRI) 用于识别日志流之间的日志记录；数据库中任何日志流内的每个日志记录都会指定一个唯一的 LRI。使用 **db2pd** 命令来确定恢复操作正在处理的 LRI。

第 8 章 恢复历史记录文件

恢复历史记录文件是随每个数据库创建的，在执行各种操作期间自动更新。

以下操作导致更新恢复历史记录文件:

- 备份了数据库或表空间
- 复原了数据库或表空间
- 前滚了数据库或表空间
- 自动重建了数据库，并且复原了多个映像
- 创建了表空间
- 改变了表空间
- 停顿表空间
- 重命名表空间
- 删除表空间
- 装入表
- 删除表（启用了恢复已废弃的表，并且您正在使用可恢复日志记录）
- 重组表
- 调用随需应变日志归档
- 写入新日志文件（使用可恢复日志记录时）
- 归档日志文件（使用可恢复日志记录时）
- 恢复数据库

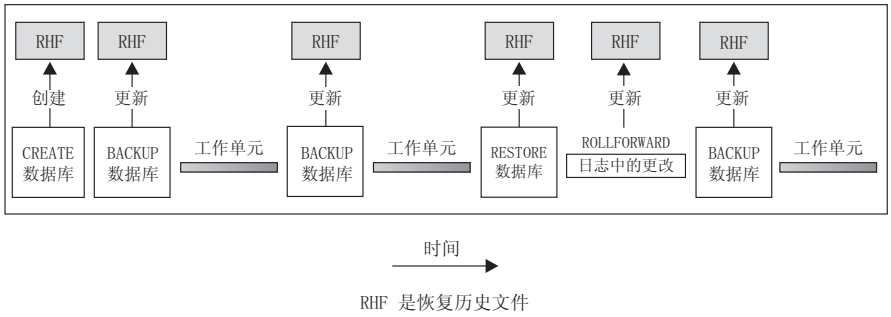


图 13. 创建并更新恢复历史记录文件

可以使用此文件中汇总的备份信息，将数据库的全部或部分恢复至给定的时间点。该文件中的信息包括:

- 唯一地标识每个条目的标识 (ID) 字段
- 已复制的部分数据库和复制方法
- 建立副本的时间
- 副本的位置（指示设备信息和访问副本的逻辑方式）
- 进行上次复原操作的时间
- 重命名表空间的时间，显示了该表空间的先前名称和当前名称

© Copyright IBM Corp. 2001, 2013

- 备份操作的状态: 活动、不活动、到期的或删除的
- 数据库备份保存的或前滚恢复操作期间处理的最后一个日志序号。

要查看恢复历史记录文件中的条目，可使用 **LIST HISTORY** 命令。

每个备份操作（数据库备份、表空间备份或增量备份）都包括复制恢复历史记录文件。该恢复历史记录文件与数据库相关联。删除数据库会删除恢复历史记录文件。将数据库复原到新位置会复原该恢复历史记录文件。复原不会覆盖现有恢复历史记录文件，除非磁盘上的文件没有任何条目。如果是这种情况，那么会根据备份映像复原数据库历史记录。

如果当前数据库不能使用或不可用，且关联的恢复历史记录文件被损坏或被删除，那么 **RESTORE** 命令中的一个选项只允许复原恢复历史记录文件。这样，可以复查该恢复历史记录文件，以提供有关要用于复原该数据库的备份的信息。

该文件的大小由 **rec_his_retentn** 配置参数控制，该参数指定该文件中条目的保留期（以天计）。即使将此参数的值设置为零 (0)，仍会保留最新的完整数据库备份（加上它的复原集）。（除去此副本的唯一方法是使用带 **FORCE** 选项的 **PRUNE HISTORY**。）保留期的缺省值是 366 天。可使用 **-1** 将保留期设置为不确定的天数。在此情况下，需要显式地删除该文件。

恢复历史记录文件条目状态

数据库管理器在恢复历史记录文件中对备份操作、复原操作、表空间创建以及其他事件创建条目。恢复历史记录文件中的每个条目都有一个相关联的状态: 活动、不活动、已到期、正在删除、已删除或 do_not_delete。

数据库管理器使用恢复历史记录文件条目的状态来确定是否需要与该条目关联的物理文件来恢复数据库。作为自动修剪的一部分，数据库管理器会更新恢复历史记录文件条目的状态。

活动数据库备份

活动的数据库备份是可使用当前日志进行复原和前滚，以恢复该数据库的当前状态的一种备份。

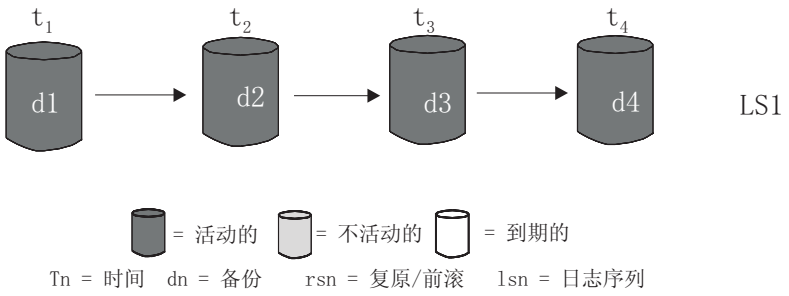


图 14. 活动数据库备份. **num_db_backups** 的值已设置为 4。

不活动数据库备份

而不活动的数据库备份是在复原时会将数据库移动回先前状态的一种备份。

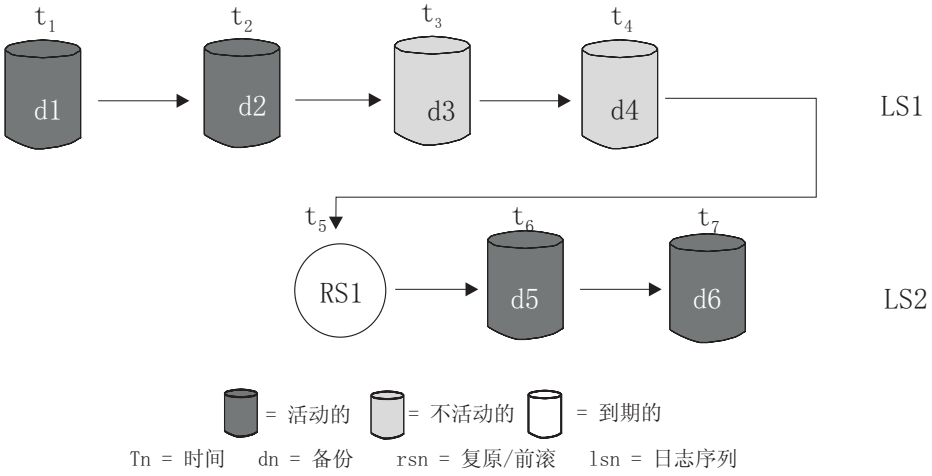


图 15. 不活动数据库备份

到期数据库备份

到期数据库备份映像是不再需要的映像，因为有更新的备份映像可用。

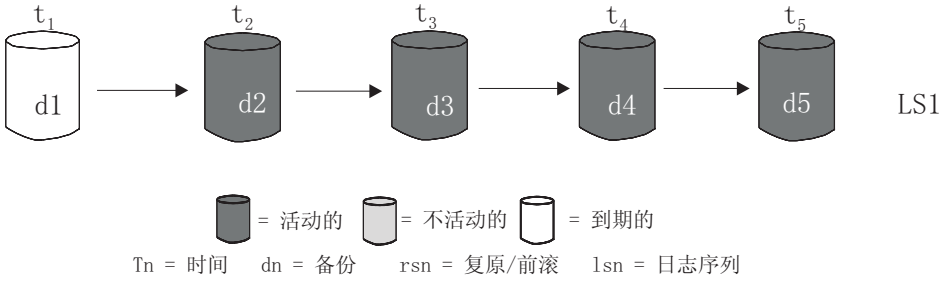


图 16. 到期数据库备份

标记为 do_not_delete 的条目

可以使用 **PRUNE HISTORY** 命令或 db2Prune API 除去（修剪）恢复历史记录文件条目。作为自动修剪的一部分，数据库管理器还会修剪恢复历史记录文件条目。

修剪标记为 do_not_delete 的条目只有三种方法:

- 调用带 **WITH FORCE** 选项的 **PRUNE HISTORY** 命令
- 调用带 **PRUNE HISTORY** 和 **WITH FORCE** 选项的 ADMIN_CMD 过程
- 调用带 DB2_PRUNE_OPTION_FORCE 选项的 db2Prune API

除非执行这三种操作之一，否则永远不会从恢复历史记录文件中修剪标记为 do_not_delete 的条目。

数据库管理器不会将恢复历史记录文件条目的状态设置为 do_not_delete。您可以使用 **UPDATE HISTORY** 命令将恢复历史记录文件条目的状态设置为 do_not_delete。

标记为“正在删除”的条目

标记为“正在删除”的条目处于正在被除去的过程中。如果修剪操作提前终止，那么该条目可能仍然存在。在这种情况下，与该条目相关联的文件可能仍然存在，也可能已不存在，但都将视为已不存在（如同已删除的条目一样）。

以下是有关不同恢复历史记录文件条目的状态的更多示例:

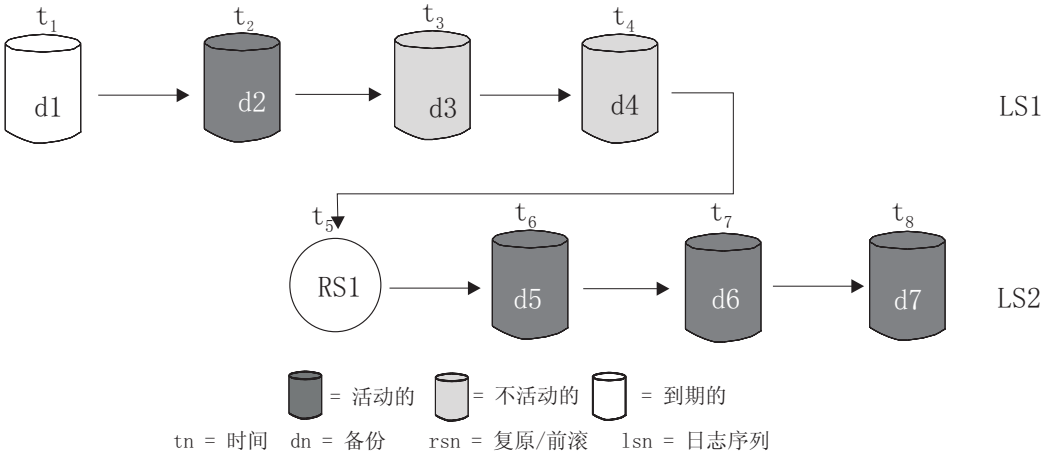


图 17. 混合活动的、不活动的和到期的数据库备份

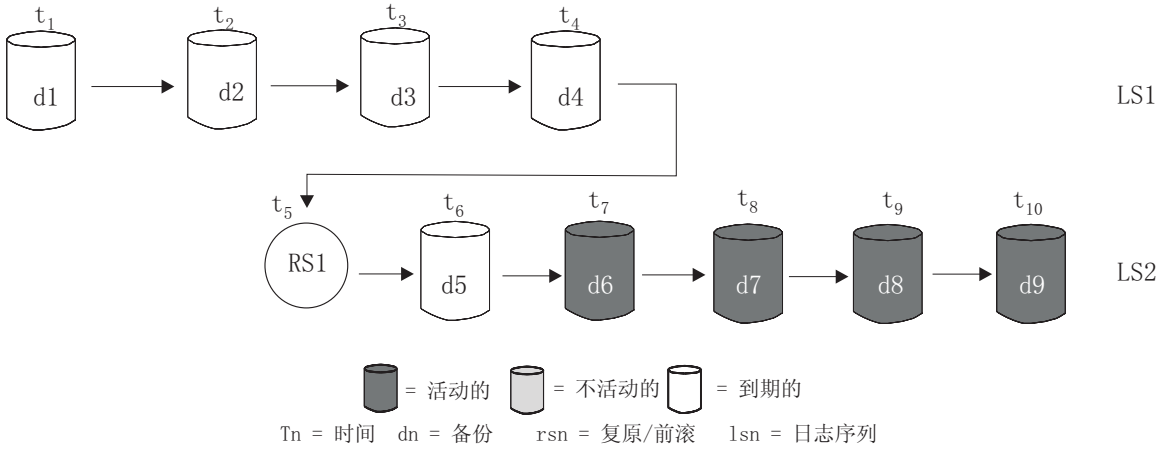


图 18. 到期日志序列

使用 DB_HISTORY 管理视图查看恢复历史记录文件条目

可以使用 DB_HISTORY() 管理视图来访问数据库历史记录文件的内容。此方法是使用 **LIST HISTORY** CLP 命令或 C 历史 API 的备用方法。

开始之前

使用此函数需要数据库连接。

关于此任务

对数据库历史记录文件执行删除和更新操作只能通过 **PRUNE HISTORY** 或 **UPDATE HISTORY** 命令进行。

过程

在 SQL SELECT 语句中使用 DB_HISTORY() 管理视图来访问您连接至的数据库的数据库历史记录文件或 **DB2NODE** 环境变量指定的数据库分区上的数据库历史记录文件。 例如，要查看历史记录文件的内容，可使用以下命令:

```
SELECT * FROM TABLE(DB_HISTORY()) AS LIST_HISTORY
```

示例

要隐藏管理视图的语法，可以按如下所示创建视图:

```
CREATE VIEW LIST_HISTORY AS
     SELECT * FROM TABLE(DB_HISTORY()) AS LIST_HISTORY
```

在创建了此视图之后，可以对此视图运行查询。例如:

```
SELECT * FROM LIST_HISTORY
```

或者

```
SELECT dbpartitionnum FROM LIST_HISTORY
```

或者

```
SELECT dbpartitionnum, start_time, seqnum, tabname, sqlstate
    FROM LIST_HISTORY
```

有关 DB_HISTORY 管理视图返回的列和列数据类型的列表，请参阅 DB_HISTORY 管理视图。

修剪恢复历史记录文件

数据库管理器在恢复历史记录文件中对备份操作、复原操作、表空间创建以及其他事件创建条目。

当由于不再需要关联的恢复对象来恢复数据库而导致恢复历史记录文件中的条目不再相关时，可能要从恢复历史记录文件中除去或*修剪*这些条目。

过程

可以使用下列方法修剪恢复历史记录文件中的条目:

- 调用 **PRUNE HISTORY** 命令
- 调用 db2Prune API
- 调用带 PRUNE_HISTORY 参数的 ADMIN_CMD 过程

下一步做什么

当使用这些方法之一来修剪恢复历史记录文件时，数据库管理器会从恢复历史记录文件中除去（修剪）早于指定时间戳记的条目。

如果某个恢复历史记录文件条目与指定修剪的条件匹配，但恢复数据库仍然需要该条目，那么除非使用 **WITH FORCE** 参数或 DB2PRUNE_OPTION_FORCE 标志，否则数据库管理器不会修剪该条目。

如果使用 **AND DELETE** 参数或 DB2PRUNE_OPTION_DELETE 标志，那么也将删除与修剪的条目关联的日志文件。

如果将 **AUTO_DEL_REC_OBJ** 数据库配置参数设置为 ON，并使用 **AND DELETE** 参数或 DB2PRUNE_OPTION_DELETE 标志，那么将删除与修剪的条目关联的日志文件、备份映像和装入副本映像。

使恢复历史记录文件修剪自动进行

可以将数据库管理器配置为自动修剪恢复历史记录文件条目并更新其状态。

可以使用 **UPDATE HISTORY** 命令、db2HistoryUpdate API 或带“UPDATE_HISTORY”参数的 ADMIN_CMD 过程来手动更新恢复历史记录文件条目的状态。可以使用 **PRUNE HISTORY** 命令、db2Prune API 或带“PRUNE_HISTORY”参数的 ADMIN_CMD 过程来从恢复历史记录文件中手动除去或修剪条目。但是，建议您配置数据库管理器来管理恢复历史记录文件，代替手动更新和修剪恢复历史记录文件。

数据库管理器会在下列情况下自动更新和修剪恢复历史记录文件条目:

- 在成功完成完整的（非增量）数据库备份操作或完整的（非增量）表空间操作后
- 在成功完成不需要前滚操作的数据库复原操作后
- 在成功完成数据库前滚操作后

在自动修剪期间，数据库管理器执行两项操作:

1. 更新恢复历史记录文件条目的状态
2. 修剪到期的恢复历史记录文件条目

数据库管理器通过以下方式更新恢复历史记录文件条目:

- 所有不再需要的活动数据库备份映像都标记为“已到期”。
- 所有标记为“不活动”并且在已到期数据库备份之前建立的数据库备份映像，也标记为“已到期”。所有相关的 “不活动”表空间备份映像和装入副本也标记为“已到期”。
- 如果复原了一个活动的数据库备份映像，但它不是历史记录文件中记录的最新数据库备份，那么属于同一个日志序列的任何后续数据库备份映像将被标记为“不活动的”。
- 如果复原了一个不活动的数据库备份映像，那么属于当前日志序列的任何不活动的数据库备份都被再次标记为“活动的”。不再在当前日志序列中的所有数据库备份映像都标记为“不活动的”。
- 与当前日志序列（也称当前日志链）不对应的任何数据库或表空间备份映像都标记为“不活动”。

当前日志序列由已复原的数据库备份映像和已处理的日志文件确定。一旦数据库备份映像复原后，所有后继的数据库备份映像都变得“不活动”，因为已复原的映像将开始一个新的日志链。（在复原备份映像时没有前滚操作的情况下就是这样。如果已发生了前滚操作，那么日志链中在中断后建立的所有数据库备份都将标记为“不活动”。可以想像，由于已对包含已损坏的当前备份映像的整个日志序列执行了 ROLLFORWARD 实用程序，所以必须复原旧的数据库备份映像。）

- 如果在表空间级的备份映像复原后应用当前日志序列无法达到数据库当前状态，该映像变为“不活动”。
- 不修剪状态为 do_not_delete 的任何条目，也不删除其关联的日志文件、备份映像和装入副本映像。
- 升级数据库时，历史记录文件中的所有联机数据库备份条目和所有联机或脱机表空间备份条目都标记为“已到期”，以便自动重建操作不会选择这些条目作为重建所需的映像。由于装入副本映像和日志归档条目不能用于恢复目的，所以也会将这些条目标记为“已到期”。

下列数据库配置参数控制数据库管理器修剪哪些条目:

num_db_backups
指定为数据库保留的数据库备份的数目。

rec_his_retentn
指定保留关于备份的历史记录信息的天数。

auto_del_rec_obj
指定数据库管理器是否应删除与修剪的恢复历史记录文件条目关联的日志文件、备份映像和装入副本映像。

要将数据库管理器配置为自动管理恢复历史记录文件，请设置下列配置参数:

- **num_db_backups**
- **rec_his_retentn**
- **auto_del_rec_obj**

如果 **auto_del_rec_obj** 设置为 ON，那么一旦有比 **num_db_backups** 配置参数更多的成功数据库备份条目，数据库管理器将自动修剪比 **rec_his_retentn** 旧的恢复历史记录文件条目。

防止恢复历史记录文件条目被修剪

通过将恢复历史记录文件条目的状态设置为 do_not_delete，可以防止重要恢复历史记录文件条目被修剪和关联的恢复对象被删除。

关于此任务

可以使用 **PRUNE HISTORY** 命令、带 PRUNE_HISTORY 的 ADMIN_CMD 过程或 db2Prune API 来除去（修剪）恢复历史记录文件条目。如果将 **AND DELETE** 参数与 **PRUNE HISTORY** 配合使用，或者将 DB2PRUNE_OPTION_DELETE 标志与 db2Prune 配合使用，而且 **auto_del_rec_obj** 数据库配置参数设置为 ON，那么关联的恢复对象也将被物理地删除。

作为自动修剪的一部分，数据库管理器还会修剪恢复历史记录文件条目。如果 **auto_del_rec_obj** 数据库配置参数设置为 ON，那么数据库管理器将删除与修剪的任何条目关联的恢复对象。

过程

要保护重要恢复历史记录文件条目和关联的恢复对象:

使用 **UPDATE HISTORY** 命令、db2HistoryUpdate API 或带“UPDATE_HISTORY”的 ADMIN_CMD 过程将重要恢复文件条目的状态设置为 do_no_delete。
修剪标记为 do_not_delete 的条目有三种方法:

- 调用带 **WITH FORCE** 选项的 **PRUNE HISTORY** 命令
- 调用带 **PRUNE HISTORY** 和 **WITH FORCE** 选项的 ADMIN_CMD 过程
- 调用带 DB2_PRUNE_OPTION_FORCE **iOption** 的 db2Prune API。

除非执行这三个过程之一，否则永远不会从恢复历史记录文件中修剪标记为 do_not_delete 的条目。
限制:

- 只有备份映像、装入副本映像和日志文件的状态可以设置为 do_not_delete。
- 备份条目的状态不会传播到与该备份操作相关的装入副本映像、非增量备份或日志文件。如果想要保存特定数据库备份条目及其相关日志文件条目，必须设置数据库备份条目和每个相关日志文件条目的状态。

第 9 章 管理恢复对象

随着您定期备份数据库，可能会累积大量数据库备份映像以及许多数据库日志和装入副本映像。IBM 数据服务器数据库管理器可简化这些恢复对象的管理。

关于此任务

存储恢复对象会占用大量存储空间。一旦运行了后续备份操作，便可以删除旧恢复对象，因为不再需要它们来复原数据库。但是，除去旧恢复对象很费时间。同时，当您删除旧恢复对象时，还可能会意外地损坏仍然需要的恢复对象。

过程

使用数据库管理器来删除复原数据库时不再需要的恢复对象有两种方法:

- 可以调用带 **AND DELETE** 参数的 **PRUNE HISTORY** 命令，或者调用带 DB2PRUNE_OPTION_DELETE 标志的 db2Prune API。
- 可以将数据库管理器配置为自动删除不需要的恢复对象。

使用 PRUNE HISTORY 命令或 db2Prune API 来删除数据库恢复对象

可以使用 **auto_del_rec_obj** 数据库配置参数和 **PRUNE HISTORY** 命令或 db2Prune API 来删除恢复对象。

关于此任务

当调用 **PRUNE HISTORY** 命令或调用 db2Prune API 时，IBM 数据服务器数据库管理器会执行以下操作:

- 从恢复历史记录文件中修剪状态不是 DB2HISTORY_STATUS_DO_NOT_DEL 的条目

当调用带 **AND DELETE** 参数的 **PRUNE HISTORY** 命令或调用带有 DB2PRUNE_OPTION_DELETE 标志的 db2Prune API 时，数据库管理器会执行以下操作:

- 从恢复历史记录文件中修剪早于指定时间戳记且状态不是 DB2HISTORY_STATUS_DO_NOT_DEL 的条目
- 删除与修剪的条目关联的物理日志文件

如果将 **auto_del_rec_obj** 数据库配置参数设置为 ON，那么当您调用带 **AND DELETE** 参数的 **PRUNE HISTORY** 命令或调用带有 DB2PRUNE_OPTION_DELETE 标志的 db2Prune API 时， 数据库管理器将执行以下操作:

- 从恢复历史记录文件中修剪状态不是 DB2HISTORY_STATUS_DO_NOT_DEL 的条目
- 删除与修剪的条目关联的物理日志文件
- 删除与修剪的条目关联的备份映像
- 删除与修剪的条目关联的装入副本映像

© Copyright IBM Corp. 2001, 2013

过程

要删除不需要的恢复对象:

1. 将 **auto_del_rec_obj** 数据库配置参数设置为 ON。
2. 调用带 **AND DELETE** 参数的 **PRUNE HISTORY** 命令，或者调用带 DB2PRUNE_OPTION_DELETE 标志的 db2Prune API。

自动管理数据库恢复对象

可以使用 **auto_del_rec_obj** 数据库配置参数和自动恢复历史记录文件修剪来配置 IBM 数据服务器数据库管理器，使它在每次完整数据库备份操作之后自动删除不需要的恢复对象。

关于此任务

每次完整（非增量）数据库备份操作成功后，数据库管理器将根据 **num_db_backup** 和 **rec_his_retentn** 配置参数的值修剪恢复历史记录文件:

- 如果恢复历史记录文件中的数据库备份条目大于 **num_db_backup** 配置参数的值，那么数据库管理器将从恢复历史记录文件中修剪早于 **rec_his_retentn** 配置参数的值且状态不是 DB2HISTORY_STATUS_DO_NOT_DEL 的条目。

如果将 **auto_del_rec_obj** 数据库配置参数设置为 ON，那么除了从恢复历史记录文件中修剪条目以外，数据库管理器还会执行以下操作:

- 删除与修剪的条目关联的物理日志文件
- 删除与修剪的条目关联的备份映像
- 删除与修剪的条目关联的装入副本映像

如果当前恢复历史记录中没有完整数据库备份映像可用（可能从未制作备份映像），那么将删除比 **rec_his_retentn** 指定的时间范围更早的映像。

如果数据库管理器因文件已不在恢复历史记录文件列出的位置而无法将其删除，那么数据库管理器将修剪该历史记录条目。

如果数据库管理器因数据库管理器与存储器管理器或设备之间的通信错误而无法删除文件，那么数据库管理器将不修剪该历史记录文件条目。错误解决后，该文件可以在下一自动修剪期间删除。

过程

要将数据库管理器配置为自动删除不需要的恢复对象:

1. 将 **auto_del_rec_obj** 数据库配置参数设置为 ON。
2. 设置 **rec_his_retentn** 和 **num_db_backups** 配置参数以启用自动恢复历史记录文件修剪。

防止恢复对象被删除

自动恢复对象管理可节省管理时间和存储空间。但是，您可能需要防止特定恢复对象被自动删除。通过将关联的恢复历史记录文件条目的状态设置为 do_not_delete，可以防止重要恢复对象被删除。

关于此任务

如果将 **auto_del_rec_obj** 数据库配置参数设置为 ON，那么修剪恢复对象的关联的恢复历史记录文件条目时，将同时删除这些恢复对象。在下列其中一种情况下，将修剪恢复历史记录文件条目:

- 调用带 **AND DELETE** 参数的 **PRUNE HISTORY** 命令
- 调用带 DB2PRUNE_OPTION_DELETE 标志的 db2Prune API
- 在每次成功完整备份表空间或数据库后，数据库管理器会自动修剪恢复历史记录文件。

无论是调用 **PRUNE HISTORY** 命令、调用 db2Prune API 还是将数据库管理器配置为自动修剪恢复历史记录文件中的条目，都不会修剪标记为 do_not_delete 的条目，也不会删除相关联的恢复对象。

限制

- 只有备份映像、装入副本映像和日志文件的状态可以设置为 do_not_delete。
- 备份条目的状态不会传播到与该备份操作相关的日志文件、装入副本映像或非增量备份。如果想要保存特定数据库备份条目及其相关日志文件条目，必须设置数据库备份条目和每个相关日志文件条目的状态。

过程

使用 **UPDATE HISTORY** 命令将关联恢复文件条目的状态设置为 do_no_delete。

管理快照备份对象

必须使用 **db2acsutil** 命令来管理快照备份对象。不要使用文件系统实用程序来移动或删除快照备份对象。

开始之前

要执行快照备份和复原操作，需要适用于存储设备的 DB2 ACS API 驱动程序。有关集成驱动程序的受支持存储硬件的列表，请参阅 Tivoli 文档，网址为: 受支持存储子系统

在可以管理快照备份对象之前，必须启用 DB2 高级副本服务 (ACS)。请参阅: 第 378 页的『启用 DB2 高级副本服务 (ACS)』。

限制

db2acsutil 命令当前仅在 AIX 和 Linux 上受支持。

过程

1. 要列示可用的快照备份对象，请使用 **QUERY** 参数。

   例如，要为名为 dbminst1 的数据库管理器实例列示可用的快照备份对象，请使用下列语法:

   ```
   db2acsutil query instance dbminst1
   ```

2. 要检查给定快照备份操作的进度，请使用 **STATUS** 参数。

例如，要了解可能当前在名为 database1 的数据库上运行的快照备份操作的进度，请使用下列语法:

```
db2acsutil query status db database1
```

3. 要删除特定快照备份对象，请使用 **DELETE** 参数。

   例如，要为名为 database1 的数据库删除已经存在 10 多天的所有快照备份对象，请使用下列语法:

   ```
   db2acsutil delete older than 10 days ago db database1
   ```

将备份映像和日志文件上载到 TSM

可以选择先将备份映像和日志文件在相对较短的时间内备份到磁盘，随后将其上载到 Tivoli Storage Manager (TSM)，同时还保留恢复历史记录信息，所以好像是将备份映像和日志文件直接备份到 TSM。

当生成备份映像的速度大于 TSM 可以写入这些备份映像的速度时，此策略可能是适当的。

示例 1: 采用策略

作为恢复计划的一部分，您决定将一组特定的映像和日志保存在磁盘上以便于恢复并以预定的时间间隔（在本例中为每周）将最旧的映像和日志上载到 TSM。该过程将查询恢复历史记录文件以获取最旧的备份映像，然后将该映像及其日志上载到 TSM。

1. 使用以下命令来查询历史记录文件以获取可用的日志和映像:

   ```
   db2 list history all for db sample
   ```

   将返回以下信息:

   ```
   List History File for sample

   Number of matching file entries = 100

   ...
   ...
   ...

   Op Obj Timestamp+Sequence Type Dev Earliest Log Current Log Backup ID
   ---- ----- ---------------------------- ------- ----- --------------------
   X D 20110403134938 1 D S0000003.LOG C0000000
   ----------------------------------------------------------------------------
   Comment:
   Start Time: 20110403134938
   End Time: 20110403135204
   Status: A
   ----------------------------------------------------------------------------
   EID: 5 Location: /home/logdir/log1/diwu/SAMPLE/NODE0000/C0000000/S0000003.LOG

   ...
   ...
   ...

   Op Obj Timestamp+Sequence Type Dev Earliest Log Current Log Backup ID
   ---- ----- ---------------------------- ------ ----- --------------------
   B D 20110404135750001 F D S0000000.LOG S0000007.LOG
   ----------------------------------------------------------------------------
   Contains 2 tablespace(s):

   00001 SYSCATSPACE
   ```

```
00002 USERSPACE1
------------------------------------------------------------------------------
Comment: DB2 BACKUP SAMPLE OFFLINE
Start Time: 20110404135750
End Time: 20110404135755
Status: A
------------------------------------------------------------------------------
EID: 10 Location: /home/backupdir

...
...
...
```

2. 使用以下命令来选择要上载的最旧日志文件:

   ```
   db2adutl upload logs between s3 and s3 db sample
   ```

   将返回以下信息:

   ```
   =====================================================
   | Upload Summary: |
   =====================================================

   1 / 1 logs were successfully uploaded

   Logs successfully uploaded:
   /home/logdir/log1/diwu/SAMPLE/NODE0000/C0000000/S0000003.LOG
   ```

3. 使用以下命令来上载最旧的映像及其日志:

   ```
   db2adutl upload images taken at 20110404135750 with logs db sample
   ```

   将返回以下信息:

   ```
   Match found, but S0000003.LOG is already on TSM

   ===========================================================
   | Upload Summary: |
   ===========================================================

   1 / 1 backup images were successfully uploaded
   4 / 4 logs were successfully uploaded

   Backup Images successfully uploaded:
   /home/backupdir/SAMPLE.0.diwu.NODE0000.CATN0000.20110404135750.001

   Logs successfully uploaded:
   /home/logdir/log1/diwu/SAMPLE/NODE0000/C0000001/S0000004.LOG
   /home/logdir/log1/diwu/SAMPLE/NODE0000/C0000001/S0000005.LOG
   /home/logdir/log1/diwu/SAMPLE/NODE0000/C0000002/S0000006.LOG
   /home/logdir/log1/diwu/SAMPLE/NODE0000/C0000002/S0000007.LOG
   ```

4. 验证结果:

   a. 使用以下命令来查询历史记录文件:

      ```
      db2 connect to sample
      ```

      将返回以下信息:

      ```
      Database server = DB2/LINUXX8664 9.7.5
      SQL authorization ID = DIWU
      Local database alias = SAMPLE
      ```

      ```
      db2 select OPERATION, OBJECTTYPE, START_TIME, SEQNUM,
      FIRSTLOG, LASTLOG, LOCATION,DEVICETYPE from
      table(ADMIN_LIST_HIST()) as T
      ```

      将返回以下信息:

```
OPERATION OBJECTTYPE START_TIME SEQNUM FIRSTLOG LASTLOG LOCATION
DEVICETYPE
---------------- ----------------- ----------------------- -------------
X D 20110403134938 - S0000003.LOG C0000000 adsm/libtsm.a A
...
...
B D 20110404135750 1 S0000000.LOG S0000007.LOG adsm/libtsm.a A
```

b. 使用以下命令来查询 TSM:

```
db2adutl query db sample
```

将返回以下信息:

```
Query for database SAMPLE


Retrieving FULL DATABASE BACKUP information.
1 Time: 20110404135750 Oldest log: S0000007.LOG DB Partition Number: 0 Sessions: 1


Retrieving INCREMENTAL DATABASE BACKUP information.
No INCREMENTAL DATABASE BACKUP images found for SAMPLE


Retrieving DELTA DATABASE BACKUP information.
No DELTA DATABASE BACKUP images found for SAMPLE


Retrieving TABLESPACE BACKUP information.
No TABLESPACE BACKUP images found for SAMPLE


Retrieving INCREMENTAL TABLESPACE BACKUP information.
No INCREMENTAL TABLESPACE BACKUP images found for SAMPLE


Retrieving DELTA TABLESPACE BACKUP information.
No DELTA TABLESPACE BACKUP images found for SAMPLE


Retrieving LOAD COPY information.
No LOAD COPY images found for SAMPLE


Retrieving LOG ARCHIVE information.
Log file: S0000003.LOG, Chain Num: 0, DB Partition Number: 0, Taken at: 2011-04-04-21.38.28
Log file: S0000004.LOG, Chain Num: 0, DB Partition Number: 0, Taken at: 2011-04-04-21.38.29
Log file: S0000005.LOG, Chain Num: 0, DB Partition Number: 0, Taken at: 2011-04-04-21.38.30
Log file: S0000006.LOG, Chain Num: 0, DB Partition Number: 0, Taken at: 2011-04-04-21.38.30
Log file: S0000007.LOG, Chain Num: 1, DB Partition Number: 0, Taken at: 2011-04-04-21.38.31
```

5. 下一周，使用以下命令来上载最旧的备份映像:

```
db2adutl upload images taken at 20110409155645 with logs db sample
```

将返回以下信息:

```
Match found, but S0000003.LOG is already on TSM

============================================================
| Upload Summary: |
============================================================

1 / 1 backup images were successfully uploaded
2 / 2 logs were successfully uploaded

Backup Images successfully uploaded:
/home/backupdir/SAMPLE.0.diwu.NODE0000.CATN0000.20110409155645.001

Logs successfully uploaded:
/home/logdir/log1/diwu/SAMPLE/NODE0000/C0000000/S0000008.LOG
/home/logdir/log1/diwu/SAMPLE/NODE0000/C0000000/S0000009.LOG
```

6. 使用以下命令来查询 TSM 以验证结果:

```
db2adutl query db sample
```

将返回以下信息:

```
Query for database SAMPLE


Retrieving FULL DATABASE BACKUP information.
1 Time: 20110404135750 Oldest log: S0000007.LOG DB Partition Number: 0 Sessions: 1
2 Time: 20110409155645 Oldest log: S0000009.LOG DB Partition Number: 0 Sessions: 1

Retrieving INCREMENTAL DATABASE BACKUP information.
No INCREMENTAL DATABASE BACKUP images found for SAMPLE


Retrieving DELTA DATABASE BACKUP information.
No DELTA DATABASE BACKUP images found for SAMPLE


Retrieving TABLESPACE BACKUP information.
No TABLESPACE BACKUP images found for SAMPLE


Retrieving INCREMENTAL TABLESPACE BACKUP information.
No INCREMENTAL TABLESPACE BACKUP images found for SAMPLE


Retrieving DELTA TABLESPACE BACKUP information.
No DELTA TABLESPACE BACKUP images found for SAMPLE


Retrieving LOAD COPY information.
No LOAD COPY images found for SAMPLE


Retrieving LOG ARCHIVE information.
Log file: S0000003.LOG, Chain Num: 0, DB Partition Number: 0, Taken at: 2011-04-04-21.38.28
Log file: S0000004.LOG, Chain Num: 0, DB Partition Number: 0, Taken at: 2011-04-04-21.38.29
Log file: S0000005.LOG, Chain Num: 0, DB Partition Number: 0, Taken at: 2011-04-04-21.38.30
Log file: S0000006.LOG, Chain Num: 0, DB Partition Number: 0, Taken at: 2011-04-04-21.38.30
Log file: S0000007.LOG, Chain Num: 0, DB Partition Number: 0, Taken at: 2011-04-04-21.38.31
Log file: S0000008.LOG, Chain Num: 0, DB Partition Number: 0, Taken at: 2011-04-09-20.21.50
Log file: S0000009.LOG, Chain Num: 0, DB Partition Number: 0, Taken at: 2011-04-09-20.21.51
```

示例 2: 上载并除去本地备份映像

1. 按以下方式来备份数据库:

   ```
   db2 backup db sample to /home/backupdir
   ```

   将返回以下信息:

   ```
   Backup successful. The timestamp for this backup image is: 20110401135620
   ```

2. 稍后，您决定使用以下命令上载该备份映像并将其从磁盘中擦除:

   ```
   db2adutl upload and remove images taken at 20110401135620 db sample
   ```

   将返回以下信息:

   ```
   File /home/backupdir/SAMPLE.0.diwu.NODE0000.CATN0000.20110401135620.001 is
   uploaded successfully.
   Would you really like to remove the original file (Y/N)
   ```

3. 输入 Y。

   注: 如果希望执行上载而不会在从磁盘中除去备份映像之前得到系统的提示，请使用以下命令:

   ```
   db2adutl upload and remove images taken at 20110401135620 db sample
   without prompting
   ```

将返回以下信息:

```
/home/backupdir/SAMPLE.0.diwu.NODE0000.CATN0000.20110401135620.001 is
successfully removed.

=====================================================
| Upload Summary: |
=====================================================

1 / 1 backup images were successfully uploaded

Backup Images successfully uploaded:
/home/backupdir/SAMPLE.0.diwu.NODE0000.CATN0000.20110401135620.001
```

示例 3: 在不指定时间戳记的情况下上载映像

1. 系统将提示您是否要上载最新的映像:

   ```
   Upload the most recent backup image?
   ```

2. 使用以下命令上载备份映像而不指定时间戳记或文件名:

   ```
   db2adutl upload images db sample
   ```

3. 输入 Y。

将返回以下信息:

```
============================================================
| Upload Summary: |
============================================================
1 / 1 backup images were successfully uploaded
Backup Images successfully uploaded:
/home/backupdir/SAMPLE.0.diwu.NODE0000.CATN0000.20110401160128.001
```

如果 TSM 上已存在最新的备份映像，那么将返回以下信息:

```
The most recent image is already on TSM.
```

示例 4: 上载日志和特定映像

因为您要上载特定备份映像并且要包括其日志，所以应该发出以下命令:

```
db2adutl upload images taken at 20110401155645 with logs db sample
```

将返回以下信息:

```
============================================================
| Upload Summary: |
============================================================

1 / 1 backup images were successfully uploaded
5 / 5 logs were successfully uploaded

Backup Images successfully uploaded:
/home/backupdir/SAMPLE.0.diwu.NODE0000.CATN0000.20110401155645.001

Logs successfully uploaded:
/home/logdir/log1/diwu/SAMPLE/NODE0000/C0000000/S0000000.LOG
/home/logdir/log1/diwu/SAMPLE/NODE0000/C0000000/S0000001.LOG
/home/logdir/log1/diwu/SAMPLE/NODE0000/C0000000/S0000002.LOG
/home/logdir/log1/diwu/SAMPLE/NODE0000/C0000000/S0000003.LOG
/home/logdir/log1/diwu/SAMPLE/NODE0000/C0000001/S0000004.LOG
```

如果要随该映像一起上载一组特定日志，那么应该指定序号范围，如以下命令所示:

```
db2adutl upload images taken at 20110401155645 logs between s3 and s7 db sample
```

将返回以下信息:

```
=========================================================
| Upload Summary: |
=========================================================

1 / 1 backup images were successfully uploaded
5 / 5 logs were successfully uploaded

Backup Images successfully uploaded:
/home/backupdir/SAMPLE.0.diwu.NODE0000.CATN0000.20110401155645.001

Logs successfully uploaded:
/home/logdir/log1/diwu/SAMPLE/NODE0000/C0000000/S0000003.LOG
/home/logdir/log1/diwu/SAMPLE/NODE0000/C0000001/S0000004.LOG
/home/logdir/log1/diwu/SAMPLE/NODE0000/C0000001/S0000005.LOG
/home/logdir/log1/diwu/SAMPLE/NODE0000/C0000002/S0000006.LOG
/home/logdir/log1/diwu/SAMPLE/NODE0000/C0000002/S0000007.LOG
```

第 10 章 监视复原操作的进度

可使用 **LIST UTILITIES** 命令来监视数据库上的复原操作。

过程

发出 **LIST UTILITIES** 命令并指定 **SHOW DETAIL** 参数

```
LIST UTILITIES SHOW DETAIL
```

结果

对于复原操作，不会给定最初估计值。而是指定 UNKNOWN。因为每个缓冲区都是从映像中读取的，所以实际读取的字节量会更新。对于可能要复原多个映像的自动增量复原操作，系统使用阶段来跟踪进度。每个阶段提供一个从增量链中复原的映像。最初，仅指示一个阶段。复原第一个映像之后，将显示阶段的总数。在复原每个映像时，会根据已处理的字节数来更新完成的阶段数。

示例

以下是用于监视复原操作性能的输出的示例:

```
ID                                   = 6
Type                                 = RESTORE
Database Name                        = SAMPLE
Partition Number     = 0
Description                          = db
Start Time                           = 08/04/2011 12:24:47.494191
State                = Executing
Invocation Type                      = User
   进度监视:
      Completed Work                 = 4096 bytes
      Start Time                     = 08/04/2011 12:24:47.494197
```

© Copyright IBM Corp. 2001, 2013

第 11 章 Backup 概述

可创建 DB2 数据库和所存储的相关数据的备份，以防止在数据库服务中断的情况下丢失数据。有几个工具可供您用来完成备份过程。

DB2 **BACKUP DATABASE** 命令的最简单格式仅要求您指定要备份的数据库的别名。例如:

```
db2 backup db sample
```

在 IBM Data Studio V3.1 或更高版本中，可以使用以下工具的任务助手：备份数据库. 任务助手可以指导您执行以下过程：设置选项、查看自动生成的命令以执行任务以及运行这些命令。有关更多详细信息，请参阅使用任务助手管理数据库。

如果成功完成命令，您将获取一个新的备份映像，它位于发出命令的路径或目录中。它位于此目录中的原因是，此示例中的命令不会显式地指定备份映像的目标位置。由 DB2 V9.5 和更高版本创建的备份映像是使用文件方式 600 生成的，这意味着在 UNIX 上仅实例所有者具有读写特权；在 Windows 上仅 DB2ADMNS组的成员（和 Administrators）对备份映像具有访问权。

注: 如果 DB2 客户机和服务器不在同一系统上，那么 DB2 数据库系统将确定客户机上的当前工作目录，并在服务器上将其用作备份目标目录。因此，建议为备份映像指定目标目录。

在调用备份实用程序时指定的目标位置创建备份映像。位置可以是:

- 目录（对于备份到磁盘或软盘）
- 设备（对于备份到磁带）
- Tivoli Storage Manager (TSM) 服务器
- 另一供应商服务器

每当调用数据库备份操作时，都会使用摘要信息自动更新恢复历史记录文件。此文件在数据库配置文件所在的目录中创建。

当不再需要旧备份映像时，如果要将它们删除，可以除去文件（如果那些备份是以文件形式存储的）。然后，如果运行指定了 **BACKUP** 选项的 **LIST HISTORY** 命令，那么还将返回有关已删除的备份映像的信息。必须使用 **PRUNE** 命令来从恢复历史记录文件中除去那些条目。

如果恢复对象是使用 Tivoli Storage Manager (TSM) 保存的，那么可以使用 **db2adutl** 实用程序来查询、抽取、验证和删除恢复对象。在 Linux 和 UNIX 上，此实用程序位于 `sqllib/adsm` 目录中；在 Windows 操作系统上，它位于 `sqllib\bin` 中。要获取快照，请使用位于 `sqllib/bin` 中的 **db2acsutil** 实用程序。

在所有操作系统上，在磁盘上创建的备份映像文件名由几个元素并置组成，这些元素由句点分隔:

```
DB_alias.Type.Inst_name.DBPARTnnn.timestamp.Seq_num
```

例如:

```
STAFF.0.DB201.DBPART000.19950922120112.001
```

© Copyright IBM Corp. 2001, 2013

数据库别名

在调用备份实用程序时指定的，由 1 至 8 个字符组成的数据库别名。

类型 备份操作的类型，其中: 0 表示完整的数据库级别备份、3 表示表空间级别的备份而 4 表示由 **LOAD COPY TO** 命令生成的备份映像。

实例名 从 **DB2INSTANCE** 环境变量中提取的、由 1 到 8 个字符组成的当前实例名。

数据库分区号

在单一分区数据库环境中，分区号始终是 DBPART000。在分区数据库环境中，它是 DBPART*xxx*，其中 *xxx* 是 db2nodes.cfg 文件中对数据库分区指定的编号。

时间戳记

执行备份操作时的日期和时间的 14 个字符表示法。该时间戳记的格式为 *yyyymmddhhnnss*，其中:

- *yyyy* 表示年份（1995 到 9999）
- *mm* 表示月份（01 到 12）
- *dd* 表示某月中的某一天（01 到 31）
- *hh* 表示小时（00 到 23）
- *nn* 表示分钟（00 到 59）
- *ss* 表示秒（00 到 59）

序号 用作文件扩展名的一个 3 位的数字。

将备份映像写到磁带时:

- 不创建文件名，但先前描述的信息会存储在备份头中以用于验证。
- 必须有一个磁带设备可通过标准操作系统接口来使用。但是，在大型分区数据库环境中，每个数据库分区服务器都有一个专用的磁带设备是不大可能的。可以将这些磁带设备与一个或多个 TSM 服务器连接，以便将对这些磁带设备的访问权提供给每个数据库分区服务器。
- 在分区数据库环境中，还可以使用提供虚拟磁带设备功能的产品，例如，REELlibrarian 4.2 或 CLIO/S。可以使用这些产品来访问通过伪磁带设备连接至其他节点（数据库分区服务器）的磁带设备。对远程磁带设备的访问权是透明地提供的，而伪磁带设备可以通过标准操作系统接口来访问。

不能备份未处于正常状态或备份暂挂状态的数据库。可以备份处于正常状态或备份暂挂状态的表空间。如果表空间未处于正常状态或备份暂挂状态，那么无法确定是否允许备份。

不允许对同一表空间进行并发备份操作。一旦对表空间启动了备份操作，任何后续尝试都将失败 (SQL2048N)。

如果数据库或表空间由于在执行复原操作期间发生了系统崩溃而处于部分复原状态，必须成功地复原该数据库或表空间才能对它们进行备份。

如果要备份的表空间的列表包含了临时表空间的名称，备份操作会失败。

备份实用程序可对建立不同数据库的备份副本的多个进程提供并行控制。此并行控制可使备份目标设备保持打开，直到所有备份操作结束。如果在备份操作期间发生错误，有一个打开的容器无法关闭，那么其他面向同一目标驱动器的备份操作就可能会

接收到访问错误。要校正这类访问错误，必须终止导致了该错误的备份操作，并与目标设备断开连接。如果正在对并发备份到磁带的操作使用备份实用程序，应确保这些进程的目标驱动器不是同一磁带。

显示备份信息

可通过 **db2ckbkp** 显示关于现有备份映像的信息。此实用程序允许您:

- 测试备份映像的完整性并确定是否可对其进行复原。
- 显示存储在备份头中的信息。
- 显示有关备份映像中对象和日志文件头的信息。

备份数据

使用 **BACKUP DATABASE** 命令来获取数据库数据副本并将其存储在另一介质上。然后在原始数据发生故障或损坏时使用此备份数据。

可以备份整个数据库，可以备份数据库分区，也可以只备份选择的表空间。

开始之前

不必连接到将要备份的数据库: 备份数据库实用程序自动建立与指定数据库的连接，而此连接会在备份操作完成时终止。如果已连接到要备份的数据库，当发出 **BACKUP DATABASE** 命令时将断开连接，备份操作将继续进行。

数据库可以是本地数据库或远程数据库。备份映像保留在数据库服务器上，除非您使用的是存储管理产品，如 Tivoli Storage Manager (TSM) 或 DB2 高级副本服务 (ACS)。

如果要执行脱机备份并且已使用 **ACTIVATE DATABASE** 命令激活数据库，那么在运行脱机备份之前必须取消激活该数据库。如果存在与该数据库的活动连接，为了成功取消激活该数据库，具有 SYSADM 权限的用户必须连接至该数据库并发出下列命令:

```
CONNECT TO database-alias
QUIESCE DATABASE IMMEDIATE FORCE CONNECTIONS;
UNQUIESCE DATABASE;
TERMINATE;
DEACTIVATE DATABASE database-alias
```

在分区数据库环境中，可以使用 **BACKUP DATABASE** 命令来逐个备份数据库分区，使用 **ON DBPARTITIONNUM** 命令参数来一次性备份多个数据库分区或使用 **ALL DBPARTITIONNUMS** 参数来同时备份所有数据库分区。可以使用 **LIST DBPARTITIONNUMS** 命令来确定具有您想要备份的用户表的数据库分区。

除非您在使用单一系统视图（SSV）备份，否则，如果您要在分区数据库环境中执行*脱机*备份，那么应将目录分区独立于所有其他数据库分区进行备份。例如，可以先备份目录分区，然后备份所有其他数据库分区。此操作是必需的，因为备份操作可能需要在目录分区上进行独占数据库连接（在此期间不能连接其他数据库分区）。如果要执行*联机*备份，那么可以同时或以任何顺序备份所有数据库分区（包括目录分区）。

在分布式请求系统上，备份操作适用于分布式请求数据库和存储在数据库目录（包装器、服务器和昵称等等）中的元数据。不备份数据源对象（表和视图），除非它们也存储在分布式请求数据库中。

如果某个数据库是使用数据库管理器的前发行版创建的且尚未升级，那么必须先升级该数据库才能对其进行备份。

限制

以下限制适用于 BACKUP 实用程序:

- 表空间备份操作和表空间复原操作不能同时运行，即使涉及的是不同的表空间。
- 如果要能够在分区数据库环境中执行前滚恢复，那么必须定期备份节点列表上的数据库。还必须具有系统中余下节点（甚至未包含该数据库的用户数据的节点）的至少一个备份映像。下列两种情况都需要在不包含数据库的用户数据的数据库分区服务器中存在数据库分区的备份映像:
  - 在建立上一个备份之后已将一个数据库分区服务器添加到数据库系统，因此需要在此数据库分区服务器上执行正向恢复。
  - 使用时间点恢复，它要求系统中的所有数据库分区都处于前滚暂挂状态。
- DMS 表空间的联机备份操作与下列操作不兼容:
  - 装入
  - 重组（联机和脱机）
  - 删除表空间
  - 截断表
  - 创建索引
  - 最初未记录任何内容（与 CREATE TABLE 和 ALTER TABLE 语句配合使用）
- 如果尝试对当前活动的数据库执行脱机备份，那么将接收到错误。在运行脱机备份之前，可以通过发出 **DEACTIVATE DATABASE** 命令来确保数据库未处于活动状态。

过程

要调用 BACKUP 实用程序，请执行以下操作:

- 在命令行处理器 (CLP) 中发出 **BACKUP DATABASE** 命令。
- 运行带 BACKUP DATABASE 参数的 ADMIN_CMD 过程。
- 使用 db2Backup 应用程序编程接口 (API)。
- 在 IBM Data Studio 中对 **BACKUP DATABASE** 命令打开任务助手。

示例

以下是通过 CLP 发出的 **BACKUP DATABASE** 命令的示例:

```
db2 backup database sample to c:\DB2Backups
```

下一步做什么

如果执行了脱机备份，那么在备份完成后，必须重新激活该数据库:

```
ACTIVATE DATABASE sample
```

执行快照备份

快照备份操作使用存储设备的快速复制技术来执行备份的数据复制部分。

开始之前

要执行快照备份和复原操作，需要适用于存储设备的 DB2 ACS API 驱动程序。有关集成驱动程序的受支持存储硬件的列表，请参阅 Tivoli 文档，网址为: 受支持存储子系统

在可以执行快照备份之前，必须启用 DB2 高级副本服务（ACS）。请参阅: 第 378 页的『启用 DB2 高级副本服务 (ACS)』。

限制

您无法使用快照备份来恢复单独的表空间。

如果您使用集成快照备份，那么不能执行重定向复原。FlashCopy® 复原会将包含所有数据库路径的一组完整卷组恢复到之前某个时间点。

过程

要执行快照备份，请使用下列其中一种方法:

- 发出带 **USE SNAPSHOT** 参数的 **BACKUP DATABASE** 命令，如以下示例所示:

```
db2 backup db sample use snapshot
```

- 调用带 **BACKUP DB** 和 **USE SNAPSHOT** 参数的 ADMIN_CMD 过程，如以下示例所示:

```
CALL SYSPROC.ADMIN_CMD
   ('backup db sample use snapshot')
```

- 发出介质类型为 SQLU_SNAPSHOT_MEDIA 的 db2Backup API，如以下示例所示:

```
int sampleBackupFunction( char dbAlias[],
                          char user[],
                          char pswd[],
                          char workingPath[] )
{
  db2MediaListStruct mediaListStruct = { 0 };

  mediaListStruct.locations = &workingPath;
  mediaListStruct.numLocations = 1;
  mediaListStruct.locationType = SQLU_SNAPSHOT_MEDIA;


  db2BackupStruct backupStruct = { 0 };

  backupStruct.piDBAlias = dbAlias;
  backupStruct.piUsername = user;
  backupStruct.piPassword = pswd;
  backupStruct.piVendorOptions = NULL;
  backupStruct.piMediaList = &mediaListStruct;

  db2Backup(db2Version950, &backupStruct, &sqlca);

       return 0;
}
```

将分割镜像用作备份映像

在 DB2 pureScale 环境外部，使用以下过程来在同一系统上的不同位置创建数据库的分割镜像，以用作备份映像。此过程可用于替代在数据库上执行备份数据库操作。

过程

要将分割镜像用作备份映像:

1. 使用以下命令来连接至主数据库:

   ```
   db2 connect to <db_name>
   ```

2. 使用以下命令来暂挂主数据库的 I/O 写操作:

   ```
   db2 set write suspend for database
   ```

   注: 当数据库处于暂挂状态时，您不应该运行其他实用程序或工具。您只应生成数据库副本。可选择在发出 **SET WRITE** SUSPEND 之前清空所有缓冲池，以将恢复时间减到最少。可以使用 FLUSH BUFFERPOOLS ALL 语句来清空所有缓冲池。

3. 使用相应的操作系统级别和储存器级别命令从主数据库创建一个或多个分割镜像。

   注:

   - 您务必复制整个数据库目录（其中包括卷目录）。还必须复制日志目录和存在于数据库目录之外的任何容器目录。如果使用多个存储器组，那么必须复制所有路径，包括这些路径的文件和子目录。要收集此信息，请参阅 DBPATHS 管理视图，该视图显示了需要分割的数据库的所有文件和目录。
   - 如果您对 **SET WRITE** 命令指定了 EXCLUDE LOGS，请勿将日志文件包括在副本中。

4. 使用以下命令来恢复主数据库上的 I/O 写操作:

   ```
   db2 set write resume for database
   ```

假定系统上将发生故障，请执行下列步骤，来复原将分割镜像数据库用作备份的数据库:

1. 使用以下命令来停止数据库实例:

   ```
   db2stop
   ```

2. 使用操作系统级别命令来复制分割数据。

   要点: 不要复制分割日志文件，因为前滚恢复需要原始日志。

3. 使用以下命令来启动数据库实例:

   ```
   db2start
   ```

4. 初始化主数据库:

   ```
   db2inidb database_alias as mirror
   ```

   其中 *database_alias* 表示数据库别名。

5. 将数据库前滚至日志末尾或某个时间点，然后停止。

在 DB2 pureScale 环境 中将分割镜像用作备份映像

在 DB2 pureScale环境中，使用以下过程来在同一系统上的不同位置创建数据库的分割镜像，以用作备份映像。此过程可用于替代在数据库上执行备份数据库操作。

过程

要将分割镜像用作备份映像:

1. 使用以下命令来连接至主数据库:

   ```
   db2 connect to <db_name>
   ```

2. 通过抽取并导入主集群中的设置，在辅助集群上配置 General Parallel File System (GPFS)。在主集群上，运行以下 GPFS 命令:

   ```
   mmfsctl filesystem syncFSconfig -n remotenodefile
   ```

   其中 *remotenodefile* 是辅助集群中主机的列表。
3. 使用以下命令来暂挂主数据库的 I/O 写操作:

   ```
   db2 set write suspend for database
   ```

   注: 当数据库处于暂挂状态时，您不应该运行其他实用程序或工具。您只应生成数据库副本。可选择在发出 **SET WRITE SUSPEND** 之前清空所有缓冲池，以将恢复时间减到最少。可以使用 FLUSH BUFFERPOOLS ALL 语句来清空所有缓冲池。
4. 使用以下命令来确定必须暂挂和复制的文件系统:

   ```
   db2cluster -cfs -list -filesystem
   ```
5. 使用以下命令来暂挂每个包含容器数据或日志数据的 GPFS 文件系统:

   ```
   /usr/lpp/mmfs/bin/mmfsctl filesystem suspend-write
   ```

   其中 *filesystem* 表示包含数据或日志数据的文件系统。

   注: 当 GPFS 文件系统处于暂挂状态时，将阻塞写操作。在此期间，您只应执行分割镜像操作，以最大程度地减少阻塞操作的时间量。
6. 使用相应的操作系统级别和储存器级别命令从主数据库创建一个或多个分割镜像。

   注:

   - 您务必复制整个数据库目录（其中包括卷目录）。还必须复制日志目录和存在于数据库目录之外的任何容器目录。如果使用多个存储器组，那么必须复制所有路径，包括这些路径的文件和子目录。要收集此信息，请参阅 DBPATHS 管理视图，该视图显示了需要分割的数据库的所有文件和目录。
   - 如果您对 **SET WRITE** 命令指定了 EXCLUDE LOGS，请勿将日志文件包括在副本中。
7. 对每个已暂挂的文件系统使用以下命令来恢复已暂挂的 GPFS 文件系统:

   ```
   /usr/lpp/mmfs/bin/mmfsctl filesystem resume
   ```

   其中 *filesystem* 表示包含数据或日志数据的暂挂文件系统。
8. 使用以下命令来恢复主数据库的 I/O 写操作:

   ```
   db2 set write resume for database
   ```

假定某种情况要求您使用分割镜像作为备份映像来复原数据库，请执行下列步骤:

1. 使用以下命令来停止主数据库实例:

   ```
   db2stop
   ```
2. 使用以下命令来列示集群管理器域:

   ```
   db2cluster -cm -list -domain
   ```
3. 使用以下命令来停止集群中每台主机上的集群管理器:

   ```
   db2cluster -cm -stop -host host -force
   ```

   注: 您关闭的最后一台主机必须是您将从中发出此命令的主机。
4. 使用以下命令来停止主数据库实例上的 GPFS 集群:

   ```
   db2cluster -cfs -stop -all
   ```

5. 使用适当的操作系统级别命令来复制主数据库的分割数据。

   要点: 不要复制分割日志文件，因为前滚恢复需要原始日志。
6. 使用以下命令来启动主数据库实例上的 GPFS 集群:

   ```
   db2cluster -cfs -start -all
   ```
7. 使用以下命令来启动集群管理器

   ```
   db2cluster -cm -start -domain domain
   ```
8. 使用以下命令来启动数据库实例:

   ```
   db2start
   ```
9. 使用以下命令来初始化主数据库:

   ```
   db2inidb database_alias as mirror
   ```
10. 将主数据库前滚至日志末尾或某个时间点，然后停止。

备份到磁带

备份数据库或表空间时，必须正确地设置块大小和缓冲区大小。如果要使用可变块大小（例如，在 AIX 上，如果块大小已设置为零），那么尤其要这样。

对备份时可使用的固定块大小的数量有限制。因为 DB2 数据库系统将备份映像头写为 4 KB 的块，所以会存在此限制。DB2 数据库系统支持的固定块大小只有 512、1024、2048 和 4096 个字节。如果使用固定块大小，那么可以指定任何备份缓冲区大小。但是您可能会发现，如果 DB2 数据库系统不支持该固定块大小，那么备份操作将无法成功完成。

如果数据库很大，那么使用固定块大小意味着您的备份操作可能花费比所预期更长的时间才能完成。为了提高性能，可以使用可变块大小。

注: 如果使用可变块大小，请确保您有经过良好测试的过程（包括明确指定的 **BACKUP** 和 **RESTORE** 命令的缓冲区大小），以使您可以使用按可变块大小创建的备份映像进行成功的恢复。

使用可变块大小时，必须指定备份缓冲区大小小于或等于正在使用的磁带设备的最大限制。要获得优化的性能，缓冲区大小必须等于正在使用的设备的最大块大小限制。

必须先发出以下命令，然后才能在 Windows 操作系统上使用磁带设备:

```
db2 initialize tape on device using blksize
```

其中:

device 是有效的磁带设备名。在 Windows 操作系统上，缺省值为 `\\.\TAPE0`。

blksize 是磁带的分块因子。它必须是 4096 的因子或倍数。缺省值是该设备的缺省块大小。

使用可变块大小从备份映像进行复原时可能会返回错误。如果发生这种情况，那么可能需要使用适当的块大小重写该映像。以下是 AIX 上的一个示例:

```
tctl -b 0 -Bn -f /dev/rmt0 read > backup_filename.file
dd if=backup_filename.file of=/dev/rmt0 obs=4096 conv=sync
```

备份映像转储到名为 `backup_filename.file` 的文件中。**dd** 命令使用块大小 4096，将映像转储到磁带上。

如果映像太大，不能转储到文件中，此方法可能会出现问题。有一种可能的解决方案是：使用 **dd** 命令将该映像从一个磁带设备转储至另一磁带设备。只要映像在一卷磁带上放得下，就可以使用这种方法。使用两个磁带设备时，**dd** 命令是:

```
dd if=/dev/rmt1 of=/dev/rmt0 obs=4096
```

如果不可能使用两台磁带设备，那么可以使用 **dd** 命令将映像转储至原始设备，然后再将该映像从原始设备转储至磁带。使用此方法的问题在于 **dd** 命令*必须*跟踪转储至原始设备的块数。此块数必须在将映像移动回磁带时指定。如果使用 **dd** 命令将映像从原始设备转储至磁带，该命令会将原始设备的整个内容转储至磁带。**dd** 实用程序不能确定使用了多少原始设备空间来保留映像。

当使用 BACKUP 实用程序时，您需要知道磁带设备的最大块大小限制。以下是一些示例:

设备	连接	块大小限制	DB2 缓冲区大小限制（以 4-KB 页计）
8 毫米	scsi	131,072	32
3420	s370	65,536	16
3480	s370	61 440	15
3490	s370	61 440	15
3490E	s370	65,536	16
7332（4 毫米）[1]	scsi	262,144	64
3490e	scsi	262,144	64
3590[2]	scsi	2,097,152	512
3570 (magstar MP)		262,144	64

注:

1. 7332 没有块大小的限制。256 KB 只是一个建议的值。块大小限制由父适配器确定。
2. 在 3590 支持 2 MB 的块大小时，如果性能完全可以满足您的需要，就可以尝试使用较低的值（例如 256 KB）。
3. 关于设备限制的信息，请查看设备文档或向设备供应商咨询。

验证磁带设备的兼容性

仅在 UNIX、Linux 和 AIX 操作系统上，要确定您的磁带设备是否支持备份 DB2 数据库，请执行以下过程:

作为数据库管理器实例所有者运行操作系统命令 **dd**，以便对磁带设备执行读写操作。如果 **dd** 命令成功，那么可以使用磁带设备来备份 DB2 数据库。

备份到命名管道

现在已支持将数据库备份至 UNIX 操作系统上的本地命名管道以及从这些命名管道复原数据库。

开始之前

命名管道的写程序与阅读器必须是同一台机器。管道必须存在并位于本地文件系统上。因为将命名管道视为本地设备，所以不需要指定目标是命名管道。

过程

1. 创建命名管道。 以下是一个 AIX 示例:

   ```
   mkfifo /u/dmcinnis/mypipe
   ```

2. 如果打算由复原实用程序来使用备份映像，那么复原操作必须在备份操作之*前*调用，才不会丢失任何数据:

   ```
   db2 restore db sample from /u/dmcinnis/mypipe into mynewdb
   ```

3. 使用此管道作为数据库备份操作的目标:

   ```
   db2 backup db sample to /u/dmcinnis/mypipe
   ```

备份分区数据库

在分区数据库环境中备份数据库可能会带来以下困难: 跟踪每个数据库分区的备份是否成功; 管理多个日志文件和备份映像; 以及确保所有数据库分区的日志文件和备份映像需要最少的恢复时间来复原数据库。使用单一系统视图（SSV）备份是备份分区数据库最简单的方法。

关于此任务

在分区数据库环境中备份数据库的方法有三种:

- 使用 **BACKUP DATABASE** 命令、带 ADMIN_CMD 过程的 **BACKUP DATABASE** 命令或 db2Backup API 函数一次备份一个数据库分区。
- 将 **db2_all** 命令和 **BACKUP DATABASE** 命令配合使用来备份指定的所有数据库分区。
- 运行单一系统视图 (SSV) 备份来同时备份部分或全部数据库分区。
- 在 IBM Data Studio 中使用任务助手来引导您完成备份数据库的过程。

每次备份一个数据库分区既耗时又容易出错。使用 **db2_all** 命令备份所有分区比逐个备份数据库分区更简单，因为通常只需进行一次命令调用。但是，当使用 **db2_all** 来备份分区数据库时，有时仍然需要多次调用 **db2_all**，因为不能同时备份包含目录的数据库分区和非目录数据库分区。无论是一次备份一个数据库还是使用 **db2_all** 进行备份，管理使用这两种方法创建的备份映像都会很难，因为每个数据库分区的备份映像的时间戳记不同，而且在数据库分区的备份映像之间协调最低恢复时间也很难。

由于先前提到的原因，在分区数据库环境中备份数据库的推荐方法是使用 SSV 备份。

过程

要使用 SSV 备份来同时备份分区数据库的部分或全部数据库分区，请执行以下操作:

1. 可选: 允许数据库保持联机，或使数据库脱机。

   数据库联机或脱机时都可以进行分区数据库备份。如果数据库联机，那么 BACKUP 实用程序将获取至其他数据库分区的共享连接，因此用户应用程序能够连接到正在备份的数据库分区。

2. 在包含数据库目录的数据库分区上，使用适当的参数对分区数据库进行备份。
   - 运行带 **ON DBPARTITIONNUMS** 参数的 **BACKUP DATABASE** 命令。
   - 使用 ADMIN_CMD 过程运行带 **ON DBPARTITIONNUMS** 参数的 **BACKUP DATABASE** 命令。
   - 带 **iAllNodeFlag** 参数调用 db2Backup API。

- 在 IBM Data Studio 中对 **BACKUP DATABASE** 命令打开任务助手。

3. 可选: 在备份映像中包含恢复所需的日志文件。

   缺省情况下，如果您执行 SSV 备份（即，您指定 **ON DBPARTITIONNUM** 参数），那么备份映像将包括日志文件。如果不希望备份映像包括日志文件，请在运行备份时使用 **EXCLUDE LOGS** 命令参数。对于非 SSV 备份，缺省情况下备份映像中将不包含日志文件。

   有关更多信息，请参阅下列主题: 第 143 页的『使用备份映像包括日志文件』。

4. 可选: 删除以前的备份映像。 用于删除旧备份映像的方法取决于存储备份映像的方式。例如，如果将备份映像存储在磁盘中，那么可以删除这些文件；如果使用 Tivoli Storage Manager 来存储备份映像，那么可以使用 **db2adutl** 实用程序来删除备份映像。如果使用的是 DB2 高级副本服务 (ACS)，那么可以使用 **db2acsutil** 来删除快照备份对象。

使用 IBM Tivoli Space Manager 分层存储管理来备份分区表

Tivoli Space Manager 分层存储管理器（HSM）客户机程序自动将合格的文件迁移到辅助存储器，以保持本地文件系统上的可用空间达到特定水平。

借助表分区功能，表数据分布在多个称为数据分区的存储对象中。HSM 支持将各个数据分区备份到辅助存储器。

在使用 SMS 表空间时，每个数据分区范围都表示为相应目录中的一个文件。因此，可以非常方便地将各个数据范围（数据分区）迁移到辅助存储器。

当使用 DMS 表空间时，每个容器都表示为文件。在此情况下，应该将不频繁访问的范围存储在它们自己的表空间中。在发出使用了 EVERY 子句的 CREATE TABLE 语句时，使用 NO CYCLE 子句来确保表级别 IN 子句中列示的表空间数与正在创建的数据分区数匹配。以下示例演示了这种情况:

示例 1

```
CREATE TABLE t1 (c INT) IN tbsp1, tbsp2, tbsp3 NO CYCLE
  PARTITION BY RANGE(c)
  (STARTING FROM 2 ENDING AT 6 EVERY 2);
```

启用自动备份

数据库可能会由于各种硬件或软件故障而变得不可用。确保有数据库的最新完整备份，这对于规划和实现系统灾难恢复策略而言至关重要。

通过在灾难恢复策略中使用数据库自动备份，DB2 就能够正确并且定期地备份数据库。

关于此任务

可以使用命令行界面或 AUTOMAINT_SET_POLICY 系统存储过程来配置自动备份。还需要启用运行状况指示器 db.db_backup_req，缺省情况下已启用该指示器。注意，进行评估时，将仅考虑活动数据库。

过程

- 要使用命令行界面来配置自动备份，请将下列每个数据库配置参数都设置为 ON:
  - **AUTO_MAINT**
  - **AUTO_DB_BACKUP**
- 要使用 IBM Data Studio 配置自动备份，请右键单击数据库并选择任务助手来配置自动备份。
- 要使用 AUTOMAINT_SET_POLICY 系统存储过程来配置自动备份:
  1. 创建指定备份媒体、是联机还是脱机备份以及备份频率等详细信息的配置 XML 输入。

     可以复制 SQLLIB/samples/automaintcfg 目录中名为 DB2DefaultAutoBackupPolicy.xml 的样本文件的内容并修改该 XML，以满足您的配置要求。
  2. 可选: 创建包含配置 XML 输入的 XML 输入文件。
  3. 调用带下列参数的 AUTOMAINT_SET_POLICY:
     - 维护类型: AutoBackup
     - 配置 XML 输入: 要么为包含配置 XML 输入文本的 BLOB; 要么为包含配置 XML 输入的文件的名称。

     有关使用 AUTOMAINT_SET_POLICY 系统存储过程的更多信息，请参阅第 58 页的『使用 SYSPROC.AUTOMAINT_SET_POLICY 或 SYSPROC.AUTOMAINT_SET_POLICYFILE 来配置自动维护策略』主题。

自动备份数据库

由于发生各种硬件或软件故障，数据库可能会变得不可用。自动备份数据库功能减轻了 DBA 的数据库备份管理任务，它始终会确保在需要时对数据库执行最新的完整备份。

它根据下面的一种或多种情况来确定是否需要执行备份操作:

- 您从未完成完整的数据库备份
- 自从上一次执行完整备份以来所经过的时间超过了指定的小时数
- 自从上一次执行备份以来所消耗的事务日志空间超过了指定的 4 KB 页数（仅适用于归档日志记录方式）。

通过为系统规划和实现灾难恢复策略来保护数据。如果能够满足您的需要，那么可以将自动备份数据库功能作为您的备份和恢复策略的一部分。

如果对数据库启用了前滚恢复（归档日志记录），那么可以对联机备份或脱机备份启用自动备份数据库功能。否则，只能进行脱机备份。自动备份数据库支持磁盘、磁带、Tivoli Storage Manager (TSM) 和供应商 DLL 介质类型。

如果选择了备份至磁盘，那么自动备份功能部件将定期从在自动数据库备份配置中指定的目录中删除备份映像。无论在自动备份策略文件中指定的完全备份数是多少，在任何给定时间只有最新的备份映像才可用。建议将此目录专用于自动备份功能部件，而不用来存储其他备份映像。

可以使用 **auto_db_backup** 和 **auto_maint** 数据库配置参数来启用或禁用自动备份数据库功能。在分区数据库环境中，如果对每个数据库分区都启用了数据库配置参数，那么每个数据库分区上都将运行自动备份数据库。

也可以使用名为 AUTOMAINT_SET_POLICY 和 AUTOMAINT_SET_POLICYFILE 的系统存储过程之一来配置自动备份。

DB2 pureScale环境中的备份和复原操作

在 DB2 pureScale环境中，如果在任何成员上发出单个 **BACKUP DATABASE** 或 **RESTORE DATABASE** 命令，那么将为所有成员启动备份或复原操作。

因为 DB2 pureScale环境只可以具有一个数据库分区，所以备份操作只需处理一组数据，并且只为整个组生成一个备份映像。在其他成员的情况下，只有数据库元数据和事务日志必须进行处理，并且这些数据会包括在单个备份映像中。

备份映像包括指定的表空间和任何必需的元数据中的数据，以及所有目前已定义的成员的配置信息。您不必对 DB2 pureScale实例中的任何其他成员执行其他备份操作。此外，您只需要单个 **RESTORE DATABASE** 命令来复原数据库及所有成员的特定于成员的元数据。您不必对任何其他成员执行其他复原操作来复原集群。连续备份映像的时间戳记是唯一的、持续增加的值，无论产生该备份映像的成员为何。

所有成员必须一致，才能尝试执行脱机备份操作。一次只能运行一个脱机备份操作，因为备份实用程序会获取所有成员对数据库的超级互斥访问权。尽管支持并发联机备份操作，但不同的备份操作不能同时复制相同的表空间，必须排队等待。

所有从数据库中读取数据和元数据的操作以及所有向备份映像写入数据的操作都发生在单个成员上。备份或复原操作与其他成员之间的交互限制为复制或更新数据库元数据（例如表空间定义、日志文件头以及数据库配置）。

注：备份之前，您需要确保日志归档路径设置为共享目录，以便所有成员都能访问日志以便执行后续前滚操作。如果无法从要在其上执行前滚的成员访问归档路径，那么会返回 SQL1273N。以下命令是将日志路径设置为共享目录的示例:

```
db2 update db cfg using logarchmeth1
DISK:/db2fs/gpfs1/svtdbm5/svtdbm5/ArchiveLOGS
```

（其中 *gpfs1* 是成员的共享目录，*ArchiveLOGS* 是归档这些日志的实际目录）。

如果另一个成员在执行联机备份操作时已脱机、变为脱机或重新联机，那么联机备份操作可以成功地继续执行 (第 270 页的表 17)。虽然数据库复原操作不受其他成员状态的影响，但是在脱机且不一致的成员上完成成员崩溃恢复时，备份操作可能需要等待一段时间。

表 17. DB2 pureScale实例中其他成员的状态对数据库备份和复原操作的影响

操作	其他成员的状态	
	脱机且一致	脱机且不一致
联机备份	备份操作成功。当备份实用程序在接近备份操作开头处访问日志文件头 (LFH) 时，或者当备份实用程序在接近备份操作末尾处访问日志流时，其他成员无法变成活动状态。	备份操作成功，但备份操作必须等待成员崩溃恢复完成并且其他成员变成活动状态或一致。当备份实用程序在接近备份操作开头处访问 LFH 时，或者当备份实用程序在接近备份操作末尾处访问日志流时，其他成员无法变成活动状态。
restore	复原操作正常完成。	复原操作正常完成。

映像和归档命名

在磁盘上创建的备份映像文件名由几个元素并置组成，这些元素由句点分隔:

DB_alias.*Type*.*Inst_name*.DBPART*nnn*.*Timestamp*.*Seq_num*

DB_alias
这是您在调用备份实用程序时所指定的数据库别名。

类型 这是备份操作的类型，其中 0 表示完整数据库备份，3 表示表空间备份，4 表示由带 **COPY** NO 选项的 **LOAD** 命令所生成的备份映像。

Inst_name
这是当前实例的名称，该名称是 **DB2INSTANCE** 环境变量的值。

nnn 数据库分区号。在 DB2 pureScale环境中，该号码始终为 000。

时间戳记
这是您执行备份操作时的日期和时间的 14 个字符表示法。该时间戳记的格式为 *yyyymmddhhnnss*，其中:

- *yyyy* 表示年份。
- *mm* 表示月份（01 到 12）。
- *dd* 表示该月份中的某一天（01 到 31）。
- *hh* 表示小时（00 到 23）。
- *nn* 表示分钟（00 到 59）。
- *ss* 表示秒（00 到 59）。

Seq_num
用作文件扩展名的一个 3 位的数字。

例如:

```
SAMPLE.0.krodger.DBPART000.200802241234.001
```

使用 INCLUDE LOGS 的联机备份

指定了 **INCLUDE LOGS** 选项（缺省值）的联机备份操作会生成一个备份映像，该备份映像包括将数据库复原并前滚至其最早恢复时间所需的日志文件范围。如果以后将此备份映像用于复原至新的数据库（可能在灾难恢复期间），并且在后续前滚操作期间只有此备份映像中的日志可用，那么带有 **TO END OF LOGS** 参数的 **ROLLFORWARD DATA-**

BASE 命令通常会返回有关丢失日志文件的错误消息 (SQL1273N)。在某些情况下，这是预期的结果，因为数据库管理器可能已检测到在备份操作之后写入了其他日志，但是当前前滚操作无法使用这些日志。也可能是将数据库及时前滚至一致时间点所必需的一个或多个日志丢失。在任何一种情况下，请验证前滚操作的结束点是否可以接受，然后发出带有 **AND STOP** 参数的 **ROLLFORWARD DATABASE**。无论是否丢失了日志文件，如果前滚操作已达到其最小恢复时间，那么带有 **AND STOP** 参数的 **ROLLFORWARD DATABASE** 应该会成功完成；否则，该命令会返回 SQL1276N（前滚操作使用此备份映像时未达到其最小恢复时间）。

在 DB2 pureScale环境中通过日志装入进行灾难恢复和获取高可用性

*日志装入*是将整个日志文件复制到备用机器上的过程，可以从归档设备上复制，也可以通过对主数据库运行的用户出口程序来复制。您可以选择将已归档的日志应用于备用数据库来使备用数据库保持最新，也可以将数据库或表空间备份映像以及日志归档保存在备用场所，而只在发生灾难后才执行复原和前滚操作。在任何一种情况下，备用场所上的前滚操作都可能会检测到一个或多个日志文件丢失，并返回 SQL1273N。验证前滚操作是否已达到可接受的时间戳记，或者执行适当的操作以解决问题。

如果在日志流合并操作期间，DB2 数据库管理器确定其中一个日志流中丢失日志文件，那么会返回错误。ROLLFORWARD 实用程序将返回 SQL1273N；db2ReadLog API 将返回 SQL2657N。如果选择将已归档的日志应用于数据库来使备用数据库保持最新，那么前滚操作可能会频繁检测到丢失了某些日志。

图 19 显示了一个示例，说明两个成员如何将日志记录写入其活动日志流中的日志文件。每个日志文件由一个框表示。请考虑主场所和备用场所都已设置高可用性的方案。在时间点 A、B 和 C 在备用场所尝试执行了带有 **END OF LOGS** 选项的 **ROLLFORWARD DATABASE** 命令。对于任何特定的时间点，已归档在该时间之前关闭的任何日志文件，并且可以在备用场所访问这些日志文件。否则，日志文件在主场所仍然处于活动状态并且对于备用场所尚不可用（如日志流 1 中的日志文件 4 在时间 B 的状态所示）。

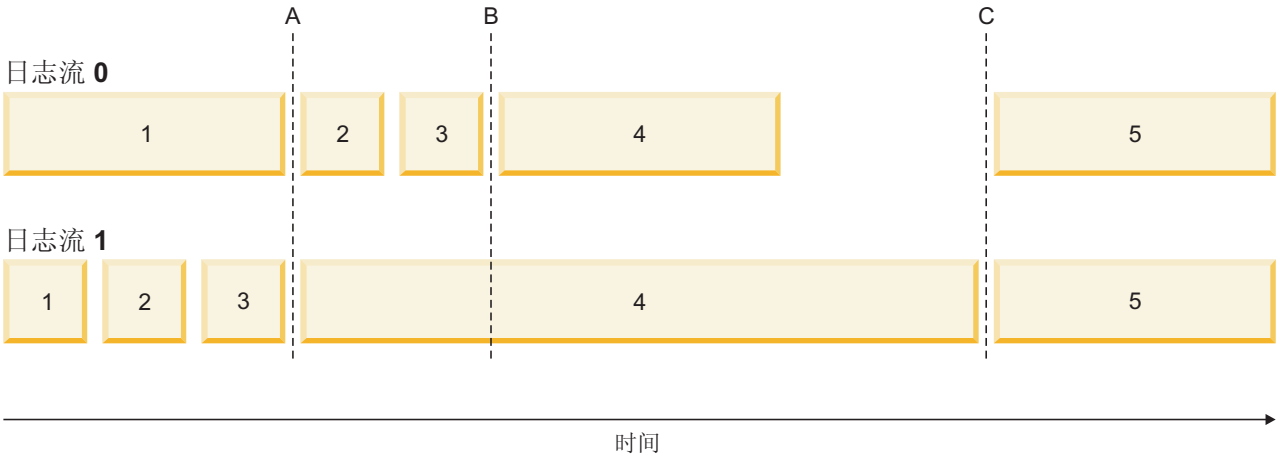


图 19. DB2 pureScale环境中的日志文件

在时间 A，**ROLLFORWARD DATABASE** 命令将成功完成，因为日志流 0 中的日志文件 1 已与日志流 1 中的日志文件 3 同时关闭并归档。然而，在时间 B，**ROLLFORWARD DATABASE** 命令将返回 v。发生此错误的原因如下：在备用场所发出该命令时，备用场所可以访问日志流 0 中的日志文件 2 和 3，但无法访问日志流 1 中的日志文件 4，这是因

为该日志文件在主场所仍然是打开的并处于活动状态。此外，因为在开始日志流 1 中的日志文件 4 的相同时间段写入了日志流 0 的文件 2 和 3 中的日志记录，所以在日志流 1 中的日志文件 4 可用之前，前滚操作无法处理日志文件 2 和 3。在时间 C，在日志流 1 中最终关闭并归档日志文件 4 时，**ROLLFORWARD DATABASE** 命令将成功完成。通过使用 **ARCHIVE LOG** 命令或者通过在所有成员上取消激活数据库，有可能强制截断和归档所有日志流中的文件。在使用 **ARCHIVE LOG** 命令的情况下，将独立截断每个日志流中的当前日志文件，而不保证将在所有成员上在完全相同的时间点进行截断。因此，即使发出了 **ARCHIVE LOG** 命令，在执行 **ROLLFORWARD DATABASE** 命令时，仍然可能发生 SQL1273N 错误。

虽然在 DB2 pureScale环境中使用日志装入时丢失日志的情况是常见和预期的，但在大多数情况下，备用场所上的每个前滚操作将在最后一个 **ROLLFORWARD DATABASE** 命令完成后取得其他进展（即使返回了 SQL1273N），因此应该预期经常会发生该错误本身。但是，主位置在成功归档其中一个日志流的日志时，可能会在归档另一个日志流的文件时遇到问题。这可能是访问一个日志流的归档存储器时发生的临时问题的结果。这样的问题会导致挂起备用场所上的日志合并和重放，从而增加发生灾难时可能丢失的事务数。要确保备用系统是最新的，请在每个返回 SQL1273N 的前滚操作之后发出带有 **QUERY STATUS** 参数的 **ROLLFORWARD DATABASE** 命令，并验证该命令是会随着时间的推移而取得进展。如果备用系统上的前滚操作在经过很长的时间段后未取得进展，请确定报告为丢失的日志文件在备用系统上不可用的原因，并解决该问题。**ARCHIVE LOG** 命令可用于截断目前正在对每个成员进行更新的日志文件，从而使这些日志文件适合于在备用系统上归档和后续重放。

如果发生灾难（例如火灾、地震、恶意破坏或其他灾难性事件），您的恢复计划可能是通过所有残存的日志来执行前滚操作，或者通过所有可用的日志来执行复原和前滚操作。如先前所述，前滚操作可能会检测到丢失了一个或多个日志文件，这是因为发生灾难时，这些日志文件已写入主系统，但尚未归档 (SQL1273N)。也可能是由于某些意外原因，导致 ROLLFORWARD 实用程序找不到已归档的日志；这也可能导致 ROLLFORWARD 实用程序返回 SQL1273N。通过使用带有 **QUERY STATUS** 参数的 **ROLLFORWARD DATABASE** 命令来验证前滚操作的结束点，并判定丢失日志的情况是否是预期的，这两个操作很重要。如果丢失日志的情况是预期的，或者结束点是可接受的，那么您可以发出带有 **STOP** 参数的 **ROLLFORWARD DATABASE** 命令来完成前滚恢复进程。

限制

在安装了 DB2 pureScale Feature的环境与尚未安装 DB2 pureScale Feature的环境之间，备份和复原操作不受支持。

在拓扑中发生了涉及添加或删除成员的更改之后，不能通过拓扑更改发生的时间点来执行前滚恢复操作。如果添加或删除成员，那么数据库将置于备份暂挂状态，您必须执行完整的数据库备份操作，才能连接到该数据库。要进行恢复，请复原此备份映像并前滚到日志末尾。如果必须在拓扑更改之前复原备份映像，那么您将只能前滚到发生拓扑更改的点。这可以通过先发出带有 **TO END OF LOGS** 的 **ROLLFORWARD DATABASE** 命令（该命令会返回 SQL1546N），然后发出带有 **STOP** 参数的 **ROLLFORWARD DATABASE** 命令来完成。此操作将不恢复拓扑更改之后更改数据库的任何事务。

在 DB2 pureScale环境中，**BACKUP DATABASE** 命令的 **ON ALL DBPARTITIONNUMS** 参数和 **ON DBPARTITION (0)** 参数是有效参数。如果您指定除 0 之外的数据库分区号，那么会返回错误 (SQL0270N)，因为不存在其他数据库分区。

此发行版存在下列限制:

- 可以将驻留在 DB2 pureScale环境外部的数据库迁移到 DB2 pureScale环境。不能使用数据库复原操作来将这样的数据库迁移到 DB2 pureScale环境。

示例

- 从任一成员备份具有 4 个成员的数据库（名为 SAMPLE）:

  ```
  BACKUP DB SAMPLE
  ```

- 复原具有 1 个成员的数据库（名为 SAMPLE）:

  ```
  RESTORE DB SAMPLE
  ```

- 使用 **RECOVER DATABASE** 命令来从任一成员复原和前滚名为 SAMPLE 的数据库:

  ```
  RECOVER DB SAMPLE TO END OF LOGS
  ```

  如果数据库不存在，请使用 **RESTORE DATABASE** 和 **ROLLFORWARD DATABASE** 命令，而不使用 **RECOVER DATABASE** 命令，因为 **RECOVER DATABASE** 命令的成功完成需要具有完整数据库历史记录的现有数据库。

监视备份操作

可使用 **LIST UTILITIES** 命令来监视数据库上的备份操作的进度。

过程

发出 **LIST UTILITIES** 命令并指定 **SHOW DETAIL** 参数:

```
list utilities show detail
```

结果

对于备份操作，将指定要处理的最初估计字节数。随备份操作进度更新要处理的字节数。显示的字节与映像的大小不相等，不应该用作备份映像大小的估计值。视映像是增量备份，还是压缩备份而定，实际映像可能小很多。

示例

以下是用于监视脱机数据库备份操作性能的输出的示例:

```
ID                                 = 3
Type                               = BACKUP
Database Name                      = SAMPLE
Partition Number     = 0
Description                        = offline db
Start Time                         = 08/04/2011 12:16:23.248367
State                = Executing
Invocation Type                    = User
Throttling:
   Priority                        = Unthrottled
Progress Monitoring:
   Estimated Percentage Complete = 31
      Total Work                   = 123147277 bytes
      Completed Work               = 37857269 bytes
      Start Time                   = 08/04/2011 12:16:23.248377
```

优化备份性能

执行备份操作时，DB2 数据库管理器会自动为缓冲区个数、缓冲区大小和并行性设置选择最佳值。这些值根据可用实用程序堆内存的数量、可用处理器数和数据库配置而定。

因此，根据系统上可用的存储量，应考虑通过增大 **util_heap_sz** 配置参数来分配更多内存。

目的是最大程度上减少完成备份操作所需的时间。除非显式地输入以下 **BACKUP DATABASE** 命令参数的值，否则 DB2 数据库管理器将为它们选择一个值:

- **WITH** *num-buffers* **BUFFERS**
- **PARALLELISM** *n*
- **BUFFER** *buffer-size*

如果未指定缓冲区数和缓冲区大小而导致 DB2 数据库管理器设置这些值，那么对大型数据库的影响应该最小。但是，对于小型数据库来说，会导致备份映像大幅增大。即使写入磁盘的最后一个数据缓冲区只包含很少数据，也会将整个缓冲区写入映像。在小型数据库中，这表示相当一部分的映像可能为空。

还可以选择执行以下任何操作来缩短完成一次备份操作所需的时间:

- 指定表空间备份。

  使用 **BACKUP DATABASE** 命令的 **TABLESPACE** 选项，可以备份（继而恢复）部分数据库。这样便于对表数据、索引和单独表空间中的长字段或大对象 (LOB) 数据进行管理。
- 增大 **BACKUP DATABASE** 命令上 **PARALLELISM** 参数的值，以使它反映正在备份的表空间数。

  PARALLELISM 参数定义在压缩备份操作期间从数据库读取数据和压缩数据时，已启动的进程或线程数。将每个进程或线程分配给特定表空间，因此，为 **PARALLELISM** 参数指定的值大于要备份的表空间数并无益处。备份完此表空间后，它会请求另一个表空间。但是应注意: 每个进程或线程都需要内存和 CPU 开销。
- 增加备份缓冲区大小。

  理想的备份缓冲区大小是表空间扩展数据块大小的倍数加一页。如果有多个扩展数据块大小不同的表空间，那么将值指定为扩展数据块大小的公倍数加一页。
- 增加缓冲区的数量。

  使用的缓冲区至少是备份目标（或会话）的两倍，以确保备份目标设备无需等待数据。
- 使用多个目标设备。

备份和复原统计信息

每次成功执行备份和复原操作都会在 db2diag.log 文件中生成单条记录，此记录提供了有关该操作的性能的信息。此日志记录可供参考，并按 diaglevel 3（这是缺省值）和 4 进行转储。

示例

备份和复原统计信息的日志记录由每个备份和复原缓冲区操纵程序 (db2bm) EDU 的相应一行和每个备份和复原媒体控制器 (db2med) EDU 的相应一行组成。

```
2012-07-30-15.41.30.012922-240 E15775E1464           LEVEL: Info
PID     : 15882                TID : 46913126656320  KTID : 16001
PROC    : db2sysc
INSTANCE: krodger              NODE : 000            DB   : SAMPLE
APPHDL  : 0-18                 APPID: *LOCAL.krodger.120730194119
AUTHID  : KRODGER              HOSTNAME: hotel74
EDUID   : 49                   EDUNAME: db2agent (SAMPLE)
FUNCTION: DB2 UDB, database utilities, sqluxLogDataStats, probe:377
MESSAGE : Performance statistics
DATA #1 : String, 951 bytes

Number of buffers = 4
Buffer size       = 16781312 (4097 4kB pages)

BM#    Total      I/O      MsgQ      WaitQ      Buffers   kBytes
---  --------  --------  --------  --------    --------  --------
000      6.30      0.02      0.00      6.18           4       640
001      5.88      4.48      0.00      1.33           9    139536
---  --------  --------  --------  --------    --------  --------
TOT     12.18      4.51      0.00      7.51          13    140176

MC#    Total      I/O      MsgQ      WaitQ      Buffers   kBytes
---  --------  --------  --------  --------    --------  --------
000      6.36      0.34      5.94      0.00           9    114748
001      6.29      0.18      5.60      0.10           6     81944
---  --------  --------  --------  --------    --------  --------
TOT     12.66      0.53     11.55      0.10          15    196692
```

各列的含义为如下所示:

BM db2bm EDU 标识

Total
每个 EDU 已存在的时间长度

I/O
对稳定存储器执行读/写 I/O 操作所耗用的时间

MsgQ
等待 I/O 缓冲区所耗用的时间

WaitQ
等待状态机控制消息所耗用的时间

Buffers
已处理的 I/O 缓冲区数

KBytes
已处理的数据的数量

MC db2med EDU 标识

示例

对于压缩备份，日志记录中还包含另外两列，用于存储有关该压缩操作的性能信息:

```
2012-07-30-15.41.47.228766-240 E38419E1913           LEVEL: Info
PID     : 15882                TID : 46913126656320  KTID : 16081
PROC    : db2sysc
```

```
INSTANCE: krodger              NODE : 000           DB   : SAMPLE
APPHDL  : 0-29                 APPID: *LOCAL.krodger.120730194132
AUTHID  : KRODGER              HOSTNAME: hotel74
EDUID   : 80                   EDUNAME: db2agent (SAMPLE)
FUNCTION: DB2 UDB, database utilities, sqluxLogDataStats, probe:377
MESSAGE : Performance statistics
DATA #1 : String, 1399 bytes

Number of buffers = 4
Buffer size       = 16781312 (4097 4K pages)

                                                                                  Compr
BM#    Total      I/O      Compr     MsgQ      WaitQ      Buffers   kBytes     kBytes
---  --------  --------  --------- --------  --------    --------  --------  ---------
000     12.08      4.36      7.18      0.00      0.37           5    139536     144941
001     11.87      0.01      0.01      0.00     11.79           1       640        640
---  --------  --------  --------- --------  --------    --------  --------  ---------
TOT     23.96      4.38      7.19      0.00     12.17           6    140176     145581

MC#    Total      I/O               MsgQ      WaitQ      Buffers   kBytes
---  --------  --------            --------  --------    --------  --------
000     12.07      0.11              11.76      0.00           4     49168
001     12.10      0.07              11.84      0.15           4     32808
---  --------  --------            --------  --------    --------  --------
TOT     24.18      0.19              23.61      0.15           8     81976
```

Compr
执行该压缩操作所耗用的时间

Compr Bytes
已压缩的先前未压缩的数据的数量

使用 backup 所需的特权、权限和授权

必须具有 SYSADM、SYSCTRL 或 SYSMAINT 权限才能使用 BACKUP 实用程序。

特权使用户能够创建或访问数据库资源。权限级别提供了对特权、较高级别数据库管理器维护和实用程序操作进行分组的方法。这两者一起用于控制对数据库管理器及其数据库对象的访问。

用户只能访问那些他们具有相应授权（即必需的特权或权限）的对象。

联机备份与其他实用程序的兼容性

某些实用程序可与联机备份实用程序同时运行，但是另一些实用程序却不行。

下列实用程序与联机备份实用程序兼容:

- **EXPORT**
- **INSPECT**

下列 SQL 语句和实用程序仅在某些情况下与联机备份兼容:

- CREATE INDEX

  在 SMS 方式下，因为 ALTER TABLE 锁定，联机索引创建与联机备份不会并行运行。联机索引创建实用程序以互斥方式获取该锁，而联机备份实用程序以共享方式获取该锁。

在 DMS 方式下，大多数情况下可以并发运行联机索引创建实用程序和联机备份实用程序。如果您在其中创建该索引的表空间中有大量表，那么联机索引创建将在内部获取联机备份锁定，此锁定将与任何并行联机备份发生冲突。

- REORG INDEX 带有 ONLINE 选项

  与联机索引创建一样，在 SMS 方式下，因为 ALTER TABLE 锁定，联机索引重组不会与联机备份并行运行。联机索引重组以互斥方式获取该锁定，而联机备份以共享方式获取该锁定。另外，联机索引重组操作在切换阶段前使表停顿并获取 Z 锁定，这将导致联机备份实用程序无法运行。但是，在获取 Z 表锁之前，ALTER TABLE 锁应防止同时运行联机备份实用程序。

  在 DMS 方式下，可以并发运行联机索引重组实用程序和联机备份实用程序。

  另外，联机索引重组实用程序在切换阶段前将使表停顿并获取 Z 锁定，这将导致联机备份实用程序无法运行。

- **IMPORT**

  导入实用程序与联机备份兼容，但以下情况除外：在发出带 **REPLACE** 参数的 **IMPORT** 命令时；在此情况下，导入实用程序将获取表的 Z 锁定并防止联机备份实用程序无法并发运行。

- TRUNCATE TABLE

  TRUNCATE 语句不与联机备份兼容，因为它在该表上获得 Z 锁定，并使联机备份无法正常运行。

- **ALLOW READ ACCESS LOAD**

  当发出指定了 **COPY NO** 参数的 **LOAD** 命令时，**ALLOW READ ACCESS** 装入操作与联机备份不兼容。在此方式下，两个实用程序都修改表空间状态，从而导致其中一个实用程序报告错误。

  当发出指定了 **COPY YES** 选项的 **LOAD** 命令时，虽然仍可能有一些兼容性问题，但 **ALLOW READ ACCESS** 装入操作与联机备份兼容。在 SMS 方式下，这两个实用程序可以并发执行，但它们将会采用不兼容的表锁定方式，从而可能会遇到要等待锁定表情况。在 DMS 方式下，这两个实用程序都采用不兼容的“内部 B”(OLB) 锁定方式，并可能遇到要等待该锁定的情况。如果这两个实用程序在同一个表空间上并发执行，那么装入实用程序可能必须等待备份实用程序完成表空间处理之后才能继续运行。

- REORG TABLE 带有 ONLINE 选项

  当联机备份实用程序运行时，无法启动联机表重组实用程序的清除阶段。根据需要，可暂停表重组以允许联机备份完成，然后继续运行联机表重组。

  对于 DMS 表空间来说，可以在对同一表空间中的表执行联机重组时启动 DMS 表空间联机备份。在截断阶段，可能会出现与重组操作相关联的锁定等待情况。

  对于 SMS 表空间来说，不能在对同一表空间中的表执行联机重组时启动 SMS 表空间联机备份。这两个操作都需要获取互斥锁定。

- 需要 Z 锁定的 DDL（例如 ALTER TABLE、DROP TABLE 和 DROP INDEX）

  联机 DMS 表空间备份与需要 Z 锁定的 DDL 兼容。

联机 SMS 表空间备份必须等待 Z 锁定被释放。

- 存储器组 DDL

  如果要通过发出下列其中一个语句来修改数据库存储器组，那么应注意使此操作与联机备份时间表协调:

  – CREATE STOGROUP
  – ALTER STOGROUP
  – DROP STOGROUP
  – RENAME STOGROUP
  – ALTER DATABASE

  如果正在进行联机备份，那么存储器组 DDL 将在该操作后等待直到它可获取适当锁定，这可能要消耗很长时间。同样，联机备份会在正在进行的存储器组 DDL 后等待直到该 DDL 落实或回滚。

- **RUNSTATS** 带有 ALLOW WRITE 或 ALLOW READ 选项

  RUNSTATS 命令与联机备份兼容，但系统目录表空间为 SMS 表空间时除外。如果系统目录驻留在 SMS 表空间中，那么 **RUNSTATS** 命令和联机备份将挂起对表的不兼容表锁定，从而导致锁定等待。

- ALTER TABLESPACE

  在表空间的联机备份期间，不允许执行用于启用/禁用自动重新调整大小功能或改变自动重新调整大小容器功能的操作。

- ALTER TABLESPACE 带有 REBALANCE 选项

  联机备份和重新平衡程序并行运行时，联机备份暂停重新平衡程序并且不等待重新平衡程序完成。

下列实用程序与联机备份实用程序不兼容:

- **REORG TABLE**
- **RESTORE DATABASE**
- **ROLLFORWARD DATABASE**
- **LOAD** 带有 **ALLOW NO ACCESS** 选项
- **SET WRITE**
- **BACKUP DATABASE** 带有 ONLINE 选项

  这适用于数据库级别的联机备份和表空间级别的联机备份（如果它们涉及相同的一个或多个表空间）。

备份示例

本主题包含不同备份策略的一些示例。

备份到 TSM

在以下示例中，数据库 SAMPLE 被备份至使用 2 个并发 TSM 客户机会话的 TSM 服务器上。BACKUP 实用程序将计算最佳的缓冲区数。根据内存量和可用的目标设备数自动计算出最佳缓冲区大小（以 4 KB 页计）。也可以根据可用处理器数和要备份的表空间数自动计算并行性设置。

```
db2 backup database sample use tsm open 2 sessions with 4 buffers

db2 backup database payroll tablespace (syscatspace, userspace1) to
      /dev/rmt0, /dev/rmt1 with 8 buffers without prompting
```

增量备份

以下是对可恢复数据库每周进行的增量备份策略的样本。包括一个每周进行的完整数据库备份操作、一个每天进行的非累积（差异）备份操作以及一个在每周中期进行的累积（增量）备份操作:

```
(Sun) db2 backup db kdr use tsm
   (Mon) db2 backup db kdr online incremental delta use tsm
   (Tue) db2 backup db kdr online incremental delta use tsm
   (Wed) db2 backup db kdr online incremental use tsm
   (Thu) db2 backup db kdr online incremental delta use tsm
   (Fri) db2 backup db kdr online incremental delta use tsm
   (Sat) db2 backup db kdr online incremental use tsm
```

备份到磁带

要在 Windows 环境中对磁带设备启动备份操作，发出:

```
db2 backup database sample to \\.\tape0
```

第 12 章 recover 概述

恢复实用程序执行必需的复原和前滚操作，以根据恢复历史记录文件中找到的信息将数据库恢复到指定时间。

当使用此实用程序时，指定将数据库恢复至时间点或日志文件的结尾。然后，实用程序将选择最适合的备份映像并执行恢复操作。

在 IBM Data Studio V3.1 或更高版本中，可以使用以下工具的任务助手：恢复数据库. 任务助手可以指导您执行以下过程：设置选项、查看自动生成的命令以执行任务以及运行这些命令。有关更多详细信息，请参阅使用任务助手管理数据库。

恢复实用程序不支持以下 **RESTORE DATABASE** 命令选项:

- **TABLESPACE** *tablespace-name*。不支持表空间复原操作。
- **INCREMENTAL**。不支持增量复原操作。
- **OPEN** *num-sessions* **SESSIONS**。无法指示将与 TSM 或其他供应商产品配合使用的 I/O 会话数。
- **BUFFER** *buffer-size*。无法设置用于复原操作的缓冲区大小。
- **DLREPORT** *filename*。无法指定用于报告文件变成取消链接的文件名。
- **WITHOUT ROLLING FORWARD**。无法指定在成功复原操作后，不要将该数据库置于前滚暂挂状态。
- **PARALLELISM** *n*。无法指示复原操作的并行度。
- **WITHOUT PROMPTING**。无法指定复原操作以无人照管方式运行。

此外，恢复实用程序不允许指定任何 **REBUILD** 选项。但是，如果恢复实用程序无法根据恢复历史记录文件中的信息来找到任何数据库备份映像，它就会自动使用适当的 **REBUILD** 选项。

对于 **RECOVER DATABASE** 命令，不能使用 TABLESPACE 选项或 **RESTORE DATABASE** 命令中的 INCREMENTAL 选项。

对于 **RECOVER DATABASE** 命令，会自动使用 restore 选项。此规则同样适用于 **RESTORE** 命令中的 REBUILD 选项。

恢复数据

RECOVER DATABASE 命令通过使用恢复历史记录文件中的信息将数据库和所有存储器组恢复至指定时间。

开始之前

对于在前滚阶段结束的未完成恢复操作，如果随后发出 **RECOVER DATABASE** 命令，那么恢复实用程序将尝试继续执行上一次恢复操作，而不会重新完成复原阶段。如果要强制恢复实用程序重新完成复原阶段，请发出带 **RESTART** 选项的 **RECOVER DATABASE** 命

© Copyright IBM Corp. 2001, 2013

令，以强制恢复实用程序忽略先前任何未能完成的恢复操作。如果使用的是应用程序编程接口 (API)，请为 **iRecoverAction** 字段指定调用程序操作 DB2RECOVER_RESTART 以强制恢复实用程序重新执行复原阶段。

如果在复原阶段中断了 **RECOVER DATABASE** 命令，那么无法继续执行该命令。必须重新发出 **RECOVER DATABASE** 命令。

不应连接到要恢复的数据库：恢复数据库实用程序会自动建立与指定数据库的连接，并且此连接将在恢复操作完成时终止。

关于此任务

数据库可以是本地数据库或远程数据库。

注： 在分区数据库环境中，必须从数据库的目录分区调用恢复实用程序。

过程

要调用恢复实用程序，请使用：

- **RECOVER DATABASE** 命令，或
- db2Recover 应用程序编程接口 (API)。

示例

以下示例显示如何通过 CLP 来使用 **RECOVER DATABASE** 命令：

```
db2 recover db sample
```

使用 db2adutl 来恢复数据

可以使用带有 **logarchopt1** 和 **vendoropt** 数据库配置参数的 **db2adutl** 命令来执行跨节点恢复。来自一些不同 Tivoli Storage Manager (TSM) 环境中的示例演示了如何执行此恢复。

在下列示例中，计算机 1 名为 bar，它正在运行 AIX 操作系统。此机器上的用户是 roecken。bar 上的数据库名为 zample。计算机 2 名为 dps。此计算机也在运行 AIX 操作系统，且用户是 regress9。

示例 1：TSM 服务器自动管理密码（PASSWORDACCESS 选项设置为 GENERATE）

此跨节点恢复示例显示如何设置两台计算机，以便当日志归档和备份都存储在 TSM 服务器上且在该服务器上使用 PASSWORDACCESS=GENERATE 选项管理密码时，您可以将数据从一台计算机恢复至另一台计算机。

注： 在更新数据库配置之后，可能需要对数据库进行脱机备份。

1. 要启用数据库以将 bar 计算机的日志归档到 TSM 服务器，请使用下列命令更新 zample 数据库的数据库配置参数 **logarchmeth1**：

   ```
   bar:/home/roecken> db2 update db cfg for zample using LOGARCHMETH1 tsm
   ```

   将返回以下信息：

   ```
   成功完成 DB20000I UPDATE DATABASE CONFIGURATION 命令。
   ```

2. 使用下列命令从数据库断开所有用户和应用程序的连接：

```
db2 force applications all
```

3. 使用下列命令验证是否已没有应用程序连接到数据库:

   ```
   db2 list applications
   ```

   您应该会接收到一条消息，说明未返回任何数据。

   注: 在分区数据库环境中，必须对所有数据库分区都执行此步骤。

4. 使用下列命令在 TSM 服务器上创建数据库备份:

   ```
   db2 backup db zample use tsm
   ```

   将返回类似以下的信息:

   ```
   Backup successful. The timestamp for this backup imagge is : 20090216151025
   ```

   注: 在分区数据库环境中，必须对所有数据库分区都执行此步骤。根据您正在执行联机备份还是脱机备份，在数据库分区上执行此步骤的顺序有所不同。有关更多信息，请参阅第 259 页的『备份数据』。

5. 使用下列命令连接至 zample 数据库:

   ```
   db2 connect to zample
   ```

6. 通过使用下列命令创建一个表并将数据装入 TSM 服务器来生成数据库的新事务日志:

   ```
   bar:/home/roecken> db2 load from mr of del modified by noheader replace
      into employee copy yes use tsm
   ```

   在此示例中，表名为 employee，并且正在从名为 mr 的定界 ASCII 文件中装入数据。指定了 **COPY** YES 选项以生成所装入的数据的副本，并且 **USE TSM** 选项指定该数据的副本存储在 TSM 服务器上。

   注: 仅当数据库启用了前滚恢复功能时才能指定 **COPY** YES 选项；即，必须将 **logarchmeth1** 数据库配置参数设置为 USEREXIT、LOGRETAIN、DISK 或 TSM。

   为了指示它的进度，装入实用程序将返回一系列消息:

   ```
   SQL3109N  实用程序正开始从"/home/roecken/mr"文件中装入数据。
   SQL3500W  实用程序正在开始"LOAD"阶段，开始时间为"02/16/2009
   15:12:13.392633"。

   SQL3519W  开始装入一致点。输入记录计数 ="0"。

      SQL3520W  装入一致点成功。
   SQL3110N  实用程序已完成处理。从输入文件
   读取了"1"行。

   SQL3519W  开始装入一致点。输入记录数 ="1"。

      SQL3520W  装入一致点成功。
   SQL3515W 实用程序已完成"LOAD"阶段，完成时间为"02/16/2009
   15:12:13.445718"。


         读取的行数        = 1
      跳过的行数        = 0
      装入的行数        = 1
      拒绝的行数        = 0
      删除的行数        = 0
      落实的行数        = 1
   ```

7. 将数据装入表中后，请通过在 zample 数据库中运行下列查询来确认在 TSM 服务器上存在一个备份映像、一个装入副本映像和一个日志文件:

   ```
   bar:/home/roecken/sqllib/adsm> db2adutl query db zample
   ```

将返回以下信息:

```
正在检索 FULL DATABASE BACKUP 信息。
 1 时间: 20090216151025  最早的日志: S0000000.LOG  日志流: 0
 会话数: 1

正在检索 INCREMENTAL DATABASE BACKUP 信息。
 找不到 ZAMPLE 的 INCREMENTAL DATABASE BACKUP 映像

正在检索 DELTA DATABASE BACKUP 信息。
 找不到 ZAMPLE 的 DELTA DATABASE BACKUP 映像

正在检索 TABLESPACE BACKUP 信息。
 找不到 ZAMPLE 的 TABLESPACE BACKUP 映像

正在检索 INCREMENTAL TABLESPACE BACKUP 信息。
 找不到 ZAMPLE 的 INCREMENTAL TABLESPACE BACKUP 映像

正在检索 DELTA TABLESPACE BACKUP 信息。
 找不到 ZAMPLE 的 DELTA TABLESPACE BACKUP 映像

正在检索 LOAD COPY 信息。
 1 时间:  20090216151213

正在检索 LOG ARCHIVE 信息。
日志文件: S0000000.LOG, 链号: 0, 日志流: 0,
生成时间: 2009-02-16-15.10.38
```

8. 要启用跨节点恢复，必须允许另一台计算机和帐户访问与 bar 计算机相关联的对象。在此示例中，使用下列命令允许计算机 dps 和用户 regress9 进行访问:

```
bar:/home/roecken/sqllib/adsm> db2adutl grant user regress9
        on nodename dps for db zample
```

将返回以下信息:

```
成功添加了 regress9 访问 dps 节点上的 ZAMPLE 的许可权。
```

注: 您可以通过发出下列命令检索当前节点的当前访问列表来确认 **db2adutl** 授权操作的结果:

```
bar:/home/roecken/sqllib/adsm> db2adutl queryaccess
```

将返回以下信息:

```
节点                 用户名             数据库名称    类型
-----------------------------------------------------------
  DPS                regress9           ZAMPLE         A
-----------------------------------------------------------
   访问类型:    B - 备份映像    L - 日志    A - 同时使用这两种访问类型
```

9. 在此示例中，尚未设置计算机 2 dps 以用于 zample 数据库的跨节点恢复。使用下列命令验证在 TSM 服务器上是否已没有数据与此用户和计算机相关联:

```
dps:/home/regress9/sqllib/adsm> db2adutl query db zample
```

将返回以下信息:

```
  --- 数据库目录是空的 ---
警告: 在 ADSM 服务器上 DB2 没有创建文件空间
警告: 在 ADSM 中找不到任何别名的 DB2 备份映像。
```

10. 使用下列命令查询 TSM 服务器以获得与用户 roecken 和计算机 bar 相关联的 zample 数据库的对象列表:

```
dps:/home/regress9/sqllib/adsm> db2adutl query db zample nodename
           bar owner roecken
```

将返回以下信息:

```
--- 数据库目录是空的 ---

对 ZAMPLE 数据库的查询

正在检索 FULL DATABASE BACKUP 信息。
 1 时间: 20090216151025  最早的日志: S0000000.LOG  日志流: 0
 会话数: 1

正在检索 INCREMENTAL DATABASE BACKUP 信息。
 找不到 ZAMPLE 的 INCREMENTAL DATABASE BACKUP 映像

正在检索 DELTA DATABASE BACKUP 信息。
 找不到 ZAMPLE 的 DELTA DATABASE BACKUP 映像

正在检索 TABLESPACE BACKUP 信息。
 找不到 ZAMPLE 的 TABLESPACE BACKUP 映像

正在检索 INCREMENTAL TABLESPACE BACKUP 信息。
 找不到 ZAMPLE 的 INCREMENTAL TABLESPACE BACKUP 映像

正在检索 DELTA TABLESPACE BACKUP 信息。
 找不到 ZAMPLE 的 DELTA TABLESPACE BACKUP 映像

正在检索 LOAD COPY 信息。
 1 时间:  20090216151213

正在检索 LOG ARCHIVE 信息。
日志文件: S0000000.LOG, 链号: 0, 日志流: 0,
生成时间: 2009-02-16-15.10.38
```

此信息与先前生成的 TSM 信息匹配，并确认可以将此映像复原到 dps 计算机上。

11. 使用下列命令将 zample 数据库从 TSM 服务器复原至 dps 计算机:

```
dps:/home/regress9> db2 restore db zample use tsm options
"'-fromnode=bar -fromowner=roecken'" without prompting
```

将返回以下信息:

```
DB20000I  已成功完成 RESTORE DATABASE 命令。
```

注: 如果 dps 上已经存在 zample 数据库，那么将忽略 **OPTIONS** 参数，而使用数据库配置参数 **vendoropt**。此配置参数将覆盖备份或复原操作的 **OPTIONS** 参数。

12. 执行前滚操作以应用在创建新表并装入新数据时记录在 zample 数据库日志文件中的事务。在此示例中，因为未指定用户和计算机信息而导致 ROLLFORWARD 实用程序找不到日志文件，所以前滚操作的下列尝试将失败:

```
dps:/home/regress9> db2 rollforward db zample to end of logs and stop
```

该命令会返回下列错误:

```
SQL4970N  数据库"ZAMPLE"上的前滚恢复不能达到指定的停止点（日志结束
或时间点），原因是在节点"0"上丢失了日志文件。
```

使用适当的 **logarchopt** 值强制 ROLLFORWARD 实用程序查找与另一台计算机相关联的日志文件。在此示例中，使用下列命令设置 **logarchopt1** 数据库配置参数并搜索与用户 roecken 和计算机 bar 相关联的日志文件:

```
dps:/home/regress9> db2 update db cfg for zample using logarchopt1
"'-fromnode=bar -fromowner=roecken'"
```

13. 通过使用下列命令来设置 **vendoropt** 数据库配置参数，使 ROLLFORWARD 实用程序能够使用备份和装入副本映像:

```
dps:/home/regress9> db2 update db cfg for zample using VENDOROPT
"'-fromnode=bar -fromowner=roecken'"
```

14. 使用下列命令通过应用记录在 zample 数据库日志文件中的事务来完成跨节点数据恢复:

```
dps:/home/regress9> db2 rollforward db zample to end of logs and stop
```

将返回以下信息:

```
                                前滚状态

输入数据库别名                  = zample
返回了状态的成员数              = 1

成员数        前滚                    要读取的              已处理的日志文件          上次落实的事务
              状态                      下一个日志
------------  ----------  ----------  ----------------------  ----------------------------
       0      未暂挂            S0000000.LOG-S0000000.LOG  2009-05-06-15.28.11.000000 UTC

DB20000I  已成功完成 ROLLFORWARD 命令。
```

已将计算机 dps 上用户 regress9 下的数据库 zample 恢复到计算机 bar 上用户 roecken 下数据库的同一位置。

示例 2: 密码由用户管理（PASSWORDACCESS 选项设置为 PROMPT）

此跨节点恢复示例显示如何设置两台计算机，以便当日志归档和备份都存储在 TSM 服务器上且在该服务器上由用户管理密码时，您可以将数据从一台计算机恢复至另一台计算机。在这些环境中，需要额外的信息，特别是用于创建对象的计算机的 TSM 节点名和密码。

1. 通过添加下列行来更新客户机 dsm.sys 文件，因为计算机 bar 是源计算机的名称

   ```
   NODENAME bar
   ```

   注: 在 Windows 操作系统上，此文件名为 dsm.opt 文件。更新此文件时，必需重新引导系统才能使更改生效。

2. 使用下列命令查询 TSM 服务器以获得与用户 roecken 和计算机 bar 相关联的对象列表:

   ```
   dps:/home/regress9/sqllib/adsm> db2adutl query db zample nodename bar
      owner roecken password *******
   ```

   将返回以下信息:

   ```
   对 ZAMPLE 数据库的查询

      正在检索 FULL DATABASE BACKUP 信息。
       1 时间: 20090216151025  最早的日志: S0000000.LOG  日志流: 0
       会话数: 1

      正在检索 INCREMENTAL DATABASE BACKUP 信息。
       找不到 ZAMPLE 的 INCREMENTAL DATABASE BACKUP 映像

      正在检索 DELTA DATABASE BACKUP 信息。
       找不到 ZAMPLE 的 DELTA DATABASE BACKUP 映像

      正在检索 TABLESPACE BACKUP 信息。
       找不到 ZAMPLE 的 TABLESPACE BACKUP 映像

      正在检索 INCREMENTAL TABLESPACE BACKUP 信息。
       找不到 ZAMPLE 的 INCREMENTAL TABLESPACE BACKUP 映像

      正在检索 DELTA TABLESPACE BACKUP 信息。
       找不到 ZAMPLE 的 DELTA TABLESPACE BACKUP 映像

     正在检索 LOAD COPY 信息。
       1 时间:  20090216151213
   ```

```
正在检索 LOG ARCHIVE 信息。
日志文件: S0000000.LOG, 链号: 0, 日志流: 0,
生成时间: 2009-02-16-15.10.38
```

3. 如果在计算机 dps 上不存在 zample 数据库，那么请执行下列步骤:
   a. 使用下列命令创建空数据库 zample:

      ```
      dps:/home/regress9> db2 create db zample
      ```

   b. 使用下列命令来更新数据库配置参数 **tsm_nodename**:

      ```
      dps:/home/regress9> db2 update db cfg for zample using tsm_nodename bar
      ```

   c. 使用下列命令来更新数据库配置参数 **tsm_password**:

      ```
      dps:/home/regress9> db2 update db cfg for zample using
      tsm_password ********
      ```

4. 尝试使用下列命令复原 zample 数据库:

   ```
   dps:/home/regress9> db2 restore db zample use tsm options
   "'-fromnode=bar -fromowner=roecken'" without prompting
   ```

   成功完成复原操作，但是发出了一条警告:

   ```
   SQL2540W  复原成功，但是在以"无中断"方式进行处理时，在"数据库复原"期间
   遇到了警告"2523"。
   ```

5. 使用下列命令来执行前滚操作:

   ```
   dps:/home/regress9> db2 rollforward db zample to end of logs and stop
   ```

   在此示例中，由于复原操作替换了数据库配置文件，因此 ROLLFORWARD 实用程序找不到正确的日志文件并且将返回下列错误消息:

   ```
   SQL1268N  由于检索节点"0"上的数据库"ZAMPLE"的日志文件"S0000000.LOG"时
   发生错误"-2112880618"，前滚恢复已停止。
   ```

   请将下列 TSM 数据库配置值重置为正确的值:
   a. 使用下列命令来设置 **tsm_nodename** 配置参数:

      ```
      dps:/home/regress9> db2 update db cfg for zample using tsm_nodename bar
      ```

   b. 使用下列命令来设置 **tsm_password** 数据库配置参数:

      ```
      dps:/home/regress9> db2 update db cfg for zample using tsm_password *******
      ```

   c. 使用下列命令来设置 **logarchopt1** 数据库配置参数，以便 ROLLFORWARD 实用程序可以找到正确的日志文件:

      ```
      dps:/home/regress9> db2 update db cfg for zample using logarchopt1
      "'-fromnode=bar -fromowner=roecken'"
      ```

   d. 使用下列命令来设置 **vendoropt** 数据库配置参数，以便在前滚操作期间也可以使用装入恢复文件:

      ```
      dps:/home/regress9> db2 update db cfg for zample using VENDOROPT
      "'-fromnode=bar -fromowner=roecken'"
      ```

6. 通过使用下列命令来执行前滚操作，可以完成跨节点恢复:

   ```
   dps:/home/regress9> db2 rollforward db zample to end of logs and stop
   ```

   将返回以下信息:

   ```
                                  前滚状态

   输入数据库别名                 = zample
   返回了状态的成员数             = 1

   成员数        前滚                要读取的                已处理的日志文件            上次落实的事务
                 状态                下一个日志
   ------------  ----------  ----------  ----------------------  ----------------------------
   ```

```
                0      未暂挂               S0000000.LOG-S0000000.LOG  2009-05-06-15.28.11.000000 UTC

DB20000I  已成功完成 ROLLFORWARD 命令。
```

已将计算机 dps 上用户 regress9 下的数据库 zample 恢复到计算机 bar 上用户 roecken 下数据库的同一位置

示例 3: TSM 服务器已配置为使用客户机代理节点

此跨节点恢复示例显示如何将两台计算机设置为代理节点，以便当日志归档和备份都存储在 TSM 服务器上且在该服务器上使用 PASSWORDACCESS=GENERATE 选项管理密码时，您可以将数据从一台计算机恢复至另一台计算机。

注: 在更新数据库配置之后，可能需要对数据库进行脱机备份。

在此示例中，计算机 bar 和 dps 注册在 clusternode 代理名称下。这些计算机已经设置为代理节点。

1. 使用下列命令将计算机 bar 和 dps 在 TSM 服务器上注册为代理节点:

   ```
   REGISTER NODE clusternode mypassword
   GRANT PROXYNODE TARGET=clusternode AGENT=bar,dps
   ```

2. 要启用数据库以将日志归档到 TSM 服务器，请使用下列命令更新 zample 数据库的数据库配置参数 **logarchmeth1**:

   ```
   bar:/home/roecken> db2 update db cfg for zample using
      LOGARCHMETH1 tsm logarchopt1 "'-asnodename=clusternode'"
   ```

   将返回以下信息:

   ```
   成功完成 DB20000I UPDATE DATABASE CONFIGURATION 命令。
   ```

3. 使用下列命令从数据库断开所有用户和应用程序的连接:

   ```
   db2 force applications all
   ```

4. 使用下列命令验证是否已没有应用程序连接到数据库:

   ```
      db2 list applications
   ```

   您应该会接收到一条消息，说明未返回任何数据。

   注: 在分区数据库环境中，必须对所有数据库分区都执行此步骤。

5. 使用下列命令在 TSM 服务器上创建数据库备份:

   ```
   db2 backup db zample use tsm options "'-asnodename=clusternode'"
   ```

   将返回类似以下的信息:

   ```
   Backup successful. The timestamp for this backup image is : 20090216151025
   ```

   如果不在 **BACKUP DATABASE** 命令中指定 **-asnodename** 选项，那么可以改为更新 **vendoropt** 数据库配置参数。

   注: 在分区数据库环境中，必须对所有数据库分区都执行此步骤。根据您正在执行联机备份还是脱机备份，在数据库分区上执行此步骤的顺序有所不同。有关更多信息，请参阅第 259 页的『备份数据』。

6. 使用下列命令连接至 zample 数据库:

   ```
   db2 connect to zample
   ```

7. 通过使用下列命令创建一个表并将数据装入 TSM 服务器来生成数据库的新事务日志:

```
bar:/home/roecken> db2 load from mr of del modified by noheader
into employee copy yes use tsmwhere
```

在此示例中，表名为 employee，并且正在从名为 mr 的定界 ASCII 文件中装入数据。指定了 **COPY** YES 选项以生成所装入的数据的副本，并且 **USE TSM** 选项指定该数据的副本存储在 TSM 服务器上。

注: 仅当数据库启用了前滚恢复功能时才能指定 **COPY** YES 选项；即，必须将 **logarchmeth1** 数据库配置参数设置为 USEREXIT、LOGRETAIN、DISK 或 TSM。

为了指示它的进度，装入实用程序将返回一系列消息:

```
SQL3109N  实用程序正开始从"/home/roecken/mr"文件中装入数据。
SQL3500W  实用程序正在开始"LOAD"阶段，开始时间为"02/16/2009
15:12:13.392633"。

SQL3519W  开始装入一致点。输入记录计数 ="0"。

   SQL3520W  装入一致点成功。
SQL3110N  实用程序已完成处理。从输入文件
读取了"1"行。

SQL3519W  开始装入一致点。输入记录数 ="1"。

   SQL3520W  装入一致点成功。
SQL3515W 实用程序已完成"LOAD"阶段，完成时间为"02/16/2009
15:12:13.445718"。


   读取的行数        = 1
跳过的行数        = 0
装入的行数        = 1
拒绝的行数        = 0
删除的行数        = 0
落实的行数        = 1
```

8. 将数据装入表中后，请通过在 zample 数据库中运行下列查询来确认在 TSM 服务器上存在一个备份映像、一个装入副本映像和一个日志文件:

```
bar:/home/roecken/sqllib/adsm> db2adutl query db zample
   options "-asnodename=clusternode"
```

将返回以下信息:

```
正在检索 FULL DATABASE BACKUP 信息。
 1 时间: 20090216151025  最早的日志: S0000000.LOG  日志流: 0
 会话数: 1

正在检索 INCREMENTAL DATABASE BACKUP 信息。
 找不到 ZAMPLE 的 INCREMENTAL DATABASE BACKUP 映像

正在检索 DELTA DATABASE BACKUP 信息。
 找不到 ZAMPLE 的 DELTA DATABASE BACKUP 映像

正在检索 TABLESPACE BACKUP 信息。
 找不到 ZAMPLE 的 TABLESPACE BACKUP 映像

正在检索 INCREMENTAL TABLESPACE BACKUP 信息。
 找不到 ZAMPLE 的 INCREMENTAL TABLESPACE BACKUP 映像

正在检索 DELTA TABLESPACE BACKUP 信息。
 找不到 ZAMPLE 的 DELTA TABLESPACE BACKUP 映像

正在检索 LOAD COPY 信息。
 1 时间:  20090216151213
```

```
正在检索 LOG ARCHIVE 信息。
日志文件: S0000000.LOG, 链号: 0, 日志流: 0,
生成时间: 2009-02-16-15.10.38
```

9. 在此示例中，尚未设置计算机 2 dps 以用于 zample 数据库的跨节点恢复。使用下列命令验证是否已没有数据与此用户和计算机相关联:

```
dps:/home/regress9/sqllib/adsm> db2adutl query db zample
```

将返回以下信息:

```
  --- 数据库目录是空的 ---
警告: 在 ADSM 服务器上 DB2 没有创建文件空间
警告: 在 ADSM 中找不到任何别名的 DB2 备份映像。
```

10. 使用下列命令查询 TSM 服务器以获得与代理节点 clusternode 相关联的 zample 数据库的对象列表:

```
dps:/home/regress9/sqllib/adsm> db2adutl query db zample
   options="-asnodename=clusternode"
```

将返回以下信息:

```
--- 数据库目录是空的 ---

对 ZAMPLE 数据库的查询

正在检索 FULL DATABASE BACKUP 信息。
 1 时间: 20090216151025  最早的日志: S0000000.LOG  日志流: 0
 会话数: 1

正在检索 INCREMENTAL DATABASE BACKUP 信息。
 找不到 ZAMPLE 的 INCREMENTAL DATABASE BACKUP 映像

正在检索 DELTA DATABASE BACKUP 信息。
 找不到 ZAMPLE 的 DELTA DATABASE BACKUP 映像

正在检索 TABLESPACE BACKUP 信息。
 找不到 ZAMPLE 的 TABLESPACE BACKUP 映像

正在检索 INCREMENTAL TABLESPACE BACKUP 信息。
 找不到 ZAMPLE 的 INCREMENTAL TABLESPACE BACKUP 映像

正在检索 DELTA TABLESPACE BACKUP 信息。
 找不到 ZAMPLE 的 DELTA TABLESPACE BACKUP 映像

正在检索 LOAD COPY 信息。
 1 时间:  20090216151213

正在检索 LOG ARCHIVE 信息。
日志文件: S0000000.LOG, 链号: 0, 日志流: 0,
生成时间: 2009-02-16-15.10.38
```

此信息与先前生成的 TSM 信息匹配，并确认可以将此映像复原到 dps 计算机上。

11. 使用下列命令将 zample 数据库从 TSM 服务器复原至 dps 计算机:

```
dps:/home/regress9> db2 restore db zample use tsm options
   "'-asnodename=clusternode'" without prompting
```

将返回以下信息:

```
DB20000I  已成功完成 RESTORE DATABASE 命令。
```

注: 如果 dps 上已经存在 zample 数据库，那么将忽略 **OPTIONS** 参数，而使用数据库配置参数 **vendoropt**。此配置参数将覆盖备份或复原操作的 **OPTIONS** 参数。

12. 执行前滚操作以应用在创建新表并装入新数据时记录在 zample 数据库日志文件中的事务。在此示例中，因为未指定用户和计算机信息而导致 ROLLFORWARD 实用程序找不到日志文件，所以前滚操作的下列尝试将失败:

```
dps:/home/regress9> db2 rollforward db zample to end of logs and stop
```

该命令会返回下列错误:

```
SQL4970N  数据库"ZAMPLE"上的前滚恢复不能达到指定的停止点（日志结束
或时间点），原因是在节点"0"上丢失了日志文件。
```

使用适当的 **logarchopt** 值强制 ROLLFORWARD 实用程序查找另一台计算机上的日志文件。在此示例中，使用下列命令设置 **logarchopt1** 数据库配置参数并搜索与用户 roecken 和计算机 bar 相关联的日志文件:

```
dps:/home/regress9> db2 update db cfg for zample using logarchopt1
   "'-asnodename=clusternode'"
```

13. 通过使用下列命令来设置 **vendoropt** 数据库配置参数，使 ROLLFORWARD 实用程序能够使用备份和装入副本映像:

```
dps:/home/regress9> db2 update db cfg for zample using VENDOROPT
   "'-asnodename=clusternode'"
```

14. 使用下列命令通过应用记录在 zample 数据库日志文件中的事务来完成跨节点数据恢复:

```
dps:/home/regress9> db2 rollforward db zample to end of logs and stop
```

将返回以下信息:

```
                                 前滚状态

输入数据库别名                  = zample
返回了状态的成员数              = 1

成员数        前滚                    要读取的                已处理的日志文件            上次落实的事务
              状态                    下一个日志
------------  ----------  ----------  ------------------------  ------------------------------
       0      未暂挂              S0000000.LOG-S0000000.LOG  2009-05-06-15.28.11.000000 UTC

DB20000I  已成功完成 ROLLFORWARD 命令。
```

已将计算机 dps 上用户 regress9 下的数据库 zample 恢复到计算机 bar 上用户 roecken 下数据库的同一位置。

示例 4: TSM 服务器已配置为使用 DB2 pureScale环境中的客户机代理节点

此示例显示如何将两个成员设置为代理节点，以便当日志归档和备份都存储在 TSM 服务器上且在该服务器上使用 PASSWORDACCESS=GENERATE 选项来管理密码时，您可以将数据从一个成员恢复至另一个成员。

注: 在更新数据库配置之后，可能需要对数据库进行脱机备份。

在此示例中，成员 member1 和 member2 注册在 clusternode 代理名称下。在 DB2 pureScale环境中，您可以从任何成员执行备份或数据恢复操作。在此示例中，会将数据从 member2 恢复

1. 使用下列命令将成员 member1 和 member2 在 TSM 服务器上注册为代理节点:

```
REGISTER NODE clusternode mypassword
GRANT PROXYNODE TARGET=clusternode AGENT=member1,member2
```

2. 要启用数据库以将日志归档到 TSM 服务器，请使用下列命令更新 zample 数据库的数据库配置参数 **logarchmeth1**:

```
member1:/home/roecken> db2 update db cfg for zample using
   LOGARCHMETH1 tsm logarchopt1 "'-asnodename=clusternode'"
```

注: 在 DB2 pureScale环境中，您可以从任何成员设置一次全局 **logarchmeth1** 数据库配置参数。

将返回以下信息:

```
成功完成 DB20000I UPDATE DATABASE CONFIGURATION 命令。
```

3. 使用下列命令从数据库断开所有用户和应用程序的连接:

```
db2 force applications all
```

4. 使用下列命令验证是否已没有应用程序连接到数据库:

```
db2 list applications global
```

您应该会接收到一条消息，说明未返回任何数据。

5. 使用下列命令在 TSM 服务器上创建数据库备份:

```
db2 backup db zample use tsm options '-asnodename=clusternode'
```

将返回类似以下的信息:

```
Backup successful. The timestamp for this backup image is : 20090216151025
```

如果不在 **BACKUP DATABASE** 命令中指定 **-asnodename** 选项，那么可以改为更新 **vendoropt** 数据库配置参数。

注: 在 DB2 pureScale环境中，您可以从任何成员运行此命令来备份数据库的所有数据。

6. 使用下列命令连接至 zample 数据库:

```
db2 connect to zample
```

7. 通过使用下列命令创建一个表并将数据装入 TSM 服务器来生成数据库的新事务日志:

```
member1:/home/roecken> db2 load from mr of del modified by noheader replace
      into employee copy yes use tsmwhere
```

在此示例中，表名为 employee，并且正在从名为 mr 的定界 ASCII 文件中装入数据。指定了 **COPY** YES 选项以生成所装入的数据的副本，并且 **USE TSM** 选项指定该数据的副本存储在 TSM 服务器上。

注: 仅当数据库启用了前滚恢复功能时才能指定 **COPY** YES 选项；即，必须将 **logarchmeth1** 数据库配置参数设置为 USEREXIT、LOGRETAIN、DISK 或 TSM。

为了指示它的进度，装入实用程序将返回一系列消息:

```
SQL3109N  实用程序正开始从"/home/roecken/mr"文件中装入数据。
SQL3500W  实用程序正在开始"LOAD"阶段，开始时间为"02/16/2009
15:12:13.392633"。

SQL3519W  开始装入一致点。输入记录计数 ="0"。

   SQL3520W  装入一致点成功。
SQL3110N  实用程序已完成处理。从输入文件
读取了"1"行。

SQL3519W  开始装入一致点。输入记录数 ="1"。

   SQL3520W  装入一致点成功。
SQL3515W 实用程序已完成"LOAD"阶段，完成时间为"02/16/2009
15:12:13.445718"。


      读取的行数          = 1
```

```
跳过的行数         = 0
装入的行数         = 1
拒绝的行数         = 0
删除的行数         = 0
落实的行数         = 1
```

8. 将数据装入表中后，请通过在 zample 数据库中运行下列查询来确认在 TSM 服务器上存在一个备份映像、一个装入副本映像和一个日志文件:

```
member1:/home/roecken/sqllib/adsm> db2adutl query db zample
   options "-asnodename=clusternode"
```

将返回以下信息:

```
正在检索 FULL DATABASE BACKUP 信息。
 1 时间: 20090216151025  最早的日志: S0000000.LOG  日志流: 0
 会话数: 1

正在检索 INCREMENTAL DATABASE BACKUP 信息。
 找不到 ZAMPLE 的 INCREMENTAL DATABASE BACKUP 映像

正在检索 DELTA DATABASE BACKUP 信息。
 找不到 ZAMPLE 的 DELTA DATABASE BACKUP 映像

正在检索 TABLESPACE BACKUP 信息。
 找不到 ZAMPLE 的 TABLESPACE BACKUP 映像

正在检索 INCREMENTAL TABLESPACE BACKUP 信息。
 找不到 ZAMPLE 的 INCREMENTAL TABLESPACE BACKUP 映像

正在检索 DELTA TABLESPACE BACKUP 信息。
 找不到 ZAMPLE 的 DELTA TABLESPACE BACKUP 映像

正在检索 LOAD COPY 信息。
 1 时间:  20090216151213

正在检索 LOG ARCHIVE 信息。

日志文件: S0000000.LOG, 链号: 1, 日志流: 1, 生成时间: 2009-02-16-13.01.10

日志文件: S0000000.LOG, 链号: 1, 日志流: 0, 生成时间: 2009-02-16-13.01.11

日志文件: S0000000.LOG, 链号: 1, 日志流: 2, 生成时间: 2009-02-16-13.01.19

日志文件: S0000001.LOG, 链号: 1, 日志流: 0, 生成时间: 2009-02-16-13.02.49

日志文件: S0000001.LOG, 链号: 1, 日志流: 1, 生成时间: 2009-02-16-13.02.49

日志文件: S0000001.LOG, 链号: 1, 日志流: 2, 生成时间: 2009-02-16-13.02.49

日志文件: S0000002.LOG, 链号: 1, 日志流: 1, 生成时间: 2009-02-16-13.03.15

日志文件: S0000002.LOG, 链号: 1, 日志流: 2, 生成时间: 2009-02-16-13.03.15

日志文件: S0000002.LOG, 链号: 1, 日志流: 0, 生成时间: 2009-02-16-13.03.16
```

9. 使用下列命令查询 TSM 服务器以获得与代理节点 clusternode 相关联的 zample 数据库的对象列表:

```
member2:/home/regress9/sqllib/adsm> db2adutl query db zample
   options="-asnodename=clusternode"
```

将返回以下信息:

```
--- 数据库目录是空的 ---

对 ZAMPLE 数据库的查询

正在检索 FULL DATABASE BACKUP 信息。
 1 时间: 20090216151025  最早的日志: S0000000.LOG  日志流: 0
 会话数: 1

正在检索 INCREMENTAL DATABASE BACKUP 信息。
 找不到 ZAMPLE 的 INCREMENTAL DATABASE BACKUP 映像
```

```
正在检索 DELTA DATABASE BACKUP 信息。
 找不到 ZAMPLE 的 DELTA DATABASE BACKUP 映像

正在检索 TABLESPACE BACKUP 信息。
 找不到 ZAMPLE 的 TABLESPACE BACKUP 映像

正在检索 INCREMENTAL TABLESPACE BACKUP 信息。
 找不到 ZAMPLE 的 INCREMENTAL TABLESPACE BACKUP 映像

正在检索 DELTA TABLESPACE BACKUP 信息。
 找不到 ZAMPLE 的 DELTA TABLESPACE BACKUP 映像

正在检索 LOAD COPY 信息。
  1 时间:  20090216151213

正在检索 LOG ARCHIVE 信息。

日志文件: S0000000.LOG, 链号: 1, 日志流: 1, 生成时间: 2009-02-16-13.01.10

日志文件: S0000000.LOG, 链号: 1, 日志流: 0, 生成时间: 2009-02-16-13.01.11

日志文件: S0000000.LOG, 链号: 1, 日志流: 2, 生成时间: 2009-02-16-13.01.19

日志文件: S0000001.LOG, 链号: 1, 日志流: 0, 生成时间: 2009-02-16-13.02.49

日志文件: S0000001.LOG, 链号: 1, 日志流: 1, 生成时间: 2009-02-16-13.02.49

日志文件: S0000001.LOG, 链号: 1, 日志流: 2, 生成时间: 2009-02-16-13.02.49

日志文件: S0000002.LOG, 链号: 1, 日志流: 1, 生成时间: 2009-02-16-13.03.15

日志文件: S0000002.LOG, 链号: 1, 日志流: 2, 生成时间: 2009-02-16-13.03.15

日志文件: S0000002.LOG, 链号: 1, 日志流: 0, 生成时间: 2009-02-16-13.03.16
```

此信息与先前生成的 TSM 信息匹配，并确认可以将此映像复原到 member2 成员上。

10. 使用下列命令从 member2 成员复原 TSM 服务器上的 zample 数据库:

```
member2:/home/regress9> db2 restore db zample use tsm options
   '-asnodename=clusternode' without prompting
```

将返回以下信息:

```
DB20000I  已成功完成 RESTORE DATABASE 命令。
```

注: 如果 member2 上已经存在 zample 数据库，那么将忽略 **OPTIONS** 参数，而使用数据库配置参数 **vendoropt**。此配置参数将覆盖备份或复原操作的 **OPTIONS** 参数。

11. 通过使用下列命令来设置 **vendoropt** 数据库配置参数，使 ROLLFORWARD 实用程序能够使用备份和装入副本映像:

```
member2:/home/regress9> db2 update db cfg for zample using VENDOROPT
   "'-asnodename=clusternode'"
```

注: 在 DB2 pureScale环境中，您可以从任何成员设置一次全局 **vendoropt** 数据库配置参数。

12. 使用下列命令通过应用记录在 zample 数据库日志文件中的事务来完成跨成员数据恢复:

```
member2:/home/regress9> db2 rollforward db zample to end of logs and stop
```

将返回以下信息:

```
                                前滚状态

输入数据库别名                  = zample
返回了状态的成员数              = 3

成员数        前滚                    要读取的                  已处理的日志文件          上次落实的事务
              状态                      下一个日志
------------  ----------  ----------  -------------------------  ----------------------------
           0  未暂挂                    S0000001.LOG-S0000012.LOG  2009-05-06-15.28.11.000000 UTC
           1  未暂挂                    S0000001.LOG-S0000012.LOG  2009-05-06-15.28.11.000000 UTC
           2  未暂挂                    S0000001.LOG-S0000012.LOG  2009-05-06-15.28.11.000000 UTC

DB20000I  已成功完成 ROLLFORWARD 命令。
```

已将成员 member2 上用户 regress9 下的数据库 zample 恢复到成员 member1 上用户 roecken 下数据库的同一位置。

恢复已删除的表

您可能会偶然删除包含仍需要的数据的表。如果是这样，那么您应该考虑在删除表操作后使关键的表成为可复原的。

可通过调用数据库复原操作恢复表数据，后跟一个前滚到删除表前的某时间点的数据库前滚操作。如果数据库很大，那么复原和前滚操作可能很耗时间，并且数据在恢复期间不可用。

已删除的表的恢复功能使您可使用表空间级的复原和前滚操作来恢复已删除的表数据。

此表空间级别恢复比数据库级别恢复快，并且数据库对用户仍然可用。

开始之前

要使已删除的表可以复原，必须对该表所在的表空间启用 DROPPED TABLE RECOVERY 选项。可在创建表空间期间启用此选项或通过调用 ALTER TABLESPACE 语句来启用此选项。DROPPED TABLE RECOVERY 选项是表空间特定的，并限于对常规表空间使用。要确定是否对表空间启用了已删除的表的恢复功能，可以查询 SYSCAT.TABLESPACES 目录表中的 DROP_RECOVERY 列。

创建表空间时，缺省情况下已删除的表的恢复选项已打开。如果不要启用表空间的已删除的表的恢复功能，那么可以在发出 CREATE TABLESPACE 语句时显式将 DROPPED TABLE RECOVERY 选项设置为 OFF，或者可以使用 ALTER TABLESPACE 语句来对现有表空间禁用已删除的表的恢复功能。如果有许多删除表操作需要恢复，或者历史记录文件很大，那么“恢复已删除表”功能可能会影响正向恢复的性能。

对表（对该表的表空间启用了已删除的表的恢复功能）运行 DROP TABLE 语句时，将在日志文件中建立另一条目（标识已删除的表）。还会在恢复历史记录文件中建立一个条目，包含可用于重新创建表的信息。

对于分区表来说，即使对一个或多个表空间中的非分区表关闭已删除的表的恢复功能，已删除的表的恢复功能也始终处于打开状态。对于分区表来说，只写一个已删除的表日志记录。此日志记录对于恢复该表的所有数据分区来说已足够了。

关于此任务

如果在删除表时它处于重组暂挂状态，那么历史记录文件中的 CREATE TABLE DDL 与导入文件的 CREATE TABLE DDL 不完全匹配。在执行 **REORG** 建议的第一个 ALTER

操作之前，导入文件将采用该表的格式，但历史记录文件中的 CREATE TABLE 语句将与包含任何 ALTER TABLE 语句结果的表的状态相匹配。

用于 LOAD 或 IMPORT 的文件类型修饰符

要通过装入或导入来恢复表，请指定下列文件类型修饰符:

- 当要恢复数据的数据类型为 GRAPHIC 或 VARGRAPHIC 时，在 **IMPORT** 或 **LOAD** 命令中应使用文件类型修饰符 **usegraphiccodepage**。 原因是它可能包含多个代码页。
- 在 **IMPORT** 或 **LOAD** 命令中，应使用 **delprioritychar** 文件类型修饰符。该修饰符允许 **LOAD** 和 **IMPORT** 命令解析在字符或图形列数据中包含换行符的行。

限制

一次只能复原一个已删除的表。

对可从已删除的表中复原的数据类型有一些限制。不可能复原:

- 不能对临时表使用 DROPPED TABLE RECOVERY 选项。
- 与行类型相关联的元数据。（数据已复原，但不是元数据。）将复原类型表的层次结构表中的数据。此数据包含的信息可能比已删除的类型表中的信息多。
- XML 数据。如果尝试恢复包含 XML 数据的已删除的表，那么相应的列数据将为空。

过程

可执行下列操作来复原已删除的表:

1. 通过调用 **LIST HISTORY DROPPED TABLE** 命令来识别已删除的表。 已删除的表标识列示在“备份标识”列中。
2. 复原在删除该表前所建立的数据库级别或表空间级别备份映像。
3. 创建包含表数据的文件将写至的导出目录。 此目录必须可供所有数据库分区访问，或者在每个数据库分区上都存在。此导出目录下的子目录是由每个数据库分区自动创建的。这些子目录的名称是 NODE*nnnn*，其中 *nnnn* 代表数据库分区或节点号。包含已废弃的表数据的数据文件（就如它存在于每个数据库分区上那样）将导出到称为 data 的较低子目录中。 例如:

   ```
   \export_directory\NODE0000\data。
   ```

4. 通过在 **ROLLFORWARD DATABASE** 命令上使用 **RECOVER DROPPED TABLE** 参数，在删除表之后前滚至某时间点。也可前滚至日志末尾，以使对表空间或数据库中的其他表进行的更新不会丢失。
5. 通过使用 CREATE TABLE 语句来根据恢复历史记录文件重新创建该表。
6. 将在前滚操作期间导出的表数据导入表中。 如果进行废弃时表处于重组暂挂状态，那么需要更改 CREATE TABLE DDL 的内容以与数据文件的内容相匹配。

崩溃恢复

对数据库执行的事务（也称工作单元）可能被意外中断。如果在作为工作单元一部分的所有更改完成、落实并写入磁盘之前发生故障，那么该数据库就会处于不一致和不可用的状态。

*崩溃恢复*是将数据库移动回一致并可用状态的进程。为此，回滚未完成的事务，并完成当发生崩溃时仍在内存中的已落实事务 (图 20)。当数据库处于一致和可用状态时，它已达到一种称为一致点的状态。

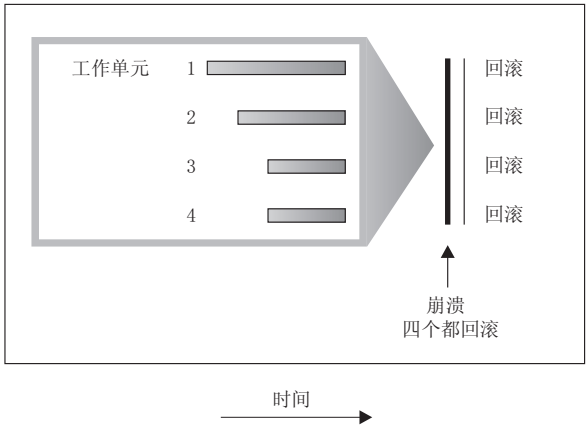


图 20. 回滚工作单元（崩溃恢复）

如果是使用 IBM DB2 pureScale Feature，那么需要了解两种特定类型的崩溃恢复：*成员崩溃恢复*和*组崩溃恢复*。成员崩溃恢复是在成员失败之后，使用单个成员的日志流来恢复部分数据库的过程。成员崩溃恢复通常在成员重新启动时自动启动，它是一种联机操作，这表示其他成员仍可以访问数据库。多个成员可以同时执行成员崩溃恢复。组崩溃恢复是在因失败而导致集群中未保留任何可用的集群高速缓存设施之后，使用多个成员的日志流来恢复数据库的过程。组崩溃恢复通常也会（在组重新启动时）自动启动，而正在进行组崩溃恢复时，无法访问数据库，因为 DB2 崩溃恢复在 DB2 pureScale 环境外部进行操作。

如果数据库或数据库管理器失败，那么数据库可能处于不一致状态。数据库内容可能包括发生失败时未完成的事务所进行的更改。数据库可能会丢失由失败之前已完成但尚未清空至磁盘的事务所进行的更改。必须执行崩溃恢复操作才能回滚部分完成的事务，并将已完成的事务先前只在内存中进行的更改写入磁盘。

必须执行崩溃恢复的情况包括:

- 机器上的断电故障，它会导致使用该机器的数据库管理器和数据库分区崩溃
- 硬件故障，例如内存、磁盘、CPU 或网络故障。
- 导致 DB2 实例异常结束的严重操作系统错误

如果您希望由数据库管理器自动执行崩溃恢复，请将 **autorestart** 数据库配置参数设置为 ON，以启用该自动重新启动参数。（这是缺省值。）如果不想要重新启动行为，那么将 **autorestart** 数据库配置参数设置为 OFF。因此，必须在数据库故障发生时发出 **RESTART DATABASE** 命令。如果数据库 I/O 在发生崩溃之前已处于暂挂状态，那么必须指定 **RESTART DATABASE** 命令的 **WRITE RESUME** 选项才能使崩溃恢复继续进行。管理通知日志记录数据库重新启动操作开始的时间。

如果崩溃恢复发生在用于前滚恢复的数据库上（即，未将 **logarchmeth1** 配置参数设置为 OFF），且在崩溃恢复期间因个别表空间而发生错误，那么会让该表空间脱机，直到修复后才能对其进行访问。崩溃恢复将继续在其他表空间上执行。在崩溃恢复完成时，该数据库中的其他表空间将是可访问的，并且可与该数据库建立连接。但是，如

果脱机的表空间包含系统目录，那么必须先修复表空间才允许进行所有连接。此行为不适用于 DB2 pureScale 环境。如果成员崩溃恢复或组崩溃恢复期间出错，那么崩溃恢复操作将失败。

恢复已损坏的表空间

损坏的表空间带有一个或多个不能访问的容器。这通常是由介质问题引起的：这些问题或者是永久性的（例如，坏的磁盘），或者是临时性的（例如，脱机磁盘或卸下的文件系统）。

如果损坏的表空间是系统目录表空间，那么不能重新启动数据库。如果不能修复容器问题但要使原始数据不受影响，可用的选项包括：

- 复原数据库
- 复原目录表空间。

  注：

  1. 因为必须前滚数据库，所以表空间复原只对可恢复数据库有效。
  2. 如果复原目录表空间，那么必须执行前滚操作至日志末尾。

如果损坏的表空间*不是*系统目录表空间，那么 DB2 for Linux, UNIX, and Windows 会尝试使尽可能多的数据库部分可用。

如果损坏的表空间只是临时的表空间，应在可进行与数据库的连接后，尽快创建新的临时表空间。创建后，可使用这个新的临时表空间，并且需要临时表空间的正常数据库操作也可以继续执行。如果希望，可删除脱机的临时表空间。使用系统临时表空间的表重组有特殊的注意事项：

- 如果数据库或数据库管理器配置参数 **indexrec** 设置为 RESTART，在数据库激活期间必须重建所有无效的索引；这包括构建阶段期间崩溃的重组中的索引。
- 如果在损坏的临时表空间中有未完成的重组请求，可能必须将 **indexrec** 配置参数设置为 ACCESS 以避免重新启动失败。

恢复可恢复数据库中的表空间

需要进行崩溃恢复时，使已损坏的表空间处于脱机状态，并且不可对它进行访问。将它处于前滚暂挂状态。如果没有其他问题，即使在数据库带有已损坏的表空间的情况下，重新启动操作也可以成功地使该数据库联机。

一旦处于联机状态，已损坏的表空间就不可用，但其余数据库可用。要修复已损坏的表空间并使它可用，请使用以下过程。

过程

要使损坏的表空间可用，使用以下过程之一：

- 方法 1
  1. 修复损坏但没有丢失原始数据的容器。
  2. 完成表空间前滚操作直至日志的结尾。

     注： 前滚操作首先尝试使表空间由脱机状态变为正常状态。
- 方法 2

1. 修复损坏且可能丢失原始数据的容器。
2. 执行表空间复原操作。
3. 完成表空间前滚操作直至日志的结尾或时间点。

恢复不可恢复数据库中的表空间

在需要进行崩溃恢复时，如果有已损坏的表空间，那么只有先删除已损坏的表空间才能成功重新启动数据库。在不可恢复的数据库中，不保留恢复已损坏的表空间所需的日志。

因此，对这类表空间可以采取的唯一有效操作是删除它们。

过程

要重新启动表空间已损坏的数据库:

1. 调用不合格的重新启动数据库操作。 如果没有损坏的表空间，那么操作将成功。如果失败 (SQL0290N)，那么可以查看管理通知日志文件以获取当前已损坏的表空间的完整列表。
2. 如果您希望删除所有已损坏的表空间，那么启动另一重新启动数据库操作，使用 DROP PENDING TABLESPACES 选项列示所有已损坏的表空间。如果损坏的表空间包括在 DROP PENDING TABLESPACES 列表中，那么表空间将进入删除暂挂状态，且必须在恢复操作完成之后删除表空间。

   重新启动操作继续，无需恢复指定的表空间。如果损坏的表空间未包括在 DROP PENDING TABLESPACES 列表中，那么重新启动数据库操作将失败，并返回 SQL0290N。

   注: 在 DROP PENDING TABLESPACES 列表中包括表空间名称并不表示该表空间将处于删除暂挂状态。仅当重新启动操作期间发现表空间损坏时才会处于此种状态。
3. 如果重新启动数据库操作成功，那么调用 **LIST TABLESPACES** 命令来找出哪些表空间处于删除暂挂状态。
4. 发出 DROP TABLESPACE 语句来删除处于删除暂挂状态的每个表空间。 删除损坏的表空间之后，您就能够收回所损坏表空间使用的空间或重新创建这些表空间。
5. 如果不想删除表空间和丢失损坏的表空间中的数据，您可以:
   - 修复损坏的容器（没有丢失原始数据）。
   - 重新发出 **RESTART DATABASE** 命令。
   - 执行数据库复原操作。

降低介质故障的影响

*介质故障*是存储设备（例如，硬盘驱动器）的读/写接口上存在的缺陷所导致的错误。此类型的故障可能难于检测和防止，因此，您必须采取措施降低所发生的故障造成的影响。

要降低介质故障的可能性，并简化从此类型的故障的恢复:

- 镜像或复制保存重要数据库的数据和日志的磁盘。
- 使用“独立磁盘冗余阵列”(RAID) 配置，例如 RAID 级别 5。

- 在分区数据库环境中，要设置一个严格的过程，来处理该目录分区上的数据和日志。因为此数据库分区是维护数据库的关键:
  - 确保它位于可靠的磁盘上
  - 复制它
  - 频繁地进行备份
  - 不要在此节点上存放用户数据。

防止磁盘故障

如果您担心可能会因为磁盘崩溃而使数据或日志损坏，请使用某种形式的磁盘故障容错。通常，这是通过使用*磁盘阵列*（一组磁盘）来完成的。

磁盘阵列有时只是指 RAID（独立磁盘冗余阵列）。磁盘阵列也可以通过处于操作系统级或应用程序级的软件来提供。硬件磁盘阵列和软件磁盘阵列的不同点在于处理输入/输出 (I/O) 请求的 CPU 处理的方式。对于硬件磁盘阵列，I/O 活动由磁盘控制器管理，对于软件磁盘阵列，由操作系统或应用程序完成。

硬件磁盘阵列

在硬件磁盘阵列中，一个磁盘控制器可使用和管理多个磁盘，且使用它自己的 CPU 来完成。管理构成此阵列的磁盘所需的所有逻辑都包含在磁盘控制器中。因此，此实现与操作系统无关。

有几种类型的 RAID 体系结构，在功能和性能上有所不同，但只有 RAID 级别 1 和级别 5 是现在常用的。

RAID 级别 1 也称为磁盘镜像和双工技术。*磁盘镜像技术*使用单个磁盘控制器将数据（一个完整的文件）从一个磁盘复制到另一个磁盘。*磁盘双工技术*类似于磁盘镜像技术，但磁盘是与另一个磁盘控制器连接的（与两个 SCSI 适配器相似）。因此可较好地保护数据：任何一个磁盘发生故障时，仍可从另一个磁盘访问数据。使用磁盘双工技术时虽然磁盘控制器可能会发生故障，但不会影响可提供的数据保护。它的性能很好，但实现它需要的磁盘数是通常磁盘数的两倍。

RAID 级别 5 涉及到跨所有磁盘，按扇区进行的数据和带奇偶校验的条带分割。奇偶性校验与数据交错，而不是存储在专用的驱动器上。数据保护性能很好：如果有任何磁盘发生故障，仍然可以使用其他磁盘上的信息以及已分割的奇偶性校验信息来访问数据。读性能良好，但写性能较差。RAID 级别 5 配置需要最少三个相同的磁盘。开销所需的磁盘空间量随阵列中磁盘的数量而变化。如果 RAID 级别 5 配置了 5 个磁盘，空间开销将是百分之二十。

RAID 级别 1+0 (10) 涉及跨至少两个磁盘的数据镜像和条带分割。制作成镜像会将数据同时写入两个或多个磁盘，提供与 RAID 级别 1 相同的故障容错能力。条带分割会将数据分割为块，再将每个块写入到独立的磁盘驱动器。这样就可以将 I/O 负载分散到多个通道和驱动器，从而达到较高的 I/O 性能，但是 RAID 级别 1+0 会使有效磁盘空间减少至一半，因为它将为所有数据生成镜像。实现 RAID 级别 10 需要至少 4 个驱动器。

RAID 级别 0+1 作为镜像阵列实现，其中的段为 RAID 0 阵列且与 RAID 级别 5 具有相同的故障容错能力。这就可以通过将 I/O 负载分散到多个通道和驱动器，从而提供

较高的 I/O 速度。但是请不要将 RAID 级别 0+1 与 RAID 级别 1+0 混淆。一个驱动器故障将导致整个阵列在实质上成为 RAID 级别 0 阵列。

使用 RAID（但不是 RAID 级别 0）磁盘阵列时，发生故障的磁盘不会影响您访问阵列上的数据。当在该阵列中使用可热插入的或可热交换的磁盘时，可以在该阵列正在使用时将替换磁盘与故障磁盘交换。使用 RAID 级别 5 时，如果有两个磁盘同时发生故障，所有数据都会丢失（但同时发生磁盘故障的可能性非常小）。

您可能会考虑对日志使用 RAID 级别 1 硬件磁盘阵列或软件磁盘阵列，因为这可以为故障点提供可恢复性并提供良好的写性能（这对日志来说很重要）。要达到此目的，可使用 *mirrorlogpath* 配置参数来指定 RAID 级别 1 文件系统上的镜像日志路径。如果（由于不能将时间花在磁盘故障后的数据恢复上）可靠性很关键，而写性能不是，那么可以考虑使用 RAID 级别 5 硬件磁盘阵列。另外，如果写性能很重要，而不太关心所需的附加磁盘空间量，那么可以考虑对您的数据以及日志使用 RAID 级别 1 硬件磁盘阵列。

有关可用的 RAID 级别的详细信息，访问以下 Web 站点：http://www.acnc.com/04_01_00.html

软件磁盘阵列

软件磁盘阵列在许多方面与硬件磁盘阵列相同，但是磁盘流量是由操作系统或在服务器上运行的应用程序来管理的。软件阵列像其他程序一样，必须争用 CPU 和系统资源。对于 CPU 受约束的系统，它不是一个好的选项，同时要记住磁盘阵列的总体性能取决于服务器的 CPU 负载和容量。

典型的软件磁盘阵列提供磁盘镜像。虽然冗余磁盘是必需的，但是由于不需要高成本的磁盘控制器，所以软件磁盘阵列的实现相对便宜。

注意:
如果操作系统引导驱动器在磁盘阵列中，那么当该驱动器发生故障时，系统将无法启动。如果在磁盘阵列运行之前该驱动器发生故障，那么磁盘阵列就不允许访问该驱动器。引导驱动器应与磁盘阵列分开。

降低事务故障的影响

*事务故障*是指在可以将事务落实到数据库之前就过早终止事务处理，这可能会导致数据丢失或破坏。

要降低事务故障的影响，应尽量确保：

- 每个 DB2 服务器配备不间断电源
- 在所有数据库分区上有足够的磁盘空间来存储数据库日志
- 在分区数据库环境中的数据库分区服务器之间有可靠的通信链路
- 分区数据库环境中的系统时钟的同步。

从分区数据库环境中的事务故障进行恢复

如果事务处理失败发生在分区数据库环境中，通常需要对发生了故障的数据库分区服务器和参与了该事务的任何其他数据库分区服务器都进行数据库恢复。

存在两种类型的数据库恢复：

- 对发生了故障的数据库分区服务器的崩溃恢复发生在更正了故障情况后。
- 对其他（仍活动的）数据库分区服务器的*数据库分区故障恢复*紧接在检测到故障后发生。

在分区数据库环境中，提交事务的数据库分区服务器是协调程序分区，而处理该事务的第一个代理程序是协调代理程序。协调代理程序负责将工作分布至其他数据库分区服务器上，并跟踪那些参与了该事务的服务器。当应用程序对一个事务发出 COMMIT 语句时，该协调代理程序使用两阶段落实协议来落实该事务。在第一阶段期间，协调程序分区将 PREPARE 请求分布至所有其他参与该事务的数据库分区服务器。然后，这些服务器用以下其中一项应答:

READ-ONLY
在此服务器中未发生任何数据更改

YES 在此服务器中发生了数据更改

NO 由于错误，服务器未准备落实

如果其中一个服务器应答 NO，那么回滚该事务。否则，协调程序分区开始第二阶段。

在第二阶段，协调程序分区写入一条 COMMIT 日志记录，然后将 COMMIT 请求分布至所有应答了 YES 的服务器。在所有其他数据库分区服务器都已落实后，它们会将 COMMIT 的应答发送至协调程序分区。当协调代理程序从所有参与服务器接收到所有 COMMIT 应答时，该事务完成。在此时间点，协调代理程序会写入一条 FORGET 日志记录。

活动数据库分区服务器上的事务故障恢复

如果任何数据库分区服务器检测到另一个服务器当机，那么与该发生故障的数据库分区服务器相关的所有工作都会停止:

- 如果仍处于活动状态的数据库分区服务器是某个应用程序的协调程序分区，且该应用程序在发生故障的数据库分区服务器上运行（尚未准备 COMMIT），那么会中断该协调代理程序，以便执行故障恢复。如果该协调代理程序处于 COMMIT 处理的第二个阶段，会将 SQL0279N 返回给应用程序，应用程序随之会丢失它的数据库连接。否则，协调代理程序将一个 ROLLBACK 请求分布至所有其他参与该事务的服务器，并将 SQL1229N 返回至该应用程序。
- 如果发生故障的数据库分区服务器是该应用程序的协调程序分区，那么仍在活动服务器上为该应用程序工作的代理程序会被中断，以便执行故障恢复。在事务未处于就绪状态的每个数据库分区上本地回滚事务。在事务处于就绪状态的那些数据库分区上，事务变得不确定。由于协调程序数据库分区不可用，所以协调程序数据库分区不知道事务在某些数据库分区上处于不确定状态。
- 如果该应用程序与发生故障的数据库分区服务器连接（在它发生故障之前），但是本地数据库分区服务器和发生故障的数据库分区服务器都不是协调程序分区，那么会中断为此应用程序工作的代理程序。协调程序分区将向其他数据库分区服务器发送 ROLLBACK 或 DISCONNECT 消息。如果协调程序分区返回 SQL0279，那么事务将仅在仍然活动的数据库分区服务器上处于不确定状态。

试图向该发生故障的服务器发送请求的任何进程（如，代理程序或死锁检测器）都会得到通知: 它不能发送该请求。

发生故障的数据库分区服务器上的事务故障恢复

如果事务失败导致数据库管理器异常结束，那么可以发出具有 RESTART 选项的 **db2start** 命令，以便在重新启动数据库分区后立即重新启动数据库管理器。如果无法重新启动数据库分区，那么可以发出 **db2start**，以便在另一数据库分区上重新启动数据库管理器。

如果数据库管理器异常结束，那么服务器上的数据库分区可能会处于不一致状态。要使它们可用，可以在数据库分区服务器上触发崩溃恢复:

- 通过 **RESTART DATABASE** 命令显式地触发
- 在 *autorestart* 数据库配置参数已设置为 ON 后，通过 CONNECT 请求隐式触发

崩溃恢复将重新应用活动日志文件中的日志记录，以确保所有已完成的事务的结果都在数据库中。重新应用了这些更改后，除不确定事务外的所有未落实的事务都将本地回滚。分区数据库环境中有两种类型的不确定事务:

- 在不是协调程序分区的数据库分区服务器上，已就绪但未落实的事务就是不确定的。
- 在协调程序分区上，已落实但还未被记录为完成（即，还未写入 FORGET 记录）的事务是不确定的。当协调代理程序未从为该应用程序工作的所有服务器接收到全部 COMMIT 应答时，会发生这种情况。

崩溃恢复试图通过以下其中一项操作解决所有不确定事务。要执行的操作取决于数据库分区服务器是否为应用程序的协调程序分区:

- 如果重新启动的服务器不是该应用程序的协调程序分区，它会将一个查询消息发送至该协调代理程序，以发现该事务的结果。
- 如果重新启动的服务器是该应用程序的协调程序分区，它会将一个消息发送至协调代理程序仍在等待它们的 COMMIT 应答的所有其他代理程序（下级代理）。

崩溃恢复可能并不能解决所有不确定事务。例如，某些数据库分区服务器可能会不可用。如果协调程序分区在参与事务的其他数据库分区之前完成崩溃恢复，那么崩溃恢复将不能解决不确定事务。因为崩溃恢复由每个数据库分区独立执行，所以上述情况是意料之中的事情。在这种情况中，会返回 SQL 警告消息 SQL1061W。由于不确定事务占用了资源（例如锁定和活动日志空间），有可能会导致不能对数据库进行任何更改，因为不确定事务占用了活动日志空间。因此，应确定在崩溃恢复之后是否还有不确定事务，并尽快恢复解决这些不确定事务所需的所有数据库分区服务器。

注: 在分区数据库环境中，会对每个节点运行 RESTART 数据库命令。为确保对所有节点重新启动该数据库，请使用以下建议命令:

```
db2_all "db2 restart database <database_name>"
```

如果解决不确定事务所需的一个或多个服务器不能及时恢复，且需要访问其他服务器上的数据库分区，可以通过作出启发式决策来手动解决这些不确定事务。可以使用 **LIST INDOUBT TRANSACTIONS** 命令来查询、落实和回滚服务器上的不确定事务。

注: **LIST INDOUBT TRANSACTIONS** 命令还用于分布式事务环境中。为了区分这两种类型的不确定事务，**LIST INDOUBT TRANSACTIONS** 命令返回的输出中的 *originator* 字段显示以下其中一项:

- DB2 企业服务器版，指示该事务始发于分区数据库环境。

- XA，它指示该事务始发于分布式环境中。

标识发生故障的数据库分区服务器

当一个数据库分区服务器发生故障时，应用程序通常会接收到下列其中一个 SQLCODE。检测哪个数据库管理器发生故障的方法取决于接收到的 SQLCODE:

SQL0279N
当在 COMMIT 处理期间终止的事务中涉及了数据库分区服务器时，会接收到此 SQLCODE。

SQL1224N
当发生故障的数据库分区服务器是该事务的协调程序分区时，会接收到此 SQLCODE。

SQL1229N
当发生故障的数据库分区服务器不是该事务的协调程序分区时，会接收到此 SQLCODE。

确定哪个发生了故障的数据库分区服务器是一个两阶段进程。

1. 通过检查 SQLCA 来查找已检测到故障的分区服务器。与 SQLCODE SQL1229N 相关的 SQLCA 在 *sqlerrd* 字段的第六个数组位置包含检测到错误的服务器的节点号。（为服务器写入的节点号与 `db2nodes.cfg` 文件中的节点号对应。）
2. 针对发生了故障的服务器的节点号，检查在步骤一中找到的有关服务器的管理通知日志。

注: 如果正在一个处理器上使用多逻辑节点，那么一个逻辑节点发生故障会导致同一个处理器上的其他逻辑节点发生故障。

从数据库分区服务器的故障恢复

您可以通过找出故障并解决导致故障的问题，从而从失败的数据库分区服务器进行恢复。

过程

要从数据库分区服务器的故障中恢复，请执行下列步骤。

1. 校正导致该故障的问题。
2. 通过从任何数据库分区服务器发出 **db2start** 命令，重新启动数据库管理器。
3. 通过在发生故障的一个或多个数据库分区服务器上发出 **RESTART DATABASE** 命令，重新启动数据库。

恢复大型机或中型服务器上的不确定事务

当 DB2 Connect 配置了 DB2 Syncpoint Manager时恢复主机上的不确定事务

如果在某个事务期间您的应用程序访问了主机或 System i 数据库服务器，那么恢复不确定事务的方法会有所不同。要访问主机或 System i 数据库服务器，使用 DB2 Connect。如果 DB2 Connect 配置了 DB2 Syncpoint Manager，那么恢复的步骤就会不同。

关于此任务

主机或 System i 服务器上不确定事务的恢复通常由“事务管理器”(TM) 和 DB2 Syncpoint Manager (SPM) 自动执行。主机或 System i 服务器上的不确定事务不占用本地 DB2 位置的任何资源，但只要该事务在主机或 System i 这一位置上是不确定的，就肯定会占用该位置的资源。如果主机或 System i 服务器的管理员确定必须作出启发式决策，那么管理员可能会与本地 DB2 数据库管理员联系（例如，通过电话）来确定是落实还是回滚主机或 System i 服务器上的事务。如果发生这种情况，那么可以使用 **LIST DRDA INDOUBT TRANSACTIONS** 命令来确定 DB2 Connect 实例中事务的状态。

过程

对于涉及 SNA 通信环境的大多数情况，可以使用下列步骤作为准则:

1. 连接至 SPM。 例如:

```
db2 => connect to db2spm

Database Connection Information

 Database product       = SPM0500
 SQL authorization ID   = CRUS
 Local database alias   = DB2SPM
```

2. 发出 **LIST DRDA INDOUBT TRANSACTIONS** 命令，以显示 SPM 已知的不确定事务。

   以下示例显示了 SPM 已知的一个不确定事务。db_name 是主机或 System i 服务器的本地别名。partner_lu 是主机或 System i 服务器的标准 LU 名。这为主机或 System i 服务器提供最佳标识，并且应该由主机或 System i 服务器的调用程序提供。luwid 为事务提供唯一标识，且在所有主机和 System i 服务器上都可用。如果显示了这个不确定的事务，那么当 uow_status 字段的值是 C（落实）或 R（回滚）时，可以使用该字段来确定事务的结果。如果发出带有 **WITH PROMPTING** 参数的 **LIST DRDA INDOUBT TRANSACTIONS** 命令，那么可以用交互式方式落实、回滚或忽略该事务。

```
db2 => list drda indoubt transactions
 DRDA Indoubt Transactions:
1.db_name: DBAS3    db_alias: DBAS3    role: AR
  uow_status: C  partner_status: I  partner_lu: USIBMSY.SY12DQA
corr_tok: USIBMST.STB3327L
 luwid: USIBMST.STB3327.305DFDA5DC00.0001
 xid: 53514C2000000017 00000000544D4442 0000000000305DFD A63055E962000000
           00035F
```

3. 如果未显示 partner_lu 和 luwid 的不确定事务，或者 **LIST DRDA INDOUBT TRANSACTIONS** 命令返回下列内容，那么已回滚该事务。

```
db2 => list drda indoubt transactions
SQL1251W  No data returned for heuristic query.
```

下一步做什么

另一种情况发生的可能性不大，但不排除有发生的可能性。如果显示了 partner_lu 的具有正确 luwid 的不确定事务，但 uow_status 为“I”，那么 SPM 不知道是要落实还是回滚该事务。在此情况下，应该使用 **WITH PROMPTING** 参数在 DB2 Connect 工作站上落实或回滚该事务。然后，允许 DB2 Connect 根据启发式决策与主机或 System i 服务器重新同步。

当 DB2 Connect 不使用 DB2 同步点管理器时恢复主机上的不确定事务

如果在某个事务期间您的应用程序访问了主机或 System i 数据库服务器，那么恢复不确定事务的方法会有所不同。要访问主机或 System i 数据库服务器，请使用 DB2 Connect。如果 DB2 Connect 配置了 DB2 同步点管理器，那么恢复步骤将有所不同。

关于此任务

从 DB2 Connect Personal Edition 或 DB2 Connect Enterprise Edition 使用 TCP/IP 连接对 DB2 for z/OS 进行多站点更新但未使用 DB2 Syncpoint Manager时，请使用本节中的信息。这种情况中的不确定事务的恢复不同于使用 DB2 Syncpoint Manager的不确定事务的恢复。当在此环境中出现不确定事务时，会在客户机、数据库服务器和/或“事务管理器”™ 数据库上生成一个警告条目，这取决于谁检测到该问题。警报条目位于 db2alert.log 文件中。

一旦 TM 和参与数据库及其连接再次全部可用，就会自动执行任何不确定事务的再同步。您应该允许在数据库服务器中自动执行再同步，而不是试探性地强制决定。但是，如果您必须这样做，可参考下列步骤。

注: 因为未包含 DB2 Syncpoint Manager，所以无法使用 **LIST DRDA INDOUBT TRANSACTIONS** 命令。

过程

1. 在 z/OS 主机上，发出命令 DISPLAY THREAD TYPE(INDOUBT)。

   从此列表中，标识您要试探性完成的事务。有关 DISPLAY 命令的详细信息，请参阅 *DB2 for z/OS Command Reference*。显示的 LUWID 可与“事务管理器数据库”中的同一个 luwid 匹配。
2. 根据您要执行的操作，发出 RECOVER THREAD(<LUWID>) ACTION (ABORT|COMMIT) 命令。

   有关 RECOVER THREAD 命令的详细信息，请参阅 *DB2 for z/OS Command Reference*。

灾难恢复

术语*灾难恢复*用于描述在发生火灾、地震、恶意破坏或其他大灾害时复原数据库所需要执行的活动。

灾难恢复计划可以包括以下其中一项或多项:

- 在紧急情况下使用的场所
- 用于恢复数据库的另一台机器
- 以非现场方式存储数据库备份和/或表空间备份以及归档日志。

如果灾难恢复计划是在另一台机器上复原整个数据库，那么至少需要完整数据库备份和该数据库的所有归档日志。当您有数据库中每个表空间的完整表空间备份时，虽然也可以重建数据库，但这种方法需要使用许多备份映像，因此，与使用完整数据库备份来进行恢复相比，这种方法更为耗时。

您可以选择将归档日志应用于数据库来使备用数据库保持最新。也可以选择将数据库备份或表空间备份以及日志归档保存在备用场所，而只在发生灾难后才执行复原和前滚操作。（在后一种情况下，最好使用最近的备份映像。）但是，发生灾难后，通常并不能将所有事务都恢复至发生灾难前的状态。

表空间备份对于灾难恢复的有用性取决于发生故障的范围。通常，如果复原整个数据库，灾难恢复就会比较简单并且不需要很长时间；因此，应该在备用场所保存一份完整数据库备份。如果该灾难是磁盘损坏，那么可以使用该磁盘上每个表空间的表空间备份来进行恢复。如果由于磁盘故障（或任何其他原因）而无法访问某个容器，那么可以将该容器复原到另一位置。

保护数据完整性或防止整个站点故障的另一种方法是实现 DB2 高可用性灾难恢复 (HADR) 功能。设置此功能后，HADR 通过将数据更改从源数据库（称为主数据库）复制到目标数据库（称为备用数据库）来防止数据丢失。

也可以使用复制功能来保护数据，以防部分站点故障或整个站点故障。复制功能允许定期地将数据复制到多个远程数据库。DB2 数据库提供了许多复制工具，这些工具允许您指定应该复制的数据、应该将数据复制到的数据库表以及复制更新内容的频率。

也可以使用诸如对等远程复制 (PPPC) 之类的存储器镜像功能来保护数据。PPPC 提供了卷或磁盘同步复制功能来预防灾难。PPPC 提供了卷或磁盘同步复制功能来预防灾难。

在进行灾难恢复规划时，DB2 数据库产品提供了若干选项。根据业务需要，您可能决定使用表空间或完整数据库备份来防止数据丢失，也可能认为您的环境更适合于使用诸如 HADR 之类的解决方案。不论作出何种选择，都应该在测试环境中测试恢复过程，然后在生产环境中实现这些过程。

版本恢复

*版本恢复*指的是使用备份操作期间创建的映像来复原数据库的先前版本。

对不可恢复数据库（即，该数据库没有归档日志）使用此方法。还可通过对 **RESTORE DATABASE** 命令使用 **WITHOUT ROLLING FORWARD** 选项，来对可恢复数据库使用此方法。

数据库复原操作将使用先前创建的备份映像来复原整个数据库。数据库备份使您可以将数据库复原到与进行备份时完全相同的状态。但是，从备份时间到故障时间之间的每个工作单元都将丢失（请参阅第 308 页的图 21）。

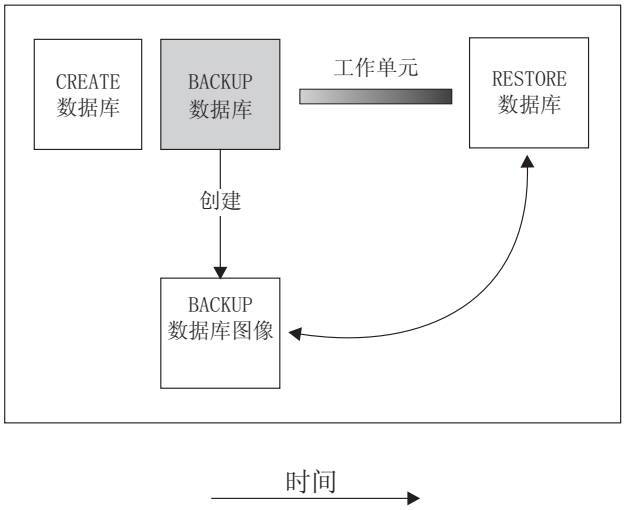


图 21. 版本恢复

使用版本恢复方法，必须定期安排和执行完整数据库备份。

在一个分区数据库环境中，数据库位于许多数据库分区服务器（或节点）上。必须复原所有数据库分区，而且用于复原数据库操作的所有备份映像必须是同时创建的。（将备份每个数据库分区并单独复原。）同时建立的每个数据库分区的备份称为*版本备份*。

前滚恢复

您可以完成数据库或表空间的前滚恢复。

要使用*前滚恢复*方法，必须已建立数据库的备份，并且已归档日志（方法是将 **logarchmeth1** 和 **logarchmeth2** 配置参数设置为 OFF 之外的值）。复原数据库并指定 **WITHOUT ROLLING FORWARD** 参数等效于使用版本恢复方法。此数据库被复原到创建脱机备份映像时的状态。如果您复原数据库，但*没有*对复原数据库操作指定 **WITHOUT ROLLING FORWARD** 参数，那么该数据库将在复原操作结束时处于前滚暂挂状态。这允许进行前滚恢复。

注：在下列情况下，不能使用 **WITHOUT ROLLING FORWARD** 参数：

- 正在从联机备份映像复原
- 正在发出表空间级复原

在恢复期间，会从归档中检索已归档的日志文件。如果已归档的日志文件已进行压缩，那么会自动将这些文件解压缩并使用这些文件。在活动日志路径或者溢出日志路径中遇到已归档的日志文件时，如果您手动复制此路径中的文件，那么也会自动将这些日志文件解压缩。

要考虑的两种恢复类型是：

- *数据库前滚恢复*。在此类型的前滚恢复中，记录在数据库日志中的事务应用到以下数据库复原操作中（请参阅第 309 页的图 22）。该数据库日志记录了对该数据库所作的所有更改。这种方法将数据库恢复到特定时间点时的状态，或恢复到故障前的状态（即，恢复到活动日志的末尾）。

在分区数据库环境中，数据库分布在许多数据库分区中，因此必须对数据库目录表所在的数据库分区（目录分区）发出 **ROLLFORWARD DATABASE** 命令。如果正在执行时间点前滚恢复，那么必须前滚所有数据库分区，以确保所有数据库分区处于同一级别。如果需要复原单个数据库分区，那么可以将前滚恢复执行至日志末尾，以将其复原到数据库中其他数据库分区所处的级别。如果前滚一个数据库分区，那么只可使用对日志末尾的恢复。时间点恢复适用于*所有*数据库分区。

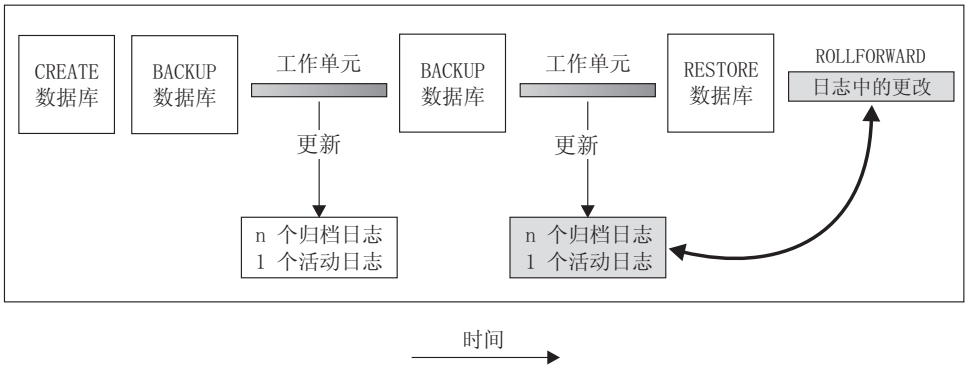


图 22. 数据库前滚恢复. 在运行时间较长的事务中，可以有多个活动日志。

- *表空间前滚恢复*。如果启用了某个数据库对它进行正向恢复，也可对它进行备份、复原并前滚表空间（请参阅第 310 页的图 23）。要执行表空间复原和前滚操作，需要整个数据库（即所有表空间）或一个或多个单独表空间的备份映像。还需要影响要恢复的表空间的日志记录。可在日志中前滚至以下两点之一:
  - 日志末尾；或
  - 一个特定时间点（称为*时间点恢复*）。

可在下列两种情况下使用表空间前滚恢复:

- 在一个表空间复原操作后，该表空间始终处于前滚暂挂状态，且必须将它前滚。调用 **ROLLFORWARD DATABASE** 命令将日志应用于表空间以使其前滚至某个时间点或日志末尾。
- 如果一个或多个表空间在崩溃恢复后处于*前滚暂挂*状态，首先应校正该表空间的问题。某些情况下，校正表空间的问题不涉及复原数据库操作。例如，掉电可能导致表空间处于前滚暂挂状态。在此情况下，复原数据库操作并不是必需的。一旦校正了该表空间的问题，可使用 **ROLLFORWARD DATABASE** 命令将日志应用于表空间，使其恢复至日志末尾。如果在进行崩溃恢复之前解决此问题，那么崩溃恢复操作足以使数据库恢复到一致并且可用的状态。

  注: 如果出错的表空间包含系统目录表，将不能启动数据库。必须复原 SYSCATSPACE 表空间，然后执行前滚恢复直至日志末尾。

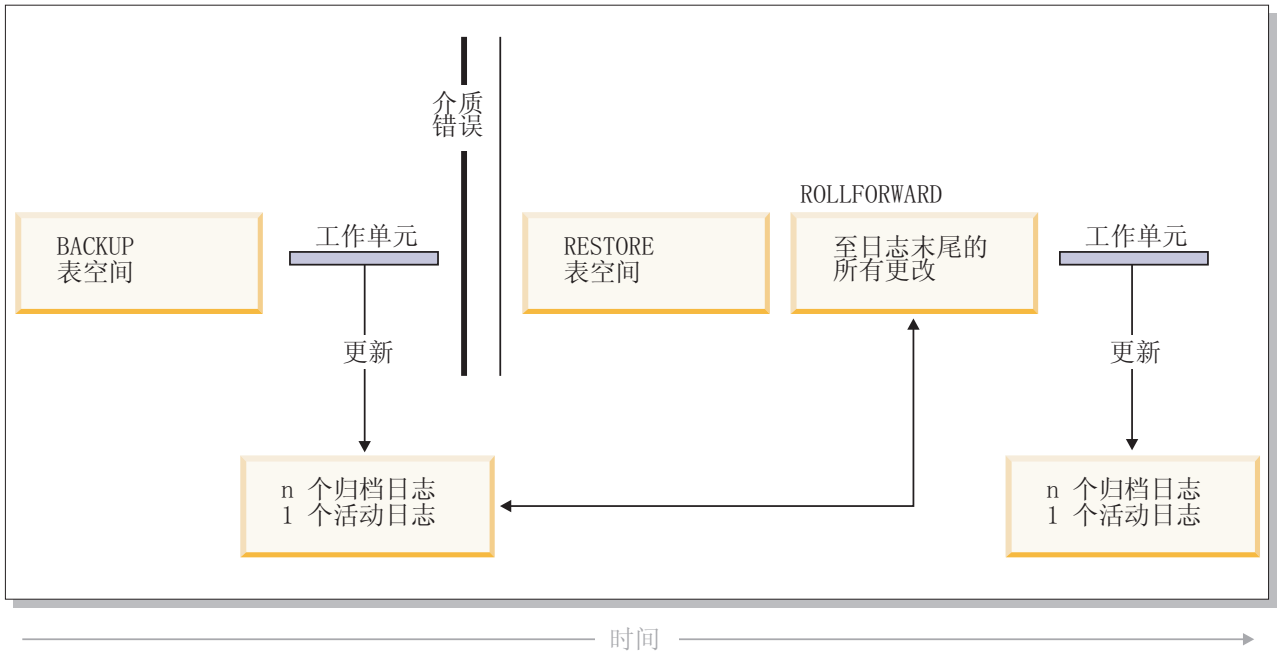


图 23. 表空间前滚恢复. 在运行时间较长的事务中，可以有多个活动日志。

在分区数据库环境中，如果要将表空间前滚*到某个时间点*，那么不必提供该表空间所在的数据库分区列表。DB2 数据库管理器将前滚请求提交到所有数据库分区。这意味着必须在表空间驻所在的所有数据库分区上复原表空间。

在分区数据库环境中，如果要将表空间前滚*至日志末尾*，而又不希望在所有数据库分区上前滚表空间，那么必须提供数据库分区列表。如果要将（所有数据库分区上）所有处于前滚暂挂状态的表空间前滚至日志末尾，那么不必提供数据库分区列表。缺省情况下，会将数据库前滚请求发送至所有数据库分区。

表空间前滚操作在 DB2 pureScale环境中有不同的行为。有关更多信息，请参阅 第 231 页的『DB2 pureScale环境中的日志流合并和日志文件管理』和 第 235 页的『DB2 pureScale环境中的日志序号』。

如果将包含任何分区表部分的表空间前滚到某个时间点，那么还必须将所有其他包含该表的表空间前滚到同一时间点。但是，可以将包含部分分区表的单个表空间前滚至日志末尾。

如果分区表带有任何已连接、已拆离或已删除数据分区，那么时间点前滚还必须包括这些数据分区的所有表空间。要确定某个分区表是否带有任何已连接、已拆离或已删除数据分区，请查询 SYSDATAPARTITIONS 目录表。

前滚恢复期间的存储器组修改

在前滚操作期间是否重做存储器组路径修改取决于您在复原过程中是否重定向了存储器组。如果您在数据库复原操作期间未重新定义存储器组，那么在前滚恢复期间将重放会影响该存储器组或其路径的日志记录。在前滚操作期间将应用这些日志记录中描述的存储路径更新、存储器组重命名操作和表空间存储器组关联更新。如果前滚操作尝试重放与添加存储路径或创建存储器组相关的日志记录，但找不到存储路径，那么将返回错误 SQL1051N。

如果您在复原操作期间重新定义了存储路径，那么前滚操作不会重做对存储路径或您重定向了其路径的存储器组的介质属性的任何更改。然而，将重做对存储器组的数据标记或名称的更改。并且将重放其他操作（包括 DROP STOGROUP 操作）的日志记录。假定已将任何显式指定的存储器组路径设置为其所需的最终路径。

如果在日志中遇到重新平衡操作，那么在前滚恢复期间将启动表空间重新平衡操作。在前滚操作期间，重新平衡操作可能未完成。在那种情况下，在完成前滚操作时将暂挂重新平衡处理，并且在您下次激活数据库时重新启动该处理。

在前滚操作期间，如果在日志中遇到 CREATE STOGROUP 语句，那么将在您发出该 CREATE STOGROUP 语句时指定的路径上创建存储器组。

增量备份与恢复

由于数据库，特别是仓库的大小持续扩展到太字节和百万兆字节范围，备份和恢复这些数据库所需的时间及硬件资源也有了大幅的增长。

因为对大数据库进行多个备份所需的存储量是巨大的，所以完整的数据库和表空间备份并不总是处理大数据库的最佳途径。

请考虑以下问题:

- 仓库中只有一小部分数据更改时，应不需要备份整个数据库。
- 将表空间追加到现有数据库然后只备份表空间是危险的作法，原因是无法保证在表空间备份之间，已备份的表空间外没有任何更改。

为了解决这些问题，DB2 提供了增量备份和恢复。

*增量备份*是一个备份映像，它只包含自上次进行备份以来有过更新的页。除更新的数据和索引页之外，每个增量备份映像还包含通常存储在完整备份映像中的所有初始数据库元数据（例如，数据库配置、表空间定义和数据库历史记录等等）。

注:

1. 如果表空间包含长型字段或大对象数据，并执行增量备份，并且如果自从上次备份之后已修改该表空间中的任何页，那么将把所有长型字段和大对象数据复制到备份映像中。
2. 如果对包含脏页（即，此页包含的数据已被更改但尚未写入磁盘）的表空间执行增量备份，就会备份所有大对象数据。对于正常数据来说，仅当这些数据已更改时，才会对它们进行备份。
3. 数据重新分发可能会为所有新数据库分区创建表空间（如果在 **REDISTRIBUTE DATABASE PARTITION GROUP** 命令上指定了 **ADD DBPARTITIONNUMS** 参数）；这可能会影响增量备份操作。

支持两种类型的增量备份:

- *增量备份*。增量备份映像是自从上次最新的、成功的完整备份操作以来，更改过的所有数据库数据的副本。也称为累积备份映像，因为进行的一系列增量备份中的每一个都会有上次增量备份映像的内容。增量备份映像的前身通常是同一对象最新的、成功的完整备份。

- *差异备份*。差异备份映像或增量差异备份映像是自从上次相关表空间的成功备份（包括完整、增量或差异备份）以来，已更改过的所有数据库数据的副本。也称为差异备份映像或非累积备份映像。差异备份映像的前身是最新的成功备份，包括差异备份映像中每个表空间的备份。

增量备份映像和差异备份映像的关键差别在于：对连续不断更改的对象进行持续备份时，它们的行为不同。每个连续增量映像都包含了前一个增量映像的完整内容以及自上一次生成完整备份后更改过的或新增的任何数据。差异备份映像只包含自上次生成任何类型的映像后更改过的页。

允许以联机和脱机操作方式组合数据库和表空间增量备份。计划备份策略时应格外小心，因为将数据库增量备份和表空间增量备份结合在一起即指示了数据库备份（或多个表空间的表空间备份）的前身不再需要是一个单独的映像，它可以是在不同时间进行的先前数据库和表空间备份的唯一集合。

要将数据库或表空间复原到一致状态，恢复过程必须从要复原的整个对象（数据库或表空间）的一致映像开始，然后必须按以下列表中描述的顺序应用每个相应的增量备份映像。

为启用对数据库更新的跟踪，DB2 支持新的数据库配置参数 **trackmod**，它可以是以下两个可接受的值中的一个：

- NO。此配置不允许使用增量备份。不会以任何方式跟踪或记录数据库页更新。这是缺省值。
- YES。增量备份允许使用此配置。启用了更新跟踪后，更改会对与数据库的第一个成功连接生效。必须选对该表空间进行完整备份，才能对特定表空间进行增量备份。

对于 SMS 和 DMS 表空间，此跟踪的详细程度为表空间级。在表空间级的跟踪中，每个表空间的标志指示该表空间中是否存在需要备份的页。如果表空间不存在任何需要备份的页，那么备份操作可以完全跳过该表空间。

对数据库的更新跟踪会对更新或插入数据的事务的运行时性能产生较小影响。

从增量备份映像复原

通过增量备份映像执行复原操作包括四个步骤。

关于此任务

1. 标识增量目标映像。

   确定要复原的最终映像，并从 DB2 复原实用程序请求增量复原操作。此映像也称为增量复原的目标映像，因为它是要复原的最后映像。增量目标映像是使用 **RESTORE DATABASE** 命令中的 **TAKEN AT** 参数指定的。
2. 复原最新的完整数据库或表空间映像以建立一个基线，以根据它来应用每个后继的增量备份映像。
3. 按产生的顺序，在“步骤 2”中复原的基线映像的顶部，复原需要的各个完整增量备份映像或表空间增量备份映像。
4. 重复“步骤 3”直到“步骤 1”中的目标映像读了两次。在整个增量复原操作期间会访问两次目标映像。在第一次访问期间，只从映像读取初始数据，而不读取任何用户数据。只在第二次访问期间才读取并处理完整的映像。

必须访问两次增量复原操作的目标映像，以确保数据库最初是使用正确的历史记录、数据库配置以及将在复原操作期间创建的数据库表空间定义来配置的。如果自从进行了最初的完整数据库备份映像以来已删除了表空间，将从备份映像中读取该映像的表空间数据，但在增量复原处理期间会忽略这些数据。

有两种方法可用来复原增量备份映像，即自动和手动:

- 对于自动增量复原，仅在指定要使用的目标映像时发出 **RESTORE DATABASE** 命令一次。然后，DB2 for Linux, UNIX, and Windows 使用数据库历史记录来确定余下的必需备份映像并将它们复原。
- 对于手动增量复原，必须对需要复原的每个备份映像都发出一次 **RESTORE DATABASE** 命令（如先前列示的步骤中概述）。

过程

- 要使用自动增量复原来复原一组增量备份映像，请发出 **RESTORE DATABASE** 命令（使用 **TAKEN AT** 参数指定要复原的最后映像的时间戳记），如下所示:

  ```
  db2 restore db sample incremental automatic taken at timestamp
  ```

  这会导致复原实用程序自动执行在本节开头描述的每个步骤。在处理的初始阶段，会读取带有指定时间戳记（以 *yyyymmddhhmmss* 格式指定）的备份映像，然后复原实用程序会验证数据库、其历史记录和表空间定义是否存在并且是否有效。

  在处理的第二阶段，将查询数据库历史记录来构建执行请求的复原操作所需的备份映像链。如果由于某些原因上述操作不可能实现，那么 DB2 for Linux, UNIX, and Windows 无法构建所需映像的完整链，复原操作会终止并返回错误消息。此时，不能进行自动增量复原，必须发出带 **INCREMENTAL ABORT** 参数的 **RESTORE DATABASE** 命令。这将清除所有余下的资源，以便可以继续手动增量复原。

  注: 强烈建议不要使用 **PRUNE HISTORY** 命令的 **WITH FORCE OPTION**。此命令的缺省操作会阻止您删除历史记录条目（从最新完整数据库备份映像恢复时可能会需要它们），但是，使用 **WITH FORCE OPTION** 可能会删除自动复原操作所需的条目。

  在处理的第三个阶段，DB2 for Linux, UNIX, and Windows 将复原生成链中余下的每个备份映像。如果在此阶段期间发生了错误，那么必须发出带 **INCREMENTAL ABORT** 选项的 **RESTORE DATABASE** 命令来清除所有余下的资源。然后，必须先确定是否能解决错误，然后重新发出 **RESTORE DATABASE** 命令或再次尝试手动增量复原。
- 要使用手动增量复原来复原一组增量备份映像，请发出 **RESTORE DATABASE** 命令（使用 **TAKEN AT** 参数指定要复原的每个映像的时间戳记），如下所示:

  1.
  ```
  db2 restore database dbname incremental taken at timestamp
  ```
  其中 *timestamp* 指向要复原的最后增量备份映像（*目标映像*）。

  2.
  ```
  db2 restore database dbname incremental taken at timestamp1
  ```
  其中 *timestamp1* 指向初始完整数据库（或表空间）映像。

  3.
  ```
  db2 restore database dbname incremental taken at timestampX
  ```
  其中 *timestampX* 指向每个增量备份映像（按创建顺序）。

4.

重复步骤 3，复原每个增量备份映像直到映像 *timestamp* 并包括该映像。

如果正在执行数据库复原操作，并已生成表空间备份映像，那么必须以表空间映像备份时间戳记的年代顺序复原这些表空间映像。

可以使用 **db2ckrst** 实用程序来查询数据库历史记录，并生成增量复原所需的备份映像时间戳记列表。同时还生成用于手动增量复原的简化了的复原语法。建议保留备份的完整记录，并仅将此实用程序用作指南。

自动增量复原的局限性

当您需要复原数据库时，自动增量复原很有用。但是，当您决定将如何恢复数据库时，应考虑自动增量复原存在的局限性以防止发生不必要的问题。

下列局限性会影响自动增量复原:

1. 如果在执行要复原的备份操作后已更改了表空间名，并且在发出表空间级别复原操作时使用了新名称，那么将无法根据数据库历史记录正确生成所需的备份映像链，并且会发生错误 (SQL2571N)。

   示例:

   ```
   db2 backup db sample —> <ts1>
   db2 backup db sample incremental —> <ts2>
   db2 rename tablespace from userspace1 to t1
   db2 restore db sample tablespace ('t1') incremental automatic taken
   at <ts2>

   SQL2571N 自动增量复原未能继续。原因码: "3"。
   ```

   建议的变通方法: 使用手动增量复原。

2. 如果删除数据库，数据库历史记录也会删除。如果复原已删除的数据库，数据库历史记录将复原到它在复原备份时的状态，而自那以后的所有历史记录条目都将丢失。如果稍后尝试执行需要使用其中任何一个已丢失的历史记录条目的自动增量复原，那么复原实用程序将尝试复原不正确的备份链，并返回“顺序不对”错误 (SQL2572N)。

   示例:

   ```
   db2 backup db sample —> <ts1>
   db2 backup db sample incremental —> <ts2>
   db2 backup db sample incremental delta —> <ts3>
   db2 backup db sample incremental delta —> <ts4>
   db2 drop db sample
   db2 restore db sample incremental automatic taken at <ts2>
   db2 restore db sample incremental automatic taken at <ts4>
   ```

   建议的变通方法:

   - 使用手动增量复原。
   - 在发出自动增量复原命令前，首先从映像 <ts4> 复原历史记录文件。

3. 如果将备份映像从一个数据库复原到另一个数据库，然后建立增量备份（差异备份），那么不能再使用自动增量复原来复原此备份映像。

   示例:

```
db2 create db a
db2 create db b

db2 update db cfg for a using trackmod on

db2 backup db a —> ts1
db2 restore db a taken at ts1 into b

db2 backup db b incremental —> ts2

db2 restore db b incremental automatic taken at ts2

QL2542N 根据提供的源数据库别名"B"和时间戳记"ts1"，找不到数据库映像文件的匹配项。
```

建议的变通方法:

- 使用如下所示的手动增量复原:

  ```
  db2 restore db b incremental taken at ts2
  db2 restore db a incremental taken at ts1 into b
  db2 restore db b incremental taken at ts2
  ```

- 在对数据库 B 执行手动复原操作之后，发出完整数据库备份以启动新的递增链

优化恢复性能

在恢复数据库期间，您可以使用一些策略来提高 DB2 性能以及缩短从 DB2 服务中断情况进行恢复所需要的时间。

当考虑恢复性能时，应注意下列各项:

- 可以将日志置于单独的设备上，以提高频繁更新的数据库的性能。在联机事务处理 (OLTP) 环境中，更常见的是需要 I/O 来将数据写至日志而不是存储数据行。将日志置于单独的设备上，可将在日志和数据库文件之间进行移动所需的磁盘臂移动最小化。

  还应该考虑该磁盘上的其他文件。例如，将日志移至一个系统中用于系统调页的磁盘，而该磁盘没有足够的实内存，这样会破坏调整。

  通过为缓冲区数目、缓冲区大小和并行性设置选择最佳值，DB2 数据库产品自动尝试将完成备份或复原操作所需的时间降至最小。此值根据可用实用程序堆内存量、可用处理器数和数据库配置而定。
- 要缩短完成一次复原操作所需的时间，请使用多个源设备。
- 如果一个表包含大量的长型字段和 LOB 数据，那么复原它可能会占用非常多的时间。如果允许数据库进行前滚恢复，那么 **RESTORE** 命令能够复原选择的表空间。如果该长型字段和 LOB 数据对于您的业务很重要，应参照完成这些表空间的备份任务所需的时间考虑复原这些表空间。通过将长型字段和 LOB 数据存储在单独的表空间中，不选择复原包含该长型字段和 LOB 数据的表空间，可以减少完成复原操作所需的时间。如果可从单独的源复制 LOB 数据，那么当创建或改变一个表以包括 LOB 列时选择 NOT LOGGED 选项。如果选择不复原包含长型字段和 LOB 数据的表空间，但需要复原包含该表的表空间，那么必须前滚至日志末尾，以便所有包含表数据的表空间都是一致的。

  注: 如果备份包含表数据的一个表空间，而未备份相关的长整数或 LOB 字段，那么不能对该表空间执行时间点前滚恢复。必须将一个表的所有表空间同时前滚至同一个时间点。

- 下列说明适用于备份和复原操作:
  - 应该使用多个设备。
  - 不要使 I/O 设备控制器带宽超负荷。
- DB2 数据库产品使用多个代理程序来执行崩溃恢复和数据库前滚恢复。您可能会发现执行这些操作期间的性能更佳（特别是在对称多处理器 (SMP) 机器上）；在数据库恢复期间使用多个代理程序可利用 SMP 机器上可用的多余 CPU。

  并行恢复引入的代理程序类型是 **db2agnsc**。DB2 数据库管理器根据机器上 CPU 的数量选择将用于数据库恢复的代理程序的数量。

  DB2 数据库管理器将日志记录分发到这些代理程序以使它们可在适当的时候并发地重新应用。例如，可通过此方法使那些与插入、删除、更新、添加键和删除键操作关联的日志记录的处理并行化。因为日志记录是在页级并行化的（相同数据页上的日志记录由相同的代理程序处理），所以即使所有工作都是对一个表完成的，性能还是会增强。
- 执行恢复操作时，DB2 数据库管理器将自动为缓冲区个数、缓冲区大小和并行性设置选择最佳值。此值根据可用实用程序堆内存的数量、可用处理器数和数据库配置而定。因此，根据系统上可用的存储量，应考虑通过增大 **util_heap_sz** 配置参数来分配更多内存。

使用 recover 所需的特权、权限和授权

必须具有 SYSADM、SYSCTRL 或 SYSMAINT 权限才能使用恢复实用程序。

特权使用户能够创建或访问数据库资源。权限级别提供了对特权、较高级别数据库管理器维护和实用程序操作进行分组的方法。这两者一起用于控制对数据库管理器及其数据库对象的访问。

用户只能访问那些他们具有相应授权（即必需的特权或权限）的对象。

第 13 章 restore 概述

您可以使用 DB2 复原工具将 DB2 数据库复原到先前状态。必须存在该数据库的备份映像之后，您才能使用这些工具。

DB2 **RESTORE DATABASE** 命令的最简单格式只需要您指定要复原的数据库的别名。例如:

```
db2 restore db sample
```

在此示例中，因为 SAMPLE 数据库存在且在发出 **RESTORE DATABASE** 命令时将被替换，所以返回以下消息:

```
SQL2539W  警告！正在复原到与备份映像数据库相同的现有数据库。数据库文件将被删除。
想要继续吗？(y/n)
```

如果指定 y，复原操作将成功完成。

数据库复原操作需要一个互斥连接: 即，启动该任务后，复原实用程序会防止其他应用程序访问数据库，直到复原操作成功完成，所以不能再对该数据库运行任何应用程序。但表空间复原操作可以联机完成。

直到复原操作（可能后跟前滚恢复）成功完成，表空间才可用。

如果有跨越多个表空间的表，那么应该一起备份并复原这个表空间集合。

执行部分或子集复原操作时，可以使用表空间级别的备份映像或完整数据库级别的备份映像，并从该映像中选择一个或多个表空间。从建立备份映像时开始的所有与这些表空间相关的日志文件必须存在。

可以将数据库从在 32 位级别创建的备份映像复原为 64 位级别，但反之则不然。

如果您要将备份从 32 位级别环境复原到 64 位级别环境，请复查数据库配置参数，以确保针对 64 位实例环境优化了这些配置参数。例如，在 32 位环境中，语句堆的缺省值低于 64 位环境中相应的缺省值。

DB2 备份实用程序和复原实用程序应该用来备份和复原数据库。建议不要将文件集从一台机器移至另一台机器，因为这可能破坏数据库的完整性。

在某些情况下，可将可传输集与 **RESTORE DATABASE** 命令配合使用来移动数据库。.

在 IBM Data Studio V3.1 或更高版本中，可以使用以下工具的任务助手: 复原数据库备份. 任务助手可以指导您执行以下过程: 设置选项、查看自动生成的命令以执行任务以及运行这些命令。有关更多详细信息，请参阅使用任务助手管理数据库。

使用复原

在发生问题（例如介质或存储器故障，或者应用程序故障）后使用 **RESTORE DATABASE** 命令来恢复数据库或表空间。如果已经备份了数据库或各个表空间，那么可以在它们由于某种原因被损坏或毁坏时对其进行重建。

© Copyright IBM Corp. 2001, 2013

开始之前

复原到*现有*数据库时，不应连接至要复原的数据库：RESTORE 实用程序自动建立与特定数据库的连接，并且此连接在复原操作完成时终止。复原到*新*数据库时，需要实例连接才能创建数据库。复原到*新远程*数据库时，必须首先连接至新数据库所在的实例。然后创建新数据库，指定服务器的代码页和地域。复原将使用备份映像的代码页来覆盖目标数据库的代码页。

关于此任务

数据库可以是本地数据库或远程数据库。

RESTORE 实用程序有以下限制:

- 仅当先前已使用 DB2 BACKUP 实用程序备份了数据库时，才能使用 RESTORE 实用程序。
- 如果非实例所有者用户（在 UNIX 上）或者 DB2ADMNS 或 Administrators 组的成员（在 Windows 上）尝试复原备份映像，那么他们将接收到错误（SQL2061N）。如果其他用户需要访问备份映像，那么需要在生成备份后更改文件许可权。
- 数据库复原操作在正在运行前滚进程时不能启动。
- 如果未指定 **TRANSPORT** 选项，那么只有当表空间当前已存在并且是同一表空间时，才可以将该表空间复原到现有数据库中。在此情况下，“同一表空间”表示在备份与复原操作之间未删除并重新创建该表空间。磁盘上的数据库与备份映像中的数据库必须相同。
- 对于表空间级备份来说，不能对其执行表空间级复原以将其复原到新数据库。
- 不能执行涉及系统目录表的联机表空间级复原操作。
- 对于在单一数据库分区环境中创建的备份来说，不能将其复原到现有分区数据库环境中。而是，必须将该备份复原到单一数据库分区环境，然后根据需要添加数据库分区。
- 将使用一个代码页的备份映像复原成使用另一个代码页的系统时，备份映像的代码页将覆盖系统代码页。
- 不能使用 **RESTORE DATABASE** 命令将未启用自动存储器的表空间转换为自动存储器启用的表空间。
- 当指定 **TRANSPORT** 选项时，会存在下列限制:
  - 如果备份映像可由复原操作进行复原且支持升级，那么可以传输该备份映像。
  - 如果使用了联机备份，那么源数据服务器和目标数据服务器都必须运行相同的 DB2 版本。
  - 必须对目标数据库发出 **RESTORE DATABASE** 命令。如果远程客户机与服务器使用的平台相同，那么模式传输既可采用本地方式在服务器上执行，也可以通过远程实例连接来执行。如果目标数据库是编目在以本地方式运行传输的实例中的远程数据库，那么对该远程目标数据库的模式传输将不受支持。
  - 只可以将表空间和模式传输到现有数据库中。传输操作将不会创建新数据库。要将数据库复原到新数据库中，可以使用 **RESTORE DATABASE** 命令而不指定 **TRANSPORT** 选项。
  - 如果源数据库中的模式受任何 DB2 安全性设置或权限保护，那么目标数据库中已传输的模式将保留这些相同设置或权限。
  - 分区数据库环境不支持传输。

- 如果模式内的任何表包含 XML 列，那么传输将失败。
- **TRANSPORT** 选项与 REBUILD 选项不兼容。
- 从快照备份映像复原功能不支持 **TRANSPORT** 选项。
- 必须对数据库恢复启用目标数据库。
- 创建登台数据库，以用于传输。该数据库不能用于其它操作。
- 登台表和目标表上的数据库配置参数需要相同，否则，传输操作会由于不兼容错误而失败。
- 必须在目标数据库上将 **auto_reval** 配置参数设置为 deferred_force，才能传输已列示为无效的对象。否则，传输将失败。
- 如果使用了联机备份映像但不包括活动日志，那么传输操作将失败。
- 如果使用联机备份，那么该备份映像必须已使用 INCLUDE LOGS 选项创建
- 如果备份映像源于前发行版，那么该映像必须是完全脱机数据库级别的备份映像。
- 如果登台数据库或目标数据库发生错误，那么必须重新发出整个恢复操作。发生的所有失败都记录在目标服务器上的 **db2diag** 日志文件中，且重新发出 **RESTORE** 命令之前应先检查这些失败。
- 如果传输客户机失败，那么登台数据库可能不会被适当清除。在这种情况下，需要删除该登台数据库。在重新发出 **RESTORE** 命令之前，请先删除所有登台数据库，以防止登台数据库中的容器阻止后续传输。
- 不支持对同一目标数据库运行并发传输。
- 表空间传输不支持生成重定向复原脚本。

- 如果已更新存储器组，那么您可以复原表空间。表空间复原期间的目标存储器组是执行 **RESTORE** 时当前与该表空间相关联的存储器组。
- 不能对早期的存储器组关联执行时间点恢复。

过程

要调用 RESTORE 实用程序，请执行以下操作:

- 发出 **RESTORE DATABASE** 命令。
- 调用 db2Restore 应用程序编程接口 (API)。
- 在 IBM Data Studio 中对 **RESTORE DATABASE** 命令打开任务助手。

示例

以下是通过 CLP 发出的 **RESTORE DATABASE** 命令的示例:

```
db2 restore db sample from D:\DB2Backups taken at 20010320122644
```

从快照备份映像复原

从快照备份复原使用存储设备的快速复制技术来执行复原的数据复制部分。

开始之前

要执行快照备份和复原操作，需要适用于存储设备的 DB2 ACS API 驱动程序。有关集成驱动程序的受支持存储硬件的列表，请参阅 Tivoli 文档，网址为: 受支持存储子系统

必须执行快照备份，然后才可以从快照备份复原。请参阅: 第 260 页的『执行快照备份』。

过程

可以使用带 **USE SNAPSHOT** 参数的 **RESTORE DATABASE** 命令或带有 SQLU_SNAPSHOT_MEDIA 介质类型的 db2Restore API 来从快照备份复原:

- **RESTORE DATABASE** 命令:

```
db2 restore db sample use snapshot
```

- db2Restore API:

```
int sampleRestoreFunction( char dbAlias[],
                           char restoredDbAlias[],
                           char user[],
                           char pswd[],
                           char workingPath[] )
{
  db2MediaListStruct mediaListStruct = { 0 };

  rmediaListStruct.locations = &workingPath;
  rmediaListStruct.numLocations = 1;
  rmediaListStruct.locationType = SQLU_SNAPSHOT_MEDIA;


  db2RestoreStruct restoreStruct = { 0 };

  restoreStruct.piSourceDBAlias = dbAlias;
  restoreStruct.piTargetDBAlias = restoredDbAlias;
  restoreStruct.piMediaList = &mediaListStruct;
  restoreStruct.piUsername = user;
  restoreStruct.piPassword = pswd;
  restoreStruct.iCallerAction = DB2RESTORE_STORDEF_NOINTERRUPT;


  struct sqlca sqlca = { 0 };


  db2Restore(db2Version900, &restoreStruct, &sqlca);

      return 0;
}
```

复原到现有的数据库

对于数据库级别复原来说，备份映像可以在别名、数据库名称或数据库*种子值*等方面与现有数据库不同。数据库种子值是数据库的唯一标识，它在该数据库的生命期内不会更改。

数据库管理器在您创建数据库时指定种子值。DB2 始终使用备份映像中的种子值。仅当表空间存在并且保持不变（这表示在备份和复原操作之间未删除并重新创建该表空间）时，您才能将该表空间复原到现有数据库中。磁盘上的数据库与备份映像中的数据库必须相同。在复原表空间时，不能修改当前定义的存储器组，也不能显式创建新的存储器组。

复原到现有数据库时，复原实用程序会执行下列操作:

- 删除现有数据库中的表、索引和长型字段数据，并将其替换为备份映像中的数据。
- 替换要复原的每个表空间的表条目。
- 保留恢复历史记录文件，除非它已损坏或者没有任何条目。如果恢复历史记录文件已损坏或不包含任何条目，数据库管理器会从备份映像中复制文件。如果要替换恢复历史记录文件，可以发出带 **REPLACE HISTORY FILE** 参数的 **RESTORE DATABASE** 命令。
- 保留现有数据库的认证类型。
- 保留现有数据库的数据库目录。这些目录定义数据库所在的位置及其编目方式。
- 比较数据库种子值。如果这些种子值不相同，那么该实用程序会执行下列操作:
  - 删除与现有数据库相关联的日志。
  - 从备份映像中复制数据库配置文件。
  - 如果您指定了 **NEWLOGPATH** 参数，那么该实用程序会将 **RESTORE DATABASE** 命令的 **NEWLOGPATH** 参数设置为 **logpath** 数据库配置参数的值。

  如果数据库种子值相同，那么该实用程序会执行下列操作:
  - 如果映像是不可恢复的数据库的映像，那么删除所有日志文件。
  - 如果映像是可恢复的数据库的映像，那么删除空的日志文件。非空日志文件将不受影响。
  - 保留当前数据库配置文件。
  - 如果您指定了 **NEWLOGPATH** 参数，那么该实用程序会将 **RESTORE DATABASE** 命令的 **NEWLOGPATH** 参数设置为 **logpath** 数据库配置参数的值。否则，该实用程序会将当前日志路径复制到数据库配置文件中。验证日志路径。如果数据库不能使用该路径，那么该实用程序会更改数据库配置以使用缺省日志路径。

复原到新的数据库

可以创建一个新数据库并向它复原完整的数据库备份映像。如果未创建新的数据库，那么 RESTORE 实用程序将创建一个数据库。

复原到新数据库时，RESTORE 实用程序:

- 使用通过目标数据库别名参数指定的数据库别名创建新数据库。（如果未指定目标数据库别名，RESTORE 实用程序会使用通过源数据库别名参数指定的相同别名来创建数据库。）
- 从备份映像复原数据库配置文件。
- 如果在 **RESTORE DATABASE** 命令上指定了 **NEWLOGPATH**，请将 **NEWLOGPATH** 设置为 **logpath** 数据库配置参数的值。验证日志路径: 如果该路径不能由数据库使用，应更改数据库配置以使用缺省日志路径。
- 从备份映像复原认证类型。
- 从备份映像中的数据库目录复原注释。
- 复原数据库的恢复历史记录文件。
- 使用备份映像的代码页来覆盖数据库代码页。

在测试和生产环境中使用增量复原

一旦对生产数据库启用增量备份与恢复，就可以使用增量或差异备份映像来创建或刷新测试数据库。

可以通过使用手动或自动增量复原来完成此任务。

要将备份映像从生产数据库复原至测试数据库，在 **RESTORE DATABASE** 命令上使用 INTO *target-database-alias* 选项。例如，在具有下列备份映像的生产数据库中:

```
backup db prod
Backup successful. The timestamp for this backup image is : ts1

   backup db prod incremental
Backup successful. The timestamp for this backup image is : ts2
```

手动增量复原的一个示例将是:

```
restore db prod incremental taken at ts2 into test without
prompting
DB20000I  已成功完成 RESTORE DATABASE 命令。
restore db prod incremental taken at ts1 into test without
prompting
DB20000I  已成功完成 RESTORE DATABASE 命令。
restore db prod incremental taken at ts2 into test without
prompting
DB20000I  已成功完成 RESTORE DATABASE 命令。
```

如果数据库 TEST 已存在，那么复原操作会覆盖该处已存在的任何数据。如果数据库 TEST 不存在，那么 RESTORE 实用程序将创建该数据库并使用备份映像中的数据对它进行填充。

因为自动增量复原操作依赖于数据库历史记录，所以根据测试数据库是否存在，复原步骤将稍微有所变化。要对数据库 TEST 执行自动增量复原，它的历史记录必须包含数据库 PROD 的备份映像历史记录。如果符合下列任一条件，那么该备份映像的数据库历史记录将替换对于数据库 TEST 已存在的所有数据库历史记录:

- 发出 **RESTORE DATABASE** 命令时数据库 TEST 不存在。
- 发出 **RESTORE DATABASE** 命令时数据库 TEST 存在，并且数据库 TEST 历史记录未包含任何数据。

以下示例显示对不存在的数据库 TEST 的自动增量复原:

```
restore db prod incremental automatic taken at ts2 into test without
prompting
DB20000I  已成功完成 RESTORE DATABASE 命令。
```

RESTORE 实用程序创建 TEST 数据库并填充该数据库。

如果数据库 TEST 确实存在且数据库历史记录不为空，那么必须删除该数据库，才能执行自动增量复原操作，如下所示:

```
drop db test
   DB20000I  已成功完成 DROP DATABASE 命令。

   restore db prod incremental automatic taken at ts2 into test without
   prompting
   DB20000I  已成功完成 RESTORE DATABASE 命令。
```

如果不想删除该数据库，那么可在发出 **RESTORE DATABASE** 命令之前发出带较远将来时间戳记和 **WITH FORCE OPTION** 参数的 **PRUNE HISTORY** 命令:

```
connect to test
Database Connection Information

Database server           = server_id
SQL authorization ID      = id
```

```
Local database alias       = TEST

   prune history 9999 with force option
DB20000I  The PRUNE command completed successfully.

connect reset
DB20000I  The SQL command completed successfully.
restore db prod incremental automatic taken at ts2 into test without
prompting
SQL2540W  Restore is successful, however a warning "2539" was
encountered during Database Restore while processing in No
Interrupt mode.
```

在此情况下，**RESTORE DATABASE** 命令的行为方式与数据库 TEST 不存在时的行为方式相同。

如果数据库 TEST 确定存在且数据库历史记录为空，不必删除数据库 TEST 就可以执行自动增量复原操作:

```
   restore db prod incremental automatic taken at ts2 into test without
prompting
SQL2540W  Restore is successful, however a warning "2539" was
encountered during Database Restore while processing in No
Interrupt mode.
```

可以继续建立测试数据库的增量或差异备份而不用先建立完整数据库备份。但是，如果需要复原其中一个增量映像或差异映像，那么必须执行手动增量复原。需要这样做是因为自动增量复原操作要求自动增量复原期间复原的每个备份映像是根据同一个数据库别名创建的。

如果在使用生产备份映像完成复原操作之后建立测试数据库的完整备份映像，那么可以建立增量或差异备份并通过使用手动或自动方式将它们复原。

执行重定向复原操作

数据库复原操作将使用数据库备份映像来重新创建数据库。

在下列任何情况下，请使用重定向复原操作:

- 如果要将备份映像复原到不同于源机器的目标机器
- 如果要将表空间容器复原到另一个物理位置
- 如果复原操作由于一个或多个容器不可访问而失败
- 如果要重新定义已定义的存储器组的路径

限制: 不能使用重定向复原操作将数据从一个操作系统移至另一个操作系统。

不能在复原过程中创建或删除存储器组。

不能在表空间复原过程中修改存储器组路径，即使正在复原与存储器组相关联的所有表空间时也是如此。

使用增量备份映像来执行重定向复原的过程与使用非增量备份映像来执行重定向复原的过程类似。请使用下列其中一种方法:

- 发出带 **REDIRECT** 参数的 **RESTORE DATABASE** 命令，并指定要用于对数据库进行增量复原的备份映像。
- 根据备份映像生成重定向复原脚本，然后按需要对该脚本进行修改。

使用 **RESTORE DATABASE** 命令的方法是一个两步骤的数据库复原过程，其中一个步骤是干预步骤，用于定义表空间容器或存储器组路径。要执行重定向复原:

1. 发出带 **REDIRECT** 参数的 **RESTORE DATABASE** 命令。
2. 执行下列其中一个步骤:
   - 通过发出 **SET TABLESPACE CONTAINERS** 命令来定义表空间容器。
   - 通过发出 **SET STOGROUP PATHS** 命令来定义要复原的数据库的存储器组路径。
3. 再次发出 **RESTORE DATABASE** 命令，这次指定 **CONTINUE** 参数。

在发出 **RESTORE CONTINUE** 命令之后，新路径生效，将用作所有相关联的表空间的表空间容器路径。如果在 **SET STOGROUP PATHS** 命令之后和 **RESTORE CONTINUE** 命令之前发出 **LIST TABLESPACE CONTAINERS** 命令或 **GET SNAPSHOT FOR TABLESPACES** 命令，那么表空间容器路径的输出不会反映您使用 **SET STOGROUP PATHS** 命令指定的新路径。

在重定向复原操作期间，如果目录和文件容器不存在，那么将自动创建目录和文件容器。数据库管理器不会自动创建设备容器。

DB2 数据库产品提供了一些 SQL 语句，用于添加、更改或除去表空间容器的非自动存储器 DMS 表空间以及自动存储器表空间的存储器组路径。重定向复原是唯一可用来修改非自动存储器 SMS 表空间容器配置的方法。

通过发出带 **REDIRECT** 参数的 **RESTORE DATABASE** 命令，可以重新定义表空间容器或修改存储器组路径。

表空间容器重定向操作为管理表空间容器提供了相当大的灵活性。您可以在从备份映像复原任何数据页之前改变数据库的存储器组配置，其方法类似于您重定向表空间容器路径的方法。如果在生成备份映像之后重命名了存储器组，那么由 **SET STOGROUP PATHS** 命令指定的存储器组名称指的是备份映像中的存储器组名称，而不是当前名称。

在分区数据库环境中执行重定向复原操作

在分区数据库环境中的重定向数据库复原期间，您只能将存储器组路径重定向到目录数据库分区中的新存储器组路径。对于所有其他数据库分区，您必须使其存储器组路径与目录分区的存储器组路径同步。

修改目录分区上的任何存储器组路径时，会将所有非目录分区置于 RESTORE_PENDING 状态。如果您重定向存储器组路径，那么必须在复原任何其他数据库分区之前复原目录分区。在复原目录数据库分区之后，可以并行复原非目录数据库分区，而不需要重定向任何存储器组路径。非目录数据库分区会自动获取您为目录数据库分区指定的新存储器组路径。在复原不同的数据库（具有不同名称、实例或种子值的数据库），如果数据库复原期间隐式更改了存储器组路径，那么也会自动获取新存储器组路径。

如果自最近一次进行备份后已修改存储器组路径，那么您仍然可以使用该备份映像（使用不同的存储器组路径）来复原任何数据库分区。此复原不会视为重定向复原。从该备份映像复原时，会导致数据库分区临时使用您在创建该备份时定义的存储器组路径。请执行前滚恢复以重新应用存储器组路径修改并使所有数据库分区再同步。

示例

示例 1

通过使用 **SET TABLESPACE CONTAINERS** 命令来定义表空间容器，可以对数据库 SAMPLE 执行表空间容器重定向复原:

```
db2 restore db sample redirect without prompting
 SQL1277W 正在执行重定向复原操作。
During a table space restore, only table spaces being restored can
have their paths reconfigured. During a database restore, storage
group storage paths and DMS table space containers can be reconfigured.

DB20000I  已成功完成 RESTORE DATABASE 命令。
   db2 set tablespace containers for 2 using (path 'userspace1.0', path
   'userspace1.1')
   DB20000I 已成功完成 SET TABLESPACE CONTAINERS 命令。
   db2 restore db sample continue
   DB20000I  已成功完成 RESTORE DATABASE 命令。
```

示例 2

可以使用 **SET STOGROUP PATHS** 命令来重新定义已定义的存储器组的路径:

```
RESTORE DB SAMPLE REDIRECT

SET STOGROUP PATHS FOR sg_hot ON '/ssd/fs1', '/ssd/fs2'
SET STOGROUP PATHS FOR sg_cold ON '/hdd/path1', '/hdd/path2'

RESTORE DB SAMPLE CONTINUE
```

示例 3

以下是对别名为 MYDB 的数据库的典型非增量重定向复原方案:

1. 发出 RESTORE DATABASE 命令，使用 REDIRECT 选项。

   ```
   db2 restore db mydb replace existing redirect
   ```

2. 对想要重新定义其容器的每个表空间发出 SET TABLESPACE CONTAINERS 命令。例如，在 Windows 环境中:

   ```
   db2 set tablespace containers for 5 using
                 (file 'f:\ts3con1'20000, file 'f:\ts3con2'20000)
   ```

   要验证复原数据库的容器是否是在此步骤中指定的那些容器，对正重新定义其容器位置的每个表空间发出 LIST TABLESPACE CONTAINERS 命令。

3. 在成功完成了步骤 1 和 2 之后，发出:

   ```
   db2 restore db mydb continue
   ```

   这是“重定向复原”操作的最后一步。

4. 如果步骤 3 失败，或者如果中止了复原操作，那么可从步骤 1 开始重新启动重定向的复原。

注:

1. 成功完成步骤 1 之后且在完成步骤 3 之前，通过发出以下命令来中止复原操作:

   ```
   db2 restore db mydb abort
   ```

2. 如果步骤 3 失败，或者如果中止了复原操作，那么可从步骤 1 开始重新启动重定向的复原。

示例 4

以下是对别名为 MYDB 且具有下列备份映像的数据库的典型手动增量重定向复原方案:

```
   backup db mydb
Backup successful. The timestamp for this backup image is : <ts1>

backup db mydb incremental
Backup successful. The timestamp for this backup image is : <ts2>
```

1. 发出带 INCREMENTAL 和 REDIRECT 选项的 RESTORE DATABASE 命令。

   ```
   db2 restore db mydb incremental taken at <ts2> replace existing redirect
   ```

2. 对必须重新定义容器的每个表空间发出 SET TABLESPACE CONTAINERS 命令。例如，在 Windows 环境中:

   ```
   db2 set tablespace containers for 5 using
                 (file 'f:\ts3con1'20000, file 'f:\ts3con2'20000)
   ```

   要验证复原数据库的容器是否是在此步骤中指定的那些容器，可发出 LIST TABLESPACE CONTAINERS 命令。

3. 在成功完成了步骤 1 和 2 之后，发出:

   ```
   db2 restore db mydb continue
   ```

4. 现在可以发出余下的增量复原命令，如下所示:

   ```
   db2 restore db mydb incremental taken at <ts1>
   db2 restore db mydb incremental taken at <ts2>
   ```

   这是“重定向复原”操作的最后一步。

注:

1. 成功完成步骤 1 之后且在完成步骤 3 之前，通过发出以下命令来中止复原操作:

   ```
   db2 restore db mydb abort
   ```

2. 成功完成步骤 3 之后且在发出步骤 4 中所有必需的命令之前，可以通过发出以下命令来中止复原操作:

   ```
   db2 restore db mydb incremental abort
   ```

3. 如果步骤 3 失败，或者如果中止了复原操作，那么可从步骤 1 开始重新启动重定向的复原。
4. 如果步骤 4 中的复原命令失败，那么可以重新发出失败的命令以继续复原过程。

示例 5

以下是对同一个数据库的典型自动增量重定向复原方案:

1. 发出带 INCREMENTAL AUTOMATIC 和 REDIRECT 选项的 RESTORE DATABASE 命令。

   ```
   db2 restore db mydb incremental automatic taken at <ts2>
                 replace existing redirect
   ```

2. 对必须重新定义容器的每个表空间发出 SET TABLESPACE CONTAINERS 命令。例如，在 Windows 环境中:

   ```
   db2 set tablespace containers for 5 using
                 (file 'f:\ts3con1'20000, file 'f:\ts3con2'20000)
   ```

   要验证复原数据库的容器是否是在此步骤中指定的那些容器，可发出 LIST TABLESPACE CONTAINERS 命令。

3. 在成功完成了步骤 1 和 2 之后，发出:

```
db2 restore db mydb continue
```

这是“重定向复原”操作的最后一步。

注:

1. 成功完成步骤 1 之后且在完成步骤 3 之前，通过发出以下命令来中止复原操作:

```
db2 restore db mydb abort
```

2. 如果步骤 3 失败，或者如果复原操作已中止，那么可以在发出以下命令之后从步骤 1 开始重新启动重定向复原:

```
db2 restore db mydb incremental abort
```

通过使用自动生成的脚本复原数据库来重新定义表空间容器

复原数据库时， RESTORE 实用程序假定物理容器布局与备份数据库时的物理容器布局相同。

如果需要更改任何物理容器的位置或大小，那么必须发出带 **REDIRECT** 选项的 **RESTORE DATABASE** 命令。使用此选项时，需要指定存储在备份映像中的物理容器的位置，并且需要提供每个将要改变的非自动表空间的全部容器。可以在备份时捕获容器信息，但是这样做会很繁琐。

为了便于执行重定向复原，RESTORE 实用程序允许根据现有备份映像生成重定向复原脚本，方法是发出带 **REDIRECT** 参数和 **GENERATE SCRIPT** 参数的 **RESTORE DATABASE** 命令。RESTORE 实用程序将检查备份映像、从备份映像中提取容器信息并生成 CLP 脚本，该脚本包含容器的所有详细信息。接着，可以在脚本中修改任何路径或容器大小，然后运行该 CLP 脚本以使用一组新容器来重新创建数据库。即使只有备份映像，而不了解容器布局，也可以使用生成的脚本来复原数据库。该脚本是在客户机上创建的。通过使用该脚本作为基础，可以确定复原后数据库的日志文件和容器所需空间的位置，并且可以相应地更改日志文件和容器路径。

生成的脚本由四节组成:

初始化 第一节设置命令选项并指定要运行命令的数据库分区。以下是第一节的示例:

```
UPDATE COMMAND OPTIONS USING S ON Z ON SAMPLE_NODE0000.out V ON;
SET CLIENT ATTACH_DBPARTITIONNUM  0;
SET CLIENT CONNECT_DBPARTITIONNUM 0;
```

其中:

- S ON 指定命令出错时应该停止执行该命令
- Z ON SAMPLE_NODE0000.out 指定应该将输出定向到名为 *dbalias*_NODE*dbpartitionnum*.out 的文件
- V ON 指定应该将当前命令打印到标准输出。

在分区数据库环境中运行脚本时，需要指定要运行脚本命令的数据库分区，这一点尤其重要。

带 REDIRECT 参数的 RESTORE DATABASE 命令

第二节启动 **RESTORE DATABASE** 命令并使用 **REDIRECT** 参数。此节可以使用所有 **RESTORE DATABASE** 命令参数，不能与 **REDIRECT** 参数配合使用的任何参数除外。以下是第二节的示例:

```
   RESTORE DATABASE SAMPLE
-- USER 'username'
-- USING 'password'
FROM '/home/jseifert/backups'
   TAKEN AT 20050906194027
-- DBPATH ON 'target-directory'
   INTO SAMPLE
-- NEWLOGPATH '/home/jseifert/jseifert/NODE0000/SQL00001/LOGSTREAM0000/'
-- WITH num-buff BUFFERS
-- BUFFER buffer-size
   -- REPLACE HISTORY FILE
   -- REPLACE EXISTING
   REDIRECT
-- PARALLELISM n
   -- WITHOUT ROLLING FORWARD
   -- WITHOUT PROMPTING
;
```

表空间定义

此节包含备份映像中包含的或命令行上指定的每个表空间的表空间定义。每个表空间都有一节，每一节都包含注释块，该注释块包含有关表空间名称、类型和大小的信息。该信息是使用表空间快照的格式提供的。可以使用提供的信息来确定所需的表空间大小。查看使用自动存储器创建的表空间的输出时，您将看不到 SET TABLESPACE CONTAINERS 子句。以下是表空间定义节的示例:

```
-- *****************************************************************
-- ** 表空间名称                          = SYSCATSPACE
-- **   表空间标识                        = 0
-- **   表空间类型                      = 系统管理空间
-- **   表空间内容类型                   = 任意数据
-- **   表空间页大小（字节）             = 4096
-- **   表空间扩展数据块大小（页）       = 32
-- **   使用自动存储器                   = No
-- **   总页数                           = 5572
-- *****************************************************************
SET TABLESPACE CONTAINERS FOR 0
-- IGNORE ROLLFORWARD CONTAINER OPERATIONS
                  USING (
  PATH   'SQLT0000.0'
);
-- *****************************************************************
-- ** 表空间名称                          = TEMPSPACE1
-- **   表空间标识                        = 1
-- **   表空间类型                      = 系统管理空间
-- **   表空间内容类型                     = 系统临时数据
-- **   表空间页大小（字节）             = 4096
-- **   表空间扩展数据块大小（页）       = 32
-- **   使用自动存储器                   = No
-- **   总页数                           = 0
-- *****************************************************************
SET TABLESPACE CONTAINERS FOR 1
-- IGNORE ROLLFORWARD CONTAINER OPERATIONS
                  USING (
  PATH   'SQLT0001.0'
);
-- *****************************************************************
-- ** 表空间名称                          = DMS
-- **   表空间标识                        = 2
-- **   表空间类型                        = 数据库管理空间
-- **   表空间内容类型                   = 任意数据
```

```
-- **   表空间页大小（字节）                = 4096
-- **   表空间扩展数据块大小（页）          = 32
-- **   使用自动存储器                      = No
-- **   启用自动调整大小                      = No
-- **   总页数                                = 2000
-- **   可用页数                              = 1960
-- **   高水位标记（页）                      = 96
-- ********************************************************************
SET TABLESPACE CONTAINERS FOR 2
-- IGNORE ROLLFORWARD CONTAINER OPERATIONS
                  USING (
  FILE   '/tmp/dms1'                                          1000
, FILE   '/tmp/dms2'                                          1000
);
```

带 CONTINUE 参数的 RESTORE DATABASE 命令

最后一节发出带 **CONTINUE** 参数的 **RESTORE DATABASE** 命令以完成重定向复原。以下是最后一节的示例:

```
RESTORE DATABASE SAMPLE CONTINUE;
```

使用自动生成的脚本来执行重定向复原

执行重定向复原操作时，必须指定备份映像中存储的物理容器的位置，并且必须为每个要改变的表空间提供一组完整的容器。

开始之前

仅当先前已使用 DB2 备份实用程序备份了数据库时，才能执行重定向复原。

关于此任务

- 如果数据库存在，您必须能够连接至该数据库才能生成脚本。因此，如果数据库需要升级或崩溃恢复，那么必须在尝试生成重定向复原脚本之前进行此操作。
- 如果正在分区数据库环境中工作，并且目标数据库不存在，那么不能运行该命令来在所有数据库分区上同时生成重定向复原脚本。相反，一次只能在一个数据库分区上运行用于生成重定向复原脚本的命令，从目录分区开始。

  或者，可以先创建一个与目标数据库具有相同名称的哑元数据库。在创建了哑元数据库之后，可以在所有数据库分区上同时生成重定向复原脚本。
- 在发出 **RESTORE DATABASE** 命令以生成脚本时，即使指定了 **REPLACE EXISTING** 参数，脚本中出现的 **REPLACE EXISTING** 参数也会被注释掉。
- 为了提高安全性，密码不会出现在生成的脚本中。您需要手动输入密码。
- 复原脚本包括您复原的每个表空间的存储器组关联。

过程

要使用脚本来执行重定向复原:

1. 使用复原实用程序来生成重定向复原脚本。 可以通过命令行处理器 (CLP) 或 db2Restore 应用程序编程接口 (API) 来调用复原实用程序。 以下是指定了 **REDIRECT** 参数和 **GENERATE SCRIPT** 参数的 **RESTORE DATABASE** 命令实例:

   ```
   db2 restore db test from /home/jseifert/backups taken at 20050304090733
               redirect generate script test_node0000.clp
   ```

   此命令将在客户机上创建名为 `test_node0000.clp` 的重定向复原脚本。
2. 在文本编辑器中打开重定向复原脚本以进行所需的修改。 可以修改:

- 复原选项
- 自动存储路径
- 容器布局和路径

3. 运行修改后的重定向复原脚本。 例如:

```
db2 -tvf test_node0000.clp
```

使用不同存储器组路径来克隆生产数据库

可能必须将生产数据库克隆到使用不同机器的测试数据库。测试机器和生产服务器可能具有不同的存储器组路径。测试系统可能没有受最新且最快的存储器磁盘支持的路径。

关于此任务

假定您具有生产数据库 proddb，其中某些数据位于存储器组 sg_hot 中，该存储器组具有 SSD 设备上的路径。您要将数据复原到较便宜且性能较低的测试数据库 testdb。测试系统没有该 SSD 设备，但其他路径是等价的。执行重定向复原时，将更改 sg_hot 在测试系统上的路径，但不会更改其他存储器组。

过程

要将数据从生产数据库复原到测试数据库，请执行下列操作:

1. 备份生产数据库。 发出下列命令:

   ```
   BACKUP DATABASE production_db TO /backup
   ```

   其中 *production_db* 是生产数据库。

2. 设置重定向复原以复原到测试数据库。 发出下列命令:

   ```
   RESTORE DATABASE testdb REDIRECT
   ```

   其中 *testdb* 是测试数据库。

3. 因为测试数据库上没有提供热存储器，所以请修改 sg_hot 的存储器路径。 发出下列命令:

   ```
   SET STOGROUP PATHS FOR sg_hot ON '/hdd/path1', '/hdd/path2'
   ```

   其中 *sg_hot* 是 sg_hot 存储器组。

4. 继续复原测试数据库。 发出下列命令:

   ```
   RESTORE DATABASE testdb CONTINUE
   ```

5. 更新存储器组名称以便与新路径匹配。 使用以下命令:

   ```
   CONNECT TO testdb
   RENAME STOGROUP sg_hot TO sg_cold
   ```

数据库重建

重建数据库是使用一组复原操作来复原数据库或它的一部分表空间的过程。数据库重建提供的功能使 DB2 数据库产品更健壮且用途更广泛，并为您提供一个更加完整的恢复解决方案。

允许从表空间备份映像重建数据库意味着您不必再进行那么多完整数据库备份。随着数据库大小的不断增长，执行完整数据库备份的机会越来越有限。由于还可以采用表

空间备份，您不必再那么频繁地进行完整数据库备份。相反，您可以更多地进行表空间备份，并计划在出现灾难时使用它们和日志文件。

在恢复情况下，如果需要使一部分表空间比其他表空间更快地处于联机状态，那么可以使用重建来完成此任务。仅使一部分表空间处于联机状态的功能在测试和生产环境中特别有用。

重建数据库涉及一系列潜在的复原操作。重建操作可以使用数据库映像和/或表空间映像。可以使用完整备份和/或增量备份。初始复原操作复原目标映像，该映像定义可以复原的数据库的结构（例如，表空间集、存储器组和数据库配置）。对于可恢复数据库，重建允许您构建可连接的数据库并且使该数据库包含需要处于联机状态的一部分表空间，而让稍后才恢复的表空间处于脱机状态。

用来重建数据库的方法取决于数据库是可恢复的还是不可恢复的。

- 如果数据库是可恢复的，使用下列其中一种方法:
  - 使用完整数据库备份映像、增量数据库备份映像或表空间备份映像作为目标映像，通过仅使用 **REBUILD** 选项从目标映像复原 SYSCATSPACE 及任何其他表空间来重建数据库。然后，可以将数据库前滚至某个时间点。
  - 使用完整数据库备份映像、增量数据库备份映像或表空间备份映像作为目标映像，通过指定使用 **REBUILD** 选项复原目标映像时在数据库中定义的表空间集来重建数据库。SYSCATSPACE 必须是此表空间集的一部分。此操作将复原指定的并且是在目标映像中定义的那些表空间，然后使用恢复历史记录文件来自动查找并复原不在目标映像中的其余表空间所需的任何其他备份映像。复原完成之后，立即将数据库前滚至某个时间点。
- 如果数据库是不可恢复的:
  - 使用完整或增量数据库备份映像作为目标映像，通过使用适当的 **REBUILD** 语法从目标映像复原 SYSCATSPACE 和任何其他表空间来重建数据库。复原完成后，可以连接至该数据库。

指定目标映像

要对数据库执行重建，一开始请发出 **RESTORE** 命令并指定用作复原操作目标的最新备份映像。此映像称为重建操作的目标映像，因为它定义要复原的数据库的结构，包括可以复原的表空间、数据库配置和日志序列。重建目标映像是使用 **RESTORE DATABASE** 命令中的 **TAKEN AT** 参数指定的。目标映像可以是任何类型的备份（完整备份、表空间备份、增量备份、联机备份或脱机备份）。创建目标映像时在数据库中定义的表空间将是可用于重建数据库的表空间。

必须使用下列其中一种方法指定想要复原的表空间:

- 指定想要复原数据库中定义的所有表空间并提供异常列表（如果想要排除一些表空间）
- 指定想要复原目标映像中具有用户数据的所有表空间并提供异常列表（如果想要排除一些表空间）
- 指定想要复原的在数据库中定义的表空间列表

一旦知道了想要重建数据库包含的表空间，就发出带有适当 **REBUILD** 选项的 **RESTORE** 命令并指定要使用的目标映像。

重建阶段

在发出带有适当 **REBUILD** 选项的 **RESTORE** 命令并成功地复原了目标映像之后，就认为数据库处于重建阶段。在复原目标映像之后，进行的任何其他表空间复原将会使数据复原到现有表空间中，如已重建的数据库中定义的那样。然后，这些表空间将在重建操作完成时与数据库一起前滚。

如果发出带有适当 **REBUILD** 选项的 **RESTORE** 命令，但数据库不存在，那么将根据目标映像中的属性创建一个新数据库。如果数据库存在，您将接收到一条警告消息，通知您重建阶段即将开始。将会询问您是否想要继续重建操作。

重建操作从目标映像复原所有初始元数据。这包括属于数据库但不属于表空间数据或日志文件的所有数据。初始元数据的示例有:

- 表空间定义
- 历史记录文件，它是记录管理操作的数据库文件

重建操作还复原数据库配置。目标映像设置日志链，它确定在重建阶段可以用于其余复原的映像。只能使用同一日志链上的映像。

如果磁盘上已经存在数据库并且想要从目标映像生成历史记录文件，那么应该指定 **REPLACE HISTORY FILE** 选项。自动逻辑使用此时磁盘上的历史记录文件来查找重建数据库所需的其余映像。

复原了目标映像之后:

- 如果数据库是可恢复的，那么数据库将处于前滚暂挂状态，并且复原的所有表空间也处于前滚暂挂状态。数据库中定义的但未复原的所有表空间都处于复原暂挂状态。
- 如果数据库是不可恢复的，那么复原的数据库和数据库表空间将进入正常状态。未复原的所有表空间将处于删除暂挂状态，因为不能再复原它们。对于这种类型的数据库来说，重建阶段完成了。

对于可恢复的数据库，当发出第一个 **ROLLFORWARD DATABASE** 命令并且 ROLLFORWARD 实用程序开始处理日志记录时，重建阶段结束。如果前滚操作在开始处理日志记录后失败，并且接着发出了复原操作，那么不认为该复原是重建阶段的一部分。应该将这种复原看作是正常表空间复原，它们不是重建阶段的一部分。

自动处理

在复原目标映像之后，RESTORE 实用程序确定是否有其余表空间需要复原。如果有，那么使用用于运行带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令的那个连接进行复原。该实用程序使用磁盘上的历史记录文件来查找在目标映像之前的最新备份映像，该备份映像包含需要复原的其余每个表空间。RESTORE 实用程序使用历史记录文件中存储的备份映像位置数据来自动复原每个映像。后续这些复原表空间级复原，只能以脱机方式执行。如果选择的映像不属于当前日志链，那么会返回错误。从该映像复原的每个表空间将处于前滚暂挂状态。

RESTORE 实用程序尝试自动复原所有必需表空间。在某些情况下，可能由于历史记录文件具有的问题而无法复原一些表空间，或者在复原其中一个必需映像时发生错误。在此情况下，可以手动完成重建，或者更正问题然后重新发出重建。

如果自动重建无法成功完成，那么 RESTORE 实用程序会将它收集的用于其余复原步骤的所有信息都写入诊断日志（**db2diag** 日志文件）中。可以使用这些信息手动完成重建。

如果正在重建某个数据库，那么仅获取属于重建过程的表空间的容器。

如果需要通过重定向复原来重新定义任何容器，那么将需要为其余复原和后续前滚操作设置新容器的新路径和大小。

如果从其余这些映像中的一个映像复原的表空间的数据不符合新容器的定义，那么该表空间将处于复原暂挂状态，并且在复原结束时会返回一则警告消息。可以在诊断日志中找到更多有关该问题的信息。

完成重建阶段

在已经复原了所有要复原的表空间后，您可以根据数据库的配置作出两种选择。如果数据库是不可恢复的，那么数据库将是可连接的，并且复原的所有表空间将处于联机状态。不能再复原处于删除暂挂状态的任何表空间，如果将来要在数据库上执行备份，那么应该删除这些表空间。

如果数据库是可恢复的，那么可以发出前滚命令来使已复原的表空间处于联机状态。如果 SYSCATSPACE 尚未复原，那么前滚将失败，并且必须在前滚操作开始之前复原此表空间。这表示必须在重建阶段复原 SYSCATSPACE。

注: 在分区数据库环境中，SYSCATSPACE 不存在于非目录分区中，因此不能在那里将它重建。但是，在目录分区上，SYSCATSPACE 必须是重建的表空间，否则前滚操作将不会成功。

前滚数据库使数据库脱离前滚暂挂状态，并使处于前滚暂挂状态的任何表空间向前滚动。不会对任何处于复原暂挂状态的表空间运行 ROLLFORWARD 实用程序。

前滚操作的停止时间必须比在重建阶段复原最新备份映像的结束时间要晚。如果给出任何其他时间，那么会发生错误。如果前滚操作无法到达复原的最早映像的备份时间，那么 ROLLFORWARD 实用程序将无法使数据库到达一个一致的时间，前滚将失败。

在最早备份映像和最新备份映像的时间范围内，必须使所有日志文件可供 ROLLFORWARD 实用程序使用。所需的日志是从最早备份映像到目标备份映像都遵循日志链的那些日志，如目标映像中的截断数组所定义的那样，否则前滚操作将失败。如果在重建阶段复原了比目标映像更新的任何备份映像，那么需要从目标映像到复原的最新备份映像的附加日志。如果日志不可用，那么前滚操作将使日志无法访问的那些表空间处于复原暂挂状态。可以发出 **LIST HISTORY** 命令来显示复原重建条目以及前滚所需的日志范围。

正确的日志文件必须可用。如果依靠 ROLLFORWARD 实用程序来检索日志，那么必须确保配置了“DB2 日志管理器”以指示检索日志文件的位置。如果日志路径或归档路径更改了，那么需要使用 **ROLLFORWARD DATABASE** 命令的 **OVERFLOW LOG PATH** 选项。

当成功完成前滚命令时，使用 **ROLLFORWARD DATABASE** 命令的 **AND STOP** 选项来使数据库可用。此时，数据库不再处于前滚暂挂状态。如果前滚操作开始了，但它未成功完

成就出现错误，那么前滚操作将在故障点处停止并返回错误。数据库仍然处于前滚暂挂状态。必须执行步骤以更正该问题（例如，修正日志文件），然后发出另一个前滚操作来继续处理。

如果无法修正该错误，那么可以通过发出 **ROLLFORWARD STOP** 命令使数据库前进到故障点。一旦使用了 **STOP** 选项，日志中超出该故障点的所有日志数据将不再可用。数据库处于该故障点处，并且已恢复的所有表空间都处于联机状态。尚未恢复的表空间处于复原暂挂状态。数据库处于正常状态。

您必须决定恢复处于复原暂挂状态的其余表空间的最佳方法。该方法可以是进行新的复原并前滚表空间，也可以是重新发出整个重建操作。这将取决于遇到的问题的类型。如果 SYSCATSPACE 是其中一个处于复原暂挂状态的表空间，那么数据库将不可连接。

数据库重建和表空间容器

重建数据库期间，只有参与重建过程的那些表空间才将获取它们的容器。将在从映像复原表空间用户数据时获取属于每个表空间的容器。

复原目标映像时，数据库在备份时知道的每个表空间只复原它们的定义。这表示通过重建创建的数据库将认识在备份时知道的那些表空间。对于同样应该从目标映像复原其用户数据的那些表空间，也会在此时获取它们的容器。

通过中间表空间复原来复原的任何其余表空间将在复原包含表空间数据的映像时获取它们的容器。

使用重定向复原重建

在重定向复原的情况下，必须在复原目标映像期间定义所有表空间容器。如果指定 **REDIRECT** 选项，那么您将重新获得重新定义表空间容器的控制权。如果使用 **SET TABLESPACE CONTAINERS** 命令重新定义了表空间容器，那么将在那时获取那些新容器。将在正常时间获取尚未重新定义的任何表空间容器，即，在从映像复原表空间用户数据时。

如果复原的表空间的数据不符合新容器的定义，那么该表空间将处于复原暂挂状态，并且在复原结束时会返回一个警告 (SQL2563W)。在 DB2 诊断日志中将有一条消息详细说明该问题。

数据库重建和临时表空间

临时表空间与其他数据库组件采用不同的方式存储在备份映像中。因为临时表空间的存储方式不同，所以在复原数据库期间将以不同的方式重建临时表空间。

DB2 备份映像通常由下列组件构成:

- 初始数据库元数据，例如，表空间定义、数据库配置文件和历史记录文件。
- 对 **BACKUP** 实用程序指定的非临时表空间的数据
- 最终数据库元数据，如日志文件头
- 日志文件（如果指定了 **INCLUDE LOGS** 选项）

在每个备份映像中，无论是数据库备份还是表空间备份，或者完整备份还是增量备份（差异备份），始终可以找到这些核心组件。

数据库备份映像将包含先前列示的所有组件，以及备份时在数据库中定义的每个表空间的数据。

表空间备份映像将始终包含先前列示的数据库元数据，但它仅包含对备份实用程序指定的那些表空间的数据。

对待临时表空间的方式与对待非临时表空间的方式不同。从不备份临时表空间数据，但这些数据的存在对于数据库的框架来说很重要。虽然从不备份临时表空间数据，但将临时表空间视为数据库的一部分，因此，在使用备份映像存储的元数据中对其进行了特别标记。这使得它们看起来与在备份映像中一样。此外，表空间定义还保存关于任何临时表空间是否存在的信息。

虽然备份映像不曾包含临时表空间的数据，但在数据库重建操作过程中，在复原目标映像时（无论映像是什么类型），同样会复原临时表空间，这样是只为了获取和分配它们的容器。获取和分配容器是作为重建过程的一部分自动完成的。因此，重建数据库时，不能排除临时表空间。

为数据库重建选择目标映像

重建目标映像应该是想要用作复原操作的起始点的最新备份映像。

此映像称为重建操作的目标映像，因为它定义要复原的数据库的结构，包括可以复原的表空间、数据库配置和日志序列。它可以是任何类型的备份（完整备份、表空间备份、增量备份、联机备份或脱机备份）。

目标映像设置日志序列（或日志链），它确定在重建阶段可以用于其余复原的映像。只能使用同一日志链上的映像。

下列示例说明如何选择应用作重建操作的目标映像的映像。

假设有一个称为 SAMPLE 的数据库，该数据库中具有下列表空间:

- SYSCATSPACE（系统目录）
- USERSP1（用户数据表空间）
- USERSP2（用户数据表空间）
- USERSP3（用户数据表空间）

第 336 页的图 24 按时间顺序显示已经进行的数据库级别备份和表空间级别备份:

1. 完整数据库备份 DB1
2. 完整表空间备份 TS1
3. 完整表空间备份 TS2
4. 完整表空间备份 TS3
5. 数据库复原并前滚至 TS1 与 TS2 之间的一个时间点
6. 完整表空间备份 TS4
7. 完整表空间备份 TS5

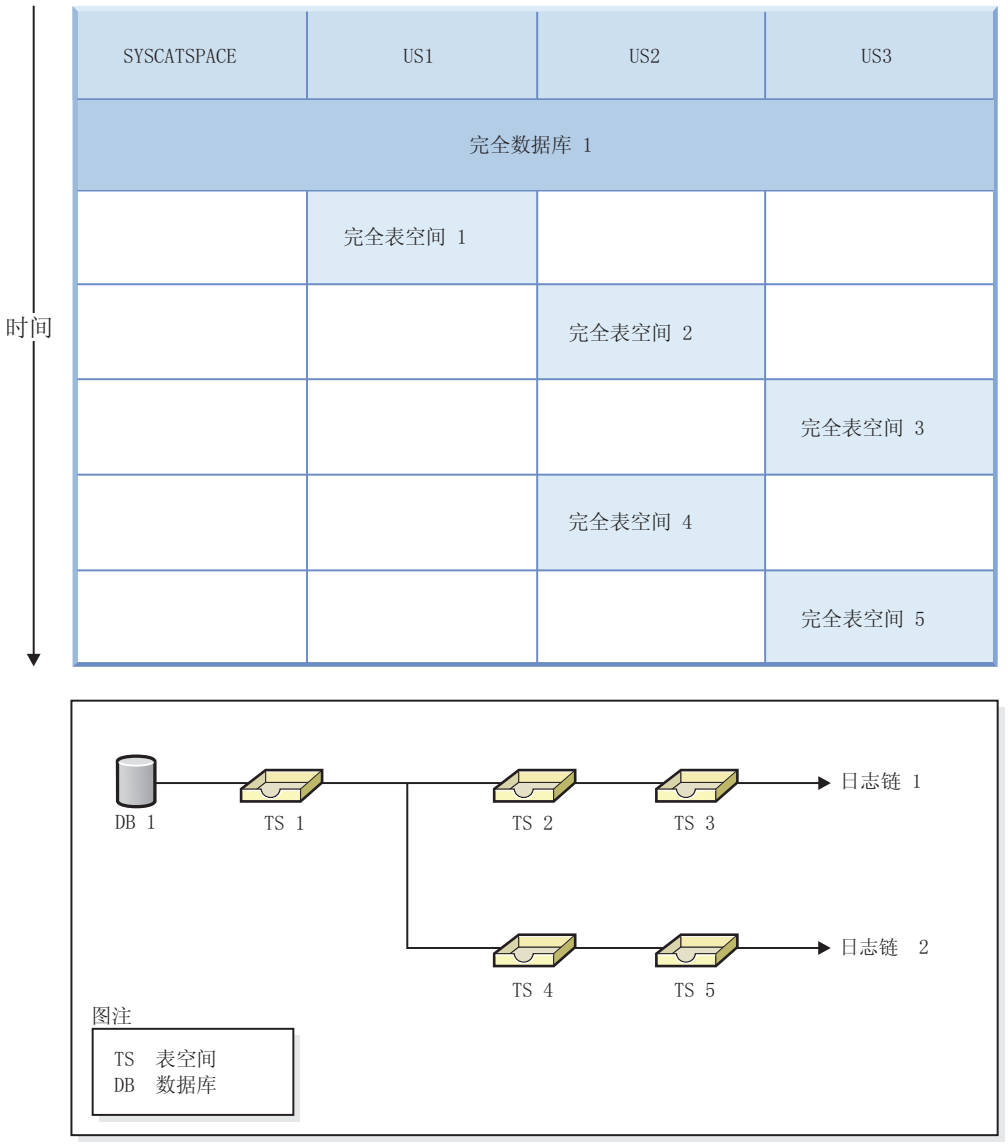


图 24. 数据库 SAMPLE 的数据库级别备份和表空间级别备份

示例 1

以下示例演示了将数据库 SAMPLE 重建至当前时间点需要发出的 CLP 命令。首先需要选择想要重建的表空间。由于目标是将数据库重建至当前时间点，因此需要选择最新备份映像作为目标映像。最新备份映像是映像 TS5，它在日志链 2 上:

```
db2 restore db sample rebuild with all tablespaces in database taken at
      TS5 without prompting
db2 rollforward db sample to end of logs
db2 rollforward db sample stop
```

这将自动复原备份映像 TS5、TS4、TS1 和 DB1，然后将数据库前滚至日志链 2 的末尾。

注: 所有属于日志链 2 的日志必须可以访问，这样前滚操作才能完成。

示例 2

第二个示例演示了将数据库 SAMPLE 重建至日志链 1 的末尾需要发出的 CLP 命令。选择的目标映像应该是日志链 1 上的最新备份映像，它是 TS3:

```
db2 restore db sample rebuild with all tablespaces in database
      taken at TS3 without prompting
db2 rollforward db sample to end of logs
db2 rollforward db sample stop
```

这将自动复原备份映像 TS3、TS2、TS1 和 DB1，然后将数据库前滚至日志链 1 的末尾。

注: 所有属于日志链 1 的日志必须可以访问，这样前滚操作才能完成。

选择错误的目标映像进行重建

假设有一个称为 SAMPLE2 的数据库，该数据库中具有下列表空间:

- SYSCATSPACE（系统目录）
- USERSP1（用户数据表空间）
- USERSP2（用户数据表空间）

图 25 显示了 SAMPLE2 的备份日志链，它由以下备份组成:

1. BK1 是完整数据库备份，它包括所有表空间
2. BK2 是 USERSP1 的完整表空间备份
3. BK3 是 USERSP2 的完整表空间备份

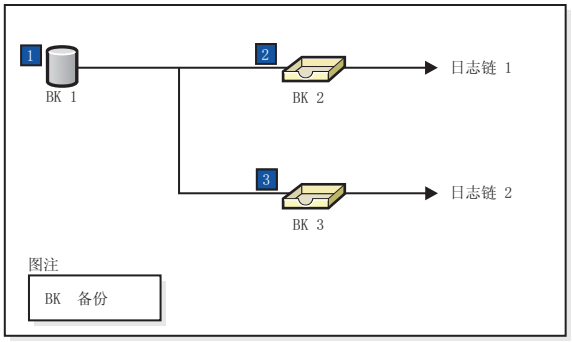


图 25. 数据库 SAMPLE2 的备份日志链

以下示例演示了使用表空间 SYSCATSPACE 和 USERSP2 从 BK3 重建数据库需要发出的 CLP 命令:

```
db2 restore db sample2 rebuild with tablespace (SYSCATSPACE,
      USERSP2) taken at BK3 without prompting
```

现在假设在此复原操作完成之后，您决定还要复原 USERSP1，因此发出以下命令:

```
db2 restore db sample2 tablespace (USERSP1) taken at BK2
```

此复原失败并提供一条消息，说明 BK2 来自错误的日志链 (SQL2154N)。如图 25 中所示，唯一可用来复原 USERSP1 的映像是 BK1。因此，需要发出以下命令:

```
db2 restore db sample2 tablespace (USERSP1) taken at BK1
```

复原成功，因此数据库可以相应地前滚。

重建所选表空间

重建数据库允许您建立包含原始数据库中的一部分表空间的数据库。

关于此任务

在下列情况下，仅重建数据库中的一部分表空间非常有用:

- 在只想对一部分表空间进行工作的测试和开发环境中。
- 在恢复情况下，如果需要使较关键的表空间比其他表空间更快地处于联机状态，那么可以先复原一部分表空间，稍后再复原其他表空间。

要重建包含原始数据库中的一部分表空间的数据库，请考虑以下示例。

在此示例中，有一个称为 SAMPLE 的数据库，它具有下列表空间:

- SYSCATSPACE（系统目录）
- USERSP1（用户数据表空间）
- USERSP2（用户数据表空间）
- USERSP3（用户数据表空间）

图 26表明已经进行了下列备份:

- BK1 是 SYSCATSPACE 和 USERSP1 的备份
- BK2 是 USERSP2 和 USERSP3 的备份
- BK3 是 USERSP3 的备份

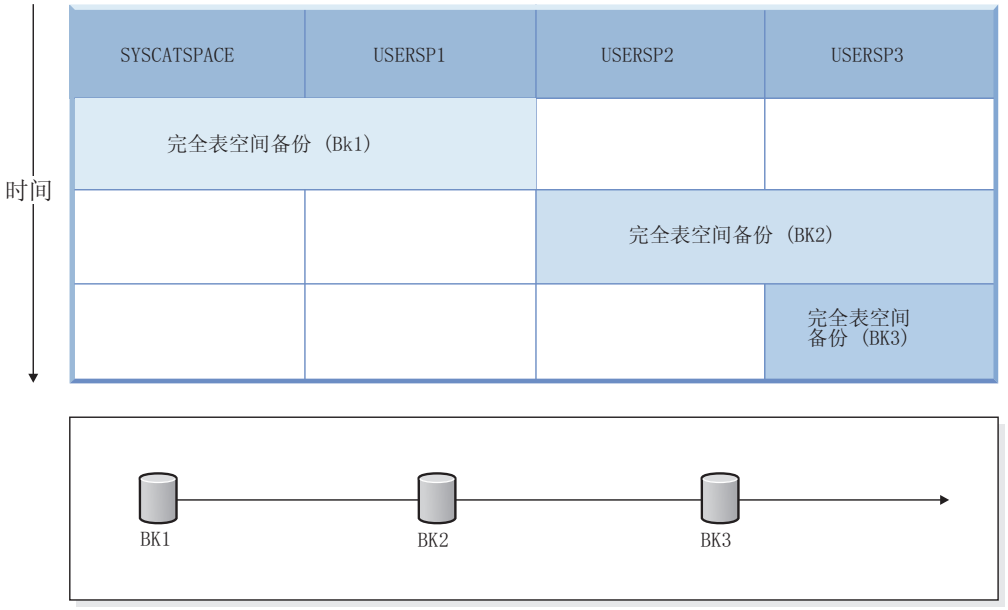


图 26. 为 SAMPLE 数据库提供的备份映像

下列过程演示了使用 CLP 发出的 **RESTORE DATABASE** 和 **ROLLFORWARD DATABASE** 命令仅将 SYSCATSPACE 和 USERSP1 重建至日志末尾:

```
db2 restore db mydb rebuild with all tablespaces in image
          taken at BK1 without prompting
db2 rollforward db mydb to end of logs
db2 rollforward db mydb stop
```

此时，该数据库是可连接的，但只有 SYSCATSPACE 和 USERSP1 处于 NORMAL 状态。USERSP2 和 USERSP3 处于复原暂挂状态。以后仍可以复原 USERSP2 和 USERSP3。

重建和增量备份映像

可以使用增量映像重建数据库。

缺省情况下，RESTORE 实用程序将尝试对所有增量映像使用自动增量复原。这表示如果不使用 **RESTORE DATABASE** 命令的 **INCREMENTAL** 选项，但目标映像又是增量备份映像，那么 RESTORE 实用程序将使用自动增量复原发出重建操作。如果目标映像不是增量映像，但另一个需要的映像是增量映像，那么 RESTORE 实用程序将确保使用自动增量复原来复原这些增量映像。无论是否指定带有 **AUTOMATIC** 选项的 **INCREMENTAL** 选项，RESTORE 实用程序的工作方式都相同。

如果指定了 **INCREMENTAL** 选项，但未指定 **AUTOMATIC** 选项，那么需要手动执行整个重建过程。RESTORE 实用程序将只从目标映像复原初始元数据，就像是正常手动增量复原一样。然后，需要使用必需的增量复原链来完成目标映像的复原。接下来，需要复原其余映像以重建数据库。

建议您使用自动增量复原来重建数据库。仅在复原失败的情况下才应尝试使用手动方法重建数据库。

重建分区数据库

要重建分区数据库，分别重建每个数据库分区。对于每个数据库分区，从目录分区开始，首先复原需要的所有表空间。未复原的所有表空间都处于复原暂挂状态。

复原了所有数据库分区之后，在目录分区上发出 **ROLLFORWARD DATABASE** 命令以前滚所有数据库分区。

关于此任务

注: 如果在将来的某一天您需要复原最初未包括在重建阶段中的任何表空间，那么需要确保在后来前滚表空间时 ROLLFORWARD 实用程序使数据库分区上的所有数据保持同步。如果在原始复原和前滚操作中丢失了某个表空间，那么直到尝试访问数据时才会检测到这种情况，将出现数据访问错误。需要复原和前滚丢失的表空间，以使它恢复与其余分区的同步。

要使用表空间级备份映像来重建分区数据库，请考虑以下示例。

在此示例中，有一个称为 SAMPLE 的可恢复数据库，它具有三个数据库分区:

- 数据库分区 1 包含表空间 SYSCATSPACE、USERSP1 和 USERSP2，它是目录分区
- 数据库分区 2 包含表空间 USERSP1 和 USERSP3
- 数据库分区 3 包含表空间 USERSP1、USERSP2 和 USERSP3

进行了下列备份，其中 BK*xy* 表示分区 *y* 上的备份编号 *x*:

- BK11 是 SYSCATSPACE、USERSP1 和 USERSP2 的备份
- BK12 是 USERSP2 和 USERSP3 的备份
- BK13 是 USERSP1、USERSP2 和 USERSP3 的备份
- BK21 是 USERSP1 的备份
- BK22 是 USERSP1 的备份
- BK23 是 USERSP1 的备份
- BK31 是 USERSP2 的备份
- BK33 是 USERSP2 的备份
- BK42 是 USERSP3 的备份
- BK43 是 USERSP3 的备份

下列过程演示了使用 CLP 发出的 **RESTORE DATABASE** 和 **ROLLFORWARD DATABASE** 命令将整个数据库重建至日志末尾。

过程

1. 在数据库分区 1 上，发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令:

   ```
   db2 restore db sample rebuild with all tablespaces in database
         taken at BK31 without prompting
   ```

2. 在数据库分区 2 上，发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令:

   ```
   db2 restore db sample rebuild with tablespaces in database
         taken at BK42 without prompting
   ```

3. 在数据库分区 3 上，发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令:

   ```
   db2 restore db sample rebuild with all tablespaces in database
         taken at BK43 without prompting
   ```

4. 在目录分区上，发出带有 **TO END OF LOGS** 选项的 **ROLLFORWARD DATABASE** 命令:

   ```
   db2 rollforward db sample to end of logs
   ```

5. 发出带有 **STOP** 选项的 **ROLLFORWARD DATABASE** 命令:

   ```
   db2 rollforward db sample stop
   ```

下一步做什么

此时，该数据库在所有数据库分区上都是可连接的，并且所有表空间都处于 NORMAL 状态。

数据库重建的限制

您可以使用 **REBUILD** 命令来完成一组 restore 命令。此命令很有帮助，但是它存在一些限制，您必须知道这些限制。

以下列表对数据库重建的限制进行了概述:

- 重建的其中一个表空间必须是目录分区上的 SYSCATSPACE。
- 必须使用命令行处理器 (CLP) 发出命令或者使用相应的应用程序编程接口 (API) 来执行重建操作。
- 除非目标映像是脱机数据库备份，否则不能对 V9.1 之前的目标映像使用 **REBUILD** 选项。如果目标映像是脱机数据库备份，那么只能将此映像中的表空间用于重建。重建操作成功完成之后，需要迁移数据库。尝试使用 V9.1 之前的任何其他类型的目标映像进行重建将导致错误。

- 除非目标映像是完整数据库备份，否则当目标映像所在的操作系统不同于正在进行复原的操作系统时，不能对该目标映像发出 **REBUILD** 选项。如果目标映像是完整数据库备份，那么只能将此映像中的表空间用于重建。如果目标映像所在的操作系统不同于正在进行复原的操作系统，那么使用任何其他类型的这种映像进行重建都将导致错误。
- TRANSPORT 选项与 REBUILD 选项不兼容。

重建会话 - CLP 示例

本主题提供许多重建操作示例。

情况 1

在下列示例中，有一个称为 MYDB 的可恢复数据库，该数据库中存在下列表空间:

- SYSCATSPACE（系统目录）
- USERSP1（用户数据表空间）
- USERSP2（用户数据表空间）
- USERSP3（用户数据表空间）

进行了下列备份:

- BK1 是 SYSCATSPACE 和 USERSP1 的备份
- BK2 是 USERSP2 和 USERSP3 的备份
- BK3 是 USERSP3 的备份

示例 1

下列命令将整个数据库重建至最新时间点:

1. 发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令:

```
db2 restore db mydb rebuild with all tablespaces in database
      taken at BK3 without prompting
```

2. 发出带有 **TO END OF LOGS** 选项的 **ROLLFORWARD DATABASE** 命令（假定所有日志都已保存并且可以访问）:

```
db2 rollforward db mydb to end of logs
```

3. 发出带有 **STOP** 选项的 **ROLLFORWARD DATABASE** 命令:

```
db2 rollforward db mydb stop
```

此时，该数据库是可连接的并且所有表空间都处于 NORMAL 状态。

示例 2

下列命令仅将 SYSCATSPACE 和 USERSP2 重建至某个时间点（BK3 的末尾比该时间点要早，而该时间点又比日志末尾要迟）:

1. 发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令并指定想要包括的表空间。

```
db2 restore db mydb rebuild with tablespace (SYSCATSPACE, USERSP2)
      taken at BK2 without prompting
```

2. 发出带有 **TO PIT** 选项的 **ROLLFORWARD DATABASE** 命令（假定所有日志都已保存并且可以访问）:

```
db2 rollforward db mydb to PIT
```

3. 发出带有 **STOP** 选项的 **ROLLFORWARD DATABASE** 命令:

```
db2 rollforward db mydb stop
```

此时，该数据库是可连接的，但只有 SYSCATSPACE 和 USERSP2 处于 NORMAL 状态。USERSP1 和 USERSP3 处于 RESTORE_PENDING 状态。

要在以后使用正常表空间复原（不带 **REBUILD** 选项）复原 USERSP1 和 USERSP3:

1. 发出*不带* **REBUILD** 选项的 **RESTORE DATABASE** 命令并指定想要复原的表空间。首先复原 USERSP1:

   ```
   db2 restore db mydb tablespace (USERSP1) taken at BK1 without prompting
   ```

2. 然后复原 USERSP3:

   ```
   db2 restore db mydb tablespace taken at BK3 without prompting
   ```

3. 发出带有 **END OF LOGS** 选项的 **ROLLFORWARD DATABASE** 命令并指定要复原的表空间（假定所有日志都已保存并且可以访问）:

   ```
   db2 rollforward db mydb to end of logs tablespace (USERSP1, USERSP3)
   ```

   前滚将所有日志最多重放至 PIT，然后对这两个表空间停止，因为自第一次前滚以来尚未对它们执行任何操作。

4. 发出带有 **STOP** 选项的 **ROLLFORWARD DATABASE** 命令:

   ```
   db2 rollforward db mydb stop
   ```

示例 3

下列命令只将 SYSCATSPACE 和 USERSP1 重建至日志末尾:

1. 发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令:

   ```
   db2 restore db mydb rebuild with all tablespaces in image
       taken at BK1 without prompting
   ```

2. 发出带有 **TO END OF LOGS** 选项的 **ROLLFORWARD DATABASE** 命令（假定所有日志都已保存并且可以访问）:

   ```
   db2 rollforward db mydb to end of logs
   ```

3. 发出带有 **STOP** 选项的 **ROLLFORWARD DATABASE** 命令:

   ```
   db2 rollforward db mydb stop
   ```

此时，该数据库是可连接的，但只有 SYSCATSPACE 和 USERSP1 处于 NORMAL 状态。USERSP2 和 USERSP3 处于 RESTORE_PENDING 状态。

示例 4

在以下示例中，备份 BK1 和 BK2 不再位于历史记录文件中所述的那个位置，但发出重建时并不知道这一点。

1. 发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令，并指定想要将整个数据库重建至最新时间点:

   ```
   db2 restore db mydb rebuild with all tablespaces in database
       taken at BK3 without prompting
   ```

   此时成功地复原了目标映像，但从 RESTORE 实用程序返回一个错误，说明无法找到所需的映像。

2. 现在必须手动完成重建。因为数据库处于重建阶段，所以可以按如下所示完成重建:

   a. 发出 **RESTORE DATABASE** 命令并指定 BK1 备份映像的位置:

```
db2 restore db mydb tablespace taken at BK1 from location
      without prompting
```

b. 发出 **RESTORE DATABASE** 命令并指定 BK2 备份映像的位置:

```
db2 restore db mydb tablespace (USERSP2) taken at BK2 from
location without prompting
```

c. 发出带有 **TO END OF LOGS** 选项的 **ROLLFORWARD DATABASE** 命令（假定所有日志都已保存并且可以访问）:

```
db2 rollforward db mydb to end of logs
```

d. 发出带有 **STOP** 选项的 **ROLLFORWARD DATABASE** 命令:

```
db2 rollforward db mydb stop
```

此时，该数据库是可连接的并且所有表空间都处于 NORMAL 状态。

示例 5

在本例中，USERSP3 表空间包含生成特定报告需要的独立数据，但是您并不希望报告的生成妨碍原始数据库。为了访问数据而不影响原始数据库，可以使用 **REBUILD** 生成只有这个表空间和 SYSCATSPACE 的新数据库。为了使数据库在进行复原和前滚操作后仍然可以连接，所以同时也需要 SYSCATSPACE。

使用 SYSCATSPACE 和 USERSP3 中最新的数据构建新的数据库:

1. 发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令，并指定将 SYSCATSPACE 和 USERSP3 表空间复原到一个新的数据库 NEWDB 中:

```
db2 restore db mydb rebuild with tablespace (SYSCATSPACE, USERSP3)
    taken at BK3 into newdb without prompting
```

2. 对 NEWDB 发出带有 **TO END OF LOGS** 选项的 **ROLLFORWARD DATABASE** 命令（假定所有日志都已保存并且可以访问）:

```
db2 rollforward db newdb to end of logs
```

3. 发出带有 **STOP** 选项的 **ROLLFORWARD DATABASE** 命令:

```
db2 rollforward db newdb stop
```

此时，新数据库是可连接的，并且只有 SYSCATSPACE 和 USERSP3 处于 NORMAL 状态。USERSP1 和 USERSP2 处于 RESTORE_PENDING 状态。

注: 如果现有数据库和新数据库之间的容器路径有问题（例如，如果因为文件系统不存在而需更改原始数据库的容器，或容器正被原始数据库使用），那么需要执行重定向复原。此示例假定对表空间使用了缺省自动存储数据库路径。

情况 2

在以下示例中，有一个称为 MYDB 的可恢复数据库，该数据库具有 SYSCATSPACE 和 1000 个名为 T*xxxx* 的用户表空间，其中 *xxxx* 表示表空间号（例如 T0001）。有一个完整数据库备份映像（BK1）

示例 6

下列命令复原除 T0999 和 T1000 之外的所有表空间:

1. 发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令:

```
db2 restore db mydb rebuild with all tablespaces in image except
    tablespace (T0999, T1000) taken at BK1 without prompting
```

2. 发出带有 **TO END OF LOGS** 选项的 **ROLLFORWARD DATABASE** 命令（假定所有日志都已保存并且可以访问）:

```
db2 rollforward db mydb to end of logs
```

3. 发出带有 **STOP** 选项的 **ROLLFORWARD DATABASE** 命令:

```
db2 rollforward db mydb stop
```

此时，该数据库是可连接的，并且除 T0999 和 T1000 之外的所有表空间都处于 NORMAL 状态。T0999 和 T1000 处于 RESTORE_PENDING 状态。

情况 3

这种情况中的示例演示了如何使用增量备份来重建可恢复数据库。在下列示例中，有一个称为 MYDB 的数据库，该数据库中存在下列表空间:

- SYSCATSPACE（系统目录）
- USERSP1（数据表空间）
- USERSP2（用户数据表空间）
- USERSP3（用户数据表空间）

进行了下列备份:

- FULL1 是 SYSCATSPACE、USERSP1、USERSP2 和 USERSP3 的完整备份
- DELTA1 是 SYSCATSPACE 和 USERSP1 的差异备份
- INCR1 是 USERSP2 和 USERSP3 的增量备份
- DELTA2 是 SYSCATSPACE、USERSP1、USERSP2 和 USERSP3 的差异备份
- DELTA3 是 USERSP2 的差异备份
- FULL2 是 USERSP1 的完整备份

示例 7

下列命令使用增量自动复原仅将 SYSCATSPACE 和 USERSP2 重建至最新时间点。

1. 发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令。**INCREMENTAL AUTO** 选项是可选的。RESTORE 实用程序将检测映像的详细程度，并在需要时使用自动增量复原。

```
db2 restore db mydb rebuild with tablespace (SYSCATSPACE, USERSP2)
     incremental auto taken at DELTA3 without prompting
```

2. 发出带有 **TO END OF LOGS** 选项的 **ROLLFORWARD DATABASE** 命令（假定所有日志都已保存并且可以访问）:

```
db2 rollforward db mydb to end of logs
```

3. 发出带有 **STOP** 选项的 **ROLLFORWARD DATABASE** 命令:

```
db2 rollforward db mydb stop
```

此时，该数据库是可连接的，但只有 SYSCATSPACE 和 USERSP2 处于 NORMAL 状态。USERSP1 和 USERSP3 处于 RESTORE_PENDING 状态。

示例 8

下列命令使用增量自动复原将整个数据库重建至最新时间点。

1. 发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令。**INCREMENTAL AUTO** 选项是可选的。RESTORE 实用程序将检测映像的详细程度，并在需要时使用自动增量复原。

```
db2 restore db mydb rebuild with all tablespaces in database
     incremental auto taken at DELTA3 without prompting
```

2. 发出带有 **TO END OF LOGS** 选项的 **ROLLFORWARD DATABASE** 命令（假定所有日志都已保存并且可以访问）：

```
db2 rollforward db mydb to end of logs
```

3. 发出带有 **STOP** 选项的 **ROLLFORWARD DATABASE** 命令:

```
db2 rollforward db mydb stop
```

此时，该数据库是可连接的并且所有表空间都处于 NORMAL 状态。

示例 9

下列命令将整个数据库（USERSP3 除外）重建至最新时间点。

1. 发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令。虽然目标映像是非增量映像，但 RESTORE 实用程序还是检测必需的重建链是否包括增量映像，并且将以增量方式自动复原这些映像。

```
db2 restore db mydb rebuild with all tablespaces in database except
     tablespace (USERSP3) taken at FULL2 without prompting
```

2. 发出带有 **TO END OF LOGS** 选项的 **ROLLFORWARD DATABASE** 命令（假定所有日志都已保存并且可以访问）：

```
db2 rollforward db mydb to end of logs
```

3. 发出带有 **STOP** 选项的 **ROLLFORWARD DATABASE** 命令:

```
db2 rollforward db mydb stop
```

情况 4

这种情况中的示例演示了如何使用包含日志文件的备份映像来重建可恢复数据库。在下列示例中，有一个称为 MYDB 的数据库，该数据库中存在下列表空间:

- SYSCATSPACE（系统目录）
- USERSP1（用户数据表空间）
- USERSP2（用户数据表空间）

示例 10

下列命令仅将数据库中的 SYSCATSPACE 和 USERSP2 重建至最新时间点。有一个完整联机数据库备份映像（BK1），它包含日志文件。

1. 发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令:

```
db2 restore db mydb rebuild with tablespace (SYSCATSPACE, USERSP2)
     taken at BK1 logtarget /logs without prompting
```

2. 发出带有 **TO END OF LOGS** 选项的 **ROLLFORWARD DATABASE** 命令（假定所有日志在 BK1 结束之后都已保存并且可以访问）：

```
db2 rollforward db mydb to end of logs overflow log path (/logs)
```

3. 发出带有 **STOP** 选项的 **ROLLFORWARD DATABASE** 命令:

```
db2 rollforward db mydb stop
```

此时，该数据库是可连接的，但只有 SYSCATSPACE 和 USERSP2 处于 NORMAL 状态。USERSP1 处于 RESTORE_PENDING 状态。

示例 11

下列命令将数据库重建至最新时间点。有两个包含日志文件的完整联机表空间备份映像:

- BK1 是 SYSCATSPACE 的备份，使用日志文件 10-45

- BK2 是 USERSP1 和 USERSP2 的备份，使用日志文件 64-80

1. 发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令:

```
db2 restore db mydb rebuild with all tablespaces in database
    taken at BK2 logtarget /logs without prompting
```

   前滚操作将从日志文件 10 开始，如果该文件不在主日志文件路径中，那么它总是位于溢出日志路径中。由于日志范围 46-63 未包含在任何备份映像中，所以需要使它们可用于前滚操作。

2. 发出带有 **TO END OF LOGS** 选项的 **ROLLFORWARD DATABASE** 命令，并使用日志文件 64-80 的溢出日志路径:

```
db2 rollforward db mydb to end of logs overflow log path (/logs)
```

3. 发出带有 **STOP** 选项的 **ROLLFORWARD DATABASE** 命令:

```
db2 rollforward db mydb stop
```

此时，该数据库是可连接的并且所有表空间都处于 NORMAL 状态。

情况 5

在下列示例中，有一个称为 MYDB 的可恢复数据库，该数据库中存在下列表空间:

- SYSCATSPACE (0)，SMS 系统目录（相对容器）
- USERSP1 (1) DMS 用户数据表空间（绝对容器 /usersp2）
- USERSP2 (2) DMS 用户数据表空间（绝对容器 /usersp3）

进行了下列备份:

- BK1 是 SYSCATSPACE 的备份
- BK2 是 USERSP1 和 USERSP2 的备份
- BK3 是 USERSP2 的备份

示例 12

下列命令使用重定向复原将整个数据库重建至最新时间。

1. 发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令:

```
db2 restore db mydb rebuild with all tablespaces in database
    taken at BK3 redirect without prompting
```

2. 对想要重新定义其容器的每个表空间发出 **SET TABLESPACE CONTAINERS** 命令。例如:

```
db2 set tablespace containers for 3 using (file '/newusersp1' 10000)
```

3.

```
db2 set tablespace containers for 4 using (file '/newusersp2' 15000)
```

4. 发出带有 **CONTINUE** 选项的 **RESTORE DATABASE** 命令:

```
db2 restore db mydb continue
```

5. 发出带有 **TO END OF LOGS** 选项的 **ROLLFORWARD DATABASE** 命令（假定所有日志都已保存并且可以访问）:

```
db2 rollforward db mydb to end of logs
```

6. 发出带有 **STOP** 选项的 **ROLLFORWARD DATABASE** 命令:

```
db2 rollforward db mydb stop
```

此时，该数据库是可连接的并且所有表空间都处于 NORMAL 状态。

情况 6

在下列示例中，有一个称为 MYDB 的数据库，它具有三个数据库分区:

- 数据库分区 1 包含表空间 SYSCATSPACE、USERSP1 和 USERSP2，它是目录分区
- 数据库分区 2 包含表空间 USERSP1 和 USERSP3
- 数据库分区 3 包含表空间 USERSP1、USERSP2 和 USERSP3

进行了下列备份，其中 BK*xy* 表示分区 *y* 上的备份编号 *x*:

- BK11 是 SYSCATSPACE、USERSP1 和 USERSP2 的备份
- BK12 是 USERSP2 和 USERSP3 的备份
- BK13 是 USERSP1、USERSP2 和 USERSP3 的备份
- BK21 是 USERSP1 的备份
- BK22 是 USERSP1 的备份
- BK23 是 USERSP1 的备份
- BK31 是 USERSP2 的备份
- BK33 是 USERSP2 的备份
- BK42 是 USERSP3 的备份
- BK43 是 USERSP3 的备份

示例 13

下列命令将整个数据库重建至日志末尾。

1. 在数据库分区 1 上，发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令:

```
db2 restore db mydb rebuild with all tablespaces in database
    taken at BK31 without prompting
```

2. 在数据库分区 2 上，发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令:

```
db2 restore db mydb rebuild with tablespaces in database taken at
    BK42 without prompting
```

3. 在数据库分区 3 上，发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令:

```
db2 restore db mydb rebuild with all tablespaces in database
    taken at BK43 without prompting
```

4. 在目录分区上，发出带有 **TO END OF LOGS** 选项的 **ROLLFORWARD DATABASE** 命令（假定在所有数据库分区上，所有日志都已保存并且可以访问）:

```
db2 rollforward db mydb to end of logs
```

5. 发出带有 **STOP** 选项的 **ROLLFORWARD DATABASE** 命令:

```
db2 rollforward db mydb stop
```

此时，该数据库在所有数据库分区上都是可连接的，并且所有表空间都处于 NORMAL 状态。

示例 14

下列示例将 SYSCATSPACE、USERSP1 和 USERSP2 重建至最新时间点。

1. 在数据库分区 1 上，发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令:

```
db2 restore db mydb rebuild with all tablespaces in database
    taken at BK31 without prompting
```

2. 在数据库分区 2 上，发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令:

```
db2 restore db mydb rebuild with all tablespaces in image taken at
    BK22 without prompting
```

3. 在数据库分区 3 上，发出带有 **REBUILD** 选项的 **RESTORE DATABASE** 命令:

```
db2 restore db mydb rebuild with all tablespaces in image taken at
    BK33 without prompting
```

   注意: 此命令省略了 USERSP1，它是完成重建操作所必需的。

4. 在目录分区上，发出带有 **TO END OF LOGS** 选项的 **ROLLFORWARD DATABASE** 命令:

```
db2 rollforward db mydb to end of logs
```

5. 发出带有 **STOP** 选项的 **ROLLFORWARD DATABASE** 命令:

```
db2 rollforward db mydb stop
```

   前滚成功，并且该数据库在所有数据库分区上都是可连接的。所有表空间都处于 NORMAL 状态，但 USERSP3 和 USERSP1 除外（USERSP3 在它所在的所有数据库分区上处于 RESTORE PENDING 状态，而 USERSP1 在数据库分区 3 上处于 RESTORE PENDING 状态）。

   尝试访问数据库分区 3 上 USERSP1 中的数据时，将发生数据访问错误。要修正此错误，需要恢复 USERSP1:

   a. 在数据库分区 3 上，发出 **RESTORE DATABASE** 命令并指定包含 USERSP1 的备份映像:

```
db2 restore db mydb tablespace taken at BK23 without prompting
```

   b. 在目录分区上，发出带有 **TO END OF LOGS** 选项和 **AND STOP** 选项的 **ROLLFORWARD DATABASE** 命令:

```
db2 rollforward db mydb to end of logs on dbpartitionnum (3) and stop
```

此时，可以访问数据库分区 3 上 USERSP1 中的数据，因为 USERSP1 处于 NORMAL 状态。

情况 7

在下列示例中，有一个称为 MYDB 的*不可恢复*数据库，它具有下列表空间:

- SYSCATSPACE (0)，SMS 系统目录
- USERSP1 (1) DMS 用户数据表空间
- USERSP2 (2) DMS 用户数据表空间

只有该数据库的一个备份 BK1:

示例 15

下列命令演示了如何对不可恢复数据库使用重建。

仅使用 SYSCATSPACE 和 USERSP1 重建数据库:

```
db2 restore db mydb rebuild with tablespace (SYSCATSPACE, USERSP1)
    taken at BK1 without prompting
```

复原之后，该数据库是可连接的。如果发出 **LIST TABLESPACES** 命令或 MON_GET_TABLESPACE 表函数，那么您将看到 SYSCATSPACE 和 USERSP1 处于 NORMAL 状态，而 USERSP2 处于 DELETE_PENDING/OFFLINE 状态。现在就可以使用处于 NORMAL 状态的那两个表空间了。

如果要备份数据库，那么首先需要使用 DROP TABLESPACE 语句来删除 USERSP2，否则备份将失败。

要在以后复原 USERSP2，那么需要从 BK1 重新发出数据库复原命令。

监视复原操作的进度

可使用 **LIST UTILITIES** 命令来监视数据库上的复原操作。

过程

发出 **LIST UTILITIES** 命令并指定 **SHOW DETAIL** 参数

```
LIST UTILITIES SHOW DETAIL
```

结果

对于复原操作，不会给定最初估计值。而是指定 UNKNOWN。因为每个缓冲区都是从映像中读取的，所以实际读取的字节量会更新。对于可能要复原多个映像的自动增量复原操作，系统使用阶段来跟踪进度。每个阶段提供一个从增量链中复原的映像。最初，仅指示一个阶段。复原第一个映像之后，将显示阶段的总数。在复原每个映像时，会根据已处理的字节数来更新完成的阶段数。

示例

以下是用于监视复原操作性能的输出的示例:

```
ID                                   = 6
Type                                 = RESTORE
Database Name                        = SAMPLE
Partition Number     = 0
Description                          = db
Start Time                           = 08/04/2011 12:24:47.494191
State                = Executing
Invocation Type                      = User
   进度监视:
      Completed Work                 = 4096 bytes
      Start Time                     = 08/04/2011 12:24:47.494197
```

优化复原性能

执行复原操作时，DB2 数据库产品将自动为缓冲区个数、缓冲区大小和并行性设置选择最佳值。此值根据可用实用程序堆内存的数量、可用处理器数和数据库配置而定。

因此，根据系统上可用的存储量，应考虑通过增大 **util_heap_sz** 配置参数来分配更多内存。目的是将完成复原操作所用的时间降至最少。除非显式地输入以下 **RESTORE DATABASE** 命令参数的值，否则 DB2 数据库产品将为它们选择一个值:

- **WITH** *num-buffers* **BUFFERS**
- **PARALLELISM** *n*
- **BUFFER** *buffer-size*

对于复原操作，始终使用备份操作所使用的缓冲区大小的倍数。在发出 **RESTORE DATABASE** 命令时可以指定缓冲区大小，但需要确保其大小是备份缓冲区大小的倍数。

还可以选择执行以下任何操作来缩短完成一次复原操作所需的时间:

- 增加复原缓冲区大小。

  复原缓冲区大小必须是在备份操作期间指定的备份缓冲区大小的正整数倍数。如果指定了不正确的缓冲区，那么分配的缓冲区将是最小的可接受大小。
- 增加缓冲区的数量。

  您指定的值必须是用于备份的缓冲区大小的倍数，否则它将向下舍入至最接近的备份缓冲区大小倍数。
- 增加 **PARALLELISM** 参数的值。

  这将增加将用来在复原操作期间写至数据库的缓冲区操纵程序 (BM) 的数目。
- 增加实用程序堆大小

  这将增大可由其他实用程序同时使用的内存。

使用 restore 所需的特权、权限和授权

要从完整的数据库备份复原到*现有*数据库，必须具有 SYSADM、SYSCTRL 或 SYSMAINT 权限。要复原到*新*数据库，必须具有 SYSADM 或 SYSCTRL 权限。

特权使用户能够创建或访问数据库资源。权限级别提供了对特权、较高级别数据库管理器维护和实用程序操作进行分组的方法。这两者一起用于控制对数据库管理器及其数据库对象的访问。

用户只能访问那些他们具有相应授权（即必需的特权或权限）的对象。

数据库模式传输

传输数据库模式包括进行数据库的备份映像和将数据库模式复原至其他现有数据库。

传输数据库模式时，将重新创建被传输的模式中的数据库对象以引用新数据库，并将数据复原至新数据库。

必须将整个数据库模式进行传输。如果表空间包含您要传输的一个模式以及另一个模式，那么必须传输这两个模式中的所有数据对象。这些没有引用其他数据库模式的模式集称为*可传输集*。可传输集中的数据（位于表空间中）和逻辑对象（位于模式中）只能引用可传输集中的表空间和模式。例如，表只能与可传输集中的其他表具有表依赖关系。

下图阐明了具有几个表空间和模式的数据库。在该图中，模式引用的表空间位于模式上方。有些模式引用多个表空间，有些表空间被多个模式引用。

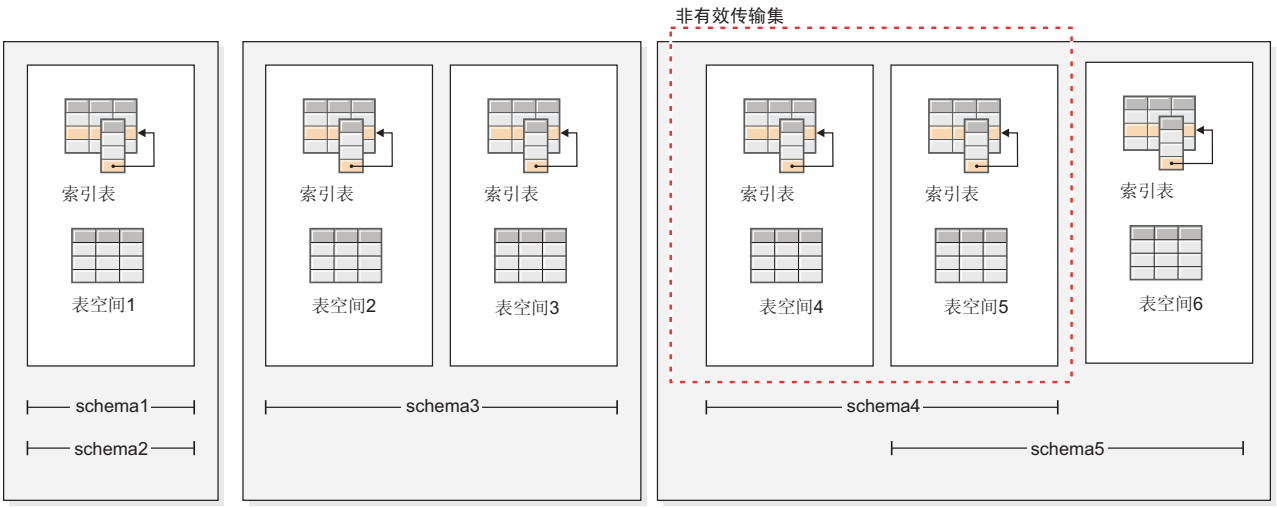


图 27. 表空间和模式集

下列表空间和模式的组合是有效的可传输集:

- 带有 schema1 和 schema2 的 tablespace1
- tablespace2 和带有 schema3 的 tablespace3
- 带有 schema4 和 schema5 的 tablespace4、tablespace5 和 tablespace6
- 有效可传输集的组合还可以组成有效的可传输集:
  - 带有 schema1、schema2 和 schema3 的 tablespace1、tablespace2 和 tablespace3

带有 schema4 的 tablespace4 和 tablespace5 集合不是有效的可传输集，因为在 tablespace5 和 schema5 之间以及 schema5 和 tablespace6 之间存在引用关系。该集合需要带有 schema5 的 tablespace6 是有效的可传输集。

可以使用带 **TRANSPORT** 参数的 **RESTORE** 命令来传输数据库模式。

传输数据库模式时，将在传输操作期间创建一个临时数据库并给其命名。传输登台数据库用于从备份映像提取逻辑对象，以便可以在目标数据库上对它们进行重新创建。如果备份映像中包括日志，那么将使用这些日志把登台数据库还原至事务性一致的点。然后，被传输的表空间的所有权将转移到目标数据库中。

有关传输数据库模式时数据库对象重新创建的注意事项

查看以下与传输数据库模式时重新创建数据库对象有关的信息:

表 blah

数据库对象	传输模式时的注意事项
SQL 例程（不是使用 SQL 的外部例程）	将在目标数据库中创建 SQL 例程的一份新副本。对于 SQL 存储过程，将占用额外的目录空间，因为会在新数据库中创建存储过程字节代码的其他副本。
外部例程	将为每个例程都创建新目录条目。此目录条目引用的二进制文件与原始的源例程引用的相同。**RESTORE** 命令不会从源系统复制外部例程二进制文件。

表 blah

数据库对象	传输模式时的注意事项
处于可导致访问问题的状态的源表	对于在生成备份映像时没有处于常规状态的表（例如处于检查暂挂状态或装入暂挂状态中的表），可能无法在目标数据库中访问那些表中的数据。要避免这种情况，可以先在源数据库中将表移至常规状态，再进行模式传输。
包含数据捕获属性的表	已启用数据捕获的源表会带有数据捕获属性被传输到目标数据库，并继续记录数据库间的数据复制信息。但是复制的表不会从此表中提取信息。**RESTORE** 命令完成后，用户可以选择注册新目标表，以让其充当复制源。
使用基于标签的访问控制 (LBAC) 的表	当传输受 LBAC 保护的数据时，传输操作会在目标数据库上重新创建 LBAC 对象。如果在目标数据库上存在具有相同名称的 LBAC 对象，那么传输操作将失败。为确保受限制的数据访问不被损坏，传输操作不会使用目标数据库上现有的 LBAC 对象。

传输表空间时，将在目标数据库上创建具有特殊格式的日志记录。此格式无法由先前的 DB2 版本读取。如果传输了表空间然后降级至低于 DB2 V9.7 FP2 的版本，那么无法恢复包含所传输的表空间的目标数据库。为确保目标数据库与先前的 DB2 版本兼容，可将目标数据库前滚至传输操作之前的时间点。

要点：如果数据库前滚操作检测到表空间模式传输日志记录，那么将使相应的已传输表空间脱机，并使其进入删除暂挂状态。这是因为数据库不具有已传输表空间的完整日志来重建已传输表空间及其内容。您可以在完成传输后制作目标数据库的完全备份，以便后续前滚操作不会将模式传输点传入日志流中。

可传输对象

将备份映像中的数据传输至目标数据库时，有两个主要结果。系统会在目标数据库中重新创建要复原的表空间中的物理和逻辑对象，并将表空间定义和容器添加至目标数据库。

将重新创建下列逻辑对象:

- 表、已创建全局临时表和具体化查询表
- 常规统计视图和统计视图
- 下列类型的生成列:
  - 表达式
  - 标识
  - 行更改时间戳记
  - 行更改标记
- 用户定义的函数和生成的函数
- 函数和过程，外部例程可执行项目除外
- 用户定义的类型
- 下列类型的约束:
  - 检查
  - 外键
  - 函数依赖性

  - 主数据库
  - 唯一
- 索引
- 触发器
- 序列
- 对象权限、特权、安全性、访问控制和审计配置
- 表统计信息、概要文件和提示
- 程序包

在目标数据库上将不创建下列模式的组件:

- 别名
- 已创建的全局变量
- 外部例程可执行文件
- 函数映射和模板
- 分层表
- 索引扩展
- 作业
- 方法
- 昵称
- OLE DB 外部函数
- 范围分区表
- 服务器
- 有源过程
- 结构化类型
- 系统目录
- 类型表和带类型视图
- 用法列表
- 包装器

传输示例

可以使用带有 TRANSPORT 选项的 **RESTORE DATABASE** 命令将一组表空间和 SQL 模式从一个数据库复制到另一个数据库。

下列示例使用名为 ORIGINALDB 的数据库作为备份映像的源并使用 TARGETDB 作为目标数据库。

下列说明显示 ORIGINALDB 的表空间和模式:

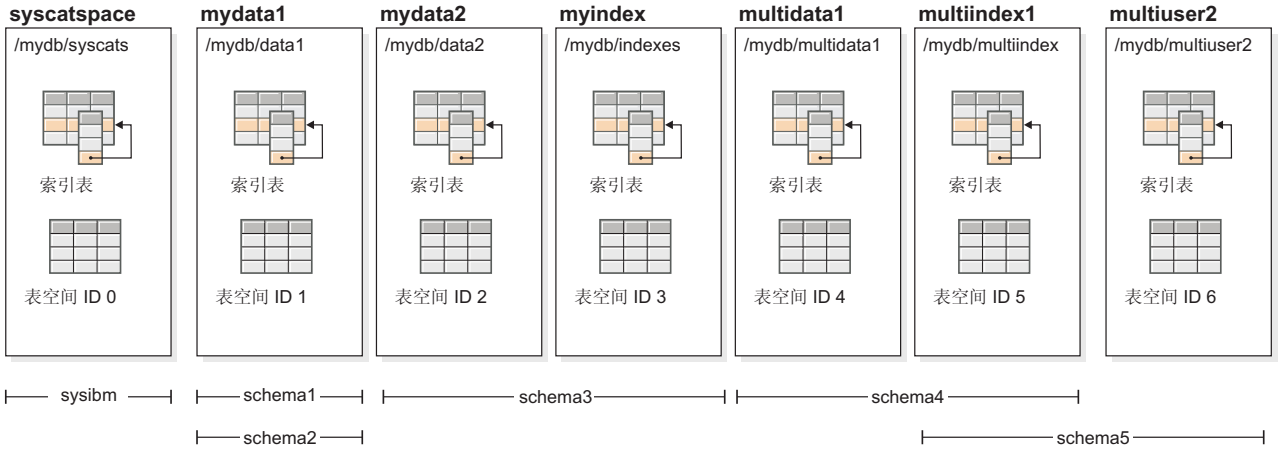


图 28. ORIGINALDB 数据库

originalDB 数据库包含下列有效的可传输集:

- mydata1; schema1 + schema2
- mydata2 + myindex; schema3
- multidata1 + multiindex1 + multiuser2; schema4 + schema5
- 有效可传输集的组合还可以组成有效的可传输集:
  - mydata1 + mydata2 + myindex; schema1 + schema + schema3

下列说明显示 TARGETDB 的表空间和模式:

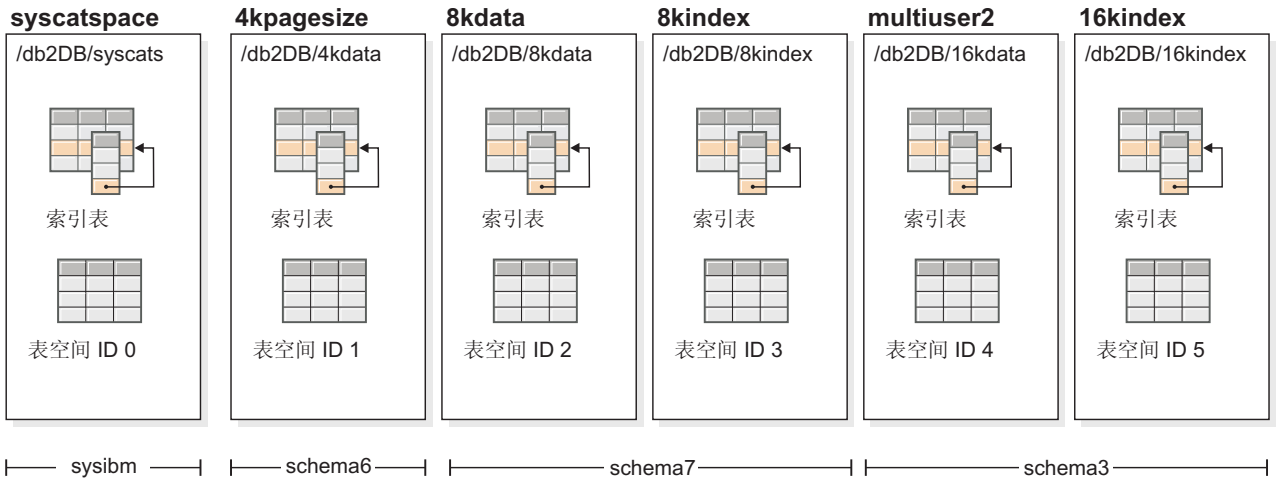


图 29. TARGETDB 数据库

如果源数据库和目标数据库包含任何具有相同模式名称的模式或任何具有相同表空间名称的表空间，那么您将无法将该模式或表空间传输到目标数据库中。对目标数据库发出包含模式（具有相同模式名称）或表空间（具有相同表空间名称）的传输操作将导致传输操作失败。例如，即使下列组是有效的可传输集，也无法将它直接传输到目标数据库:

- mydata2 + myindex; schema3（schema3 存在于源数据库和目标数据库中）

如果有 ORIGINALDB 的单个联机备份映像（包含数据库中的所有表空间），那么这将是传输的源。这也适用于表空间级别备份映像。

您可以重定向正在传输的表空间的容器路径。如果已使用了数据库相对路径，那么这特别重要。

示例

示例 1: 将 mydata1 表空间中的模式 schema1 和 schema2 成功传输到 TARGETDB 中。

```
db2 restore db originaldb tablespace (mydata1) schema(schema1,schema2)
  from <Media_Target_clause> taken at <date-time>
  transport into targetdb redirect
db2 list tablespaces
db2 set tablespace containers for <tablespace ID for mydata1>
using (path '/db2DB/data1')

db2 restore db originaldb continue
```

结果 TARGETDB 将包含 mydata1 表空间和 schema1 与 schema2。

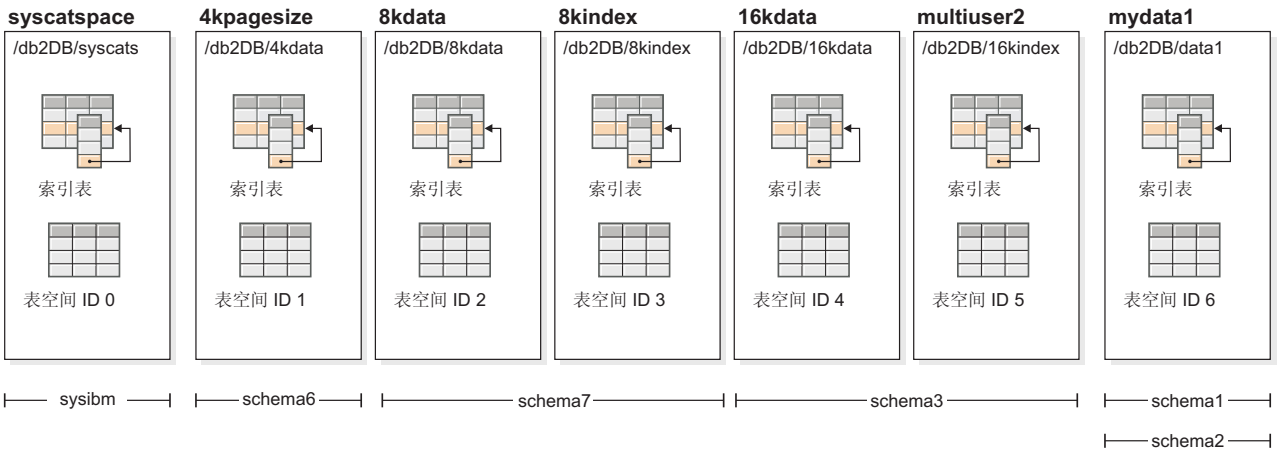


图 30. 传输后的 TARGETDB 数据库

示例 2: 将 mydata2 和 myindex 表空间中的模式 schema3 传输到 TARGETDB 中。您不能传输目标数据库上已存在的模式。

```
db2 restore db originaldb tablespace (mydata2,myindex) schema(schema3)
  transport into targetdb
```

传输操作将失败，因为模式 schema3 已存在于目标数据库上。 TARGETDB 将保持不变。SQLCODE=SQL2590N rc=3。

示例 3: 将 multidata1、multiindex1 和 multiuser2 表空间中的模式 schema4 和 schema5 传输到 TARGETDB 中。您不能传输目标数据库上已存在的表空间。

```
db2 restore db originaldb tablespace (multidata1,multiindex1,multiuser2)
  schema(schema4,schema5) transport into targetdb
```

传输操作将失败且 TARGETDB 将保持不变，因为表空间 multiuser2 已存在于目标数据库上。 SQLCODE=SQL2590N rc=3。

示例 4: 将 myindex 表空间传输到 TARGETDB。不能传输部分模式。

```
db2 restore db originaldb tablespace (myindex) schema(schema3)
  transport into targetdb
```

正在传输的表空间和模式的列表不是有效的可传输集。传输操作将失败，且 TARGETDB 将保持不变。SQLCODE=SQL2590N rc=1。

示例 5: 将 syscatspace 表空间复原到 TARGETDB。不能传输系统目录。

```
db2 restore db originaldb tablespace (syscatspace) schema(sysibm)
  transport into targetdb
```

传输操作将失败，因为不能传输系统目录。SQLCODE=SQL2590N rc=4。您可传输用户定义的表空间，或在不指定传输选项的情况下通过 RESTORE DATABASE 命令复原系统目录。

示例 6: 您无法复原到系统上不存在的目标数据库中。

```
db2 restore db originaldb tablespace (mydata1) schema(schema1,schema2)
  transport into notexists
```

传输操作将失败。无法将表空间传输到不存在的目标数据库。

故障诊断: 传输模式

如果登台数据库或目标数据库发生错误，那么必须重新执行整个复原操作。发生的所有失败都记录在目标服务器上的 db2diag log 日志文件中。请在重新发出 **RESTORE** 命令之前检查 **db2diag** 日志。

处理错误

可以采用各种方式来处理复原期间发生的错误，具体取决于正在复制的对象的类型和传输阶段。可能存在各种没有清除所有内容的情况（例如断电）。

传输操作由下列阶段构成:

- 登台数据库创建
- 物理表空间容器复原
- 前滚过程
- 模式验证
- 表空间容器所有权的传输
- 目标数据库中的模式重新创建
- 删除登台数据库（如果未指定 **STAGE IN** 参数）

如果在模式重新创建阶段结束时记录了关于传输物理对象的任何错误，那么复原操作会失败并返回错误。目标数据库上的所有对象创建都将回滚，并且所有内部创建的表在登台数据库上都将被清除。在重新创建阶段结束时将执行回滚，以允许将所有可能的错误都记录在 **db2diag** 日志文件中。您可以先研究返回的所有错误，然后再重新发出命令。

成功或失败后将自动删除登台数据库。然而，如果指定了 **STAGE IN** 参数，那么万一失败，也不会删除登台数据库。必须先删除登台数据库才能复用登台数据库名称。

第 14 章 rollforward 概述

您无法恢复在使用复原工具创建备份映像之后发生的事务。然而，您可以使用 rollforward 命令来恢复自从完成上一个 backup 命令以来发生的事务。您必须启用数据库日志记录以使这些命令生效。

ROLLFORWARD DATABASE 命令的最简单的格式只需要您指定想要前滚恢复的数据库的别名，如下列示例中所示:

```
db2 ROLLFORWARD DB sample
```

在 IBM Data Studio V3.1 或更高版本中，可以使用以下工具的任务助手：前滚数据库. 任务助手可以指导您执行以下过程：设置选项、查看自动生成的命令以执行任务以及运行这些命令。有关更多详细信息，请参阅使用任务助手管理数据库。

以下是一种可以用来执行前滚恢复操作的方法:

1. 在不指定 **STOP** 选项的情况下调用 ROLLFORWARD 实用程序。
2. 在指定 **QUERY STATUS** 选项的情况下调用 ROLLFORWARD 实用程序

   如果指定恢复至日志末尾，而返回的时间点又早于您预期的时间点，那么 **QUERY STATUS** 选项会指示丢失了一个或多个日志文件。

   如果指定时间点恢复，那么 **QUERY STATUS** 选项将有助于您确保前滚操作已在正确的时间点完成。
3. 在指定 **STOP** 选项的情况下调用 ROLLFORWARD 实用程序。在操作停止后，不可能前滚其他更改。

另一种可以用来执行前滚恢复操作的方法是:

1. 在指定 **AND STOP** 选项的情况下调用 ROLLFORWARD 实用程序。
2. 是否需要执行其他步骤取决于前滚操作的输出:
   - 如果前滚操作成功，前滚操作就会完成，并且数据库将是可连接并可使用的。此时，不可能前滚其他更改。
   - 如果返回了任何错误，那么执行需要的操作以解决问题。例如，如果缺少日志文件：就需要查找该日志文件，或者，如果发生了检索错误：那么应确保日志归档功能正常运行。然后，在指定 **AND STOP** 选项的情况下重新运行 ROLLFORWARD 实用程序。

必须先使用复原实用程序成功地复原数据库，然后才能前滚它，但表空间就不一样。表空间可以暂时处于前滚暂挂状态，但不需要执行复原操作以撤销该状态（例如在断电后执行该操作）。

调用 ROLLFORWARD 实用程序时:

- 如果数据库处于前滚暂挂状态，将前滚该数据库。如果任何表空间是从备份映像复原的，而该备份映像是在数据库备份映像之后生成的，并且该表空间当前处于前滚暂挂状态，那么还将前滚该表空间。如果任何表空间是在数据库级别备份之前生成的，并且在复原该数据库级别备份之后已进行复原，那么该表空间将仍保留前滚暂挂状态。您必须发出后续表空间级别前滚命令以对它们进行恢复。

© Copyright IBM Corp. 2001, 2013

- 如果数据库不处于前滚暂挂状态，但该数据库中的表空间处于前滚暂挂状态:
  - 如果指定表空间列表，将只前滚那些表空间。
  - 如果不指定表空间列表，将前滚处于前滚暂挂状态的所有表空间。

数据库前滚操作是脱机运行的。直到前滚操作成功完成数据库才可用，除非在调用实用程序时指定了 **STOP** 选项，否则该操作不能完成。

表空间前滚操作可脱机运行。直到前滚操作成功完成数据库才可用。当达至日志末尾时或在调用实用程序时指定了 **STOP** 选项，会发生这种情况。

只要表空间中未包括 SYSCATSPACE，就可对表空间执行*联机*前滚操作。对表空间执行联机前滚操作时，该表空间不可用，但数据库中的其他表空间是可用的。

首次创建数据库时，只对它启用了循环日志记录。这意味着日志是可复用的，而不是被保存或归档。使用循环日志记录，是不能进行前滚恢复的: 只能进行崩溃恢复或版本恢复。归档日志将建立备份后对数据库进行的更改记录成文档。通过将 **logarchmeth1** 数据库配置参数设置为其缺省值 OFF 以外的值来启用日志归档（和前滚恢复）。将 **logarchmeth1** 设置为 OFF 以外的值时，数据库将处于备份暂挂状态，而且必须脱机备份该数据库才能再次使用。

注: 在恢复历史记录文件中为前滚操作中使用的每个日志文件生成条目。
在此示例中，命令返回:

在分区数据库环境和 DB2 pureScale环境中，将为每个数据库分区或成员返回此状态信息:

```
  db2 rollforward db mydb to end of logs

                                前滚状态

输入数据库别名                  = mydb
返回了状态的成员数              = 3

成员标识  前滚                  要读取的           已处理的日志文件          上次落实的事务
          状态                    下一个日志
--------- ------------  ------------  -------------------------  -------------------------
        0  DB  工作          S0000001.LOG         S0000000.LOG-S0000000.LOG  2009-05-06-15.28.11.000000 UTC
        1  DB  工作          S0000010.LOG         S0000000.LOG-S0000009.LOG  2009-05-06-15.28.20.000000 UTC
        2  DB  工作          S0000005.LOG         S0000000.LOG-S0000004.LOG  2009-05-06-15.27.33.000000 UTC

 DB20000I  已成功完成 ROLLFORWARD 命令。
```

使用前滚

使用 **ROLLFORWARD DATABASE** 命令将数据库日志文件中记录的事务应用于复原的数据库备份映像或表空间备份映像。

开始之前

不应连接至要前滚恢复的数据库。前滚实用程序会自动建立与指定数据库的连接，并且此连接会在前滚操作完成时终止。

关于此任务

不要在未取消正在进行的前滚操作的情况下复原表空间。否则，您可能已设置以下表空间，在此表空间中，某些表空间处于正在执行前滚状态，某些表空间处于前滚暂挂状态。正在进行的前滚操作仅作用于处于正在执行前滚状态的表空间。

数据库可以是本地数据库或远程数据库。

ROLLFORWARD 实用程序有以下限制:

- 每次只能调用一个前滚操作。如果有多个要复原的表空间，可在同一操作中指定所有表空间。
- 如果在最近的备份操作后有重命名的表空间，应确保在前滚表空间时使用新名称。先前表空间名不被识别。
- 不能取消正在运行的前滚操作。只能取消已完成但那些尚未指定 **STOP** 参数的前滚操作或未完成就已失败的前滚操作。
- 指定的时间戳记小于前一时间戳记时，不能*继续*将表空间前滚至某时间点。如果未指定时间点，将使用前一时间点。可以只指定 STOP 来发出一个在指定时间点前结束的前滚操作，但只有在涉及的表空间都是从同一脱机备份映像复原时才能这样做。在此情况下，不需要日志处理。如果在正在进行的前滚操作完成或取消前使用另一表空间列表启动另一前滚操作，会返回一条错误消息（SQL4908）。对所有数据库分区调用 **LIST TABLESPACES** 命令（或使用 MON_GET_TABLESPACE 表函数），以确定当前正在前滚（“前滚进行中”状态）的表空间，以及已准备好进行前滚（“前滚暂挂”状态）的表空间。有 3 种选择:
  - 完成所有表空间上正在进行的前滚操作。
  - 完成表空间子集上正在进行的前滚操作。（如果前滚操作将继续到特定时间点，从而需要涉及到所有数据库分区，那么可能无法完成表空间子集上正在进行的前滚操作。）
  - 取消正在进行的前滚操作。
- 在分区数据库环境中，必须从数据库的目录分区调用 ROLLFORWARD 实用程序。
- DB2 V9.1 客户机中引入了表空间的时间点前滚。应将所有客户机升级至 V10.1 以将表空间前滚至某个时间点。
- 无法从前发行版前滚日志。

过程

要调用前滚实用程序，请使用:

- **ROLLFORWARD DATABASE** 命令，或者
- db2Rollforward 应用程序编程接口 (API)。
- 在 IBM Data Studio 中对 **ROLLFORWARD DATABASE** 命令打开任务助手。

示例

以下是通过 CLP 发出的 **ROLLFORWARD DATABASE** 命令的示例:

```
db2 rollforward db sample to end of logs and stop
```

继续已停止或失败的前滚操作

发生下列任一情况时可继续前滚操作: 先前前滚失败; 先前前滚中断; 或先前前滚完成但命令未指定 STOP 或 COMPLETE。

开始之前

不应连接至要前滚恢复的数据库。前滚实用程序会自动建立与指定数据库的连接，并且此连接会在前滚操作完成时终止。

关于此任务

继续前滚操作时您的一个选择是使用*强制停止*，您可通过发出带 **STOP** 选项的 **ROLLFORWARD DATABASE** 命令（但指定 **TO**）来完成此操作。强制停止意味着前滚实用程序忽略某些错误（如果忽略它们是安全的）。例如，缺少日志文件错误、校验和错误、日志链错误以及未达到时间点都被视为可忽略错误。如果 DB2 确定停止是安全的，那么前滚操作将完成撤销阶段，并且数据库将可用于正常连接。如果 DB2 确定停止是不安全的，那么前滚操作将失败，并且数据库保持前滚暂挂状态。

限制

如果要继续要达到某个时间点的前滚操作，那么新的前滚必须为下列其中一项:

- 至某个时间点的前滚
- 至稍后时间点的前滚
- 至日志结尾的前滚
- 带有 STOP 或 COMPLETE 选项但没有时间点、END OF LOGS 或 END OF BACKUP 选项的前滚

过程

要继续前滚操作，请使用以下步骤:

结果

示例

下一步做什么

前滚表空间中的更改

如果数据库启用了正向恢复，您可以选择备份、复原和前滚表空间，而不必前滚整个数据库。

您可以独立于数据库中的其他表空间来前滚对某个表空间所作的更改，您也可以一次前滚对所有表空间所作的更改。

您可能需要对各个表空间实现恢复策略，因为这可节省时间: 恢复数据库的一部分所需的时间比恢复整个数据库所需的时间少。

例如，如果一个磁盘损坏且它只包含一个表空间，那么可以复原并前滚该表空间，而不必恢复整个数据库，也不会影响用户访问该数据库的其余部分，除非损坏的表空间包含了系统目录表; 在此情况下，您不能连接至数据库。（如果包含系统目录表空间的表空间级备份映像是可用的，那么可以独立复原系统目录表空间。）表空间级的备份还允许您较为频繁地备份数据库的关键部分，因此所需的时间比备份整个数据库的时间少。

复原表空间后，它始终处于前滚暂挂状态。要使该表空间可以使用，必须对它执行前滚恢复。在大多数情况中，可以选择前滚至日志末尾或前滚至特定的时间点。但是不能将包含系统目录表的表空间前滚至某时间点。这些表空间必须前滚至日志末尾，以确保数据库中的所有表空间保持一致。

如果您想要跳过已知道不包含会影响表空间的任何日志记录的日志文件，请确保将 **DB2_COLLECT_TS_REC_INFO** 注册表变量设置为 ON（这是缺省值）。必须在创建和使用日志文件之前设置此注册表变量，以便收集跳过日志文件所需要的信息。如果 **DB2_COLLECT_TS_REC_INFO** 设置为 OFF，那么在前滚表空间时，DB2 将处理所有日志文件，即使它们不包含会影响该表空间的日志记录。

注: 在 DB2 pureScale 环境中不支持跳过日志。

数据库目录中的表空间更改历史记录文件 (DB2TSCHG.HIS) 记录应针对每个表空间进行处理的日志。通过使用 **db2logsForRfwd** 实用程序可以查看此文件的内容，通过使用 **PRUNE HISTORY** 命令可以从其中删除条目。执行数据库复原操作期间，从备份映像复原 DB2TSCHG.HIS，然后在数据库前滚操作期间对它进行更新。如果没有任何信息可用于日志文件，那么就好像每个表空间的恢复都需要该日志文件一样来对其进行处理。

因为在日志不活动之后每个日志文件的信息都会清空至磁盘，所以崩溃可能会导致此信息丢失。为弥补这一点，如果恢复操作从日志文件的中间开始，那么就好像该日志包含对系统中每个表空间的修改来处理整个日志。在这之后，将处理活动日志并重建它们的信息。如果在崩溃情况下丢失较旧的或归档的日志文件的信息，且它们在数据文件中没有任何信息，那么就好像它们包含表空间恢复操作期间对每个表空间的修改一样来处理这些日志文件。

将表空间前滚之前，使用 MON_GET_TABLESPACE 表函数来确定*最早恢复时间*，它是可将表空间前滚至的最早时间点。对表空间或表空间中的表运行数据定义语言 (DDL) 语句时，将更新此最早恢复时间。必须将该表空间至少前滚至最早恢复时间，以便与系统目录表中的信息同步。如果恢复多个表空间，表空间必须至少前滚至要恢复的所有表空间中的最高“最早恢复时间”。您不能将表空间前滚至早于备份时间戳记的时间。 在分区数据库环境中，必须至少将表空间前滚至所有数据库分区上的所有表空间中最晚的“最早恢复时间”。

如果要将表空间前滚到某时间点，而且有一个表包含在多个表空间中，那么必须同时前滚所有这些表空间。例如，如果该表数据包含在一个表空间中，而该表的索引包含在另一个表空间中，那么必须同时将这两个表空间前滚至相同的时间点。

如果表中的数据和长对象位于各自的表空间中，并且已将该长对象数据重组，那么必须同时复原并前滚数据和长对象的表空间。在重组该表之后，应该为受影响的表空间建立备份。

如果要将一个表空间前滚到某时间点，那么表空间中的表是:

- 另一个表空间中的具体化查询表或登台表的基础表
- 另一个表空间中的表的具体化查询表或登台表

应该将两个表空间前滚到同一时间点。否则，在前滚操作结束时，具体化查询表或登台表将处于设置完整性暂挂状态。具体化查询表将需要完全刷新，并且登台表将被标记为不完整。

如果要将一个表空间前滚至一个时间点，而该表空间中的一个表与另一个表空间所包含的另一个表存在引用完整性关系，那么应同时将这两个表空间前滚至相同的时间点。否则，在前滚操作结束时，引用完整性关系中的子表将处于设置完整性暂挂状态。以后检查子表是否存在约束违例时，需要对整个表进行检查。如果下列其中任何一个表存在，那么它们将与子表一起处于设置完整性暂挂状态:

- 子表的任何派生具体化查询表
- 子表的任何后代登台表
- 子表的任何后代外键表

这些表将需要完全完整性处理，以使它们脱离设置完整性暂挂状态。如果同时前滚这两个表空间，那么该约束将在该时间点前滚操作结束时仍然有效。

应确保时间点表空间前滚操作不会导致在某些表空间中回滚一个事务，而在另一些表空间中落实该事务。这种情况可能在下列时候发生:

- 对一个事务已更新的表空间的子集执行时间点前滚操作，且该时间点在事务落实的时间之前。
- 要前滚至某个时间点的表空间中所包含的任何表有一个关联的触发器，或者这个表由一个触发器来更新，该触发器影响的表空间不是要前滚的表空间。

解决方案是: 找到一个可阻止这种情况发生的适当的时间点。

可以发出 **QUIESCE TABLESPACES FOR TABLE** 命令来创建事务一致的时间点，以将表空间前滚。停顿请求（是共享的，用于更新或互斥方式）等待（通过锁定）对那些表空间运行的所有事务完成，并阻拦新请求。准许停顿请求时，表空间处于一致状态。要确定停止前滚操作的适当时间，可查看恢复历史记录文件以找出抑制点，并检查它是否是在最小复原时间后发生的。

表空间时间点前滚操作完成后，表空间将置于备份暂挂状态。必须建立该表空间的备份，因为在前滚到的时间点与当前时间之间对该表空间所作的所有更新都已被除去。再也不能从先前的数据库级别或表空间级别备份映像将该表空间前滚到当前时间。以下示例显示表空间级的备份映像为什么是必需的，以及如何使用。（要使表空间可用，可以备份整个数据库、处于备份暂挂状态的表空间，或包括处于备份暂挂状态的表空间的表空间集合。）

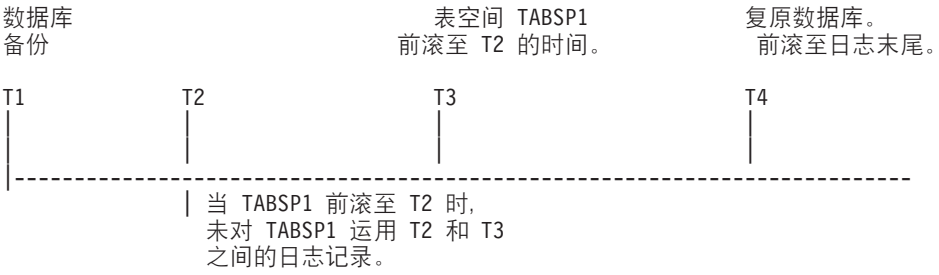


图 31. 表空间备份要求

在上述示例中，在 T1 时备份数据库。然后在 T3 时，表空间 TABSP1 前滚至某特定的时间点 (T2)，表空间在时间 T3 后备份。因为表空间处于备份暂挂状态，所以备份操作是必需的。表空间备份映像的时间戳记在 T3 后，但表空间是在 T2。T2 和 T3 之间的日志记录没有应用于 TABSP1。在时间 T4 处，将使用在 T1 处创建的备份映像复原数据库，并前滚至日志末尾。表空间 TABSP1 将在时间 T3 处置于复原暂挂状态，因为数据库管理器假设在 T3 和 T4 之间，对 TABSP1 执行操作，而无需对表空间应用了 T2 和 T3 之间的日志更改。如果这些日志更改是作为对数据库的前滚操作的一部分应用的，该假设将不成立。必须在表空间前滚到某时间点后建立的表空间级的备份，使您可将那个表空间前滚至前一时间点前滚操作之前（在上例中为 T3）。

假定要将表空间 TABSP1 复原到 T4，应该从 T3 之后建立的备份映像（必需的备份或稍后的一个）复原该表空间，然后将 TABSP1 前滚至日志末尾。

在前面的示例中，将数据库复原到时间 T4 最有效的方法应是按下列顺序执行必需的步骤:

1. 复原数据库。
2. 复原表空间。
3. 前滚数据库。

因为您在前滚数据库之前复原表空间，所以在前滚数据库时将不使用资源向表空间应用日志记录。

如果找不到时间 T3 后的 TABSP1 备份映像，或者想将 TABSP1 复原到 T3（或更早），可以:

- 将表空间前滚至 T3。无需再次复原表空间，因为它已根据数据库备份映像复原。
- 再次使用在时间 T1 建立的数据库备份来复原表空间，然后将该表空间前滚至时间 T3 之前的一个时间。
- 删除表空间。

在分区数据库环境中:

- 必须同时将一个表空间的所有部分前滚至同一时间点。这可确保该表空间在各数据库分区中是一致的。
- 如果某些数据库分区处于前滚暂挂状态，而且在其他数据库分区上，某些表空间处于前滚暂挂状态（但数据库分区不是），您必须首先前滚数据库分区，然后前滚表空间。
- 如果打算将表空间前滚至日志末尾，不必在每个数据库分区上复原：只需在需要恢复的数据库分区上复原即可。但如果打算将表空间前滚到某时间点，那么必须在每个数据库分区上将其复原。

在包含分区表的数据库中:

- 如果将包含任何分区表部分的表空间前滚到某个时间点，那么还必须将所有其他包含该表的表空间前滚到同一时间点。但是，允许将包含部分分区表的单个表空间前滚至日志末尾。如果分区表带有任何已连接、已拆离或已删除数据分区，那么时间点前滚还必须包括这些数据分区的所有表空间。要确定某个分区表是否带有任何已连接、已拆离或已删除的数据分区，请查询 SYSCAT.DATAPARTITIONS 目录视图。

DB2 pureScale 环境中的数据库前滚操作

在 DB2 pureScale 环境中，每个成员都有自己的日志流；但是，成功执行 **ROLLFORWARD DATABASE** 命令需要来自所有成员的日志流。

在数据库前滚操作期间，来自所有日志流的日志记录会合并并重放以使数据库保持一致。您在 **ROLLFORWARD DATABASE** 命令上指定的时间点与合并的日志流相关。要将数据库复原到一致状态，指定的时间必须晚于*最早恢复时间* (MRT)。MRT 是前滚操作期间数据库目录中列示对象与磁盘上实际存在的对象相匹配的最早时间。例如，如果要从联机备份操作期间创建的映像复原，那么针对前滚操作指定的时间点必须晚于联机备份操作的完成时间。这将确保数据库一致性。

针对后续数据库前滚操作指定的时间点不得早于合并的日志流中的 MRT；否则，该前滚操作失败 (SQL1276N)，并且会返回此 MRT 的时间戳记及错误消息。或者，可使用 END OF BACKUP 选项来自动前滚至 MRT。

建议使成员时钟同步；但是，可能无法使它们一直同步。这可能导致日志记录具有相同时间戳记，而合并的日志流包含的日志记录的时间戳记顺序混乱。在 DB2 pureScale 环境中，时间点数据库前滚操作在遇到其时间戳记晚于来自任何日志流的指定时间戳记的第一个日志记录时停止，并已处理对应数据库的 MRT 的日志记录。

未完成的或中断的前滚操作使数据库处于前滚暂挂状态。在此情况下，请再次发出 **ROLLFORWARD DATABASE** 命令。在 DB2 pureScale 环境中，可对同一成员或另一成员运行后续 **ROLLFORWARD DATABASE** 命令。

在 DB2 pureScale 环境中，如果要在新数据库中使用联机数据库备份映像执行数据库复原操作，那么正确方法取决于是所有日志文件还是只有备份映像中的日志文件可用。

- 如果可访问预先存在的日志文件或已归档的日志文件，那么以下前滚操作适用:

  ```
  db2 rollforward db dbname to end of logs and stop
  ```

  注: 备份之前，您需要确保日志归档路径设置为共享目录，以便所有成员都能访问日志以便执行后续前滚操作。如果无法从要在其上执行前滚的成员访问归档路径，那么会返回 SQL1273N。以下命令是将日志路径设置为共享目录的示例:

  ```
  db2 update db cfg using logarchmeth1
  DISK:/db2fs/gpfs1/svtdbm5/svtdbm5/ArchiveLOGS
  ```

  （其中 *gpfs1* 是成员的共享目录，*ArchiveLOGS* 是归档这些日志的实际目录）。

- 如果可访问的文件仅仅来自备份映像，那么以下前滚操作适用:

  ```
  db2 rollforward db dbname to end of backup and stop
  ```

  此命令重放所有必需日志记录以实现一致数据库状态（它在备份操作结束时生效）。如果可访问预先存在的日志文件或归档的日志文件，那么也可使用此命令，但它将在备份操作结束时停止；它不会使用备份操作结束后生成的任何额外日志。

  在此情况下，如果使用指定了 END OF LOGS 选项的 **ROLLFORWARD DATABASE** 命令，那么会返回 SQL1273N。后续带 STOP 选项的 **ROLLFORWARD DATABASE** 命令会成功，并且数据库将可用（如果不需要缺少的日志文件）。但是，如果需要缺少的日志文件（并且停止操作会不安全），那么前滚操作将再次返回 SQL1273N。

示例

假设有两个成员，M1 和 M2。M2 的时钟比 M1 的时钟快 5 秒。M2 的日志流包含以下日志记录:

A1 at 2010-04-03-14.21.56

A2 at 2010-04-03-14.21.56

B at 2010-04-03-14.21.58

C at 2010-04-03-14.22.01

M1 的日志流包含以下日志记录:

D at 2010-04-03-14.21.55

E at 2010-04-03-14.21.56

F at 2010-04-03-14.21.57

M2 上的数据库最早恢复时间 (MRT) 为 2010-04-03-14.21.55。因为 M1 的时钟慢 5 秒，所以日志记录 D、E 和 F 在合并的日志流中出现在较后位置:

```
MRT: 2010-04-03-14.21.55 (M2)
A1:  2010-04-03-14.21.56 (M2)
A2:  2010-04-03-14.21.56 (M2)
B:   2010-04-03-14.21.58 (M2)
D:   2010-04-03-14.21.55 (M1) --> corresponding time on M2 is 14.22.00
C:   2010-04-03-14.22.01 (M2)
E:   2010-04-03-14.21.56 (M1) --> corresponding time on M2 is 14.22.01
F:   2010-04-03-14.21.57 (M1) --> corresponding time on M2 is 14.22.02
```

字母字符（A1、 A2、B 等等）代表的顺序即在运行时（在成员间）实际写入对应日志记录的顺序。请注意，来自成员 M2 的日志记录 A1 和 A2 具有相同的时间戳记；DB2 数据服务器尝试在将日志缓冲区中的数据写至日志文件时包括来自多个事务的落实日志记录来优化性能。

以下命令返回 SQL1276N（数据库“test”不能脱离前滚暂挂状态，除非前滚通过“2010-04-03-14.21.55”或之后的时间点）:

```
db2 rollforward db test to 2010-04-03-14.21.54
```

但是，以下命令最多将数据库前滚至日志记录 A2 并包括该日志记录:

```
db2 rollforward db test to 2010-04-03-14.21.56
```

因为日志记录 A1 和 A2 都具有不晚于此命令中指定的时间的时间戳记，所以它们都会重放。日志记录 B 的时间戳记 (2010-04-03-14.21.58) 晚于指定值 (2010-04-03-14.21.56)，此日志记录会停止前滚操作并且不会重放。日志记录 D 也不会重放，即使其时间戳记早于指定值，原因是先遇到了日志记录 B 的较高值 (2010-04-03-14.21.58)。以下命令最多将数据库前滚至日志记录 D 并包括该日志记录:

```
db2 rollforward db test to 2010-04-03-14.21.58
```

日志记录 C 的时间戳记 (2010-04-03-14.22.01) 晚于指定值 (2010-04-03-14.21.58)，此日志记录会停止前滚操作并且不会重放。日志记录 E 也不会重放，即使其时间戳记早于指定值也是如此。

监视前滚操作

您可以使用 **db2pd** 或 **LIST UTILITIES** 命令来监视数据库前滚操作的进度。

过程

- 发出 **LIST UTILITIES** 命令并指定 **SHOW DETAIL** 参数

  ```
  LIST UTILITIES SHOW DETAIL
  ```

- 发出 **db2pd** 命令并指定 **-recovery** 参数:

  ```
  db2pd -recovery
  ```

结果

对于前滚恢复，进度监视分为两个阶段: FORWARD 和 BACKWARD。FORWARD 阶段期间中，读取日志文件并将日志记录应用于数据库。对于前滚恢复，当此阶段开始时，将工作总量估计值指定为 UNKNOWN。按字节计的已处理工作量将随进程的继续而更新。

在 BACKWARD 阶段中，回滚 FORWARD 阶段期间中应用任何未落实的更改。提供了要处理的日志数据量的估计值（按字节计）。当进程继续运行时，会更新已处理的工作量（按字节计）。

示例

以下示例是使用 **db2pd** 命令监视前滚操作性能所获得的输出:

```
Recovery:
Recovery Status     0x00000401
Current Log         S0000005.LOG
Current LSN         0000001F07BC
Current LSO         000002551BEA
Job Type            ROLLFORWARD RECOVERY
Job ID              7
Job Start Time      (1107380474) Wed Feb  2 16:41:14 2005
Job Description     Database Rollforward Recovery
Invoker Type        User
Total Phases        2
Current Phase       1

Progress:
Address            PhaseNum Description StartTime               CompletedWork  TotalWork
0x0000000200667160 1        Forward     Wed Feb  2 16:41:14 2005 2268098 bytes  Unknown
0x0000000200667258 2        Backward    NotStarted              0 bytes        Unknown
```

以下示例是使用附带 SHOW DETAIL 选项的 **LIST UTILITIES** 命令监视数据库前滚操作性能所获得的输出:

```
ID                                = 7
Type                              = ROLLFORWARD RECOVERY
Database Name                     = TESTDB
Member Number                     = 0
Description                       = Database Rollforward Recovery
Start Time                        = 01/11/2012 16:56:53.770404
State                             = Executing
Invocation Type                   = User
Progress Monitoring:
   Estimated Percentage Complete = 50
   Phase Number                   = 1
      Description                 = Forward
      Total Work                  = 928236 bytes
      Completed Work              = 928236 bytes
      Start Time                  = 01/11/2012 16:56:53.770492

   Phase Number [Current]         = 2
      Description                 = Backward
      Total Work                  = 928236 bytes
      Completed Work              = 0 bytes
      Start Time                  = 01/11/2012 16:56:56.886036
```

以下示例是使用附带 SHOW DETAIL 选项的 **LIST UTILITIES** 命令监视表空间前滚操作性能所获得的输出:

```
ID                                = 17
Type                              = ROLLFORWARD RECOVERY
Database Name                     = TESTDB
Member Number                     = 0
Description                       = Offline Tablespace Rollforward Recovery: 3
Start Time                        = 01/11/2012 17:04:27.269171
State                             = Executing
Invocation Type                   = User
Progress Monitoring:
   Estimated Percentage Complete = 63
   Phase Number                   = 1
      Description                 = Forward
      Total Work                  = 142
      Completed Work              = 90
```

```
   Start Time                 = 01/11/2012 17:04:27.269283

Phase Number [Current]        = 2
   Description                = Backward
   Total Work                 = 0
   Completed Work             = 0
   Start Time                 = Not Started
```

rollforward 所需的授权

必须具有 SYSADM、SYSCTRL 或 SYSMAINT 权限才能使用 ROLLFORWARD 实用程序。

特权使用户能够创建或访问数据库资源。权限级别提供了对特权、较高级别数据库管理器维护和实用程序操作进行分组的方法。这两者一起用于控制对数据库管理器及其数据库对象的访问。

用户只能访问那些他们具有相应授权（即必需的特权或权限）的对象。

前滚会话 - CLP 示例

您可以从命令行提示符处发出 rollforward 命令。在发出 rollforward 命令之前，您可能会发现复查某些样本会话很有帮助。

示例 1

ROLLFORWARD DATABASE 命令允许每次指定多个操作，各个操作由关键字 AND 隔开。例如，要前滚至日志末尾，然后完成，可将下列独立的命令:

```
db2 rollforward db sample to end of logs
db2 rollforward db sample complete
```

组合为:

```
db2 rollforward db sample to end of logs and complete
```

虽然这两种方法是等效的，但建议您分两个步骤来完成此类操作。在停止前滚操作前验证它是否具有预期的进度，以免丢失任何日志，这一点很重要。

如果前滚命令遇到错误，前滚操作就无法完成。在此情况下，将返回该错误，这样，您就能够修正该错误并重新发出以上命令。但是，如果无法修正该错误，那么可以通过发出以下命令强制前滚完成:

```
db2 rollforward db sample complete
```

此命令使数据库联机并复原到发生故障前日志点。

示例 2

将数据库前滚至日志末尾（已复原了两个表空间）:

```
db2 rollforward db sample to end of logs
db2 rollforward db sample to end of logs and stop
```

这两个语句是等效的。不需要 AND STOP 或 AND COMPLETE 表空间就可以前滚恢复至日志末尾。不需要表空间名称。如果未指定，将包括所有需要前滚恢复的表空间。如果将只前滚这些表空间的一个子集，那么必须指定它们的名称。

示例 3

复原了 3 个表空间后，将其中一个前滚至日志末尾，另两个前滚到某时间点，所有操作都要联机执行:

```
    db2 rollforward db sample to end of logs tablespace(TBS1) online
db2 rollforward db sample to 1998-04-03-14.21.56 and stop
        tablespace(TBS2, TBS3) online
```

应注意，两个前滚操作不能并发运行。只有在成功地完成了第一个前滚操作后，才能调用第二个命令。

示例 4

复原数据库后，前滚到某时间点，使用 OVERFLOW LOG PATH 来指定用户出口用来保存已归档日志的目录:

```
db2 rollforward db sample to 1998-04-03-14.21.56 and stop
        overflow log path (/logs)
```

示例 5

在以下示例中，有一个称为 sample 的数据库。备份该数据库，在备份映像中包含恢复日志；复原该数据库；然后，将该数据库前滚到备份时间戳记末尾。

备份该数据库，在备份映像中包含恢复日志:

```
db2 backup db sample online include logs
```

使用备份映像复原该数据库:

```
db2 restore db sample
```

将该数据库前滚到备份时间戳记末尾:

```
db2 rollforward db sample to end of backup
```

示例 6（分区数据库环境）

有三个数据库分区: 0、1 和 2。在所有数据库分区上定义表空间 TBS1，在数据库分区 0 和 2 上定义表空间 TBS2。在数据库分区 1 上复原了数据库，并在数据库分区 0 和 2 上复原了 TBS1 之后，在数据库分区 1 上前滚数据库:

```
db2 rollforward db sample to end of logs and stop
```

这会返回警告 SQL1271（已恢复数据库，但数据库分区 0 和 2 上的一个或多个表空间处于脱机状态）。

```
db2 rollforward db sample to end of logs
```

此命令在数据库分区 0 和 2 上前滚 TBS1。在此情况下，子句 TABLESPACE(TBS1) 是可选的。

示例 7（分区数据库环境）

在以下示例中，有一个称为 sample 的分区数据库。使用单系统视图备份备份所有数据库分区；在所有数据库分区上复原该数据库；将该数据库前滚到备份时间戳记末尾。

执行单系统视图 (SSV) 备份:

```
db2 backup db sample on all nodes online include logs
```

在所有数据库分区上复原该数据库:

```
db2_all "db2 restore db sample taken at 1998-04-03-14.21.56"
```

将该数据库前滚到备份时间戳记末尾:

```
db2 rollforward db sample to end of backup on all nodes
```

示例 8（分区数据库环境）

在以下示例中，有一个称为 sample 的分区数据库。使用 db2_all，通过一个命令备份所有数据库分区；在所有数据库分区上复原该数据库；并将该数据库前滚到备份时间戳记末尾。

使用 db2_all，通过一个命令备份所有数据库分区:

```
db2_all "db2 backup db sample include logs to //dir/"
```

在所有数据库分区上复原该数据库:

```
db2_all "db2 restore db sample from //dir/"
```

将该数据库前滚到备份时间戳记末尾:

```
db2 rollforward db sample to end of backup on all nodes
```

示例 9（分区数据库环境）

只在数据库分区 0 和 2 上复原表空间 TBS1 之后，在数据库分区 0 和 2 上前滚 TBS1:

```
    db2 rollforward db sample to end of logs
```

忽略数据库分区 1。

```
db2 rollforward db sample to end of logs tablespace(TBS1)
```

此命令失败，因为 TBS1 未对在数据库分区 1 上进行前滚恢复作好准备。报告 SQL4906N。

```
db2 rollforward db sample to end of logs on
         dbpartitionnums (0, 2) tablespace(TBS1)
```

成功完成。

```
db2 rollforward db sample to 1998-04-03-14.21.56 and stop
         tablespace(TBS1)
```

此命令失败，因为 TBS1 未对在数据库分区 1 上进行前滚恢复做好准备；必须将所有段一起前滚。

注: 在将表空间前滚到某时间点之后，将不接受 dbpartitionnum 子句。前滚操作必须在表空间所在的所有数据库分区上进行。

在数据库分区 1 上复原 TBS1 后:

```
db2 rollforward db sample to 1998-04-03-14.21.56 and stop
         tablespace(TBS1)
```

成功完成。

示例 10（分区数据库环境）

在所有数据库分区上复原表空间后前滚至 PIT2，但不指定 AND STOP。前滚操作仍在进行中。取消并前滚至 PIT1:

```
db2 rollforward db sample to pit2 tablespace(TBS1)
   db2 rollforward db sample cancel tablespace(TBS1)

** restore TBS1 on all dbpartitionnums **

   db2 rollforward db sample to pit1 tablespace(TBS1)
   db2 rollforward db sample stop tablespace(TBS1)
```

示例 11（分区数据库环境）

前滚恢复 db2nodes.cfg 文件中列示的 8 个数据库分区（3 至 10）上的表空间:

```
db2 rollforward database dwtest to end of logs tablespace (tssprodt)
```

前滚恢复至日志末尾（而不是时间点）的操作成功完成。不必指定表空间所在的数据库分区。实用程序缺省到 db2nodes.cfg 文件。

示例 12（分区数据库环境）

前滚恢复单一数据库分区数据库分区组（在数据库分区 6 上）上的 6 个小表空间:

```
db2 rollforward database dwtest to end of logs on dbpartitionnum (6)
         tablespace(tsstore, tssbuyer, tsstime, tsswhse, tsslscat, tssvendor)
```

前滚恢复至日志末尾（而不是时间点）的操作成功完成。

示例 13（分区表 - 在所有数据分区上前滚至日志末尾）

使用表空间 tbsp1、tbsp2 和 tbsp3 创建了分区表，索引在 tbsp0 中。后来，用户对 tbsp4 中的表添加了数据分区，并从 tbsp5 中的表连接了数据分区。可以将所有表空间前滚至日志末尾。

```
db2 rollforward db PBARDB to END OF LOGS and stop
        tablespace(tbsp0, tbsp1, tbsp2, tbsp3, tbsp4, tbsp5)
```

成功完成。

示例 14（分区表 - 在一个表空间上前滚至日志末尾）

最初使用表空间 tbsp1、tbsp2 和 tbsp3 创建了分区表，索引在 tbsp0 中。后来，用户对 tbsp4 中的表添加了数据分区，并从 tbsp5 中的表连接了数据分区。表空间 tbsp4 损坏并要求复原和前滚至日志末尾。

```
db2 rollforward db PBARDB to END OF LOGS and stop tablespace(tbsp4)
```

成功完成。

示例 15（分区表 - 在所有数据分区上前滚到时间点，这些数据分区包括那些已添加、已连接、已拆离或带索引的数据分区）

使用表空间 tbsp1、tbsp2 和 tbsp3 创建了分区表，索引在 tbsp0 中。后来，用户对 tbsp4 中的表添加了数据分区，从 tbsp5 中的表连接了数据分区，并从 tbsp1 拆离了数据分区。用户对分区表使用的所有表空间（包括 INDEX IN 子句中指定的那些表空间）执行前滚到 PIT 操作。

```
db2 rollforward db PBARDB to 2005-08-05-05.58.53 and stop
        tablespace(tbsp0, tbsp1, tbsp2, tbsp3, tbsp4, tbsp5)
```

成功完成。

示例 16（分区表 - 在一小部分表空间上前滚到 PIT）

使用三个表空间（tbsp1、tbsp2 和 tbsp3）创建了分区表。后来，用户从 tbsp3 中拆离了所有数据分区。只允许在 tbsp1 和 tbsp2 上前滚到 PIT。

```
db2 rollforward db PBARDB to 2005-08-05-06.02.42 and stop
         tablespace( tbsp1, tbsp2)
```

成功完成。

第 15 章 使用 IBM Tivoli Storage Manager (TSM) 进行数据恢复

调用 **BACKUP DATABASE** 或 **RESTORE DATABASE** 命令时，可以指定要使用 IBM Tivoli Storage Manager (TSM) 产品来管理数据库或表空间备份或复原操作。

除了在下列平台上之外，需要的 TSM 客户机 API 的最低级别为 V4.2.0:

- 64 位 Solaris 系统，它需要 TSM 客户机 API V4.2.1。
- 64 位 Windows 操作系统，它需要 TSM 客户机 API V5.1。
- 所有 Windows x64 系统，这些系统需要 TSM 客户机 API V5.3.2。
- 32 位 Linux for IBM Power Systems™ 和 Linux for pSeries，它们需要 TSM 客户机 API V5.1.5 或更高版本。
- 64 位 Linux for IBM Power Systems 和 Linux for pSeries，它们需要 TSM 客户机 API V5.2.2 或更高版本。
- 64 位 Linux on AMD Opteron 系统，它们需要 TSM 客户机 API V5.2.0 或更新版本。
- Linux for zSeries，它们需要 TSM 客户机 API V5.2.2 或更新版本。

配置 Tivoli Storage Manager 客户机

要使 DB2 数据库管理器能够使用 IBM Tivoli Storage Manager (TSM) 客户机来管理数据库、表空间备份或复原操作，必须首先配置 TSM 环境。

开始之前

必须安装和配置可用的 TSM 客户机和服务器。另外，必须在每台 DB2 数据库服务器上安装 TSM 客户机 API。如果已将 TSM 服务器配置为支持 TSM 客户机代理节点，那么 TSM 客户机代理节点将受支持。有关服务器配置和代理节点支持的信息，请参阅第 375 页的『使用 Tivoli Storage Manager 时的注意事项』，或参阅 Tivoli 文档。

过程

配置 DB2 数据库系统使用的 TSM 环境:

1. 设置 TSM 客户机 API 使用的环境变量:

   DSMI_DIR
   标识 API 可信代理程序文件（dsmtca）所在的用户定义目录路径。

   DSMI_CONFIG
   标识 dsm.opt 文件（它包含 TSM 用户选项）的用户定义目录路径。与另外两个变量不同，此变量应包含标准路径和文件名。

   DSMI_LOG
   标识将在其中创建错误日志（dsierror.log）的用户定义目录路径。

   注: 在多分区数据库环境中，这些设置必须在 sqllib/userprofile 文件中指定。

© Copyright IBM Corp. 2001, 2013

2. 如果对这些环境变量进行了任何更改，并且数据库管理器正在运行，那么先停止然后重新启动数据库管理器。 例如:
   - 使用 **db2stop** 命令停止数据库管理器。
   - 使用 **db2start** 命令启动数据库管理器。
3. 根据服务器配置的不同，Tivoli 客户机可能需要密码才能与 TSM 服务器交互。

   如果将 TSM 环境配置为使用 PASSWORDACCESS=generate，那么 Tivoli 客户机需要建立它的密码。

   将可执行文件 dsmapipw 安装在实例所有者的 sqllib/adsm 目录中。此可执行文件允许您建立和重设 TSM 密码。

   要执行 **dsmapipw** 命令，必须作为本地管理员或 『root』 用户登录。当执行此命令时，将提示您输入下列信息:
   - *旧密码*，它是 TSM 服务器识别的该 TSM 节点的当前密码。第一次执行此命令时，此密码是 TSM 管理员在 TSM 服务器上注册节点时所提供的密码。
   - *新密码*，它是 TSM 节点的新密码，存储在 TSM 服务器上。（将提示您输入新密码两次，以检查是否有输入错误。）

   注: 调用 **BACKUP DATABASE** 或 **RESTORE DATABASE** 命令的用户不需要知道此密码。只需要在为初始连接建立密码时，以及 TSM 服务器上的密码重新设置后，才需要运行 **dsmapipw** 命令。

下一步做什么

如果想要使用代理节点，那么根据备份和日志归档策略，您可能需要执行附加步骤来配置 TSM 客户机。代理节点允许您将存在于多个客户机节点上或多个用户下的数据库的备份或日志归档合并到 TSM 服务器上的共用目标节点名。当执行备份的管理员或计算机可随时间更改（例如，随集群更改）时，此配置非常有用。asnodename 选项还允许从其他计算机复原数据或从除执行备份的用户外的其他用户复原数据。

如果要在 DB2 pureScale环境中使用 TSM，那么建议使用代理节点配置，因为每个成员都可以表示为 TSM 客户机或节点，并可以映射至公共代理节点。

如果在缺省情况下不想使用代理节点，那么不需要其他客户机设置。当您想要使用代理节点执行备份或复原操作时，请在调用 **BACKUP DATABASE** 或 **RESTORE DATABASE** 命令时在 **OPTIONS** 参数中指定 asnodename 值。

如果在缺省情况下想要使用 TSM 代理节点，请使用下列方法:
- 更新数据库配置参数，以对不同的数据库使用不同的代理节点。
- 更新 dsm.sys 文件，以对机器上的所有用户和数据库使用同一代理节点。

  注: 使用相同 TSM 代理名称的每个用户与主机组合对于 TSM 将表现为同一个 DB2 实例。这意味着，如果多个 DB2 实例在 TSM 客户机节点代理配置中使用相同的数据库名称，那么它们可能覆盖彼此的日志归档和备份映像。要避免出现这种情况，请执行以下操作:
  - 为每个 DB2 实例创建一个不同的代理主机名。
  - 如果多个 DB2 实例可能使用相同 TSM 代理名称来创建数据库，那么不要使用 TSM 的客户机节点代理功能部件。

使用 vendoropt、logarchopt1 和 logarchopt2 的 TSM 客户机设置

可以设置下列数据库配置参数中的一个或多个参数，以对每个数据库启用不同的代理节点设置:

- 要启用使用 TSM 的命令（例如，backup 和 restore）以使用代理节点，请在 **vendoropt** 数据库配置参数中指定 asnodename 选项，如下所示:

  ```
  db2 update db cfg for dbname using vendoropt "'-asnodename=proxynode'"
  ```

  其中 *proxynode* 是共享 TSM 代理节点的名称。

- 要配置将日志归档到 TSM 服务器，请将 **logarchmeth1** 数据库配置参数设置为 TSM 并将代理节点的名称指定为 **logarchopt1** 数据库配置参数中的 asnodename 值，如下所示:

  ```
  db2 update db cfg for dbname using logarchmeth1 tsm
  logarchopt1 "'-asnodename=proxynode'"
  ```

  其中 *proxynode* 是共享 TSM 代理节点的名称。

  您可以对 **logarchmeth2** 和 **logarchopt2** 数据库配置参数做类似的更新。

在 DB2 pureScale环境中，这些数据库配置参数都是全局参数，您可以从任何成员中对这些数据库配置参数进行设置。

使用 dsm.sys 文件的 TSM 客户机设置方法

1. 编辑 dsm.sys 文件并添加代理节点信息，如下所示:

   ```
   asnodename proxynode
   ```

   其中 *proxynode* 是共享 TSM 代理节点的名称。

2. 请确保在 **DSMI_CONFIG** 路径中指定的 dsm.opt 文件包含 TSM 服务器的名称，如下所示:

   ```
   servername servername
   ```

   其中 *servername* 是 TSM 服务器名称。

使用 Tivoli Storage Manager 时的注意事项

您可以使用 Tivoli Storage Manager 来创建 DB2 数据库的备份映像。当您决定对数据库使用哪个备份工具时，必须考虑每个可用选项的功能和限制。

- 要使用 Tivoli Storage Manager (TSM) 中的特定功能部件，可能需要指定使用该功能部件的对象的标准路径名。记住在 Windows 操作系统上，将使用 \ 来代替 /。以下是不同对象的标准路径名:
  - 完整数据库恢复对象为: /*database*/DBPART*nnn*/FULL_BACKUP.timestamp.seq_no
  - 增量数据库恢复对象为: /*database*/DBPART*nnn*/DB_INCR_BACKUP.timestamp.seq_no
  - 增量差异数据库恢复对象为: /*database*/DBPART*nnn*/DB_DELTA_BACKUP.timestamp.seq_no
  - 完整表空间恢复对象为: /*database*/DBPART*nnn*/TSP_BACKUP.timestamp.seq_no
  - 增量表空间恢复对象为: /*database*/DBPART*nnn*/TSP_INCR_BACKUP.timestamp.seq_no

– 增量差异表空间恢复对象为: /*database*/DBPART*nnn*/TSP_DELTA_BACKUP.timestamp.seq_no

其中 *database* 是数据库别名，而 DBPART*nnn* 是数据库分区编号。显示的大写名称必须输入为大写。

- 对于多个备份映像使用同一个数据库别名的情况，时间戳记和序号就成为标准名称中的专有部分。需要查询 TSM 以确定要使用的备份版本。
- 如果执行联机备份操作并指定了 USE TSM 选项和 INCLUDE LOGS 选项，那么当两个进程尝试同时写同一台磁带机时，就会发生死锁。如果正在将磁带机用作日志和备份映像的存储设备，那么需要为 TSM 定义两个独立的磁带池，一个用于存储备份映像，另一个用于存储归档日志。
- 要使用客户机代理节点，TSM 管理员必须在 TSM 服务器上完成下列步骤:
  1. 如果 DB2 客户机尚未向 TSM 服务器注册，请使用 **register node** TSM 命令注册每个客户机。
  2. 请使用 **register node** TSM 命令向 TSM 服务器注册要由客户机组使用的（虚拟）常用 TSM 节点名。
  3. 请使用 **grant proxynode** TSM 命令授予组中的所有计算机代理权限。

  有关如何设置代理节点客户机的信息，请参阅第 373 页的『配置 Tivoli Storage Manager 客户机』，或参阅 Tivoli 文档。
- 在仅修改了少量页面的情况下执行增量备份时，可能必须增大 TSM 参数 **IDLETIMEOUT**，以使它大于完成对最大表空间进行备份所需的时间。这可以防止 TSM 在完成增量备份之前关闭会话。

第 16 章 DB2 高级副本服务 (ACS)

DB2 高级副本服务 (ACS) 允许您使用存储设备的快速复制技术来执行备份和复原操作的数据复制部分。

在传统的备份或复原操作中，数据库管理器使用操作系统调用将数据复制到磁盘或者从磁盘或存储设备复制数据。能够使用存储设备来执行数据复制可以使备份和复原操作更为快速。使用 DB2 ACS 的备份操作称为快照备份。

要执行快照备份和复原操作，需要适用于存储设备的 DB2 ACS API 驱动程序。有关集成驱动程序的受支持存储硬件的列表，请参阅 Tivoli 文档，网址为: 受支持存储子系统

DB2 ACS 与存储解决方案（例如，从 V10.1 开始与 DB2 捆绑在一起的 Tivoli Storage FlashCopy Manager）交互。有关安装和使用 Tivoli Storage FlashCopy Manager 的详细指示信息，请参阅 Tivoli 文档，网址为: http://www.ibm.com/developerworks/wikis/display/tivolidoccentral/Tivoli+Storage+FlashCopy+Manager

DB2 高级副本服务 (ACS) 最佳实践

在安装和配置 DB2 高级副本服务 (ACS) 时，请考虑下列最佳实践。

为日志路径指定专用卷组

建议要包括在自己的快照卷中的日志路径独立于数据库目录和数据库容器。

为每个数据库分区指定一个卷组

在分区数据库环境中，每个数据库分区必须位于独立于其他数据库分区的一组快照卷中。

嵌入式版本的 Tivoli Storage FlashCopy Manager 的限制

如果将 DB2 高级副本服务 (ACS) 与随 DB2 交付的 Tivoli Storage FlashCopy Manager 版本配合使用，那么应知道以下限制。

不支持卷共享。如果一个数据库分区与任何其他数据库分区位于相同存储卷上，那么不允许执行快照操作。此外，为使用没有许可证文件的 Tivoli Storage FlashCopy Manager（即嵌入式版本），数据库和日志卷必须位于支持冻结和解冻请求的文件系统上。表 18 列示可通过获取 IBM Tivoli Storage Manager (TSM) 的完整许可证来避免的许多功能限制。

表 18. 将随 DB2 交付的嵌入式 Tivoli Storage FlashCopy Manager 的受支持功能与完整版本的 IBM Tivoli Storage Manager (TSM) 产品的受支持功能进行比较

功能项目	嵌入式 Tivoli Storage FlashCopy Manager 版本支持	Tivoli Storage FlashCopy Manager 支持
本地快照备份版本	最多支持两个快照版本。	无产品限制。Tivoli Storage FlashCopy Manager 支持存储设备和可用资源允许的任意数目的版本。

© Copyright IBM Corp. 2001, 2013

表 18. 将随 DB2 交付的嵌入式 Tivoli Storage FlashCopy Manager 的受支持功能与完整版本的 IBM Tivoli Storage Manager (TSM) 产品的受支持功能进行比较 (续)

功能项目	嵌入式 Tivoli Storage FlashCopy Manager 版本支持	Tivoli Storage FlashCopy Manager 支持
与镜像集成的快照支持	不支持。	对于 AIX 逻辑卷管理器 (LVM) 镜像，支持来自源镜像集或目标镜像集的快照。
将快照映像集成备份到磁带	无集成支持。传统备份与快照备份互补，但未集成。	提供将快照映像备份到 TSM 的完全集成支持。一个备份命令就可以将快照备份与其他备份一起备份到 TSM。
备份到从生产服务器卸载的磁带	无备份到磁带的集成支持。	提供从另一台主机执行备份到 TSM 的完全集成支持。
Tivoli Storage FlashCopy Manager 存储库的集成备份	不支持。存储库不活动或关闭时可以执行外部备份。	自动备份至 TSM。

有关 Tivoli Storage FlashCopy Manager 的更多信息，请查看最新文档，网址为: Tivoli 文档中心。

启用 DB2 高级副本服务 (ACS)

要使用 DB2 高级副本服务 (ACS) 或执行快照备份操作，您必须安装、激活和配置 DB2 ACS。

开始之前

要执行快照备份和复原操作，需要适用于存储设备的 DB2 ACS API 驱动程序。有关集成驱动程序的受支持存储硬件的列表，请参阅 Tivoli 文档，网址为: 受支持存储子系统

过程

1. 安装 DB2 ACS。请参阅: 第 379 页的『安装 DB2 高级副本服务 (ACS)』。
2. 创建将与 DB2 ACS 配合使用的数据库管理器实例。

   当创建新的数据库管理器实例时，会在新的实例 `sqllib` 目录中创建名为 `acs` 的目录。因为每个数据库管理器实例均具有 `acs` 目录，所以您可以采用不同方式配置每个数据库管理器实例。
3. 对于将与 DB2 ACS 配合使用的每个数据库管理器实例，请执行下列步骤:
   a. 激活 DB2 ACS。请参阅: 第 379 页的『手动激活 DB2 高级副本服务 (ACS)』。
   b. 配置 DB2 ACS。请参阅: 第 380 页的『配置 DB2 高级副本服务 (ACS)』。

结果

启用 DB2 ACS 后，必须先配置存储解决方案，才能执行快照备份。有关配置和使用随 DB2 产品嵌入的 Tivoli Storage FlashCopyManager 版本的指示信息，请参阅最新文档，网址为: Tivoli 文档中心。

安装 DB2 高级副本服务 (ACS)

只有在典型安装和定制安装期间，才会安装 DB2 Advanced Copy Services (ACS) 所需的文件和库。

开始之前

安装 ACS 之前，必须已安装了下列库:

在 AIX 上:

- `ln -s /opt/freeware/lib/powerpc-ibm-aix5.3.0//libgcc_s.a /usr/lib/libgcc_s.a`

还应查看以下主题:

- “DB2 ACS 安装和配置最佳实践”。请参阅 第 377 页的『DB2 高级副本服务 (ACS) 最佳实践』
- “嵌入式版本的 Tivoli Storage FlashCopy Manager 的限制”。请参阅第 377 页的『嵌入式版本的 Tivoli Storage FlashCopy Manager 的限制』

在 Red Hat Enterprise Linux 上:

- `ln -s libssl.so.0.9.7xxx libssl.so.0.9.7`
- `ln -s libcrypto.so.0.9.7xxx libcrypto.so.0.9.7`
- `ln -s libssl.so.0.9.7xxx libssl.so`
- `ln -s libssl.so.0.9.7xxx libssl.so.0`

限制

DB2 ACS 支持 IBM 数据服务器所支持的硬件和操作系统的子集。有关 DB2 ACS 支持的硬件和操作系统的列表，请参阅 Tivoli 文档，网址为: 受支持的存储子系统

过程

1. 使用 **db2_install** 命令、典型安装、定制安装或定制响应文件安装（其使用 **ACS** 响应文件关键字）来安装 DB2 for Linux, UNIX, and Windows。

   要点: 不推荐使用 **db2_install** 命令，在将来的发行版中可能会除去该命令。请改为将 **db2setup** 命令与响应文件配合使用。

   注: 精简安装期间，不会再像之前发行版一样自动安装 DB2 ACS。如果已完成精简安装且稍后必须安装 DB2 ACS，请使用带 ACS 关键字的定制响应文件安装。
2. 在 TCP/IP 服务文件中为 DB2 ACS 代理程序添加端口。 例如:

   ```
   db2acs  5400/tcp # DB2 ACS service port
   ```

下一步做什么

安装 DB2 ACS 后，必须激活 DB2 ACS 并配置 DB2 ACS。请参阅『手动激活 DB2 高级副本服务 (ACS)』和第 380 页的『配置 DB2 高级副本服务 (ACS)』。

手动激活 DB2 高级副本服务 (ACS)

必须先对给定数据库管理器实例激活 DB2 高级副本服务 (ACS) 功能，才能使用 DB2 ACS 对该实例执行快照备份。

在创建数据库管理器实例期间和升级 IBM Data Server 时，数据库管理器会自动调用 **setup_db2.sh** 以激活 DB2 ACS 功能；但是，您也可通过手动运行脚本来激活 DB2 ACS。

开始之前

必须已安装 DB2 ACS（此操作通常已完成，除非使用了精简安装），并且必须已创建将与 DB2 ACS 配合使用的数据库管理器实例，才能激活 DB2 ACS。

关于此任务

在创建数据库管理器实例期间和您升级 IBM Data Server 时，数据库管理器会自动调用 **setup_db2.sh** 以激活 DB2 ACS 功能。在 AIX 系统上，它还会自动安装嵌入式版本的 Tivoli Storage FlashCopy Manager；在 Linux 系统上，您必须显式发出安装脚本以安装 Tivoli Storage FlashCopy Manager。

还可以手动激活 DB2 ACS。

过程

要手动激活 DB2 ACS，请以具有 root 用户权限的用户身份带适当参数运行 **setup_db2.sh** 脚本激活 DB2 ACS。
有关 **setup_db2.sh** 的更多信息，请参阅：第 381 页的『setup_db2.sh 脚本』。

结果

运行 **setup_db2.sh** 脚本的一个重要结果是验证 sqllib/acs 目录中 DB2 ACS 可执行文件的所有权和许可权。

下一步做什么

在激活 DB2 ACS 之后，必须先配置 DB2 ACS，然后才可以执行快照备份操作。

配置 DB2 高级副本服务 (ACS)

在可以使用 DB2 高级副本服务 (ACS) 来执行快照备份之前，必须配置 DB2 ACS。可以使用配置文件来配置 DB2 ACS。

开始之前

必须先执行下列任务才能配置 DB2 ACS:

1. 安装 DB2 ACS。请参阅第 379 页的『安装 DB2 高级副本服务 (ACS)』。
2. 创建将与 DB2 ACS 配合使用的数据库管理器实例。.
3. 激活 DB2 ACS。请参阅第 379 页的『手动激活 DB2 高级副本服务 (ACS)』。

过程

从 sqllib/acs 目录不带任何参数运行 **setup_db2.sh** 脚本。 这将引导您通过基于文本的交互式向导来配置 DB2 ACS。该向导将在机器上创建一个配置概要文件并修改 /etc/initab，以触发 DB2 ACS 守护程序的启动。

配置 DB2 高级副本服务 (ACS) 目录

当创建新的数据库管理器实例时，会在新的实例 sqllib 目录中创建名为 acs 的目录。DB2 高级副本服务 (ACS) 使用此 acs 目录来存储配置文件，例如目标卷控制文件和恢复对象的共享库。

目前对改变或配置此 acs 目录的方法有一些限制。

关于此任务

1. 在任何 DB2 ACS 或快照备份操作中，均不得涉及 acs 目录。
2. 在使用 IBM Tivoli Storage Manager (TSM) 的所有数据库分区和快照备份的备份系统上，acs 目录可以是 NFS 导出的和 NFS 共享的。

setup_db2.sh 脚本

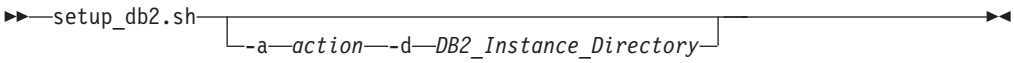
setup_db2.sh 脚本激活并配置 DB2 高级副本服务 (ACS)，并手动安装 Tivoli Storage FlashCopy Manager

位置

脚本 **setup_db2.sh** 位于 sqllib/acs 目录。

语法

此处是 **setup_db2.sh** 的语法:



其中 *action* 可以是下列其中一项:

disable
此选项用于停止 Tivoli Storage FlashCopy Manager 并从 /etc/inittab 中除去所有条目。要使用此选项，您必须具有 root 用户权限或者是实例所有者。

install
此选项用于安装 Tivoli Storage FlashCopy Manager。要使用此选项，您必须具有 root 用户权限。

start
此选项用于启动先前已安装并配置的 Tivoli Storage FlashCopy Manager 版本。要使用此选项，您必须具有 root 用户权限或者是实例所有者。

stop
此选项用于停止当前运行的 Tivoli Storage FlashCopy Manager 版本。要使用此选项，您必须具有 root 用户权限或者是实例所有者。

用法

在创建数据库管理器实例期间和您升级 IBM Data Server 时，数据库管理器会自动调用 **setup_db2.sh** 以激活 DB2 ACS 功能。在 AIX 系统上，它还会自动安装嵌入式版本的 Tivoli Storage FlashCopy Manager; 在 Linux 系统上，您必须显式发出安装脚本以安装 Tivoli Storage FlashCopy Manager。

可手动调用 **setup_db2.sh** 脚本以执行以下任务:

激活 DB2 ACS

可通过以具有 root 用户权限的用户身份带上文所述的参数运行 **setup_db2.sh** 脚本来激活 DB2 ACS。

配置 DB2 ACS

可通过不带任何参数运行 **setup_db2.sh** 脚本来配置 DB2 ACS。如果不带任何参数运行 **setup_db2.sh** 脚本，那么向导将引导您完成 DB2 ACS 配置。

安装 Tivoli Storage FlashCopy Manager

在 Linux 系统上，您需要手动运行 **setup_db2.sh** 脚本来安装 Tivoli Storage FlashCopy Manager。

运行 **setup_db2.sh** 脚本的一个重要结果是验证 sqllib/acs 目录中 DB2 ACS 可执行文件的所有权和许可权。

卸载 DB2 高级副本服务 (ACS)

当卸载 DB2 产品时，将自动卸载 DB2 ACS。从 V9.7 FP2 开始，您可使用 **db2_deinstall** 命令、“DB2 安装”向导或响应文件来仅卸载 DB2 ACS。

过程

要卸载 DB2 ACS，请使用下列其中一种方法:

- 输入带 **-F ACS** 参数的 **db2_deinstall** 命令，如下所示:

  ```
  db2_deinstall -F ACS
  ```

- 在“DB2 安装”向导中，单击**使用现有**，并从已安装的 DB2 副本中除去选择的 DB2 ACS 组件。
- 将 **ACS** 关键字添加到您的响应文件，如下所示:

  ```
  REMOVE_COMP = ACS
  ```

下一步做什么

检查日志文件中的消息。日志文件位于以下目录中:

- 对于 root 用户安装: /tmp/db2_deinstall.log.*processID*，其中 *processID* 表示 DB2 安装程序的进程标识
- 对于非 root 用户安装: /tmp/db2_deinstall_*userID*.log，其中 *userID* 表示拥有非 root 用户安装的用户标识

使用 **db2ls** 命令列出所有已安装的组件从而确保已除去了 DB2 ACS，如下所示:

```
db2ls -q -a -b base-install-path
```

其中 *base-install-path* 表示您正在查询的目录。

手动安装 Tivoli Storage FlashCopy Manager (Linux)

必须在 Linux 操作系统上手动安装随 IBM 产品嵌入的 Tivoli Storage FlashCopy Manager 的版本。它在 AIX 操作系统上是自动安装的。

开始之前

为了安装 Tivoli Storage FlashCopy Manager，您需要具有 root 用户权限。此外，必须已创建您要与 DB2 ACS 配合使用的 DB2 实例。

关于此任务

此任务指导您如何获取 Tivoli Storage FlashCopy Manager 的特定于 Linux 的程序包，以及如何发出安装脚本，此脚本将二进制文件复制到特定于实例的安装目录并为这些二进制文件设置适当访问权。

过程

1. 在标签为 `fcm_linux` 的单独 CD 上找到 Tivoli Storage FlashCopy Manager 的 Linux 程序包。或者，可从您获取产品映像的 Web 站点中下载 `fcm_linux` 程序包。
2. 将这些程序包解压缩至安装路径中的 ACS 目录（例如，`/opt/IBM/db2/V10.1/acs`）
3. 按如下所示调用 **setup_db2.sh** 安装脚本:

   ```
   setup_db2.sh -a install -d Instance_directory
   ```

结果

您现在应该能够执行快照备份。您应该知道的是，随 DB2 产品嵌入的 Tivoli Storage FlashCopy Manager 版本有一些限制（与随 IBM Tivoli Storage Manager 产品获取的完整产品版本相比）。

DB2 高级副本服务 (ACS) API

DB2 高级副本服务 (ACS) 应用程序编程接口 (API) 定义了一组功能，数据库管理器可使用它们来与存储硬件通信，从而执行快照备份操作。

要执行快照备份和复原操作，需要适用于存储设备的 DB2 ACS API 驱动程序。有关集成驱动程序的受支持存储硬件的列表，请参阅 Tivoli 文档，网址为: 受支持存储子系统

DB2 高级副本服务 API 函数

数据库管理器通过 DB2 ACS API 函数来向存储硬件传递 DB2 ACS 请求。

db2ACSQueryApiVersion - 返回 DB2 高级副本服务 (ACS) API 的当前版本

返回 DB2 高级副本服务 (ACS) API 的当前版本。

API 包含文件

```
db2ACSApi.h
```

API 和数据结构语法

```
db2ACS_Version db2ACSQueryApiVersion();
```

参数

无。

使用说明

可能的返回值:

- DB2ACS_API_VERSION1
- DB2ACS_API_VERSION_UNKNOWN

db2ACSInitialize - 初始化 DB2 高级副本服务 (ACS) 会话

初始化新的 DB2 高级副本服务 (ACS) 会话。此调用在数据库管理器的 DB2 ACS 库和存储硬件的 DB2 ACS API 驱动程序之间建立通信。

包含文件

```
db2ACSApi.h
```

语法和数据结构

```
/* ============================================================================
 * Session Initialization
 * ============================================================================ */
db2ACS_RC db2ACSInitialize(
               db2ACS_CB               * pControlBlock,
               db2ACS_ReturnCode       * pRC );
```

参数

pControlBlock

数据类型: db2ACS_CB *

db2ACS_CB 包含初始化和终止 DB2 ACS 会话所需的基本信息。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在调用 db2ACSInitialize() 之前，数据库管理器填充下列字段:

pControlBlock->session
pControlBlock->options

在返回之前，DB2 ACS API 驱动程序填充下列字段:

pControlBlock->handle
pControlBlock->vendorInfo

pRC 数据类型: db2ACS_ReturnCode *

db2ACS_ReturnCode 包含诊断信息，其中包括特定于存储硬件的消息文本和错误代码。用于 DB2 ACS API 函数调用的 db2ACS_ReturnCode 参数的内容将记录在数据库管理器诊断日志中。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在返回之前，DB2 ACS API 驱动程序填充 **pRC** 的字段。

返回码

表 19. 返回码

返回码	描述	注意
DB2ACS_RC_OK	操作已成功。	
DB2ACS_RC_INIT_FAILED	数据库管理器尝试初始化 DB2 ACS 会话，但初始化失败。	
DB2ACS_RC_INV_ACTION	数据库管理器请求来自 DB2 ACS API 驱动程序的无效操作。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_INV_DEV_HANDLE	数据库管理器传递无效的存储设备句柄。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_DEV_ERROR	存在与诸如磁带机之类的存储设备有关的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_IO_ERROR	DB2 ACS API 驱动程序遇到输入或输出操作所产生的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_COMM_ERROR	存在与诸如磁带机之类的存储设备的通信错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_NO_DEV_AVAIL	当前不存在诸如磁带机之类的存储设备可用。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。

如果 DB2 ACS API 驱动程序遇到错误，那么该驱动程序可能会中止 DB2 ACS 操作。DB2 ACS 会话不能用于除以下各项之外的任何操作:

- 如果先前对 db2ACSBeginQuery() 的调用成功了，那么数据库管理器可以调用 db2ACSEndQuery()
- 如果先前对 db2ACSBeginOperation() 的调用成功了，那么数据库管理器可以调用 db2ACSEndOperation()
- 如果先前对 db2ACSInitialize() 的调用成功了，那么数据库管理器可以调用 db2ACSTerminate()

有关 DB2 ACS API 返回码的更多信息，请参阅主题: 第 422 页的『DB2 高级副本服务 (ACS) API 返回码』。

使用说明

在数据库管理器可以进行任何 DB2 ACS API 调用（除了调用 db2ACSQueryAPIVersion() 之外）之前，数据库管理器必须调用 db2ACSInitialize()。一旦数据库管理器通过调用 db2ACSInitialize() 建立了 DB2 ACS 会话，该数据库管理器就可以执行 DB2 ACS 查询、读取、写入或删除操作的任何组合。通过调用 db2ACSTerminate()，数据库管理器可以终止 DB2 ACS 会话。

db2ACSTerminate - 终止 DB2 高级副本服务 (ACS) 会话

终止 DB2 高级副本服务 (ACS) 会话。

包含文件

```
db2ACSApi.h
```

语法和数据结构

```
/* ============================================================================
 * Session Termination
 * ============================================================================ */
db2ACS_RC db2ACSTerminate(
              db2ACS_CB               * pControlBlock,
              db2ACS_ReturnCode       * pRC );
```

参数

pControlBlock

数据类型: `db2ACS_CB *`

`db2ACS_CB` 包含初始化和终止 DB2 ACS 会话所需的基本信息。

在调用 db2ACSInitialize() 之前，数据库管理器为此参数分配内存。在调用 db2ACSTerminate() 之后，数据库管理器负责释放此内存。

在调用 db2ACSTerminate() 之前，数据库管理器填充下列字段:

pControlBlock->options

DB2 ACS API 驱动程序可能在 `pControlBlock->vendorInfo.vendorCB` 中使内存无效或释放内存。

pRC 数据类型: `db2ACS_ReturnCode *`

`db2ACS_ReturnCode` 包含诊断信息，其中包括特定于存储硬件的消息文本和错误代码。用于 DB2 ACS API 函数调用的 `db2ACS_ReturnCode` 参数的内容将记录在数据库管理器诊断日志中。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在返回之前，DB2 ACS API 驱动程序填充 **pRC** 的字段。

返回码

表 20. 返回码

返回码	描述	注意
`DB2ACS_RC_OK`	操作已成功。	释放为此会话分配的所有内存并将其终止。
`DB2ACS_INV_ACTION`	数据库管理器请求来自 DB2 ACS API 驱动程序的无效操作。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。

如果 DB2 ACS API 驱动程序遇到错误，那么该驱动程序可能会中止 DB2 ACS 操作。DB2 ACS 会话不能用于除以下各项之外的任何操作:

- 如果先前对 db2ACSBeginQuery() 的调用成功了，那么数据库管理器可以调用 db2ACSEndQuery()
- 如果先前对 db2ACSBeginOperation() 的调用成功了，那么数据库管理器可以调用 db2ACSEndOperation()

- 如果先前对 db2ACSInitialize() 的调用成功了，那么数据库管理器可以调用 db2ACSTerminate()

有关 DB2 ACS API 返回码的更多信息，请参阅主题：第 422 页的『DB2 高级副本服务 (ACS) API 返回码』。

使用说明

DB2 ACS API 驱动程序应该在 db2ACSTerminate() 中释放驱动程序分配给 DB2 ACS 会话的所有内存。

不论 db2ACSTerminate() 是否成功完成，在事先未调用 db2ACSInitialize() 的情况下，数据库管理器均无法再次对此 DB2 ACS 会话调用 DB2 ACS 函数。

db2ACSPrepare - 准备执行快照备份操作。

当执行快照备份时，数据库管理器会暂挂数据库。db2ACSPrepare() 执行所有步骤来准备进行快照备份操作，直至（但不包括）数据库管理器暂挂数据库这一刻。

包含文件

```
db2ACSApi.h
```

语法和数据结构

```
/* ============================================================================
 * Prepare
 * ============================================================================ */
db2ACS_RC db2ACSPrepare(
               db2ACS_GroupList        * pGroupList,
               db2ACS_CB               * pControlBlock,
               db2ACS_ReturnCode       * pRC );
```

参数

pGroupList

数据类型：db2ACS_GroupList *

db2ACS_GroupList 包含在快照备份操作中要包含的组的列表。

如果 **pGroupList** 为 NULL，那么快照备份操作中将包含所有组（路径）。

如果 **pGroupList** 不是 NULL:

- **pGroupList** 包含快照备份操作中要包括的组（路径）的列表。
- 数据库管理器负责分配和释放 **pGroupList** 的内存。
- 数据库管理器填充下列字段，然后才将 **pGroupList** 传递到 db2ACSPrepare():

  pGroupList->numGroupID
  pGroupList->id

pControlBlock

数据类型：db2ACS_CB *

db2ACS_CB 包含初始化和终止 DB2 ACS 会话所需的基本信息。

在调用 db2ACSPrepare() 之前，数据库管理器填充下列字段:

pControlBlock->handle
pControlBlock->vendorInfo
pControlBlock->options

pRC 数据类型: db2ACS_ReturnCode *

db2ACS_ReturnCode 包含诊断信息，其中包括特定于存储硬件的消息文本和错误代码。用于 DB2 ACS API 函数调用的 db2ACS_ReturnCode 参数的内容将记录在数据库管理器诊断日志中。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在返回之前，DB2 ACS API 驱动程序填充 **pRC** 的字段。

返回码

表 21. 返回码

返回码	描述	注意
DB2ACS_RC_OK	操作已成功。	
DB2ACS_RC_INV_ACTION	数据库管理器请求来自 DB2 ACS API 驱动程序的无效操作。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_INV_DEV_HANDLE	数据库管理器传递无效的存储设备句柄。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_DEV_ERROR	存在与诸如磁带机之类的存储设备有关的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_IO_ERROR	DB2 ACS API 驱动程序遇到输入或输出操作所产生的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。

如果 DB2 ACS API 驱动程序遇到错误，那么该驱动程序可能会中止 DB2 ACS 操作。DB2 ACS 会话不能用于除以下各项之外的任何操作:

- 如果先前对 db2ACSBeginQuery() 的调用成功了，那么数据库管理器可以调用 db2ACSEndQuery()
- 如果先前对 db2ACSBeginOperation() 的调用成功了，那么数据库管理器可以调用 db2ACSEndOperation()
- 如果先前对 db2ACSInitialize() 的调用成功了，那么数据库管理器可以调用 db2ACSTerminate()

有关 DB2 ACS API 返回码的更多信息，请参阅主题: 第 422 页的『DB2 高级副本服务 (ACS) API 返回码』。

使用说明

如果 db2ACSPrepare() 成功执行，那么数据库管理器将暂挂数据库，然后才调用 db2ACSSnapshot()。

db2ACSBeginOperation - 开始 DB2 高级副本服务 (ACS) 操作。

开始 DB2 高级副本服务 (ACS) 操作。

包含文件

db2ACSApi.h

语法和数据结构

```
/* ============================================================================
 * Operation Begin
 *
 * A valid ACS operation is specified by passing an ObjectType OR'd with one of
 * the following Operations, such as:
 *
 *    (DB2ACS_OP_CREATE | DB2ACS_OBJTYPE_SNAPSHOT)
 * ============================================================================ */
db2ACS_RC db2ACSBeginOperation(
               db2ACS_Operation           operation,
               db2ACS_CB                * pControlBlock,
               db2ACS_ReturnCode        * pRC );
```

参数

operation

数据类型: db2ACS_Operation。

operation 是位屏蔽，指示要开始的 DB2 ACS 操作和所涉及对象的类型。

操作类型:

DB2ACS_OP_CREATE
DB2ACS_OP_READ
DB2ACS_OP_DELETE

对象类型:

DB2ACS_OBJTYPE_BACKUP
DB2ACS_OBJTYPE_LOG
DB2ACS_OBJTYPE_LOADCOPY
DB2ACS_OBJTYPE_SNAPSHOT

例如: (DB2ACS_OP_CREATE | DB2ACS_OBJTYPE_SNAPSHOT) 或 (DB2ACS_OP_DELETE | DB2ACS_OBJTYPE_LOADCOPY)。

数据库管理器将 **operation** 传递到 db2ACSBeginOperation() 函数调用。

pControlBlock

数据类型: db2ACS_CB *

db2ACS_CB 包含初始化和终止 DB2 ACS 会话所需的基本信息。

在调用 db2ACSBeginOperation() 之前，数据库管理器填充下列字段:

pControlBlock->handle
pControlBlock->vendorInfo
pControlBlock->options

如果 **operation** 是 DB2ACS_OP_CREATE 或 DB2ACS_OP_READ，那么数据库管理器也会填充下列字段:

pControlBlock->operation

pControlBlock->operation 中包含的信息仅在特定的 DB2 ACS 操作的上下文中才有效。pControlBlock->operation 将在 db2ACSBeginOperation() 期间进行设置，并且在 db2ACSEndOperation() 返回之前将保持不变。数据库管理器和 DB2 ACS API 驱动程序均不应该引用 DB2 ACS 操作范围外的 pControlBlock->operation。

pRC 数据类型: db2ACS_ReturnCode *

db2ACS_ReturnCode 包含诊断信息，其中包括特定于存储硬件的消息文本和错误代码。用于 DB2 ACS API 函数调用的 db2ACS_ReturnCode 参数的内容将记录在数据库管理器诊断日志中。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在返回之前，DB2 ACS API 驱动程序填充 **pRC** 的字段。

返回码

表 22. 返回码

返回码	描述	注意
DB2ACS_RC_OK	操作已成功。	
DB2ACS_RC_INV_OPTIONS	数据库管理器指定了无效选项。	
DB2ACS_RC_INV_ACTION	数据库管理器请求来自 DB2 ACS API 驱动程序的无效操作。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。

如果 DB2 ACS API 驱动程序遇到错误，那么该驱动程序可能会中止 DB2 ACS 操作。DB2 ACS 会话不能用于除以下各项之外的任何操作:

- 如果先前对 db2ACSBeginQuery() 的调用成功了，那么数据库管理器可以调用 db2ACSEndQuery()
- 如果先前对 db2ACSBeginOperation() 的调用成功了，那么数据库管理器可以调用 db2ACSEndOperation()
- 如果先前对 db2ACSInitialize() 的调用成功了，那么数据库管理器可以调用 db2ACSTerminate()

有关 DB2 ACS API 返回码的更多信息，请参阅主题: 第 422 页的『DB2 高级副本服务 (ACS) API 返回码』。

使用说明

无。

db2ACSEndOperation - 结束 DB2 高级副本服务 (ACS) 操作。

结束 DB2 高级副本服务 (ACS) 操作。

包含文件

db2ACSApi.h

语法和数据结构

```
/* ============================================================================
 * Operation End
 * ============================================================================ */
db2ACS_RC db2ACSEndOperation(
               db2ACS_EndAction          endAction,
               db2ACS_CB               * pControlBlock,
               db2ACS_ReturnCode       * pRC );
```

参数

endAction

数据类型: db2ACS_EndAction。

endAction 是位屏蔽，指示 DB2 ACS API 驱动程序应如何结束 DB2 ACS 操作。

值:

DB2ACS_END_COMMIT
DB2ACS_END_ABORT

数据库管理器将 **endAction** 传递到 db2ACSEndOperation() 函数调用。

pControlBlock

数据类型: db2ACS_CB

db2ACS_CB 包含初始化和终止 DB2 ACS 会话所需的基本信息。

在调用 db2ACSEndOperation() 之前，数据库管理器填充下列字段:

pControlBlock->handle
pControlBlock->vendorInfo
pControlBlock->options

pRC

数据类型: db2ACS_ReturnCode *

db2ACS_ReturnCode 包含诊断信息，其中包括特定于存储硬件的消息文本和错误代码。用于 DB2 ACS API 函数调用的 db2ACS_ReturnCode 参数的内容将记录在数据库管理器诊断日志中。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在返回之前，DB2 ACS API 驱动程序填充 **pRC** 的字段。

返回码

表 23. 返回码

返回码	描述	注意
DB2ACS_RC_OK	操作已成功。	
DB2ACS_RC_INV_ACTION	数据库管理器请求来自 DB2 ACS API 驱动程序的无效操作。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_COMMIT_FAILED	DB2 ACS API 驱动程序无法落实事务。	
DB2ACS_RC_ABORT_FAILED	数据库管理器尝试中止 DB2 ACS 操作，但尝试中止失败。	

如果 DB2 ACS API 驱动程序遇到错误，那么该驱动程序可能会中止 DB2 ACS 操作。DB2 ACS 会话不能用于除以下各项之外的任何操作:

- 如果先前对 db2ACSBeginQuery() 的调用成功了，那么数据库管理器可以调用 db2ACSEndQuery()
- 如果先前对 db2ACSBeginOperation() 的调用成功了，那么数据库管理器可以调用 db2ACSEndOperation()
- 如果先前对 db2ACSInitialize() 的调用成功了，那么数据库管理器可以调用 db2ACSTerminate()

有关 DB2 ACS API 返回码的更多信息，请参阅主题: 第 422 页的『DB2 高级副本服务 (ACS) API 返回码』。

使用说明

如果数据库管理器将 DB2ACS_END_ABORT 作为 **endAction** 参数来传递，那么结果应该是删除快照备份对象。

db2ACSBeginQuery - 开始有关快照备份对象的查询

开始有关可用于复原操作的快照备份对象的 DB2 高级副本服务 (ACS) 查询操作。

包含文件

db2ACSApi.h

语法和数据结构

```
db2ACS_RC db2ACSBeginQuery(
              db2ACS_QueryInput       * pQueryInput,
              db2ACS_CB               * pControlBlock,
              db2ACS_ReturnCode       * pRC );
```

参数

pQueryInput

数据类型: db2ACS_QueryInput *

db2ACS_QueryInput 具有与 db2ACS_ObjectInfo 相同的字段。db2ACS_ObjectInfo 包含有关使用 DB2 高级副本服务 (ACS) API 创建的对象的信息。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在调用 db2ACSBeginQuery() 之前，数据库管理器填充 **pQueryInput** 的字段。

DB2 ACS API 驱动程序必须支持在查询中使用下列通配符:

- 字符串字段中的 DB2ACS_WILDCARD
- 用于数据库分区字段的 DB2ACS_ANY_PARTITIONNUM
- 用于 32 位无符号的整数（Uint32）字段的 DB2ACS_ANY_UINT32

pControlBlock

数据类型: db2ACS_CB *

db2ACS_CB 包含初始化和终止 DB2 ACS 会话所需的基本信息。

在调用 db2ACSBeginQuery() 之前，数据库管理器填充下列字段:

pControlBlock->handle
pControlBlock->vendorInfo
pControlBlock->options

pRC 数据类型：db2ACS_ReturnCode *

db2ACS_ReturnCode 包含诊断信息，其中包括特定于存储硬件的消息文本和错误代码。用于 DB2 ACS API 函数调用的 db2ACS_ReturnCode 参数的内容将记录在数据库管理器诊断日志中。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在返回之前，DB2 ACS API 驱动程序填充 **pRC** 的字段。

返回码

表 24. 返回码

返回码	描述	注意
DB2ACS_RC_OK	操作已成功。	
DB2ACS_RC_INV_ACTION	数据库管理器请求来自 DB2 ACS API 驱动程序的无效操作。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_INV_DEV_HANDLE	数据库管理器传递无效的存储设备句柄。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_DEV_ERROR	存在与诸如磁带机之类的存储设备有关的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_IO_ERROR	DB2 ACS API 驱动程序遇到输入或输出操作所产生的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。

如果 DB2 ACS API 驱动程序遇到错误，那么该驱动程序可能会中止 DB2 ACS 操作。DB2 ACS 会话不能用于除以下各项之外的任何操作:

- 如果先前对 db2ACSBeginQuery() 的调用成功了，那么数据库管理器可以调用 db2ACSEndQuery()
- 如果先前对 db2ACSBeginOperation() 的调用成功了，那么数据库管理器可以调用 db2ACSEndOperation()
- 如果先前对 db2ACSInitialize() 的调用成功了，那么数据库管理器可以调用 db2ACSTerminate()

有关 DB2 ACS API 返回码的更多信息，请参阅主题：第 422 页的『DB2 高级副本服务 (ACS) API 返回码』。

使用说明

db2ACSBeginQuery() 不会返回任何查询数据。

db2ACSGetNextObject - 列示下一个可用于复原的快照备份对象

返回可用于复原操作的快照备份对象列表中的下一项。

包含文件

```
db2ACSApi.h
```

语法和数据结构

```
db2ACS_RC db2ACSGetNextObject(
              db2ACS_QueryOutput      * pQueryOutput,
              db2ACS_CB               * pControlBlock,
              db2ACS_ReturnCode       * pRC );
```

参数

pQueryOutput

数据类型: db2ACS_QueryOutput *

db2ACS_QueryOutput 包含有关快照备份对象的查询结果信息。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在返回之前，DB2 ACS API 驱动程序填充 **pQueryOutput** 的字段。

pControlBlock

数据类型: db2ACS_CB *

db2ACS_CB 包含初始化和终止 DB2 ACS 会话所需的基本信息。

在调用 db2ACSGetNextObject() 之前，数据库管理器填充下列字段:

pControlBlock->handle
pControlBlock->vendorInfo
pControlBlock->options

pRC

数据类型: db2ACS_ReturnCode *

db2ACS_ReturnCode 包含诊断信息，其中包括特定于存储硬件的消息文本和错误代码。用于 DB2 ACS API 函数调用的 db2ACS_ReturnCode 参数的内容将记录在数据库管理器诊断日志中。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在返回之前，DB2 ACS API 驱动程序填充 **pRC** 的字段。

返回码

表 25. 返回码

返回码	描述	注意
DB2ACS_RC_OK	操作已成功。	
DB2ACS_RC_INV_ACTION	数据库管理器请求来自 DB2 ACS API 驱动程序的无效操作。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_INV_DEV_HANDLE	数据库管理器传递无效的存储设备句柄。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_DEV_ERROR	存在与诸如磁带机之类的存储设备有关的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。

表 25. 返回码 (续)

返回码	描述	注意
DB2ACS_RC_IO_ERROR	DB2 ACS API 驱动程序遇到输入或输出操作所产生的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_OBJ_NOT_FOUND	DB2 ACS API 驱动程序找不到数据库管理器指定的快照备份对象。	函数调用并未失败，但不存在与传递到 db2ACSBeginQuery() 的条件相符的快照备份对象。
DB2ACS_RC_END_OF_DATA	DB2 ACS API 驱动程序找不到更多的快照备份对象。	函数调用并未失败，但不存在更多与传递到 db2ACSBeginQuery() 的条件相符的快照备份对象。
DB2ACS_RC_MORE_DATA	有更多的数据要从存储位置传送到数据库管理器。	有关与传递到 db2ACSBeginQuery() 的条件相符的快照备份对象的信息返回后，存在更多与传递到 db2ACSBeginQuery() 的条件相符的快照备份对象。

如果 DB2 ACS API 驱动程序遇到错误，那么该驱动程序可能会中止 DB2 ACS 操作。DB2 ACS 会话不能用于除以下各项之外的任何操作:

- 如果先前对 db2ACSBeginQuery() 的调用成功了，那么数据库管理器可以调用 db2ACSEndQuery()
- 如果先前对 db2ACSBeginOperation() 的调用成功了，那么数据库管理器可以调用 db2ACSEndOperation()
- 如果先前对 db2ACSInitialize() 的调用成功了，那么数据库管理器可以调用 db2ACSTerminate()

有关 DB2 ACS API 返回码的更多信息，请参阅主题: 第 422 页的『DB2 高级副本服务 (ACS) API 返回码』。

使用说明

数据库管理器必须先调用 db2ACSBeginQuery()，然后才调用 db2ACSGetNextObject()。数据库管理器在传递到 db2ACSBeginQuery() 的 db2ACS_QueryInput 参数中指定搜索条件。

db2ACSGetNextObject() 返回有关与传递到 db2ACSBeginQuery() 的搜索条件相符的一个快照备份对象的信息。如果 db2ACSGetNextObject() 返回 DB2ACS_RC_MORE_DATA，那么数据库管理器可以再次调用 db2ACSGetNextObject()，以接收有关与搜索条件相符的另一快照备份对象的信息。如果 db2ACSGetNextObject() 返回 DB2ACS_RC_END_OF_DATA，那么不存在更多与搜索条件相符的快照备份对象。

db2ACSEndQuery - 结束有关快照备份对象的查询

数据库管理器使用 DB2 高级副本服务 (ACS) API 函数 db2ACSBeginQuery() 和 db2ACSGetNextObject() 来查询有关可用于复原操作的快照备份对象的信息。db2ACSEndQuery() 终止该 DB2 ACS 查询会话。

包含文件

db2ACSApi.h

语法和数据结构

```
db2ACS_RC db2ACSEndQuery(
              db2ACS_CB               * pControlBlock,
              db2ACS_ReturnCode       * pRC );
```

参数

pControlBlock

数据类型: db2ACS_CB *

db2ACS_CB 包含初始化和终止 DB2 ACS 会话所需的基本信息。

在调用 db2ACSEndQuery() 之前，数据库管理器填充下列字段:

pControlBlock->handle
pControlBlock->vendorInfo
pControlBlock->options

pRC 数据类型: db2ACS_ReturnCode *

db2ACS_ReturnCode 包含诊断信息，其中包括特定于存储硬件的消息文本和错误代码。用于 DB2 ACS API 函数调用的 db2ACS_ReturnCode 参数的内容将记录在数据库管理器诊断日志中。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在返回之前，DB2 ACS API 驱动程序填充 **pRC** 的字段。

返回码

表 26. 返回码

返回码	描述	注意
DB2ACS_RC_OK	操作已成功。	
DB2ACS_RC_INV_ACTION	数据库管理器请求来自 DB2 ACS API 驱动程序的无效操作。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_INV_DEV_HANDLE	数据库管理器传递无效的存储设备句柄。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_DEV_ERROR	存在与诸如磁带机之类的存储设备有关的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_IO_ERROR	DB2 ACS API 驱动程序遇到输入或输出操作所产生的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。

如果 DB2 ACS API 驱动程序遇到错误，那么该驱动程序可能会中止 DB2 ACS 操作。DB2 ACS 会话不能用于除以下各项之外的任何操作:

- 如果先前对 db2ACSBeginQuery() 的调用成功了，那么数据库管理器可以调用 db2ACSEndQuery()
- 如果先前对 db2ACSBeginOperation() 的调用成功了，那么数据库管理器可以调用 db2ACSEndOperation()

- 如果先前对 db2ACSInitialize() 的调用成功了，那么数据库管理器可以调用 db2ACSTerminate()

有关 DB2 ACS API 返回码的更多信息，请参阅主题: 第 422 页的『DB2 高级副本服务 (ACS) API 返回码』。

使用说明

在没有再次事先调用 db2ACSBeginQuery() 的情况下，数据库管理器无法再次对此 DB2 ACS 会话调用 db2ACSGetNextObject()。

db2ACSSnapshot - 执行 DB2 高级副本服务 (ACS) 操作

执行 DB2 高级副本服务 (ACS) 操作。

包含文件

```
db2ACSApi.h
```

语法和数据结构

```
typedef union db2ACS_ReadList
{
   db2ACS_GroupList           group;
} db2ACS_ReadList;


db2ACS_RC db2ACSSnapshot(
               db2ACS_Action             action,
               db2ACS_ObjectID           objectID,
               db2ACS_ReadList         * pReadList,
               db2ACS_CB               * pControlBlock,
               db2ACS_ReturnCode       * pRC );
```

参数

action 数据类型: db2ACS_Action

要执行的 DB2 ACS 操作的类型。值:

DB2ACS_ACTION_WRITE
DB2ACS_ACTION_READ_BY_OBJECT
DB2ACS_ACTION_READ_BY_GROUP

数据库管理器将 **action** 传递到 db2ACSSnapshot()。

objectID

数据类型: db2ACS_ObjectID

db2ACS_ObjectID 是由查询返回到存储库的每个存储对象的唯一标识。保证 db2ACS_ObjectID 是唯一的，且仅在单一 DB2 ACS 会话时间框架中持久存在。

如果数据库管理器在调用 db2ACSBeginOperation() 时将 DB2ACS_OP_READ 或 DB2ACS_OP_DELETE 指定为 **operation**，那么数据库管理器会将 **objectID** 的值传递到 db2ACSSnapshot()。

pReadList

数据类型: db2ACS_ReadList *

db2ACS_ReadList 包含组列表。

仅在 **action** 为 DB2ACS_ACTION_READ_BY_GROUP 时，才使用 **pReadList**。

如果 **action** 是 DB2ACS_ACTION_READ_BY_GROUP，那么数据库管理器负责分配内存以及填充 **pReadLIst** 的字段，然后调用 db2ACSSnapshot() 和释放 **pReadList** 的内存。

pControlBlock

数据类型：db2ACS_CB *

db2ACS_CB 包含初始化和终止 DB2 ACS 会话所需的基本信息。

在调用 db2ACSSnapshot() 之前，数据库管理器填充下列字段:

pControlBlock->handle
pControlBlock->vendorInfo
pControlBlock->options

pRC 数据类型：db2ACS_ReturnCode *

db2ACS_ReturnCode 包含诊断信息，其中包括特定于存储硬件的消息文本和错误代码。用于 DB2 ACS API 函数调用的 db2ACS_ReturnCode 参数的内容将记录在数据库管理器诊断日志中。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在返回之前，DB2 ACS API 驱动程序填充 **pRC** 的字段。

返回码

表 27. 返回码

返回码	描述	注意
DB2ACS_RC_OK	操作已成功。	
DB2ACS_RC_INV_ACTION	数据库管理器请求来自 DB2 ACS API 驱动程序的无效操作。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_INV_DEV_HANDLE	数据库管理器传递无效的存储设备句柄。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_DEV_ERROR	存在与诸如磁带机之类的存储设备有关的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_IO_ERROR	DB2 ACS API 驱动程序遇到输入或输出操作所产生的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。

如果 DB2 ACS API 驱动程序遇到错误，那么该驱动程序可能会中止 DB2 ACS 操作。DB2 ACS 会话不能用于除以下各项之外的任何操作:

- 如果先前对 db2ACSBeginQuery() 的调用成功了，那么数据库管理器可以调用 db2ACSEndQuery()
- 如果先前对 db2ACSBeginOperation() 的调用成功了，那么数据库管理器可以调用 db2ACSEndOperation()

- 如果先前对 db2ACSInitialize() 的调用成功了，那么数据库管理器可以调用 db2ACSTerminate()

有关 DB2 ACS API 返回码的更多信息，请参阅主题: 第 422 页的『DB2 高级副本服务 (ACS) API 返回码』。

使用说明

数据库管理器调用 db2ACSBeginOperation()，然后调用 db2ACSPartition()、db2ACSPrepare() 和 db2ACSSnapshot()。数据库管理器在调用 db2ACSBeginOperation() 时，在 **operation** 参数中指定 DB2 ACS API 驱动程序应执行的 DB2 ACS 操作的类型。

db2ACSPartition - 将数据库分区的目标数据分组在一起

将组标识与数据库管理器列示的每个路径相关联，使这些路径从属于数据库分区。

包含文件

db2ACSApi.h

语法和数据结构

```
/* ============================================================================
 * Partition
 * ============================================================================ */
db2ACS_RC db2ACSPartition(
               db2ACS_PathList         * pPathList,
               db2ACS_CreateObjectInfo * pCreateObjInfo,
               db2ACS_CB               * PControlBlock,
               db2ACS_ReturnCode       * pRC );
```

参数

pPathList

数据类型: db2ACS_PathList

db2ACS_PathList 包含数据库路径列表，包括有关特定于 DB2 ACS 操作的每个这些路径的一些额外信息。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

db2ACS_PathList 结构的 **entry** 字段是一组类型为 db2ACS_PathEntry 的元素。db2ACS_PathEntry 包含有关数据库路径的信息。

在调用 db2ACSPartition 之前，数据库管理器填充 **pPathList** 中每个 db2ACS_PathEntry 条目的下列字段:

- **path**
- **type**
- **toBeExcluded**

DB2 ACS API 驱动程序为属于该数据库分区的数据库管理器标识的每个路径提供了组标识。在返回之前，DB2 ACS API 驱动程序填充 **pPathList** 中每个 db2ACS_PathEntry 的 **groupID** 字段。

pCreateObjInfo

数据类型: db2ACS_CreateObjectInfo

db2ACS_CreateObjectInfo 包含有关创建 DB2 ACS 备份对象的信息。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在调用 db2ACSPartition 之前，数据库管理器填充 **pCreateObjInfo** 的字段。

pControlBlock

数据类型：db2ACS_CB *

db2ACS_CB 包含初始化和终止 DB2 ACS 会话所需的基本信息。

在调用 db2ACSPartition() 之前，数据库管理器填充下列字段:

pControlBlock->handle
pControlBlock->vendorInfo
pControlBlock->options

pRC 数据类型：db2ACS_ReturnCode *

db2ACS_ReturnCode 包含诊断信息，其中包括特定于存储硬件的消息文本和错误代码。用于 DB2 ACS API 函数调用的 db2ACS_ReturnCode 参数的内容将记录在数据库管理器诊断日志中。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在返回之前，DB2 ACS API 驱动程序填充 **pRC** 的字段。

返回码

表 28. 返回码

返回码	描述	注意
DB2ACS_RC_OK	操作已成功。	
DB2ACS_RC_INIT_FAILED	数据库管理器尝试初始化 DB2 ACS 会话，但初始化失败。	
DB2ACS_RC_INV_ACTION	数据库管理器请求来自 DB2 ACS API 驱动程序的无效操作。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_INV_DEV_HANDLE	数据库管理器传递无效的存储设备句柄。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_DEV_ERROR	存在与诸如磁带机之类的存储设备有关的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_IO_ERROR	DB2 ACS API 驱动程序遇到输入或输出操作所产生的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_OBJ_OUT_OF_SCOPE	数据库管理器尝试对不被 DB2 ACS API 驱动程序管理的恢复对象执行 DB2 ACS 操作。	

如果 DB2 ACS API 驱动程序遇到错误，那么该驱动程序可能会中止 DB2 ACS 操作。DB2 ACS 会话不能用于除以下各项之外的任何操作:

- 如果先前对 db2ACSBeginQuery() 的调用成功了，那么数据库管理器可以调用 db2ACSEndQuery()
- 如果先前对 db2ACSBeginOperation() 的调用成功了，那么数据库管理器可以调用 db2ACSEndOperation()
- 如果先前对 db2ACSInitialize() 的调用成功了，那么数据库管理器可以调用 db2ACSTerminate()

有关 DB2 ACS API 返回码的更多信息，请参阅主题：第 422 页的『DB2 高级副本服务 (ACS) API 返回码』。

使用说明

DB2 高级副本服务自动处理单一数据库分区上的数据。即，一个数据库分区的数据将一起备份或复原，并且独立于其他数据库分区 - 即使当该操作是涉及多个数据库分区的操作的一部分时也是如此。db2ACSPartition 将单一数据库分区的数据库路径信息分组在一起。

数据库管理器调用 db2ACSPartition，然后调用 db2ACSSnapshot。数据库管理器通过 **pPathList** 参数来列示与此数据库分区相关联的所有路径。通过在 **pReadList** 参数中指定传递到 db2ACSSnapshot 的路径子集，数据库管理器可以对这些列示在 **pPathList** 中的路径子集执行 DB2 ACS 操作。

db2ACSVerify - 验证 DB2 高级副本服务 (ACS) 操作是否已成功完成

验证 DB2 高级副本服务 (ACS) 操作是否成功

包含文件

```
db2ACSApi.h
```

语法和数据结构

```
/* ============================================================================
 * Verify
 * ============================================================================ */
db2ACS_RC db2ACSVerify(
               db2ACS_PostObjectInfo   * pPostObjInfo,
               db2ACS_CB               * pControlBlock,
               db2ACS_ReturnCode       * pRC );
```

参数

pPostObjInfo

数据类型：db2ACS_PostObjectInfo

db2ACS_DB2ID 是一组数据，在快照备份对象创建时无法知道，但必须在对象存储库中对其进行维护。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在调用 db2ACSVerify 之前，数据库管理器填充 **pPostObjInfo** 的字段。在 DB2 ACS 操作之后，**pPostObjInfo** 包含相关的信息。例如，在成功的快照备份之后，**pPostObjInfo** 可能包含第一个活动日志文件。如果在 DB2 ACS 操作后没有相关数据，那么数据库管理器会将 **pPostObjInfo** 设置为 NULL。

pControlBlock

数据类型: db2ACS_CB *

db2ACS_CB 包含初始化和终止 DB2 ACS 会话所需的基本信息。

在调用 db2ACSVerify() 之前，数据库管理器填充下列字段:

pControlBlock->handle
pControlBlock->vendorInfo
pControlBlock->options

pRC 数据类型: db2ACS_ReturnCode *

db2ACS_ReturnCode 包含诊断信息，其中包括特定于存储硬件的消息文本和错误代码。用于 DB2 ACS API 函数调用的 db2ACS_ReturnCode 参数的内容将记录在数据库管理器诊断日志中。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在返回之前，DB2 ACS API 驱动程序填充 **pRC** 的字段。

返回码

表 29. 返回码

返回码	描述	注意
DB2ACS_RC_OK	操作已成功。	
DB2ACS_RC_INV_ACTION	数据库管理器请求来自 DB2 ACS API 驱动程序的无效操作。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_INV_DEV_HANDLE	数据库管理器传递无效的存储设备句柄。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_DEV_ERROR	存在与诸如磁带机之类的存储设备有关的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_IO_ERROR	DB2 ACS API 驱动程序遇到输入或输出操作所产生的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。

如果 DB2 ACS API 驱动程序遇到错误，那么该驱动程序可能会中止 DB2 ACS 操作。DB2 ACS 会话不能用于除以下各项之外的任何操作:

- 如果先前对 db2ACSBeginQuery() 的调用成功了，那么数据库管理器可以调用 db2ACSEndQuery()
- 如果先前对 db2ACSBeginOperation() 的调用成功了，那么数据库管理器可以调用 db2ACSEndOperation()
- 如果先前对 db2ACSInitialize() 的调用成功了，那么数据库管理器可以调用 db2ACSTerminate()

有关 DB2 ACS API 返回码的更多信息，请参阅主题: 第 422 页的『DB2 高级副本服务 (ACS) API 返回码』。

使用说明

如果 db2ACSVerify 返回“快照备份操作成功”，这表示快照备份生成的恢复对象可用于复原操作。

db2ACSDelete - 删除使用 DB2 高级副本服务 (ACS) 创建的恢复对象

删除使用 DB2 高级副本服务 (ACS) 创建的恢复对象

包含文件

```
db2ACSApi.h
```

语法和数据结构

```
/* ============================================================================
 * Delete
 * ============================================================================ */
db2ACS_RC db2ACSDelete(
               db2ACS_ObjectID           objectID,
               db2ACS_CB               * pControlBlock,
               db2ACS_ReturnCode       * pRC );
```

参数

objectID

数据类型：db2ACS_ObjectID

db2ACS_ObjectID 是由查询返回到存储库的每个存储对象的唯一标识。保证 db2ACS_ObjectID 是唯一的，且仅在单一 DB2 ACS 会话时间框架中持久存在。

数据库管理器可以使用 db2ACSQuery() 来获取有效的 **objectID** 以传递到 db2ACSDelete()。

pControlBlock

数据类型：db2ACS_CB *

db2ACS_CB 包含初始化和终止 DB2 ACS 会话所需的基本信息。

在调用 db2ACSDelete() 之前，数据库管理器填充下列字段:

pControlBlock->handle
pControlBlock->vendorInfo
pControlBlock->options

pRC 数据类型：db2ACS_ReturnCode *

db2ACS_ReturnCode 包含诊断信息，其中包括特定于存储硬件的消息文本和错误代码。用于 DB2 ACS API 函数调用的 db2ACS_ReturnCode 参数的内容将记录在数据库管理器诊断日志中。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在返回之前，DB2 ACS API 驱动程序填充 **pRC** 的字段。

返回码

表 30. 返回码

返回码	描述	注意
DB2ACS_RC_OK	操作已成功。	删除了指定的对象。无法对该对象执行其他 DB2 ACS 操作。
DB2ACS_RC_DELETE_FAILED	DB2 ACS API 驱动程序无法成功地删除数据库管理器指定的快照备份对象。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_INV_DEV_HANDLE	数据库管理器传递无效的存储设备句柄。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_DEV_ERROR	存在与诸如磁带机之类的存储设备有关的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_IO_ERROR	DB2 ACS API 驱动程序遇到输入或输出操作所产生的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_OBJ_NOT_FOUND	DB2 ACS API 驱动程序找不到数据库管理器指定的快照备份对象。	

如果 DB2 ACS API 驱动程序遇到错误，那么该驱动程序可能会中止 DB2 ACS 操作。DB2 ACS 会话不能用于除以下各项之外的任何操作:

- 如果先前对 db2ACSBeginQuery() 的调用成功了，那么数据库管理器可以调用 db2ACSEndQuery()
- 如果先前对 db2ACSBeginOperation() 的调用成功了，那么数据库管理器可以调用 db2ACSEndOperation()
- 如果先前对 db2ACSInitialize() 的调用成功了，那么数据库管理器可以调用 db2ACSTerminate()

有关 DB2 ACS API 返回码的更多信息，请参阅主题: 第 422 页的『DB2 高级副本服务 (ACS) API 返回码』。

使用说明

当数据库管理器调用 db2ACSDelete 时，DB2 ACS API 驱动程序删除由 **objectID** 标识的恢复对象。

当用户调用带 DELETE 参数的 **db2acsutil** 时，数据库管理器将调用 db2ACSDelete。

db2ACSStoreMetaData - 存储使用 DB2 高级副本服务 (ACS) 生成的恢复对象的元数据

存储有关使用 DB2 高级副本服务 (ACS) 创建的恢复对象的元数据

包含文件

db2ACSApi.h

语法和数据结构

```
db2ACS_RC db2ACSStoreMetaData(
              db2ACS_MetaData         * pMetaData,
              db2ACS_CB               * pControlBlock,
              db2ACS_ReturnCode       * pRC );
```

参数

pMetaData

数据类型：db2ACS_MetaData

db2ACS_MetaData 存储快照备份元数据。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

存储在 **pMetaData** 的 **data** 字段中的元数据是数据库管理器固有的，可能会随着时间的推移而发生变化，因此 DB2 ACS API 驱动程序只需将此数据视为二进制流。

pControlBlock

数据类型：db2ACS_CB *

db2ACS_CB 包含初始化和终止 DB2 ACS 会话所需的基本信息。

在调用 db2ACSStoreMetaData() 之前，数据库管理器填充下列字段:

pControlBlock->handle
pControlBlock->vendorInfo
pControlBlock->options

pRC

数据类型：db2ACS_ReturnCode *

db2ACS_ReturnCode 包含诊断信息，其中包括特定于存储硬件的消息文本和错误代码。用于 DB2 ACS API 函数调用的 db2ACS_ReturnCode 参数的内容将记录在数据库管理器诊断日志中。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在返回之前，DB2 ACS API 驱动程序填充 **pRC** 的字段。

返回码

表 31. 返回码

返回码	描述	注意
DB2ACS_RC_OK	操作已成功。	
DB2ACS_RC_INV_ACTION	数据库管理器请求来自 DB2 ACS API 驱动程序的无效操作。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_INV_DEV_HANDLE	数据库管理器传递无效的存储设备句柄。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_DEV_ERROR	存在与诸如磁带机之类的存储设备有关的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。

表 31. 返回码 (续)

返回码	描述	注意
DB2ACS_RC_IO_ERROR	DB2 ACS API 驱动程序遇到输入或输出操作所产生的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。

如果 DB2 ACS API 驱动程序遇到错误，那么该驱动程序可能会中止 DB2 ACS 操作。DB2 ACS 会话不能用于除以下各项之外的任何操作:

- 如果先前对 db2ACSBeginQuery() 的调用成功了，那么数据库管理器可以调用 db2ACSEndQuery()
- 如果先前对 db2ACSBeginOperation() 的调用成功了，那么数据库管理器可以调用 db2ACSEndOperation()
- 如果先前对 db2ACSInitialize() 的调用成功了，那么数据库管理器可以调用 db2ACSTerminate()

有关 DB2 ACS API 返回码的更多信息，请参阅主题: 第 422 页的『DB2 高级副本服务 (ACS) API 返回码』。

使用说明

快照备份操作由若干 DB2 ACS API 函数调用组成，例如: db2ACSInitialize、db2ACSBeginOperation、db2ACSPrepare 和 db2ACSSnapshot。db2ACSStoreMetaData 也是整体操作的一部分。为了使快照备份操作成功，包括 db2ACSStoreMetaData 在内的所有这些 API 调用均必须成功。如果 db2ACSStoreMetaData 失败，那么由 DB2 ACS 备份操作生成的恢复对象不可使用。

db2ACSRetrieveMetaData - 检索有关使用 DB2 高级副本服务 (ACS) 生成的恢复对象的元数据

检索有关使用 DB2 高级副本服务 (ACS) 创建的恢复对象的元数据

包含文件

```
db2ACSApi.h
```

语法和数据结构

```
db2ACS_RC db2ACSRetrieveMetaData(
               db2ACS_MetaData         * pMetaData,
               db2ACS_ObjectID           objectID,
               db2ACS_CB               * pControlBlock,
               db2ACS_ReturnCode       * pRC );
```

参数

pMetaData

数据类型: db2ACS_MetaData

db2ACS_MetaData 存储快照备份元数据。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

存储在 **pMetaData** 的 **data** 字段中的元数据是数据库管理器固有的，可能会随着时间的推移而发生变化，因此 DB2 ACS API 驱动程序只需将此数据视为二进制流。

objectID

数据类型：db2ACS_ObjectID

db2ACS_ObjectID 是由查询返回到存储库的每个存储对象的唯一标识。保证 db2ACS_ObjectID 是唯一的，且仅在单一 DB2 ACS 会话时间框架中持久存在。

数据库管理器可以使用 db2ACSQuery() 来获取有效的 **objectID**，以便传递到 db2ACSRetrieveMetaData()。

pControlBlock

数据类型：db2ACS_CB *

db2ACS_CB 包含初始化和终止 DB2 ACS 会话所需的基本信息。

在调用 db2ACSRetrieveMetaData() 之前，数据库管理器填充下列字段:

pControlBlock->handle
pControlBlock->vendorInfo
pControlBlock->options

pRC 数据类型：db2ACS_ReturnCode *

db2ACS_ReturnCode 包含诊断信息，其中包括特定于存储硬件的消息文本和错误代码。用于 DB2 ACS API 函数调用的 db2ACS_ReturnCode 参数的内容将记录在数据库管理器诊断日志中。

数据库管理器为此参数分配内存，并将指向例示对象的指针传递到该函数。数据库管理器负责释放此内存。

在返回之前，DB2 ACS API 驱动程序填充 **pRC** 的字段。

返回码

表 32. 返回码

返回码	描述	注意
DB2ACS_RC_OK	操作已成功。	
DB2ACS_RC_INV_ACTION	数据库管理器请求来自 DB2 ACS API 驱动程序的无效操作。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_INV_DEV_HANDLE	数据库管理器传递无效的存储设备句柄。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_DEV_ERROR	存在与诸如磁带机之类的存储设备有关的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_IO_ERROR	DB2 ACS API 驱动程序遇到输入或输出操作所产生的错误。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。
DB2ACS_RC_OBJ_NOT_FOUND	DB2 ACS API 驱动程序找不到数据库管理器指定的快照备份对象。	DB2 ACS API 驱动程序遇到错误。数据库管理器无法使用 DB2 ACS API 会话。

如果 DB2 ACS API 驱动程序遇到错误，那么该驱动程序可能会中止 DB2 ACS 操作。DB2 ACS 会话不能用于除以下各项之外的任何操作:

- 如果先前对 db2ACSBeginQuery() 的调用成功了，那么数据库管理器可以调用 db2ACSEndQuery()
- 如果先前对 db2ACSBeginOperation() 的调用成功了，那么数据库管理器可以调用 db2ACSEndOperation()
- 如果先前对 db2ACSInitialize() 的调用成功了，那么数据库管理器可以调用 db2ACSTerminate()

有关 DB2 ACS API 返回码的更多信息，请参阅主题: 第 422 页的『DB2 高级副本服务 (ACS) API 返回码』。

使用说明

无。

DB2 高级副本服务 (ACS) API 数据结构

要调用 DB2 高级副本服务 (ACS) API 函数，必须使用 DB2 ACS API 数据结构。

db2ACS_BackupDetails DB2 高级副本服务 (ACS) API 数据结构

db2ACS_BackupDetails 包含有关快照备份操作的信息。

```
/* ------------------------------------------------------------------------ */
typedef struct db2ACS_BackupDetails
{
   /* A traditional DB2 backup can consist of multiple objects (logical tapes),
    * where each object is uniquely numbered with a non-zero natural number.
    * ------------------------------------------------------------------------ */
   db2Uint32                  sequenceNum;

   char                       imageTimestamp[SQLU_TIME_STAMP_LEN + 1];
} db2ACS_BackupDetails;
```

sequenceNum

数据类型: db2Uint32。

通过其唯一的编号来标识备份对象。

imageTimestamp

数据类型: char[]。

长度 SQLU_TIME_STAMP_LEN + 1 的字符串。

db2ACS_CB DB2 高级副本服务 (ACS) API 数据结构

db2ACS_CB 包含初始化和终止 DB2 ACS 会话所需的基本信息。

```
/* ============================================================================
 * DB2 Backup Adapter Control Block
 * ============================================================================ */
typedef struct db2ACS_CB
{
   /* Output:  Handle value for this session.
    * ------------------------------------------------------------------------ */
   db2Uint32                  handle;
   db2ACS_VendorInfo          vendorInfo;
```

```
   /* Input fields and parameters.
    * ------------------------------------------------------------------------ */
   db2ACS_SessionInfo         session;
   db2ACS_Options             options;

   /* Operation info is optional, possibly NULL, and is only ever valid
    * within the context of an operation (from call to BeginOperation() until
    * the EndOperation() call returns).
    *
    * The operation info will be present during creation or read operations
    * of snapshot and backup objects.
    * ------------------------------------------------------------------------ */
   db2ACS_OperationInfo     * operation;
} db2ACS_CB;
```

handle

数据类型：db2Uint32。

用于引用 DB2 ACS 会话的句柄。

vendorInfo

数据类型：db2ACS_VendorInfo。

db2ACS_VendorInfo 包含有关 DB2 ACS API 驱动程序的信息。

session

数据类型：db2ACS_SessionInfo。

db2ACS_SessionInfo 包含有关 DB2 ACS 会话的所有信息。

options

数据类型：db2ACS_Options。

db2ACS_Options 指定要用于 DB2 ACS 操作的选项。此字符串的内容特定于 DB2 ACS API 驱动程序。

operation

数据类型：db2ACS_OperationInfo *。

db2ACS_OperationInfo 包含有关快照备份操作的信息。

db2ACS_CreateObjectInfo DB2 高级副本服务 (ACS) API 数据结构

db2ACS_CreateObjectInfo 包含有关创建 DB2 ACS 备份对象的信息。

```
/* ============================================================================
 * Object Creation Parameters.
 * ============================================================================ */
typedef struct db2ACS_CreateObjectInfo
{
   db2ACS_ObjectInfo          object;
   db2ACS_DB2ID               db2ID;

   /* ------------------------------------------------------------------------
    * The following fields are optional information for the database manager
    * to use as it sees fit.
    * ------------------------------------------------------------------------ */

   /* Historically both the size estimate and management
    * class parameters have been used by the TSM client API for traditional
    * backup objects, log archives, and load copies, but not for snapshot
    * backups.
    * ------------------------------------------------------------------------ */
   db2Uint64                  sizeEstimate;
   char                       mgmtClass[DB2ACS_MAX_MGMTCLASS_SZ + 1];
```

```
   /* The appOptions is a copy of the iOptions field of flags passed to DB2's
    * db2Backup() API when this execution was initiated. This field will
    * only contain valid data when creating a backup or snapshot object.
    * ------------------------------------------------------------------------ */
   db2Uint32                 appOptions;
} db2ACS_CreateObjectInfo;
```

object 数据类型: db2ACS_ObjectInfo

db2ACS_ObjectInfo 包含有关使用 DB2 高级副本服务 (ACS) API 创建的对象的信息。

db2ID 数据类型: db2ACS_DB2ID

db2ACS_DB2ID 标识了 IBM 数据服务器。

sizeEstimate

数据类型: db2Uint64。

正在创建的备份对象大小的估计值。此估计值不适用于日志归档、装入副本或快照备份对象。

mgmtClass

数据类型: db2ACS_MgmtClass。

长度 db2ACS_MAX_MGMTCLASS_SZ + 1 的字符串。

这不适用于快照备份对象。

appOptions

数据类型: db2Uint32。

将备份选项的副本传递至初始化快照备份的备份命令。

db2ACS_DB2ID DB2 高级副本服务 (ACS) API 数据结构

db2ACS_DB2ID 标识了 IBM 数据服务器。

```
/* ============================================================================
 * DB2 Data Server Identifier
 * ============================================================================ */
typedef struct db2ACS_DB2ID
{
   db2Uint32                 version;
   db2Uint32                 release;
   db2Uint32                 level;
   char                      signature[DB2ACS_SIGNATURE_SZ + 1];
} db2ACS_DB2ID;
```

version

数据类型: db2Uint32。

IBM 数据服务器的版本。例如: 9

release

数据类型: db2Uint32。

IBM 数据服务器的发行版级别。例如: 5

level 数据类型: db2Uint32。

IBM 数据服务器的级别标识。例如: 0

signature

数据类型: char[]。

长度 DB2ACS_SIGNATURE_SZ + 1 的字符串。例如: “SQL09050”

db2ACS_GroupList DB2 高级副本服务 (ACS) API 数据结构

db2ACS_GroupList 包含在快照备份操作中要包含的组的列表。

```
/* ============================================================================
 * Snapshot Group List
 *
 * This is an array of size 'numGroupIDs', indicating the set of groups that
 * are to be included in the snapshot operation.
 * ============================================================================ */
typedef struct db2ACS_GroupList
{
   db2Uint32                 numGroupIDs;
   db2Uint32               * id;
} db2ACS_GroupList;
```

numGroupIDs

数据类型: db2Uint32。

数组 **id** 中组的编号。

id 数据类型: db2Uint32 *。

组标识的数组。已标识的组是快照备份操作中要包含的组（或路径列表）。

db2ACS_LoadcopyDetails DB2 高级副本服务 (ACS) API 数据结构

db2ACS_LoadcopyDetails 包含有关装入副本操作的信息。

```
/* ---------------------------------------------------------------------- */
typedef struct db2ACS_LoadcopyDetails
{
   /* Just like the BackupDetails, a DB2 load copy can consist of multiple
    * objects (logical tapes), where each object is uniquely numbered with a
    * non-zero natural number.
    * ---------------------------------------------------------------------- */
   db2Uint32                 sequenceNum;

   char                      imageTimestamp[SQLU_TIME_STAMP_LEN + 1];
} db2ACS_LoadcopyDetails;
```

sequenceNum

数据类型: db2Uint32。

通过其唯一的编号来标识备份对象。

imageTimestamp

数据类型: char[]。

长度 SQLU_TIME_STAMP_LEN + 1 的字符串

db2ACS_LogDetails DB2 高级副本服务 (ACS) API 数据结构

db2ACS_LogDetails 包含用于标识特定数据库日志文件的信息。

```
/* ---------------------------------------------------------------------- */
typedef struct db2ACS_LogDetails
{
   db2Uint32                 fileID;
   db2Uint32                 chainID;
} db2ACS_LogDetails;
```

fileID 数据类型: db2Uint32。

一个编号，它是数据库日志文件的文件名。

chainID

数据类型：db2Uint32。

一个编号，它标识了数据库日志文件 **fileID** 所属的数据库日志文件链。

db2ACS_ObjectInfo DB2 高级副本服务 (ACS) API 数据结构

db2ACS_ObjectInfo 包含有关使用 DB2 高级副本服务 (ACS) API 创建的对象的信息。

```
/* ============================================================================
 * Object Description and Associated Information.
 *
 * This structure is used for both input and output, and its contents define
 * the minimum information that must be recorded about any object created
 * through this interface.
 * ============================================================================ */
typedef struct db2ACS_ObjectInfo
{
   db2ACS_ObjectType          type;
   SQL_PDB_NODE_TYPE          dbPartitionNum;

   char                       db[SQL_DBNAME_SZ + 1];
   char                       instance[DB2ACS_MAX_OWNER_SZ + 1];
   char                       host[SQL_HOSTNAME_SZ + 1];
   char                       owner[DB2ACS_MAX_OWNER_SZ + 1];

   union
   {
      db2ACS_BackupDetails    backup;
      db2ACS_LogDetails       log;
      db2ACS_LoadcopyDetails  loadcopy;
      db2ACS_SnapshotDetails  snapshot;
   } details;
} db2ACS_ObjectInfo;
```

type 数据类型：db2ACS_ObjectType。

指定快照备份对象类型。值：

DB2ACS_OBJTYPE_ALL
DB2ACS_OBJTYPE_BACKUP
DB2ACS_OBJTYPE_LOG
DB2ACS_OBJTYPE_LOADCOPY
DB2ACS_OBJTYPE_SNAPSHOT

DB2ACS_OBJTYPE_ALL 只能用作查询的过滤器。不存在类型为 0 的对象。

dbPartitionNum

数据类型：SQL_PDB_NODE_TYPE。

此数据库分区的标识。

db 数据类型：char[]。

长度 SQL_DBNAME_SZ + 1 的字符串。

instance

数据类型：char[]。

长度 DB2ACS_MAX_OWNER_SZ + 1 的字符串。

host 数据类型：char[]。

长度 SQL_HOSTNAME_SZ + 1 的字符串。

owner 数据类型：char[]。

长度 DB2ACS_MAX_OWNER_SZ + 1 的字符串。

details

backup

数据类型：db2ACS_BackupDetails

db2ACS_BackupDetails 包含有关快照备份操作的信息。

log 数据类型：db2ACS_LogDetails

db2ACS_LogDetails 包含用于标识特定数据库日志文件的信息。

loadcopy

数据类型：db2ACS_LoadcopyDetails

db2ACS_LoadcopyDetails 包含有关装入副本操作的信息。

snapshot

数据类型：db2ACS_SnapshotDetails

db2ACS_SnapshotDetails 包含有关快照备份操作的信息。

db2ACS_ObjectStatus DB2 高级副本服务 (ACS) API 数据结构

db2ACS_ObjectStatus 包含有关快照备份操作的状态或进度的信息，或包含有关快照备份对象的状态或可用性的信息。

```
typedef struct db2ACS_ObjectStatus
{
   /* The total and completed bytes refer only to the ACS snapshot backup
    * itself, not to the progress of any offloaded tape backup.
    *
    * A bytesTotal of 0 indicates that the progress could not be determined.
    * ------------------------------------------------------------------------ */
   db2Uint64                bytesCompleted;
   db2Uint64                bytesTotal;
   db2ACS_ProgressState     progressState;
   db2ACS_UsabilityState    usabilityState;
} db2ACS_ObjectStatus;
```

bytesCompleted

数据类型：db2Uint64。

已完成的快照备份的数量（以字节计）。

bytesTotal

数据类型：db2Uint64。

已完成的快照备份的大小（以字节计）。

progressState

数据类型：db2ACS_ProgressState。

快照备份操作的状态。值：

DB2ACS_PSTATE_UNKNOWN
DB2ACS_PSTATE_IN_PROGRESS
DB2ACS_PSTATE_SUCCESSFUL
DB2ACS_PSTATE_FAILED

usabilityState

数据类型：db2ACS_UsabilityState。

快照备份对象的状态以及如何可以使用快照备份对象。值:

DB2ACS_USTATE_UNKNOWN
DB2ACS_USTATE_LOCALLY_MOUNTABLE
DB2ACS_USTATE_REMOTELY_MOUNTABLE
DB2ACS_USTATE_REPETITIVELY_RESTORABLE
DB2ACS_USTATE_DESTRUCTIVELY_RESTORABLE
DB2ACS_USTATE_SWAP_RESTORABLE
DB2ACS_USTATE_PHYSICAL_PROTECTION
DB2ACS_USTATE_FULL_COPY
DB2ACS_USTATE_DELETED
DB2ACS_USTATE_FORCED_MOUNT
DB2ACS_USTATE_BACKGROUND_MONITOR_PENDING
DB2ACS_USTATE_TAPE_BACKUP_PENDING
DB2ACS_USTATE_TAPE_BACKUP_IN_PROGRESS
DB2ACS_USTATE_TAPE_BACKUP_COMPLETE

db2ACS_OperationInfo DB2 高级副本服务 (ACS) API 数据结构

db2ACS_OperationInfo 包含有关快照备份操作的信息。

```
/* ============================================================================
 * Operation Info
 *
 * The information contained within this structure is only valid within the
 * context of a particular operation. It will be valid at the time
 * BeginOperation() is called, and will remain unchanged until EndOperation()
 * returns, but must not be referenced outside the scope of an operation.
 * ============================================================================ */
typedef struct db2ACS_OperationInfo
{
   db2ACS_SyncMode            syncMode;

   /* List of database and backup operation partitions.
    *
    * For details, refer to the db2ACS_PartitionList definition.
    * ------------------------------------------------------------------------- */
   db2ACS_PartitionList     * dbPartitionList;
} db2ACS_OperationInfo;
```

syncMode

数据类型：db2ACS_SyncMode。

单独的数据库分区上备份操作之间的同步级别。

值:

DB2ACS_SYNC_NONE

在多个数据库分区上的相关操作之间没有同步。在操作期间，如果没有利用多个数据库分区之间的任何同步，那么使用该值。

DB2ACS_SYNC_SERIAL

当在多个数据库分区上执行并发快照备份操作时使用该值。当快照备份操作发出后，每个数据库分区将暂挂其输入和输出（IO），然后以串行方式而不是并发方式恢复数据库分区上的 I/O。

SYNC_PARALLEL

对多个分区并发执行快照操作。一旦快照备份操作涉及的所有数据库分区完成快照备份操作准备，输入和输出（IO）将在所有数据库分区上暂挂。剩余快照备份步骤将在所有涉及的数据库分区上并发执行。

dbPartitionList

数据类型：db2ACS_PartitionList *。

db2ACS_PartitionList 包含有关数据库中的数据库分区以及 DB2 ACS 操作中涉及的数据库分区的信息。

db2ACS_Options DB2 高级副本服务 (ACS) API 数据结构

db2ACS_Options 指定要用于 DB2 ACS 操作的选项。此字符串的内容特定于 DB2 ACS API 驱动程序。

```
/* ****************************************************************************
 * DB2 Backup Adapter User Options
 * ************************************************************************** */
typedef struct db2ACS_Options
{
   db2Uint32                  size;
   void                     * data;
} db2ACS_Options;
```

size 数据类型：db2Uint32。

data 的大小（以字节计）。

data 数据类型：void *。

指向包含选项的内存块的指针。

db2ACS_PartitionEntry DB2 高级副本服务 (ACS) API 数据结构

db2ACS_PartitionEntry 是 db2ACS_PartitionList 的元素。

```
typedef struct db2ACS_PartitionEntry
{
   SQL_PDB_NODE_TYPE          num;
   char                       host[SQL_HOSTNAME_SZ + 1];
} db2ACS_PartitionEntry;
```

num 数据类型：SQL_PDB_NODE_TYPE。

此数据库分区条目的标识。

host 数据类型：char[]。

长度 SQL_HOSTNAME_SZ + 1 的字符串。

db2ACS_PartitionList DB2 高级副本服务 (ACS) API 数据结构

db2ACS_PartitionList 包含有关数据库中的数据库分区以及 DB2 ACS 操作中涉及的数据库分区的信息。

```
typedef struct db2ACS_PartitionList
{
   db2Uint64                  numPartsInDB;
   db2Uint64                  numPartsInOperation;
   db2ACS_PartitionEntry    * partition;
} db2ACS_PartitionList;
```

numPartsInDB

数据类型：db2Uint64。

数据库中的数据库分区数目。

numPartsInOperation

数据类型：db2Uint64。

DB2 ACS 操作中涉及的数据库分区的数目。

partition

数据类型：db2ACS_PartitionEntry *。

db2ACS_PartitionEntry 是 db2ACS_PartitionList 的元素。

db2ACS_PathEntry DB2 高级副本服务 (ACS) API 数据结构

db2ACS_PathEntry 包含有关数据库路径的信息。

```
typedef struct db2ACS_PathEntry
{
   /* INPUT: The path and type will be provided by the database server, as well
    *        as a flag indicating if the path is to be excluded from the backup.
    * ------------------------------------------------------------------------ */
   char                       path[DB2ACS_MAX_PATH_SZ + 1];
   db2ACS_PathType            type;
   db2Uint32                  toBeExcluded;

   /* OUTPUT: The group ID is to be provided by the backup adapter for use by
    *         the DB2 server. The group ID will be used during with snapshot
    *         operations as an indication of which paths are dependent and must
    *         be included together in any snapshot operation. Unique group IDs
    *         indicate that the paths in those groups are independent for the
    *         purposes of snapshot operations.
    * ------------------------------------------------------------------------ */
   db2Uint32                  groupID;
} db2ACS_PathEntry;
```

path 数据类型：char[]。

长度 DB2ACS_MAX_PATH_SZ + 1 的字符串。

type 数据类型：db2ACS_PathType。

路径类型。值:

DB2ACS_PATH_TYPE_UNKNOWN
DB2ACS_PATH_TYPE_LOCAL_DB_DIRECTORY
DB2ACS_PATH_TYPE_DBPATH
DB2ACS_PATH_TYPE_DB_STORAGE_PATH
DB2ACS_PATH_TYPE_TBSP_CONTAINER
DB2ACS_PATH_TYPE_TBSP_DIRECTORY
DB2ACS_PATH_TYPE_TBSP_DEVICE
DB2ACS_PATH_TYPE_LOGPATH
DB2ACS_PATH_TYPE_MIRRORLOGPATH

toBeExcluded

数据类型：db2Uint32。

指示快照备份中是否包含给定路径的标志。值:

- 0 - 快照备份中包含路径
- 1 - 快照备份中不包含路径

groupID

数据类型：db2Uint32。

组标识。

db2ACS_PathList DB2 高级副本服务 (ACS) API 数据结构

db2ACS_PathList 包含数据库路径列表，包括有关特定于 DB2 ACS 操作的每个这些路径的一些额外信息。

```
/* ============================================================================
 * Snapshot File List
 *
 * This is an array of 'numEntries' db2ACS_PathEntry's, where each path entry is
 * a path to some storage on the DB2 server which is in use by the current
 * database.
 * ============================================================================ */

typedef struct db2ACS_PathList
{
   db2Uint32                  numEntries;
   db2ACS_PathEntry         * entry;
} db2ACS_PathList;
```

numEntries

数据类型：db2Uint32。

entry 数组中路径条目的数目。

entry 数据类型：db2ACS_PathEntry。

db2ACS_PathEntry 包含有关数据库路径的信息。

db2ACS_PostObjectInfo DB2 高级副本服务 (ACS) API 数据结构

db2ACS_DB2ID 是一组数据，在快照备份对象创建时无法知道，但必须在对象存储库中对其进行维护。

```
/* ============================================================================
 * The PostObjectInfo is a set of data that can not be known at object
 * creation time, but which must be maintained in the object repository. This
 * is an optional field on the Verify() call, which may be NULL if there are
 * no post-operation updates to be made.
 * ============================================================================ */
typedef struct db2ACS_PostObjectInfo
{
   /* The first active log will only be valid when creating a backup or
    * snapshot object. It will indicate the file number and chain id of the
    * first log required for recovery using this object.
    * ------------------------------------------------------------------------ */
   db2ACS_LogDetails          firstActiveLog;
} db2ACS_PostObjectInfo;
```

firstActiveLog

数据类型：db2ACS_LogDetails。

db2ACS_LogDetails 包含用于标识特定数据库日志文件的信息。

db2ACS_QueryInput 和 db2ACS_QueryOutput DB2 高级副本服务 (ACS) API 数据结构

db2ACS_QueryInput 包含要查询的对象的标识信息。 db2ACS_QueryOutput 包含有关快照备份对象的查询结果信息。

```
/* ============================================================================
 * Unique Querying.
 *
 * When using this structure as query input, to indicate the
 * intention to supply a 'wildcard' search criteria, DB2 will supply:
```

```
 *
 *    -- character strings as "*".
 *    -- numeric values as (-1), cast as the appropriate signed or unsigned
 *       type.
 * ========================================================================== */

typedef struct db2ACS_ObjectInfo db2ACS_QueryInput;

typedef struct db2ACS_QueryOutput
{
   db2ACS_ObjectID            objectID;
   db2ACS_ObjectInfo          object;
   db2ACS_PostObjectInfo      postInfo;
   db2ACS_DB2ID               db2ID;
   db2ACS_ObjectStatus        status;

   /* Size of the object in bytes.
    * ------------------------------------------------------------------  */
   db2Uint64                  objectSize;

   /* Size of the metadata associated with the object, if any, in bytes.
    * ------------------------------------------------------------------  */
   db2Uint64                  metaDataSize;

   /* The creation time of the object is a 64bit value with a definition
    * equivalent to an ANSI C time_t value (seconds since the epoch, GMT).
    *
    * This field is equivalent to the file creation or modification time in
    * a traditional filesystem. This should be created and stored
    * automatically by the BA subsystem, and a valid time value should be
    * returned with object query results, for all object types.
    * ------------------------------------------------------------------  */
   db2Uint64                  createTime;
} db2ACS_QueryOutput;
```

objectID

数据类型：db2ACS_ObjectID。

db2ACS_ObjectID 是由查询返回到存储库的每个存储对象的唯一标识。保证 db2ACS_ObjectID 是唯一的，且仅在单一 DB2 ACS 会话时间框架中持久存在。

object 数据类型：db2ACS_ObjectInfo

db2ACS_ObjectInfo 包含有关使用 DB2 高级副本服务 (ACS) API 创建的对象的信息。

postInfo

数据类型：db2ACS_PostObjectInfo。

db2ACS_DB2ID 是一组数据，在快照备份对象创建时无法知道，但必须在对象存储库中对其进行维护。

db2ID 数据类型：db2ACS_DB2ID。

db2ACS_DB2ID 标识了 IBM 数据服务器。

status 数据类型：db2ACS_ObjectStatus。

db2ACS_ObjectStatus 包含有关快照备份操作的状态或进度的信息，或包含有关快照备份对象的状态或可用性的信息。

objectSize

数据类型：db2Uint64。

对象大小（以字节计）。

metaDataSize

数据类型：db2Uint64。

与对象（如果有）相关的元数据的大小（以字节计）。

createTime

数据类型：db2Uint64。

对象的创建时间。**createTime** 的值等于 ANSI C time_t 值。

db2ACS_ReadList DB2 高级副本服务 (ACS) API 数据结构

db2ACS_ReadList 包含组列表。

```
/* The ReadList will only be used for snapshots where the action is READ, and
 * where one of the granularity modifiers other than BY_OBJ has been specified.
 * In the typical usage scenario of ( READ | BY_OBJ ) the ReadList parameter
 * should be ignored.
 *
 * When the action is DB2ACS_ACTION_BY_GROUP the union is to be interpreted
 * as a group list.
 * ------------------------------------------------------------------------- */
typedef union db2ACS_ReadList
{
   db2ACS_GroupList           group;
} db2ACS_ReadList;
```

group 数据类型：db2ACS_GroupList。

db2ACS_GroupList 包含在快照备份操作中要包含的组的列表。

db2ACS_ReturnCode DB2 高级副本服务 (ACS) API 数据结构

db2ACS_ReturnCode 包含诊断信息，其中包括特定于存储硬件的消息文本和错误代码。用于 DB2 ACS API 函数调用的 db2ACS_ReturnCode 参数的内容将记录在数据库管理器诊断日志中。

```
/* ========================================================================
 * Storage Adapter Return Code and Diagnostic Data.
 *
 * These will be recorded in the DB2 diagnostic logs, but are intended to be
 * internal return and reason codes from the storage layers which can be used
 * in conjunction with the DB2ACS_RC to provide more detailed diagnostic info.
 * ======================================================================== */
typedef struct db2ACS_ReturnCode
{
   int                        returnCode;
   int                        reasonCode;
   char                       description[DB2ACS_MAX_COMMENT_SZ + 1];
} db2ACS_ReturnCode;
```

returnCode

数据类型：int。

特定于存储硬件的返回码。

reasonCode

数据类型：int。

特定于存储硬件的原因码。

description

数据类型：char[]。

长度 DB2ACS_MAX_COMMENT_SZ + 1 的字符串。

db2ACS_SessionInfo DB2 高级副本服务 (ACS) API 数据结构

db2ACS_SessionInfo 包含有关 DB2 ACS 会话的所有信息。

```
/* ============================================================================
 * Session Info
 * ============================================================================ */
typedef struct db2ACS_SessionInfo
{
   db2ACS_DB2ID               db2ID;

   /* Fields identifying the backup session originator.
    * ------------------------------------------------------------------------ */
   SQL_PDB_NODE_TYPE          dbPartitionNum;
   char                       db[SQL_DBNAME_SZ + 1];
   char                       instance[DB2ACS_MAX_OWNER_SZ + 1];
   char                       host[SQL_HOSTNAME_SZ + 1];
   char                       user[DB2ACS_MAX_OWNER_SZ + 1];
   char                       password[DB2ACS_MAX_PASSWORD_SZ + 1];

   /* The fully qualified ACS vendor library name to be used.
    * ------------------------------------------------------------------------ */
   char                       libraryName[DB2ACS_MAX_PATH_SZ + 1];
} db2ACS_SessionInfo;
```

db2ID 数据类型: db2ACS_DB2ID

db2ACS_DB2ID 标识了 IBM 数据服务器。

dbPartitionNum

数据类型: SQL_PDB_NODE_TYPE

数据库分区的唯一的数字标识。

db 数据类型: char[]。

长度 SQL_DBNAME_SZ + 1 的字符串。

instance

数据类型: char[]。

长度 DB2ACS_MAX_OWNER_SZ + 1 的字符串。

host 数据类型: char[]。

长度 SQL_HOSTNAME_SZ + 1 的字符串。

user 数据类型: char[]。

长度 DB2ACS_MAX_OWNER_SZ + 1 的字符串。

password

数据类型: char[]。

长度 DB2ACS_MAX_PASSWORD_SZ + 1 的字符串。

libraryName

数据类型: char[]。

长度 DB2ACS_MAX_PATH_SZ + 1 的字符串。

db2ACS_SnapshotDetails DB2 高级副本服务 (ACS) API 数据结构

db2ACS_SnapshotDetails 包含有关快照备份操作的信息。

```
typedef struct db2ACS_SnapshotDetails
{
   char                       imageTimestamp[SQLU_TIME_STAMP_LEN + 1];
} db2ACS_SnapshotDetails;
```

imageTimestamp

数据类型：char[]。

长度 SQLU_TIME_STAMP_LEN + 1 的字符串。

db2ACS_MetaData DB2 高级副本服务 (ACS) API 数据结构

db2ACS_MetaData 存储快照备份元数据。

```
/* ****************************************************************************
 * The metadata structure itself is internal to DB2 and is to be treated by
 * the storage interface as an unstructured block of data of the given size.
 * **************************************************************************** */
typedef struct db2ACS_MetaData
{
   db2Uint64                  size;
   void                     * data;
} db2ACS_MetaData;
```

size 数据类型：db2Uint32。

data 的大小（以字节计）。

data 数据类型：void *。

指向数据库管理器用于存储快照备份元数据的内存块的指针。

db2ACS_VendorInfo DB2 高级副本服务 (ACS) API 数据结构

db2ACS_VendorInfo 包含有关 DB2 ACS API 驱动程序的信息。

```
/* ****************************************************************************
 * Storage Vendor Identifier
 * **************************************************************************** */
typedef struct db2ACS_VendorInfo
{
   void                     * vendorCB;             /* Vendor control block  */
   db2Uint32                  version;              /* Current version       */
   db2Uint32                  release;              /* Current release       */
   db2Uint32                  level;                /* Current level         */
   char                       signature[DB2ACS_MAX_VENDORID_SZ + 1];
} db2ACS_VendorInfo;
```

vendorCB

数据类型：void *。

特定于 DB2 ACS API 驱动程序的控制块的指针。

version

数据类型：db2Uint32。

DB2 ACS API 驱动程序的版本。

release

数据类型：db2Uint32。

DB2 ACS API 驱动程序的发行版级别。

level 数据类型：db2Uint32。

DB2 ACS API 驱动程序的级别标识。

signature

数据类型：db2ACS_VendorSignature。

长度 DB2ACS_MAX_VENDORID_SZ + 1 的字符串。

DB2 高级副本服务 (ACS) API 返回码

DB2 高级副本服务 (ACS) API 函数返回已定义的一组可能的返回码。

表 33. DB2 高级副本服务 (ACS) API 返回码

返回码	描述
DB2ACS_RC_OK	操作已成功。
DB2ACS_RC_LINK_EXIST	先前已激活会话。
DB2ACS_RC_COMM_ERROR	存在与诸如磁带机之类的存储设备的通信错误。
DB2ACS_RC_INV_VERSION	数据库管理器的 DB2 ACS 库版本与 DB2 ACS API 驱动程序版本不兼容。
DB2ACS_RC_INV_ACTION	数据库管理器请求来自 DB2 ACS API 驱动程序的无效操作。
DB2ACS_RC_NO_DEV_AVAIL	当前不存在诸如磁带机之类的存储设备可用。
DB2ACS_RC_OBJ_NOT_FOUND	DB2 ACS API 驱动程序找不到数据库管理器指定的快照备份对象。
DB2ACS_RC_OBJS_FOUND	DB2 ACS API 驱动程序找到多个与数据库管理器指定的规范相符的快照备份对象。
DB2ACS_RC_INV_USERID	数据库管理器将无效的用户标识传递到 DB2 ACS API 驱动程序。
DB2ACS_RC_INV_PASSWORD	数据库管理器将无效的密码传递到 DB2 ACS API 驱动程序。
DB2ACS_RC_INV_OPTIONS	数据库管理器指定了无效选项。
DB2ACS_RC_INIT_FAILED	数据库管理器尝试初始化 DB2 ACS 会话，但初始化失败。
DB2ACS_RC_INV_DEV_HANDLE	数据库管理器传递无效的存储设备句柄。
DB2ACS_RC_BUFF_SIZE	数据库管理器指定了无效的缓冲区大小。
DB2ACS_RC_END_OF_DATA	DB2 ACS API 驱动程序找不到更多的快照备份对象。
DB2ACS_RC_END_OF_TAPE	存储设备意外地到达磁带备份介质的末端。
DB2ACS_RC_DATA_RESEND	诸如磁带机之类的存储设备请求数据库管理器重新发送最近的数据缓冲区。
DB2ACS_RC_COMMIT_FAILED	DB2 ACS API 驱动程序无法落实事务。
DB2ACS_RC_DEV_ERROR	存在与诸如磁带机之类的存储设备有关的错误。
DB2ACS_RC_WARNING	存储硬件返回警告。查看数据库管理器诊断日志，以了解更多信息。
DB2ACS_RC_LINK_NOT_EXIST	先前未激活会话。
DB2ACS_RC_MORE_DATA	有更多的数据要从存储位置传送到数据库管理器。
DB2ACS_RC_ENDOFMEDIA_NO_DATA	存储设备到达存储介质的末端而未找到任何数据。
DB2ACS_RC_ENDOFMEDIA	存储设备到达存储介质的末端。
DB2ACS_RC_MAX_LINK_GRANT	已经建立最大数目的链接。数据库管理器无法建立更多链接。
DB2ACS_RC_IO_ERROR	DB2 ACS API 驱动程序遇到输入或输出操作所产生的错误。

表 33. DB2 高级副本服务 (ACS) API 返回码 (续)

返回码	描述
DB2ACS_RC_DELETE_FAILED	DB2 ACS API 驱动程序无法成功地删除数据库管理器指定的快照备份对象。
DB2ACS_RC_INV_BKUP_FNAME	数据库管理器为快照备份对象指定了无效的文件名。
DB2ACS_RC_NOT_ENOUGH_SPACE	DB2 ACS API 驱动程序估计没有足够的存储空间来执行数据库管理器指定的数据库的快照备份。
DB2ACS_RC_ABORT_FAILED	数据库管理器尝试中止 DB2 ACS 操作，但尝试中止失败。
DB2ACS_RC_UNEXPECTED_ERROR	DB2 ACS API 驱动程序遇到严重的未知错误。
DB2ACS_RC_NO_DATA	DB2 ACS API 驱动程序未将任何数据返回到数据库管理器。
DB2ACS_RC_OBJ_OUT_OF_SCOPE	数据库管理器尝试对不被 DB2 ACS API 驱动程序管理的恢复对象执行 DB2 ACS 操作。
DB2ACS_RC_INV_CALL_SEQUENCE	数据库管理器按照无效的顺序来调用 DB2 ACS API 函数。例如，数据库管理器必须先调用 db2ACSInitialize，然后才调用除 db2ACSQueryAPIVersion 之外的其他任何 DB2 ACS API 函数。
DB2ACS_RC_SHARED_STORAGE_GROUP	数据库管理器尝试针对另一数据库或应用程序正在使用的存储对象执行快照操作。

第 3 部分 附录

© Copyright IBM Corp. 2001, 2013

附录 A. DB2 技术信息概述

DB2 技术信息以多种可以通过多种方法访问的格式提供。

您可以通过下列工具和方法获得 DB2 技术信息:

- DB2 信息中心
  - 主题（任务、概念和参考主题）
  - 样本程序
  - 教程
- DB2 书籍
  - PDF 文件（可下载）
  - PDF 文件（在 DB2 PDF DVD 中）
  - 印刷版书籍
- 命令行帮助
  - 命令帮助
  - 消息帮助

注: DB2 信息中心主题的更新频率比 PDF 书籍或硬拷贝书籍的更新频率高。要获取最新信息，请安装可用的文档更新或者参阅 ibm.com 上的 DB2 信息中心。

您可以在线访问 ibm.com 上的其他 DB2 技术信息，例如技术说明、白皮书和 IBM Redbooks® 出版物。请访问以下网址处的 DB2 信息管理软件资料库站点: http://www.ibm.com/software/data/sw-library/。

文档反馈

我们非常重视您对 DB2 文档的反馈。如果您想就如何改善 DB2 文档提出建议，请向 db2docs@ca.ibm.com 发送电子邮件。DB2 文档小组将阅读您的所有反馈，但无法直接给您答复。请尽可能提供具体的示例，这样我们才能更好地了解您所关心的问题。如果您要提供有关具体主题或帮助文件的反馈，请加上标题和 URL。

请不要使用以上电子邮件地址与 DB2 客户支持机构联系。如果您遇到文档无法解决的 DB2 技术问题，请与您当地的 IBM 服务中心联系以获得帮助。

硬拷贝或 PDF 格式的 DB2 技术库

下列各表描述 IBM 出版物中心（网址为 www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss）所提供的 DB2 资料库。可从 www.ibm.com/support/docview.wss?rs=71&uid=swg2700947 下载 PDF 格式的 DB2 V10.1 手册的英文版本和翻译版本。

尽管这些表标识书籍有印刷版，但可能未在您所在国家或地区提供。

每次更新手册时，表单号都会递增。确保您正在阅读下面列示的手册的最新版本。

注: *DB2 信息中心*的更新频率比 PDF 或硬拷贝书籍的更新频率高。

© Copyright IBM Corp. 2001, 2013

表 34. DB2 技术信息

书名	书号	是否提供印刷版	最近一次更新时间
Administrative API Reference	SC27-3864-00	是	2012 年 4 月
Administrative Routines and Views	SC27-3865-01	否	2013 年 1 月
Call Level Interface Guide and Reference Volume 1	SC27-3866-01	是	2013 年 1 月
Call Level Interface Guide and Reference Volume 2	SC27-3867-01	是	2013 年 1 月
Command Reference	SC27-3868-01	是	2013 年 1 月
数据库管理概念和配置参考	S151-1758-01	是	2013 年 1 月
Data Movement Utilities Guide and Reference	SC27-3869-01	是	2013 年 1 月
数据库监视指南和参考	S151-1759-01	是	2013 年 1 月
数据恢复及高可用性指南与参考	S151-1755-01	是	2013 年 1 月
数据库安全性指南	S151-1753-02	是	2013 年 1 月
DB2 Workload Management Guide and Reference	SC27-3891-01	是	2013 年 1 月
开发 ADO.NET 和 OLE DB 应用程序	S151-1765-01	是	2013 年 1 月
开发嵌入式 SQL 应用程序	S151-1763-01	是	2013 年 1 月
Developing Java Applications	SC27-3875-01	是	2013 年 1 月
Developing Perl, PHP, Python, and Ruby on Rails Applications	SC27-3876-00	否	2012 年 4 月
开发 IBM Data Servers	S151-1901-00	是	2013 年 1 月
开发用户定义的例程（SQL 和外部例程）	S151-1761-01	是	2013 年 1 月
数据库应用程序开发入门	G151-1764-01	是	2013 年 1 月
Linux 和 Windows 上的 DB2 安装和管理入门	G151-1769-00	是	2012 年 4 月
全球化指南	S151-1757-00	是	2012 年 4 月
安装 DB2 服务器	G151-1768-01	是	2013 年 1 月
安装 IBM Data Server Client	G151-1751-00	否	2012 年 4 月
消息参考第 1 卷	S151-1767-01	否	2013 年 1 月
消息参考第 2 卷	S151-1766-01	否	2013 年 1 月

表 34. DB2 技术信息 (续)

书名	书号	是否提供印刷版	最近一次更新时间
Net Search Extender 管理和用户指南	S151-1905-01	否	2013 年 1 月
分区和集群指南	S151-1754-01	是	2013 年 1 月
Preparation Guide for DB2 10.1 Fundamentals Exam 610	SC27-4540-00	否	2013 年 1 月
Preparation Guide for DB2 10.1 DBA for Linux, UNIX, and Windows Exam 611	SC27-4541-00	否	2013 年 1 月
pureXML 指南	S151-1775-01	是	2013 年 1 月
Spatial Extender User's Guide and Reference	SC27-3894-00	否	2012 年 4 月
《SQL 过程语言: 应用程序启用和支持》	S151-1762-01	是	2013 年 1 月
SQL Reference Volume 1	SC27-3885-01	是	2013 年 1 月
SQL Reference Volume 2	SC27-3886-01	是	2013 年 1 月
Text Search Guide	SC27-3888-01	是	2013 年 1 月
故障诊断和调整数据库性能	S151-1760-01	是	2013 年 1 月
升级到 DB2 V10.1	S151-1770-01	是	2013 年 1 月
DB2 V10.1 新增内容	S151-1752-01	是	2013 年 1 月
XQuery 参考	S151-1774-01	否	2013 年 1 月

表 35. 特定于 DB2 Connect 的技术信息

书名	书号	是否提供印刷版	最近一次更新时间
DB2 Connect 安装和配置 DB2 Connect Personal Edition	S151-1773-00	是	2012 年 4 月
DB2 Connect 安装和配置 DB2 Connect 服务器	S151-1772-01	是	2013 年 1 月
DB2 Connect 用户指南	S151-1771-01	是	2013 年 1 月

从命令行处理器显示 SQL 状态帮助

DB2 产品针对可能充当 SQL 语句结果的条件返回 SQLSTATE 值。SQLSTATE 帮助说明 SQL 状态和 SQL 状态类代码的含义。

过程

要启动 SQL 状态帮助，请打开命令行处理器并输入:

```
? sqlstate or ? class code
```

其中，*sqlstate* 表示有效的 5 位 SQL 状态，*class code* 表示该 SQL 状态的前 2 位。例如，`? 08003` 显示 08003 SQL 状态的帮助，而 `? 08` 显示 08 类代码的帮助。

访问不同版本的 DB2 信息中心

您可以在 ibm.com® 上的不同信息中心中找到其他版本 DB2 产品的文档。

关于此任务

对于 DB2 V10.1 主题，*DB2 信息中心* URL 是 http://publib.boulder.ibm.com/infocenter/db2luw/v10r1。

对于 DB2 V9.8 主题，*DB2 信息中心* URL 是 http://publib.boulder.ibm.com/infocenter/db2luw/v9r8/。

对于 DB2 V9.7 主题，*DB2 信息中心* URL 是 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/。

对于 DB2 V9.5 主题，*DB2 信息中心* URL 是 http://publib.boulder.ibm.com/infocenter/db2luw/v9r5。

对于 DB2 V9.1 主题，*DB2 信息中心* URL 是 http://publib.boulder.ibm.com/infocenter/db2luw/v9/。

对于 DB2 V8 主题，请转至 *DB2 信息中心* URL: http://publib.boulder.ibm.com/infocenter/db2luw/v8/。

更新安装在计算机或内部网服务器上的 DB2 信息中心

安装在本地的 DB2 信息中心必须定期进行更新。

开始之前

必须已安装 DB2 V10.1 信息中心。有关详细信息，请参阅安装 DB2 服务器中的“使用 DB2 安装向导来安装 DB2 信息中心”主题。所有适用于安装信息中心的先决条件和限制同样适用于更新信息中心。

关于此任务

可以自动或手动更新现有的 DB2 信息中心:

- 自动更新将更新现有的信息中心功能部件和语言。自动更新的一个优点是，与手动更新相比，信息中心的不可用时间较短。另外，自动更新可设置为作为定期运行的其他批处理作业的一部分运行。
- 可以使用手动更新方法来更新现有的信息中心功能部件和语言。自动更新可以缩短更新过程中的停机时间，但如果您想添加功能部件或语言，那么必须执行手动过程。例如，如果本地信息中心最初安装的是英语和法语版，而现在还要安装德语版；那么手动更新将安装德语版，并更新现有信息中心的功能和语言。但是，手动更新要求您手动停止、更新和重新启动信息中心。在整个更新过程期间信息中心不可用。在自动更新过程中，信息中心仅在更新完成后停止工作以重新启动信息中心。

此主题详细说明了自动更新的过程。有关手动更新的指示信息，请参阅“手动更新安装在您的计算机或内部网服务器上的 DB2 信息中心”主题。

过程

要自动更新安装在计算机或内部网服务器上的 DB2 信息中心:

1. 在 Linux 操作系统上,
   a. 浏览至信息中心的安装位置。缺省情况下，DB2 信息中心安装在 /opt/ibm/db2ic/V10.1 目录中。
   b. 从安装目录浏览至 doc/bin 目录。
   c. 运行 update-ic 脚本:

      ```
      update-ic
      ```
2. 在 Windows 操作系统上,
   a. 打开命令窗口。
   b. 浏览至信息中心的安装位置。缺省情况下，DB2 信息中心安装在 <Program Files>\IBM\DB2 Information Center\V10.1 目录中，其中 <Program Files> 表示 Program Files 目录的位置。
   c. 从安装目录浏览至 doc\bin 目录。
   d. 运行 update-ic.bat 文件:

      ```
      update-ic.bat
      ```

结果

DB2 信息中心将自动重新启动。如果更新可用，那么信息中心会显示新的以及更新后的主题。如果信息中心更新不可用，那么会在日志中添加消息。日志文件位于 doc\eclipse\configuration 目录中。日志文件名称是随机生成的编号。例如，1239053440785.log。

手动更新安装在计算机或内部网服务器上的 DB2 信息中心

如果您已在本地安装 DB2 信息中心，那么可从 IBM 获取文档更新并进行安装。

关于此任务

手动更新安装在本地的 *DB2 信息中心*要求您:

1. 停止计算机上的 *DB2 信息中心*，然后以独立方式重新启动信息中心。如果以独立方式运行信息中心，那么网络上的其他用户将无法访问信息中心，因而您可以应用更新。DB2 信息中心的工作站版本总是以独立方式运行。
2. 使用“更新”功能部件来查看可用的更新。如果有您必须安装的更新，那么请使用“更新”功能部件来获取并安装这些更新。

   注: 如果您的环境要求在一台未连接至因特网的机器上安装 *DB2 信息中心*更新，请使用一台已连接至因特网并已安装 *DB2 信息中心*的机器将更新站点镜像至本地文件系统。如果网络中有许多用户将安装文档更新，那么可以通过在本地也为更新站点制作镜像并为更新站点创建代理来缩短每个人执行更新所需要的时间。
   如果提供了更新包，请使用“更新”功能部件来获取这些更新包。但是，只有在单机方式下才能使用“更新”功能部件。

3. 停止独立信息中心，然后在计算机上重新启动 *DB2 信息中心*。

注: 在 Windows 2008、Windows Vista 和更高版本上，稍后列示在此部分的命令必须作为管理员运行。要打开具有全面管理员特权的命令提示符或图形工具，请右键单击快捷方式，然后选择**以管理员身份运行**。

过程

要更新安装在您的计算机或内部网服务器上的 *DB2 信息中心*:

1. 停止 *DB2 信息中心*。
   - 在 Windows 上，单击**开始 > 控制面板 > 管理工具 > 服务**。右键单击 **DB2 信息中心**服务，并选择**停止**。
   - 在 Linux 上，输入以下命令:

     ```
     /etc/init.d/db2icdv10 stop
     ```
2. 以独立方式启动信息中心。
   - 在 Windows 上:
     a. 打开命令窗口。
     b. 浏览至信息中心的安装位置。缺省情况下，*DB2 信息中心*安装在 *Program_Files*\IBM\DB2 Information Center\V10.1 目录中，其中 *Program Files* 表示 Program Files 目录的位置。
     c. 从安装目录浏览至 doc\bin 目录。
     d. 运行 help_start.bat 文件:

     ```
     help_start.bat
     ```
   - 在 Linux 上:
     a. 浏览至信息中心的安装位置。缺省情况下，*DB2 信息中心*安装在 /opt/ibm/db2ic/V10.1 目录中。
     b. 从安装目录浏览至 doc/bin 目录。
     c. 运行 help_start 脚本:

     ```
     help_start
     ```

   系统缺省 Web 浏览器将打开以显示独立信息中心。
3. 单击**更新**按钮（  ）。（必须在浏览器中启用 JavaScript。）在信息中心的右边面板上，单击**查找更新**。将显示现有文档的更新列表。
4. 要启动安装过程，请检查您要安装的选项，然后单击**安装更新**。
5. 在安装进程完成后，请单击**完成**。
6. 要停止独立信息中心，请执行下列操作:
   - 在 Windows 上，浏览至安装目录中的 doc\bin 目录并运行 help_end.bat 文件:

     ```
     help_end.bat
     ```

     注: help_end 批处理文件包含安全地停止使用 help_start 批处理文件启动的进程所需的命令。不要使用 Ctrl-C 或任何其他方法来停止 help_start.bat。
   - 在 Linux 上，浏览至安装目录中的 doc/bin 目录并运行 help_end 脚本:

     ```
     help_end
     ```

注: help_end 脚本包含安全地停止使用 help_start 脚本启动的进程所需的命令。不要使用任何其他方法来停止 help_start 脚本。

7. 重新启动 *DB2 信息中心*。
   - 在 Windows 上，单击**开始 > 控制面板 > 管理工具 > 服务**。右键单击 **DB2 信息中心**服务，并选择**启动**。
   - 在 Linux 上，输入以下命令:

     ```
     /etc/init.d/db2icdv10 start
     ```

结果

更新后的 *DB2 信息中心*将显示新的以及更新后的主题。

DB2 教程

DB2 教程帮助您了解 DB2 数据库产品的各个方面。这些课程提供了逐步指示信息。

开始之前

您可以在信息中心中查看 XHTML 版的教程: http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/。

某些课程使用了样本数据或代码。有关其特定任务的任何先决条件的描述，请参阅教程。

DB2 教程

要查看教程，请单击标题。

*pureXML 指南*中的『**pureXML®**』
: 设置 DB2 数据库以存储 XML 数据以及对本机 XML 数据存储器执行基本操作。

DB2 故障诊断信息

我们提供了各种各样的故障诊断和问题确定信息来帮助您使用 DB2 数据库产品。

DB2 文档
: 您可以在《*故障诊断和调整数据库性能*》或者 *DB2 信息中心*的“数据库基础”部分中找到故障诊断信息，这些信息包含以下内容:
  - 有关如何使用 DB2 诊断工具和实用程序来隔离和确定问题的信息。
  - 一些最常见问题的解决方案。
  - 旨在帮助您解决 DB2 数据库产品使用过程中可能会遇到的其他问题的建议。

IBM Support Portal
: 如果您遇到问题并且希望得到帮助以查找可能的原因和解决方案，请访问 IBM Support Portal 。这个技术支持站点提供了指向最新 DB2 出版物、技术说明、授权程序分析报告（APAR 或错误修订）、修订包和其他资源的链接。可搜索此知识库并查找问题的可能解决方案。

  访问 IBM 支持门户网站网址为: http://www.ibm.com/support/entry/portal/Overview/Software/Information_Management/DB2_for_Linux,_UNIX_and_Windows

信息中心条款和条件

如果符合以下条款和条件，那么授予您使用这些出版物的许可权。

适用性: 用户需要遵循 IBM Web 站点的使用条款及以下条款和条件。

个人使用: 只要保留所有的专有权声明，您就可以为个人、非商业使用复制这些出版物。未经 IBM 明确同意，您不可以分发、展示或制作这些出版物或其中任何部分的演绎作品。

商业使用: 只要保留所有的专有权声明，您就可以仅在企业内复制、分发和展示这些出版物。未经 IBM 明确同意，您不可以制作这些出版物的演绎作品，或者在您的企业外部复制、分发或展示这些出版物或其中的任何部分。

权利: 除非本许可权中明确授予，否则不得授予对这些出版物或其中包含的任何信息、数据、软件或其他知识产权的任何许可权、许可证或权利，无论是明示的还是暗含的。

IBM 保留根据自身的判断，认为对出版物的使用损害了 IBM 的权益（由 IBM 自身确定）或未正确遵循以上指示信息时，撤回此处所授予权限的权利。

只有您完全遵循所有适用的法律和法规，包括所有的美国出口法律和法规，您才可以下载、出口或再出口该信息。

IBM 对这些出版物的内容不作任何保证。这些出版物“按现状”提供，不附有任何种类的（无论是明示的还是暗含的）保证，包括但不限于暗含的关于适销和适用于某种特定用途的保证。

IBM 商标: IBM、 IBM 徽标和 ibm.com 是 International Business Machines Corp., 在全球许多管辖区域注册的商标或注册商标。其他产品和服务名称可能是 IBM 或其他公司的商标。当前的 IBM 商标列表，可从 Web 站点 www.ibm.com/legal/copytrade.shtml获得

附录 B. 声明

本信息是为在美国提供的产品和服务编写的。有关非 IBM 产品的信息是基于首次出版此文档时的可获信息且会随时更新。

IBM 可能在其他国家或地区不提供本文档中讨论的产品、服务或功能特性。有关您当前所在区域的产品和服务的信息，请向您当地的 IBM 代表咨询。任何对 IBM 产品、程序或服务的引用并非意在明示或暗示只能使用 IBM 的产品、程序或服务。只要不侵犯 IBM 的知识产权，任何同等功能的产品、程序或服务，都可以代替 IBM 产品、程序或服务。但是，评估和验证任何非 IBM 产品、程序或服务，则由用户自行负责。

IBM 公司可能已拥有或正在申请与本文档内容有关的各项专利。提供本文档并未授予用户使用这些专利的任何许可。您可以用书面方式将许可查询寄往:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

有关双字节字符集 (DBCS) 信息的许可查询，请与您所在国家或地区的 IBM 知识产权部门联系，或用书面方式将查询寄往:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

本条款不适用英国或任何这样的条款与当地法律不一致的国家或地区: International Business Machines Corporation“按现状”提供本出版物，不附有任何种类的（无论是明示的还是暗含的）保证，包括但不限于暗含的有关非侵权、适销和适用于某种特定用途的保证。某些国家或地区在某些交易中不允许免除明示或暗含的保证。因此本条款可能不适用于您。

本信息中可能包含技术方面不够准确的地方或印刷错误。此处的信息将定期更改；这些更改将编入本资料的新版本中。IBM 可以随时对本资料中描述的产品和/或程序进行改进和/或更改，而不另行通知。

本信息中对非 IBM Web 站点的任何引用都只是为了方便起见才提供的，不以任何方式充当对那些 Web 站点的保证。那些 Web 站点中的资料不是此 IBM 产品资料的一部分，使用那些 Web 站点带来的风险将由您自行承担。

IBM 可以按它认为适当的任何方式使用或分发您所提供的任何信息而无须对您承担任何责任。

© Copyright IBM Corp. 2001, 2013

本程序的被许可方如果要了解有关程序的信息以达到如下目的: (i) 允许在独立创建的程序和其他程序（包括本程序）之间进行信息交换，以及 (ii) 允许对已经交换的信息进行相互使用，请与下列地址联系:

IBM Canada Limited
U59/3600
3600 Steeles Avenue East
Markham, Ontario L3R 9Z7
CANADA

只要遵守适当的条款和条件，包括某些情形下的一定数量的付费，都可获得这方面的信息。

本资料中描述的许可程序及其所有可用的许可资料均由 IBM 依据 IBM 客户协议、IBM 国际软件许可协议或任何同等协议中的条款提供。

此处包含的任何性能数据都是在受控环境中测得的。因此，在其他操作环境中获得的数据可能会有明显的不同。有些测量可能是在开发级的系统上进行的，因此不保证与一般可用系统上进行的测量结果相同。此外，有些测量是通过推算而估计的，实际结果可能会有差异。本文档的用户应当验证其特定环境的适用数据。

涉及非 IBM 产品的信息可从这些产品的供应商、其出版说明或其他可公开获得的资料中获取。IBM 没有对这些产品进行测试，也无法确认其性能的精确性、兼容性或任何其他关于非 IBM 产品的声明。有关非 IBM 产品性能的问题应当向这些产品的供应商提出。

所有关于 IBM 未来方向或意向的声明都可随时更改或收回，而不另行通知，它们仅仅表示了目标和意愿而已。

本信息可能包含在日常业务操作中使用的数据和报告的示例。为了尽可能完整地说明这些示例，示例中可能会包括个人、公司、品牌和产品的名称。所有这些名称都是虚构的，与实际商业企业所用的名称和地址的任何雷同纯属巧合。

版权许可:

本信息包括源语言形式的样本应用程序，这些样本说明不同操作平台上的编程方法。如果是为按照在编写样本程序的操作平台上的应用程序编程接口 (API) 进行应用程序的开发、使用、经销或分发，您可以任何形式对这些样本程序进行复制、修改、分发，而无须向 IBM 付费。这些示例并未在所有条件下作全面测试。因此，IBM 不能担保或暗示这些程序的可靠性、可维护性或功能。此样本程序“按现状”提供，且不附有任何种类的保证。对于使用此样本程序所引起的任何损坏，IBM 将不承担责任。

凡这些样本程序的每份拷贝或其任何部分或任何衍生产品，都必须包括如下版权声明:

© (*your company name*) (*year*). 此部分代码是根据 IBM 公司的样本程序衍生出来的。
© Copyright IBM Corp. *_enter the year or years_*. All rights reserved.

商标

IBM 商标: IBM、 IBM 徽标和 ibm.com 是 International Business Machines Corp., 在全球许多管辖区域注册的商标或注册商标。其他产品和服务名称可能是 IBM 或其他公

司的商标。当前的 IBM 商标列表，可从 Web 站点 www.ibm.com/legal/copytrade.shtml 上“版权和商标信息”部分获取。

下列各项是其他公司的商标或注册商标

- Linux 是 Linus Torvalds 在美国和/或其他国家或地区的注册商标。
- Java 和所有基于 Java 的商标和徽标是 Oracle 和/或其子公司的商标或注册商标。
- UNIX 是 The Open Group 在美国和其他国家或地区的注册商标。
- Intel、Intel 徽标、Intel Inside、Intel Inside 徽标、Celeron、Intel SpeedStep、Itanium 和 Pentium 是 Intel Corporation 或其子公司在美国和其他国家或地区的商标或注册商标。
- Microsoft、Windows、Windows NT 和 Windows 徽标是 Microsoft Corporation 在美国和/或其他国家或地区的商标。

其他公司、产品或服务名称可能是其他公司的商标或服务标记。

索引

[B]

[C]

© Copyright IBM Corp. 2001, 2013

[D]

[F]

[G]

[H]

[N]

[P]

[Q]

[R]

[S]

[T]

A

B

C

D

H

I

L

M

N

O

R

S

T

V

W

Printed in China

S151-1755-01

Spine information:

IBM DB2 10.1 for Linux, UNIX, and Windows

数据恢复及高可用性指南与参考