

第24章 备份与恢复

本章要点：

- 备份策略

- 理解物理数据丢失与逻辑数据丢失

- 使用逻辑备份

- 使用冷物理备份

- 使用热物理备份

- 使用恢复管理器进行物理备份

- 从逻辑备份中恢复

- 使用物理恢复

- 测试策略

- 问题解决策略

- 项目：被破坏的归档日志的不同蕴涵

24.1 备份策略

正如大部分人已经知道的那样，数据备份几乎是任何计算机系统绝对必需的组成部分。磁盘头毁损、电路失效甚至数据中心的灾难性丢失都是现实情况，数据库管理员必须对此有所准备。所幸的是，Oracle是一个非常坚固的数据库系统，经过多年实践检验过的备份与恢复技术能够在不可避免的事情发生时保证你的数据安全。作为一名专业的数据库管理员，你必需采取措施保证你的RDBMS中包含的数据免遭各种类型的破坏。

假设在年底处理前的两周，你的数据库系统丢失了一个磁盘驱动器。CEO与CFO都是你办公室中的成员。他们想要知道系统备份需要花费多长时间并且还提醒你不能丢失财务系统中的任何数据。本章将给出你将要告诉他们的信息：“系统备份不会花费很长时间”以及“当然，不会丢失任何数据”。你还将学到一些备份系统的常见非应急用法。

提示 告诉你的用户与管理人员备份能够做什么和不能够做什么以及它一般要花费多长时间恢复系统。你有可能使所有人的希望成为现实。

还要指出的是，备份既可以在应急情况下用于数据恢复，也可以在受人工操纵的情况下用于数据恢复。在受人工操纵的情况下，你可以把数据从一个实例简单地移动到另一个实例、交换机器或重新组织一个数据库。

本章的重点是帮助你确定、设计并测试你的备份策略。恢复你的数据库的难度与实施备份策略直接有关。假如你对恢复环境不满意，你应当利用Oracle技术支持满足你的具体恢复要求，因为具体的细节根据不同的情况而变化。当你设计、实现并使用你的备份策略时，请记住如下这些要点。

- 设计你的备份与恢复策略。设计如何使用你的备份以便恢复你的数据库。

- 定期测试你的备份与恢复策略（对于新的维护成员特别重要）。

确保操作系统级的备份仍然脱离数据库服务器。数据库备份不能保护 init.ora 文件、Oracle 设备或操作系统。

在重要的修改以前或以后执行适当的数据库备份。假如此修改包括删除段或表空间的话，执行适当的数据库备份就显得特别重要。

含有动态数据的表空间比多数静态表空间需要更为频繁地备份。

手头上保存较老的备份。有时几个月或几年也不会有表或一行已经被删除。

定期导出你的数据库进行附加的保护。

考虑适用于高利用率应用程序的分布式数据库备份。

24.2 理解物理数据丢失与逻辑数据丢失

数据丢失能够被分为物理丢失和逻辑丢失两类。物理数据丢失是指操作系统级的数据库组件的丢失，例如数据文件、控制文件、重做日志以及存档日志。引起物理数据丢失的原因可能是磁盘驱动器毁损，有人意外删除了一个数据文件或者是改写了任何关键的数据库文件所造成的配置变化。逻辑数据丢失是诸如表、索引或表记录的数据库级组件的丢失。引起逻辑数据丢失的原因可能是有人意外删除了不该删除的表、应用出错或一条 DELETE 语句中不适当的 WHERE 子句。毫无疑问，Oracle 能够实现物理数据备份与逻辑数据备份。尽管两种备份方案可以互相替代，但是你会发现有效的备份计划必须把两种方案都包含在内才能使你完全避免数据丢失。

物理数据备份是下面各项的拷贝：

- 数据文件。

- 控制文件。

- 可用的归档重做日志。

假如你丢失了一个磁盘驱动器或者需要来回移动一些数据文件，最直接的方法是恢复。物理备份通常按照预定的时间间隔运行以防止数据库的物理丢失。当然，如果你想保证能够把系统恢复到最后一次提交时的状态，你必须以物理备份为基础，同时你还必须有自上次物理备份以来累积的归档日志与重做日志。

物理备份可以进一步分为冷（cold）备份与热（hot）备份。当你有权关闭数据库足够长的时间以备份系统时可以使用冷备份。在数据库打开并可被用户使用时，热备份提供了一个物理备份。这使一个 Oracle 数据库能够 1 周 7 天、1 天 24 小时地运行，同时仍然可以定时得到备份。如果你不必使数据库不间断地运行，那么最好使用冷备份。正如你将看到的那样，在使用热备份以前，它必须得到充分测试。

请注意术语“冷备份”是“脱机备份（offline backup）”的同义词，就像“热备份”是“联机备份（online backup）”的同义词一样。

备份频率与 ARCHIVELOG 模式

计划备份的两项主要工作是为你的数据库选择适当的备份频率以及决定是否在 ARCHIVELOG 模式下运行数据库。

当选择备份间隔时，总是应问自己“我能承担得起多少数据的丢失？”假如你的数据库没有运行在 ARCHIVELOG 模式下，你会丢失自最后一次备份以来的所有事务。假如你的业务需求使数据库能忍受丢失一周的事务，你需要每周至少备份数据库一次。

在成品OLTP系统中你不能失去任何事务。非常希望数据库能够在 ARCHIVELOG模式下运行，因为即使你的最后一次物理备份发生在两天以前，它也能够把你的数据库完全恢复到最后一次提交时的状态。

即便在ARCHIVELOG模式下，你仍然必须对系统执行定期的物理备份。为了完全恢复一个数据库，你必须有自最后一次物理备份以来的所有归档重做日志。定期物理备份减少了用于存储归档重做日志的空间数量，同时也减少了由于归档重做日志的丢失或损坏而导致丢失事务的风险。

物理备份应用于相同的机器、相同的Oracle版本以及引起物理备份的实例上的恢复。因此，物理备份被认为是不可移植的。它们通常仅用于保护同一台机器和实例中的数据免遭数据丢失。

此规则的一个例外是你想要把一个数据库从一个系统完全转移到另一个系统。只要两种机器是相同的体系结构、相同的操作系统版本以及相同的 Oracle版本，你就能把数据库的物理成分从系统A拷贝到系统B。假如这些条件中的任何一个不满足或者你还不确定，那么就使用逻辑备份来移动你的数据。

逻辑数据备份通常是SQL语句的集合，这些SQL语句以一种ASCII格式重新创建数据库对象（数据库本身、表、索引、授权、角色等）和记录。

当你需要在实例之间移动指定的数据时，或者需要在不同的系统结构、操作系统版本或Oracle版本之间拷贝一个实例的全部数据时可以使用 Oracle的逻辑备份系统。尽管除了物理备份以外确实需要定期的逻辑备份，但是逻辑备份通常必须由数据库管理员执行。

提示 Oracle传统的备份工具通常不适宜用于归档数据库系统外的陈旧数据（特别是在数据能够被装入一个非Oracle系统的情况下）。来自BMC和Platinum厂商等的第三方工具为这些特殊情况的逻辑备份提供了很大的灵活性和可移植性。

在开发者环境中，应用开发者将与数据库管理员一同工作以设计一个逻辑备份策略。一般存在两个数据库拷贝，一个是成品数据库，另一个由开发者使用，用来调试他们的程序。开发数据库将与成品数据库定期地保持同步。

现在看一些实际问题。你必须保证你的数据库在 ARCHIVELOG模式下运行并且正在运行归档程序以便把你的系统恢复到最后一次提交时的水平。下面的几节中列举了常见的备份要求和推荐使用的备份方法。本章的余下部分讲述由 Oracle提供的备份与恢复方法的技术细节。

下面提供了几个例子对应说明了每一种备份系统的正确使用。

1. 物理备份

物理备份是所采用的最主要的备份方式。如果发生物理数据库丢失或崩溃，物理备份用于保证数据库在最小的数据丢失或没有数据丢失的情况下得到恢复。

2. 冷物理备份

冷物理备份提供了最简单和最直接的方法保护数据库免遭物理损坏或丢失。正确使用冷物理备份的示例如下：

数据库已经在一台数据库服务器上创建，你想按常规对数据库执行一个备份。数据库在备份过程中不必处于可用状态。

系统管理员将对你的磁盘进行升级，每张磁盘的容量从 4GB升为9GB。你需要备份系

统并在新驱动器中相同的目录结构下恢复它。

3. 热物理备份

热物理备份提供了相同的保护方法用于免遭物理数据库损坏。备份过程在数据库打开并且用户可以使用的环境下进行。需要执行热物理备份的情形如下：

你的业务需求要求你的用户群体可以 1周7天、1天24小时地使用数据库。

研究结果表明你的数据库需要一整天的处理以达到一个高缓存命中率。在系统中使用许多 GB 内存的特大型 Oracle 数据库中，往往需要花费几个小时的处理时间才能使 Oracle 缓存整个数据工作组。尽管系统能够在夜晚关闭，但是如果数据库不在夜晚关闭的话，其早晨的性能会更好。

尽管你的用户群体一天仅需要系统运行 12 小时，然而剩余的 12 小时几乎都花费在批处理上；没有足够的停机时间用于批处理与系统备份两方面上。

4. 逻辑备份

逻辑备份用于实现数据库对象的恢复并且它是一个全面备份策略的必要的组成部分，用来保证数据库能够从无意中的修改（例如 DELETE、DROP 或 UPDATE）中恢复出来。

5. 完全逻辑备份

完全逻辑备份把整个数据库导出到一个 Oracle 格式文件中，该文件可以在不同的 Oracle 版本、操作系统和硬件平台之间进行移植。如下特殊情况或许需要执行一个完全逻辑备份：

你没有另外的作为测试平台的物理系统，但是你有处于同一台机器上的成品与测试实例并且你希望它们不时地进行同步。

新服务器刚安装完并且将用来替换你的老服务器。你的老数据库服务器与新数据库服务器使用不同的平台（例如，Sun 与 HP）。

6. 指定表的逻辑备份

Oracle 的逻辑备份工具可完成指定表的逻辑备份。此功能可用于如下情形：

你需要把表 ABC 从 JSMITH 模式移动到 TGASPER 模式。

尽管你想保存表 ABC 的备份拷贝，然而你将要删除它。

7. 指定用户的逻辑备份

逻辑备份可由一个或多个指定用户（或模式）组成。当出现如下情形时，你或许想要使用此功能：

你刚被告知可以删除用户 JSMITH。该用户拥有几个表，而你认为某些人有可能以后会需要这些表，所以你想要备份 JSMITH 模式。

有一个应用，该应用使用的表完全包含在一个单独的模式中。你计划对此应用进行升级，必须运行一个脚本对表和索引进行“更新”以便新版本能够正常工作。

提示 这些具体的事例说明了不同的备份方法适合于在你的数据库工作中所遇到的不同情形。不要只依赖一种备份方法。尽管大多数数据库管理员知道需要物理备份的必要性，但仍有许多人不会定期运行逻辑备份。这使他们容易受到一条无意间使用的 DROP 命令的损害。记住，DROP 命令是立即执行的，没有回滚。在 Oracle 技术支持的帮助下，使用物理备份恢复一张已被删除的表是可能的。此恢复过程非常耗时间并且代价很大，该过程或许可以通过使用逻辑备份来避免。记住，系统不必为逻辑备份而关闭。

在你运行一个没有经过仔细检查的 SQL 脚本以前，执行逻辑备份是非常重要的事

情。因为DROP或DELETE命令或许出人意料地隐藏起来，等待时机对你的数据库“发动攻击”。

假如你把热备份作为你的物理备份的话，那么有机会时你还要运行冷备份。由于热备份的复杂性，使得许多数据库管理员（包括我自己）错误地认为他们已经有了一个相当近期的冷备份，这样的话，他们不能有足够的备份。

24.3 使用逻辑备份

通过EXP工具，Oracle为你提供了逻辑备份（IMP用于逻辑恢复）。你要么在完全批处理模式下完全交互式地使用EXP工具，要么用一些指定的关键字交互式地使用EXP工具。在研究EXP的操作特征以前，首先查看可用的关键字。运行 `exp HELP=Y` 就会显示所有的关键字，如清单24-1中所示。

清单24-1 EXP关键字

oreo:./815/bin\$ exp help=y			
Keyword	Description (Default)	Keyword	Description (Default)
USERID	username/password	FULL	export entire file (N)
BUFFER	size of data buffer	OWNER	list of owner usernames
FILE	output file (EXPDAT.DMP)	TABLES	list of table names
COMPRESS	import into one extent (Y)	RECORDLENGTH	length of IO record
GRANTS	export grants (Y)	INCTYPE	incremental export type
INDEXES	export indexes (Y)	RECORD	track incr. export (Y)
ROWS	export data rows (Y)	PARFILE	parameter filename
CONSTRAINTS	export constraints (Y)	CONSISTENT	cross-table consistency
LOG	log file of screen output	STATISTICS	analyze objects (ESTIMATE)
DIRECT	direct path (N)	TRIGGERS	export triggers (Y)
FEEDBACK	display progress every x rows (0)		
QUERY	select clause used to export a subset of a table		

清单24-1中的关键字在下文进行解释。

USERID是EXP使用的用户标识/口令，用于登录到数据库以便执行导出操作。要使用EXP，你至少必须具有CONNECT SESSION特权。假如你打算导出另一个用户的对象，你必须具有EXP_FULL_DATABASE特权。数据库管理员用户当然可以做任何事情。

提示 你可以把EXP用于使用普通USERID/PASSWORD@REMOTE_NAME约定的远程数据库。

FULL指定是否正在导出整个数据库。它的缺省值是 N。

BUFFER用来指定在提交到导出文件以前内存中将装入多少行。缺省值根据操作系统而定。一般说来，你的缓冲区越大，导出操作运行得也就越快（同时将使用更多的系统内存）。你可以用如下公式计算将要装入到缓冲区中的行数：

行数=缓冲区容量/最大的行容量

假如该表包含一个LONG列，那么每次仅能有一行装入缓冲区。

OWNER列出了你想要导出的模式。你可以使用此关键字导出完整模式。假如 ROWS=N，那么仅仅导出用来重新生成模式的SQL代码——没有表行被导出。

FILE是导出文件的名称。你应当使用一个合理描述所包含的数据的文件名。在有些操作系统（特别是UNIX操作系统）中，你可以把数据直接导出到一个磁带设备中。当你没有足够

的磁盘空间用于导出文件时，此操作对大型数据库的备份来说特别有用。

警告 当你使用FILE关键字指定一个文件名时，根据你正在导出的内容，这些文件可能相当大。在诸如UNIX的操作系统中，你可以把数据直接导出到磁带上，这样就消除了上述的问题。假如不能这样做，你就必须把导出的内容分解为足够小的片，然后要么装入几个不同的文件系统中，要么装入一个地方，以便在随后的 EXP运行中每一片能够在下一片创建以前移动到磁带中。你可能想在工作时间之外运行 EXP以减轻文件系统的压力。

TABLES列出了你想要导出的单个表。假如 ROWS=N，那么仅仅导出用来重新生成指定表的SQL代码，而不导出行。

当导入数据时，COMPRESS有许多含义。COMPRESS=Y将引发IMP构造一个新对象的初始区间，使之足以容纳所有对象的数据。例如，假如一张表有 4个区间，每一个区间的容量是 20MB，那么新表的初始区间近似为 80MB。假如COMPRESS=N，新表将有4个20MB的区间，就像初始时一样。

警告 当你试图重新组织一张表（减少区间的数量以提高性能）时，尽管COMPRESS=Y非常有用，然而在使用此选项以前，你需要做一些准备工作。假设你有一张名叫 ORDERS的表，该表有如下这些区间容量： 30MB、100MB、1000MB、1500MB、1500MB和1000MB。此表的总容量略微超过5GB。现在假设你的机器仅有2GB和4GB的磁盘驱动器。你的系统能够有的最大的区间是4GB，然而Oracle希望找到一个5GB的区域。那么，导入进程不会工作，因为 Oracle不能指定一个 5GB的区域。在使用 COMPRESS=Y以前，在你打算导出的每张表中使用如下的SQL语句：

```
SELECT SUM(BYTES) FROM DBA_EXTENTS WHERE SEGMENT_NAME=<table name>
AND OWNER=<table owner>
```

你可以使用此语句调节一个区间中的表。

RECORDLENGTH用于把导出文件移动到另一个操作系统上的 Oracle数据库中。假如你正在从系统 A移动到系统 B，你必须知道系统 B中的记录容量。此信息保存在由 Oracle提供的操作系统文件中。

GRANTS指示EXP是否导出与被导出的对象有关的授权。

INCTYPE指定COMPLETE（完全）、CUMULATIVE（累积）或INCREMENTAL（增量）备份。你必须使用一个具有 EXP_FULL_DATABASE的USEID，因为此选项仅在FULL=Y时才有用。在大多数OLTP产品环境中很少使用此能力。

注意 假如一张表已经改变了的话，CUMULATIVE或INCREMENTAL导出操作导出整张表。EXP不能只导出已变化的行。假如你选择此选项的话，确保定期地执行一个 COMPLETE导出操作以保持你的当前导出基线。

INDEXES指定要导出正在被导出的表中的索引。

RECORD指定Oracle是否记录 COMPLETE、CUMULATIVE或INCREMENTAL导出。这用于下面的情况中：

你通常在星期天执行一次 COMPLETE导出，从星期一至星期五执行 CUMULATIVE导出。星期二晚上你计划给测试系统重新装入成品系统的 COMPLETE导出。当你在星期二晚上导出

成品系统时，如果 RECORD=Y，那么星期三的 CUMULATIVE 导出是基于星期二晚上的 COMPLETE 导出而不是星期天的 COMPLETE 导出。假如你正在按与通常排定的导出顺序不一致的情况下在数据库上运行导出的话，一般使用 RECORD=N。

ROWS 指示是否要导出你正在导出的表中所包含的数据。假如 ROWS=N，仅仅导出用于重新创建该表的 SQL 代码。

PARFILE 是一个参数文件的文件名，该参数文件包含用于 EXP 的任何有效的关键字设置值。下面是一个参数文件的示例：

```
TABLES=CUSTOMER,VEDOR  
OWNER=SUSAN  
COMPRESS=N  
RECORD=N
```

正如你看到的那样，参数文件没什么难理解的。它仅在每行上有一个关键字和数值。假如你有一个需要定期运行的标准导出的话，参数文件就非常有用，因为你不必担心忘记设置关键字。

CONSTRAINTS 表明是否导出与你正在导出的对象有关的约束。

假如你正在导出同时也正在被更新的表时，CONSISTENT 是有用的。CONSISTENT=Y 保证导出中的每张表在任何时刻都是一致的。假如 EXP 在导出一张表时遇到了一个已更新的行，那么它使用回滚信息并导出老版本。假如遇到插入的一个新行，EXP 将忽略它。只有在需要时才使用 CONSISTENT=Y，因为 EXP 在整个导出期间保存了一个回滚段。因为这在你的数据库上加了额外负载，所以该选项最好在业余时间使用。

假如你遇到一个快照太旧错误，表明你或许需要在系统负载较轻期间运行你的导出。你甚至需要在 RESTRICT 模式下打开数据库的情况下运行导出以使用户在导出期间不对数据库进行更新。假如你需要几个表的一致性视图，你或许也要在 RESTRICT 模式下执行导出操作。CONSISTENT=Y 仅在每张表上提供读取一致性。

LOG 是用来保存导出信息和错误信息的文件的文件名。通常，EXP 仅在你的屏幕上显示此信息。然而，因为此信息显示的速度太快以致于你来不及阅读，所以你可能想要把 EXP 日志输出保存到文件中。所有的信息和错误显示在你的屏幕上，甚至当你指定 LOG 文件名时也是这样。

提示 假如你正在使用一个 UNIX 操作系统，如果你不愿看到信息在你的监视器上飞滚，你可以把 EXP 屏幕输出重新定向到 /dev/null。假如 EXP 正在显示大量信息的话，把输出重新定向到 /dev/null 通常会使运行速度更快。这是因为对于 UNIX 操作系统来说，把信息简单地抛弃比把信息发送到屏幕上更快。

STATISTICS 指定当导出文件被导入时如何产生统计值。选项有 COMPUTE、ESTIMATE 或 NONE。假如你的导出文件较大，为了节省时间，把 STATISTICS 设置为 ESTIMATE 或 NONE（NONE 选项最节省时间）。你总是可以以后使用 ANALYZE TABLE 命令产生统计值。

DIRECT 用于增强 EXP。通过直接读取数据块（而不是通过普通的数据库引擎），DIRECT 极大地提高了导出性能。如果你有此选项的话，你通常应该使用它。你的导出操作会运行得更快。

TRIGGERS 指示触发器 PL/SQL 代码是否同表一起导出。缺省值为 Y，该缺省值适用于大多数用途。

FEEDBACK是EXP中新增加的。假如你设置 FEEDBACK=n (n是某个正整数), EXP为它导出的每n行显示一个点 (.)。设置FEEDBACK=0禁用此功能。

QUERY是Oracle 8i中新引入的。它使你能够仅导出一个基于 SELECT子句的表的子集。使用Oracle工具参考手册查阅有关这个新关键字的完整描述。

你可以在命令行上以任何顺序键入你想要的关键字。使用如下语法：

```
exp [<keyword>=<value>[,<value>,...] [<keyword>=<value>[,<value>,...] ...]]
```

我一般更喜欢在批处理模式下使用 EXP, 因为我能够在不修改或轻微修改该命令的情况下随意地重复使用它。假如 EXP所需的信息比你在命令行上所提供的信息要多的话, 例如用户标识/口令, 它提示你键入该信息。要完全交互式地运行没有关键字的 EXP, 只要在系统的命令提示符下键入 “ exp ” 即可。EXP提示你它所需要的全部信息。

24.3.1 完全逻辑备份

EXP特别适用于完全逻辑数据库备份。唯一的要求是你必须有 EXP_FULL_DATABASE特权以及必须有足够的存储空间用于存储导出文件。这里有一个写入缺省输出文件中的完全逻辑备份的例子：

```
exp USERID=SYSTEM/MANAGER FULL=Y
```

下面的例子显示了一个完全逻辑备份, 该备份直接保存到 UNIX系统中的磁带上：

```
exp USERID=SYSTEM/MANAGER FULL=Y FILE=/dev/rmt0
```

假如一个对象 (用户、表、索引等) 被无意中删除、破坏或修改的话, 那么新近执行的完全逻辑备份会给数据库管理员一个补救的机会。尽管不能把你所要的对象带回它被删除前的状态, 然而你至少有了得到恢复的对象。

24.3.2 指定用户模式的逻辑备份

EXP很容易地支持指定用户模式的备份。导出一个指定的用户组缺省情况下会导出用户定义、所有的表、索引、约束、授权以及相关的数据。如果必须改变用户的 USERID, 或者一个用户将要被删除, 而你仅希望在手头上保存一个拷贝的话, 这就非常有用。你可以要么指定一个用户, 要么指定一组用户。这里是一个用于把 TGASPER模式导出到标准输出文件中的命令行示例：

```
exp USERID=SYSTEM/MANAGER OWNER=TGASPER
```

现在假定你正在清理老用户帐户, 打算删除 JSMITH、JKLEIN、MJONES和BGEORGE帐户。作为一名谨慎的数据库管理员, 你决不会真正删除任何东西。你需要为任何情况做准备, 例如有人下个月来找你并且说: “ 我现在需要 JKLEIN帐户的 XYZ表。我知道我们已要求你删除了那个模式, 但是假如我们不能得到那张表的数据, 我们会处于困境中。” 下面这条命令用于将多个用户模式导出到一个 UNIX磁带设备中：

```
exp USERID=SYSTEM/MANAGER OWNER=JSMITH,JKLEIN,MJONES,BGEORGE FILE=/dev/rmt0
```

现在你已准备好了。

提示 无论何时, 只要你把将要删除的信息导出到磁带中, 在从数据库中实际删除该信息以前, 你或许希望测试导出磁带。

24.3.3 指定表的逻辑备份

导出一组表的处理方法与用户导出的方法一样。通过使用 TABLES关键字列举表的名字，你可以选择导出一个表或一组表。下面的例子把 ABC表与XYZ表从BGEORGE模式中导出到缺省输出文件中：

```
exp USERID=SYSTEM/NANAGER OWNER=BGEORGE TABLES=ABC,XYZ
```

举例来说，假如你需要在相同的或不同的 Oracle实例中的模式之间移动指定的表，对指定的表进行备份很有用。在从系统中删除一个表以前，你或许还想保存它的逻辑备份。

提示 不管算是经验还是偏执，在没有首先生成一个逻辑备份以前，我总是不愿意真正删除任何东西。你的环境有可能需要也有可能不需要此项措施，但是要注意恢复一个已删除的对象很难且非常花费时间。

注意 EXP缺乏一些主要的功能，许多熟悉其他RDBMS的数据库管理员或许对此感到惊奇。EXP使你不能导出一条基于SQL WHERE子句的信息。它仅使你能够导出全部对象、全部模式或整个数据库。Oracle的导出文件实际上是一种专有格式。你不能存取原始文本数据或用于重新生成数据库、表、授权等的 SQL代码。市场上有许多第三方的工具提供了这些缺少的功能。

24.4 使用冷物理备份

冷物理备份是数据库管理员用来保护数据库系统整体上免遭各种类型数据破坏的工具。定期的冷备份以及一组好的归档重做日志与当前重做日志使你能够把数据库恢复到任何一个时间点上——甚至是恰好在磁盘驱动器失败前的最后一次提交。

在Windows NT桌面环境出现以前，几乎完全使用诸如 tar (UNIX) 的操作系统工具执行冷物理备份。唯一的数据库交互作用是关闭数据库并在以后重新启动它。在 UNIX中仍然是这样。因为使用Windows NT的计算机不是面向命令行的，所以 Oracle创建了一个叫做备份管理器的产品，备份管理器使你能够使用一个标准的 Windows NT图形用户界面程序执行物理备份与恢复。从Oracle 8开始，Oracle引入了一个称为恢复管理器（或简称为 RWAN）的新的备份与恢复系统。该系统尽管有点复杂，但是可以满足所有的备份 /恢复要求。

注意 不要把此恢复管理器与Oracle的NT恢复管理器混淆；名字非常相似，但是它们完全是不同的产品。

每种备份系统各有一组不同的特性，尽管最终执行相同的功能，但是方法不同。命令行和桌面备份方法在 24.4节和24.5节中各有一小节。因为RWAN被设计为更奇特的物理备份工具，所以它的使用在稍后的 24.6一节中讨论。

不管你的环境或方法是怎样的，冷物理备份将包含如下数据库元素：

- 控制文件。
- 数据文件。
- 重做日志文件。
- 归档日志文件。

24.4.1 命令行驱动冷物理备份

UNIX系统数据库管理员或许会有被 Oracle抛弃的感觉，因为 Oracle缺乏用于管理诸如备份或恢复操作的图形用户界面工具。但是假如你做一些完善某些脚本的工作的话，那么在命令行中执行的备份和恢复将比通过恢复管理器执行的备份和恢复更加灵活且不会出错。

冷物理备份一般涉及生成所有必要的数据库元素（配置文件、数据文件、控制文件、重做日志和归档日志）的一个文件级拷贝。在备份过程开始以前，你需要确保你的数据库能够足够长时间地停留在脱机状态以便得到完全备份。假如你没有足够的时间，应考虑使用热备份。

执行一个冷备份所需要的基本步骤如下：

- 1) 建立一个你需要备份的操作系统级文件的列表。
- 2) 选择使用NORMAL或IMMEDIATE方式关闭数据库。
- 3) 执行步骤1中文件列表的操作系统级备份。
- 4) 正常启动数据库。

警告 在系统突然关闭后不要对它进行备份。仅在使用IMMEDIATE或NORMAL模式关闭数据库后才对它进行备份。

此外，当你已经备份了你的归档日志文件后，你可能会从系统中删除它们（但要确保在某个地方长期保存归档日志文件）。我通常喜欢保存系统中最近七天的归档日志。假设你每天晚上备份你的数据库，即便你的备份已经在最近的这七天失效了，此项操作仍然可以保证你能够恢复该数据库。设想一下在感恩节的周末，大部分公司从星期四到星期天关门。假如在星期三你的备份脚本由于某个原因而失效了，有可能直到下星期一早晨你才发现某些东西失效了。即使你有一组一直轮流值班的操作人员，他们或许也不能像你一样密切地监控备份。通过保存系统中最近七天的归档日志文件，即便数据库在星期一下午的某个时间失效了，你仍然可以恢复它。

提示 对于具有关键使命的关键数据库来说，至少在整个ANALYZES或者数据库导出期间保存归档日志是很重要的。其原因是假如出现了一个坏的数据块，你可以把数据文件恢复到一个较为合适的时刻（当你的最后一次数据库分析或导出表明它是完全有效时）并且使用归档日志把系统前滚到最近的提交。在你已经正确运行了一个针对于数据库中所有表的ANALYZE TABLE VALIDATE STRUCTURE语句后，你可以认为你的数据库摆脱了任何坏数据块并且较老的归档日志不必从坏数据块中恢复。

1. 建立你的文件列表

不管数据库管理员尽了多大努力，他们看起来总是不能对组成数据库的全套文件保持一个固定的处理方法。这很不幸，因为如果冷备份不是数据库文件的完整集合的话，那么它几乎毫无价值。在你关闭数据库以前，或许很值得对每样东西多次检查一下以确保你有需要备份的文件的完整列表。

当数据库启动时，你要么明确地要么隐含地读取一个init<instance>.ora文件。你必须备份此文件。你还要查看一下此文件是否有一个ifile行，在读取init.ora文件时，该行基本上通知Oracle包含另一个文件。假如有一个使用ifile关键字列出的文件名，你必须备份那个文件。

要生成由数据库使用的控制文件的列表，你可以查看可用的init.ora文件或者运行下面的

SQL语句：

```
SELECT VALUE FROM v$PARAMETER WHERE NAME='control_files';
```

Oracle返回控制文件的一个逗号分隔的列表。

可以使用如下SQL语句获得所有数据库数据文件的列表：

```
SELECT FILENAME FROM DBA_DATA_FILES;
```

Oracle返回一个列举了用于数据库中的表空间的所有操作系统级文件名的查询结果。

假如数据库被正常关闭的话，重做日志不包含 Oracle所需的信息。从技术上讲，它们不需要被备份——但是你仍然必须在恢复期间创建新的重做日志，所以无论如何你或许要备份它们。如下的SQL语句生成了重做日志文件的清单：

```
SELECT MEMBER FROM V$LOGFILE;
```

归档日志一般可由归档进程自动写到一个目录中。init.ora文件中的log_archive_dest关键字被设置为目的目录，归档日志将被写入这里。你也可以使用如下 SQL语句获得相同的信息：

```
SELECT VALUE FROM V$PARAMETER WHERE NAME='log_archive_dest';
```

提示 在一个大型数据库中，追踪单个的文件非常消耗时间并且容易出错。你对上述两个问题的最好解决方法是采用一个好的文件组织结构。可能的话，使用 Oracle的OFA（光学灵活结构）准则以便容易地维护文件结构。

使用OFA结构，你能够快速地区别需要得到备份的数据库的所有关键组件。OFA还能够帮助数据库管理员确信控制文件、重做日志和其他冗余的数据库组件被映射到不止一个物理设备中。

2. 关闭Oracle

这里你不需要做任何特别的事情，仅仅正常关闭数据库即可。不要使用 SHUTDOWN ABORT，可以使用SHUTDOWN NORMAL，或者SHUTDOWN IMMEDIATE关闭数据库。

3. 执行备份

你准备使用手头现有的或从步骤1中得到的文件列表对数据库进行备份。使用你的操作系统级选择工具，你需要以任何顺序把文件备份到一个备份设备中（通常是磁带）。

对于UNIX操作系统而言，选择工具趋向于使用 tar、cpio或vdump，而tar是最通用的。按照正常步骤启动Oracle数据库。

提示 尽管冷备份消耗时间，然而你可以使用几个选项来减少数据库必须关闭的时间。

假如你有大量的磁盘空间，可以仅在磁盘中的一个临时区中执行配置文件、数据文件、控制文件、重做日志以及归档日志的文件拷贝。然后你可以再次启动你的数据库并开始把临时区备份到磁带上。你至少需要两倍数量的磁盘空间用于该数据库，它会大量减少备份所需的时间。

考虑把多个磁带设备增加到你的系统中。在多操作系统下，你可以同时运行备份程序的多个实例。这使你能够同时备份数据库的几个不同区域。实际上，四个磁带驱动器备份一个数据库所花费的时间通常是一个磁带驱动器备份该数据库所花费时间的三分之一。

24.4.2 桌面驱动冷备份

仅当图形用户界面已经简化了字处理方法、电子表格、电子邮件、甚至是系统管理时，

它才把许多有用的功能提供给数据库管理员。Oracle的备份管理器服务器作为一个工具用于Oracle数据库的备份。原来由脚本和大量基于操作系统的工具完成的工作现在可以由 Oracle提供并支持的一个程序来完成。图 24-1显示了一个通用但很简单的备份管理器界面。

备份管理器的界面相当容易使用。请注意你的备份管理器的窗口或许与你在图 24-1中所看到的有点不同。该窗口或许仅为你提供了可用于数据库状态的选项（正在运行 / 没有运行，归档/非归档日志模式）。

你需要在此工具外完成的唯一工作是确定将你的脱机备份存储在哪里。备份管理器使你既可以将脱机备份存储到磁盘上的一个目录中也可以存储到一个磁带驱动器中。注意在图24-1的左下角，备份管理器确切地告诉你备份该数据库所需花费的空间。为了使备份操作正常工作，你需要一个目录或足够容量的磁带设备。

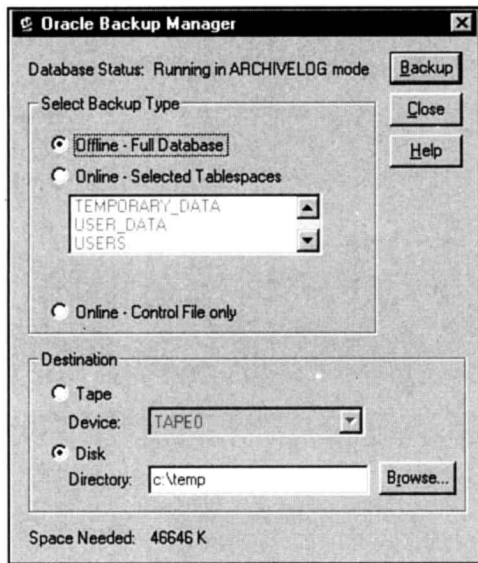


图24-1 用于Windows NT的Oracle备份管理器有一个简单的图形用户界面

执行一个冷物理备份包含如下步骤：

- 1) 确保数据库已经启动并且运行正常（参见随后的警告）。
- 2) 打开备份管理器。
- 3) 从Select Backup Type框中选中Offline-Full Database选项。
- 4) 通过在Destination框中选中Tape或Disk把数据备份到磁盘或磁带中。假如你把数据库备份到磁带中，你还需要指定使用哪一个磁带设备。另外，你需要指定你想把数据备份到哪一个目录中。
- 5) 单击Backup按钮。备份管理器完成剩余的工作。
- 6) 重新启动数据库。

警告 不要把数据备份到你的数据库所驻留的物理设备中。

警告 因为通常在数据库脱机时执行冷备份，所以在步骤1中确保数据库正在运行看起来有点奇怪。这是因为备份管理器需要在由于备份而关闭数据库以前对该数据库进行访问以便它能建立一个需要备份的数据库文件列表。在没有数据库运行的情况下启动备份管理器可能产生一个不完全备份集。

提示 假如存在打开的会话（包括服务器管理器），备份管理器不会执行脱机备份。假如你很难防止会话在Oracle中打开，你可以在启动备份管理器以前在RESTRICT模式下打开数据库。不要忘记关闭你的服务器管理器会话！

备份管理器不备份init.ora、config.ora或归档日志文件。尽管这些文件一般会在普通的操作系统备份过程中得到备份，然而你应与系统管理员一起仔细检查以确保这些非常重要的文件定期得到备份。

24.5 使用热物理备份

数据库管理员受委托维护的数据库经常是公司的命根子。这些数据库有时一周 7天、一天 24小时地用于访问。在这种情况下，只有具有工业强度的 RDBMS才有充分的能力。Oracle就是这样一种数据库，这是已经经过实践证明的。

热备份要求你对 Oracle和支持 Oracle的操作系统具有较高的专业知识以及适应水平。由于热备份的复杂性，你还必须投入更多的时间和精力用来测试你的备份策略。只有当你从一个测试系统中的大量失败中恢复出来时，热备份系统才被认为是可靠的。在开始实施一个热备份方案以前，确信值得为增加可用的时间而增加成本与令人头疼的事。

在继续热备份以前，你应当保证熟悉冷备份。你要依赖在以前的小节中所讨论过的冷备份。特别是，本节利用了你的冷备份知识并且讨论数据库在启动和运行时对它进行备份的附加的复杂性。Oracle提供了两种热备份方法，一种用于命令行 UNIX环境，另一种用于桌面 (Windows) 环境。由于此原因，下面将分别讨论每种热备份方法。

24.5.1 理解复杂性

系统管理员与数据库管理员经常提出的一个问题是“热备份为何如此重要？我们备份有用户的操作系统不会有任何问题，所以我们也可以在系统中有用户时备份 Oracle数据文件。对吗？”回答是，错。要阐述热备份的复杂性需要简要地讨论一下 Oracle如何存储与修改信息。

回忆一下，Oracle的基本存储单元是数据块（可以在创建数据库时配置它的容量）。数据块中存储数据，数据块头存储包含在块中的数据的的信息。

进行正常的冷备份时，每一个数据文件被逐位地拷贝到备份设备中。假如数据库处于联机状态时，麻烦的情况是备份程序和 Oracle都工作于同一个块上。假定当 Oracle写新的数据和

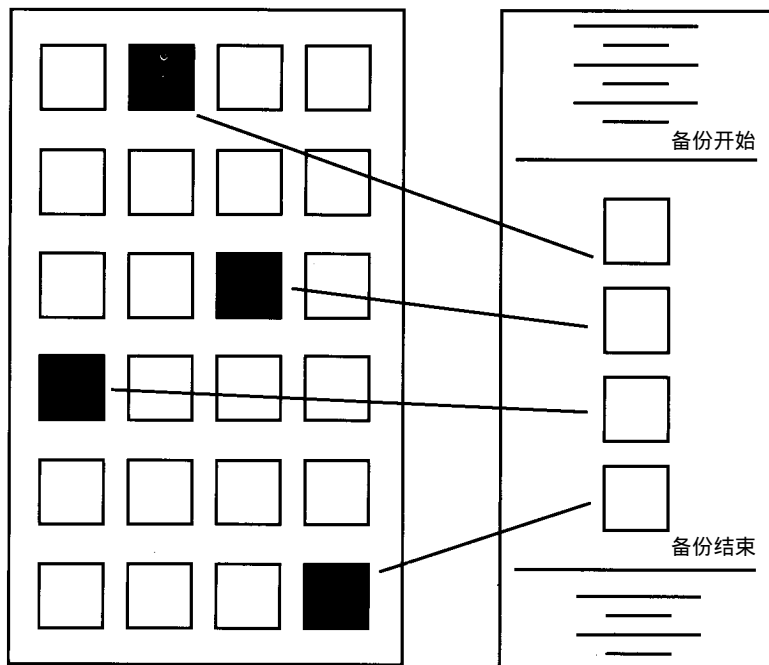


图24-2 Oracle的备份模式在恢复期间对可能受到破坏的数据块进行修补

更新数据块头时，备份程序仅完成了对指定块的块头部分的备份。在备份程序对指定的数据块完成备份之后，该块具有老的块头和新的数据。这个不纯的数据块使 Oracle 不知道如何对它进行处理。

Oracle 的解决方案是实现一种称为备份模式的特别模式，该模式在各个表空间上依次被启用或禁用。当启用该模式并且 Oracle 更新一个数据块时，已修改块的完整拷贝被写入到重做日志中。这与 Oracle 的普通过程仅把修改写入重做日志形成对比。

Oracle 数据文件的联机备份仍然存在不纯的数据块，但有效的拷贝存在于重做日志中（最终会在归档日志中）。在恢复状态下，数据文件得到恢复并且 Oracle 使用重做/归档日志来用有效块“修补”数据文件。这样，最终的结果是一个有效且一致的数据库。在图 24-2 中对此系统进行了举例说明。

24.5.2 命令行驱动热物理备份

通过把 SQL 与操作系统脚本结合使用实现基于命令行的 UNIX 环境下的热备份。热备份的具体细节在不同的操作系统下会有所变化，下面是热备份的操作顺序：

- 1) 建立一张需要备份的表空间的列表。
- 2) 对于该列表中的每一个表空间，执行如下操作：
 - a. 建立一个数据文件列表。
 - b. 让表空间处于备份模式下。
 - c. 把数据文件列表中的数据文件备份到一个备份设备中。
 - d. 使该表空间脱离备份模式。
- 3) 使用 Oracle 生成一个“可备份的”控制文件。
- 4) 把步骤 3 中的控制文件备份到一个备份设备中。
- 5) 强制执行一个日志交换。
- 6) 等待重做日志得到归档。
- 7) 把归档日志目录备份到一个备份设备中。

1. 建立一个表空间列表

如下的示例 SQL 语句在一个 Oracle 数据库中产生表空间列表：

```
SELECT TABLESPACE_NAME FROM DBA_TABLESPACES;
```

2. 为表空间建立一个数据文件列表

你需要得到表空间的数据文件列表以便确切地知道哪一个文件要备份：

```
SELECT FILE_NAME FROM DBA_DATA_FILES WHERE TABLESPACE_NAME='<tablespace name>';
```

3. 让表空间处于备份模式下

在备份表空间的数据文件以前，需要使用如下的 SQL 语句让表空间处于备份模式下：

```
ALTER TABLESPACE <tablespace name> BEGIN BACKUP;
```

4. 把数据文件备份到备份设备中

这是一个较大的操作系统问题。你知道使用 SQL*Plus 为每个表空间生成一张需要备份的文件列表。本书附带的 CD-ROM 包含一个用于 UNIX 的工作脚本，它举例说明了此过程。

5. 从备份模式中取出表空间

就像你使用 SQL 语句让表空间处于备份模式中一样，应使用一条类似的语句使它脱离备

份模式：

```
ALTER TABLESPACE <tablespace name> END BACKUP;
```

6. 生成一个“可备份的控制文件”

因为有一个Oracle实例正在运行，这必然意味着控制文件是打开的并且可能在任何时间更新，所以你需要Oracle产生一个保证是一致的且不会被更新的控制文件拷贝。如下 SQL语句将完成此任务：

```
ALTER DATABASE BACKUP CONTROL FILE TO <output filename> REUSE;
```

其中，<output filename>是任何有效的文件名，它指定了你希望写入控制文件拷贝的地方。REUSE意指如果该文件已经存在的话，它将被改写而不是放弃。

7. 强迫执行一个日志交换

当一个表空间上的备份开始和结束时，Oracle在重做日志中做标记。因为你没有备份重做日志，所以你需要归档当前的重做日志（该重做日志内含你刚刚备份的表空间的“备份结束”标记符）。当你备份归档日志目录时，你还要为每个表空间备份带有“开始备份”和“结束备份”的日志。日志交换用如下语句实现：

```
ALTER SYSTEM SWITCH LOGFILE;
```

8. 等待对交换重做日志进行归档

通常当一个日志交换发生时，Oracle的归档进程立刻把当前重做日志拷贝到归档日志目录。这个过程要花费时间并且Oracle并不通知你重做日志何时成功归档。一般来说，在备份脚本中应安排一个两到五分钟的固定延迟时间，在大多数情况下此时间足够了。然而，假如你的重做日志特别大或者日志归档到一个速度较慢的介质中，你应当确保Oracle能够得到足够的延迟时间用于归档一个重做日志。

9. 把归档日志目录备份到一个备份设备中

就像你在冷备份中所做的那样，你需要把包含归档日志的目录备份到一个备份设备中。

提示 因为Oracle对于每个写操作把一个完整的数据块写入到重做日志中，而不是写入改变了的信息中，因此当表空间处于备份模式时，重做日志填充得更快。你应当让你的数据库中的活动最少时进行你的联机备份。尤其要注意已调度的批处理作业。它们趋向于进行非常集中的写操作并且容易被忽略，因为它们经常自动地由调度程序运行。

24.5.3 桌面驱动热物理备份

Windows NT环境下的Oracle数据库管理员会发现热备份比命令行备份更容易。你将使用Oracle的备份管理器执行实际的备份操作。当你在数据库正在运行时启动备份管理器时，一个类似于图24-3的窗口会显示出来。

你需要决定把你的数据库备份存放在哪里。你要么把它备份到磁盘上的一个目录中，要么备份到一个磁带设备中。备份管理器能够指出用你选择的备份方式备份数据库时所需要的空间容量。此信息有助于你决定把数据库备份到哪里。采用如下这些步骤以执行数据库的联机备份：

- 1) 确保你的Oracle数据库运行正常。
- 2) 启动备份管理器。

3) 在 Select Backup Type 框中选中 Online-Selected Tablespaces 选项。

4) 选择你希望备份的表空间。对于一个完全备份来说,你必须选择所有的表空间。在你完成此项操作后,对已选择的表空间进行备份所需要的空间数量显示在窗口的左下角。

5) 在 Destination 框中,选择备份到磁带还是磁盘中。假如你选择备份到磁盘中,还应提供了用于存储数据库备份的目录名称。

6) 单击 Backup 按钮。

7) 当备份结束后,在 Select Backup Type 框中选中 Online-Control File only 选项。

8) 在 Destination 框中,选中 Disk 选项并提供你希望存储控制文件备份的目录名称。尽管你可以备份到磁带中,然而,通常不值得把小容量的控制文件存储到磁带中。

在此过程结束后,你的数据库已经成功地得到备份。备份管理器已经完成了对你的数据库的备份并且已经使所有的表空间脱离备份模式。

警告 备份管理器不能备份用于启动数据库的 init.ora、config.ora 或归档日志文件。尽管这些文件一般在正常的操作系统备份过程中得到备份,然而你应该和系统管理员仔细检查一下以确保这些非常重要的文件定期得到备份。

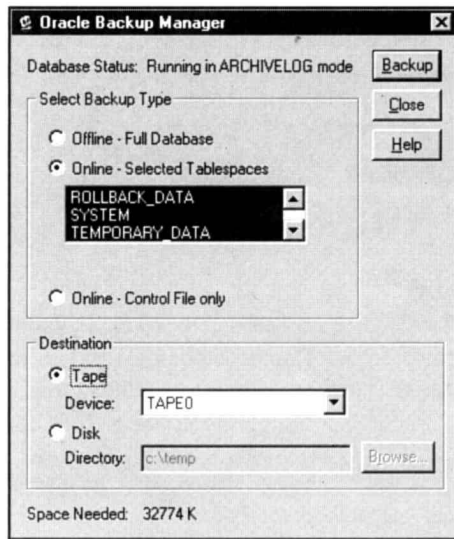


图24-3 备份管理器能够用来执行热备份

24.6 使用恢复管理器进行物理备份

Oracle8中引入的恢复管理器(RMAN)很有可能开创了一个全新的方法用于执行 Oracle 数据库的备份。它提供了从命令行与 GUI 用户界面执行的所有热备份和冷备份功能。恢复管理器同磁盘和磁带备份介质一同工作并提供了如下优于传统的基于操作系统的物理备份方法的特点:

热备份不会导致与传统的 BEGIN/END 备份方法有关的重做日志产生率。

作为备份过程的一个集成部分,检查数据库块是否残缺不全。这可以减少对整个数据库频繁使用 ANALYZE TABLE VALIDATE STRUCTURE 语句的要求。

支持增量物理备份。

支持多线程备份(仅限企业版)。

恢复管理器提供一个集成的分类系统,该分类系统在鉴别所需的备份磁带时能够把混乱减到最小。

恢复管理器明显提供了许多优于传统的备份方法的特点。然而,许多用户由于下述原因而不使用恢复管理器:

尽管 Oracle 已经花费很大力气使恢复管理器尽可能地简明易懂,然而在处理简单的备份要求时,恢复管理器必须提供的灵活度仍然使它有点不方便。

作为一种较新的产品,它有丢失重要数据的潜在可能性。在完全依赖它以前,有些单

位可能只想仅在更有限和不太关键的系统中使用它。

没有经过广泛的测试，管理员不愿改变备份方法；那些拥有经过严格编写的备份脚本的单位或许不愿接受一个新的、相对比较年轻的备份系统；等到该系统的优点超过了初始实现的成本时，这种情况也许会有变化。

对恢复管理器的使用介绍几乎可以占用一本书。在本节中，将向你介绍一个基本的完全系统备份过程。要想得到进一步的信息，参阅 Oracle 8 备份与恢复向导中的“ Recovery Manager Concepts ”一章。所示的例子将使用恢复管理器的命令行版本。图形用户界面版本可通过企业管理器的备份管理器软件得到。

在使用恢复管理器以前，必须在将要得到备份的数据库以外的数据库中建立一个恢复目录（Catalog）。在不同的数据库（有可能是一个不同的节点）中建立恢复目录的原因是，确保当编目的数据库不存在（这种情况在恢复介质失效时发生）时目录是可用的。假如你不止有一个数据库节点（例如 Mars 和 Venus），你可能使用 Mars 的数据库保存 Venus 的目录，或者使用 Venus 的数据库保存 Mars 的目录。

警告 尽管恢复管理器不要求使用目录，然而建议使用目录。

恢复管理器一般需要发出两个命令行参数（参见本节中说明参数使用的恢复管理器会话示例）；

TARGET 指定一个用户标识 / 口令 @ 数据库名，它在将要使用恢复管理器的数据库上具有管理权限。

RCVCAT 指定一个用于存储目标数据库的恢复目录的用户标识 / 口令 @ 数据库名。这应当是在一个与目标数据库不同的数据库上（可能也不是同一主机）。

第一次使用恢复管理器需要采用如下步骤：

- 1) 选择一个存放恢复目录的数据库。
- 2) 在步骤 1 中选取的数据库上用你选好的口令创建一个用户恢复管理器。把连接和资源授予恢复管理器。确保选择合适的缺省表空间与临时表空间。
- 3) 作为恢复管理器注册到目录数据库并运行和 CATALOG.SQL 与 CATPROC.SQL 处于同一目录中的 CATRMAN.SQL 脚本。
- 4) 你必须像如下例子所显示的那样对要在目录中备份的数据库进行注册。

```
[tgasper@oreo tgasper]$ rman target=sys/manager rcvcat=rman/rman@remote_db
```

```
Recovery Manager: Release 8.0.5.0.0 - Production
```

```
RMAN-06005: connected to target database: TST8
```

```
RMAN-06008: connected to recovery catalog database
```

```
RMAN> register database;
```

```
RMAN-03022: compiling command: register
```

```
RMAN-03023: executing command: register
```

```
RMAN-08006: database registered in recovery catalog
```

```
RMAN-03023: executing command: full resync
```

```
RMAN-08029: snapshot controlfile name set to default value: ?/dbs/snapcf_@.f
```

```
RMAN-08002: starting full resync of recovery catalog
```

```
RMAN-08004: full resync complete
```

```
RMAN> exit;
```

你现在准备使用恢复管理器备份你的数据库。下面的示例表明如果数据库没有打开（然而，该数据库必须被安装）的话，恢复管理器将执行一个冷数据库备份；如果数据库打开的

话，恢复管理器将执行一个热备份。热备份和传统的备份一样，仅能在运行于 ARCHIVELOG 模式下的数据库系统中执行。

```

RMAN> run {
2> allocate channel c1 type disk;
3> backup full filesper set 3
4> (database format 'rm_%p%d.%s');
5> }

RMAN-03022: compiling command: allocate
RMAN-03023: executing command: allocate
RMAN-08030: allocated RMAN-08500: channel c1: sid=13 devtype=DISK
RMAN-03022: compiling command: backup
RMAN-03023: executing command: backup
RMAN-08008: channel c1: starting datafile backupset
RMAN-08502: set_count=3 set_stamp=362111548
RMAN-08010: channel c1: including datafile 1 in backupset
RMAN-08011: channel c1: including current controlfile in backupset
RMAN-08010: channel c1: including datafile 2 in backupset
RMAN-08013: channel c1: piece 1 created
RMAN-08503: piece handle=rm_1TST8.3 comment=NONE
RMAN-08008: channel c1: starting datafile backupset
RMAN-08502: set_count=4 set_stamp=362111606
RMAN-08010: channel c1: including datafile 4 in backupset
RMAN-08010: channel c1: including datafile 3 in backupset
RMAN-08013: channel c1: piece 1 created
RMAN-08503: piece handle=rm_1TST8.4 comment=NONE
RMAN-03023: executing command: partial resync
RMAN-08003: starting partial resync of recovery catalog
RMAN-08005: partial resync complete
RMAN-08031: released channel: c1

RMAN> exit;

```

24.7 从逻辑备份中恢复

你可以使用由 Oracle 提供的 EXP 工具创建逻辑备份。因为导出存在于一个专有格式中，所以仅有一个免费赠送的工具 IMP。IMP 不会从别的系统中导入数据。适合于此的工具是 SQL*Loader。

IMP 与 EXP 的语法相同并且许多选项是完全相同的。清单 24-2 显示了 IMP HELP=Y 的输出结果。

清单 24-2 IMP HELP=Y 的输出结果

Keyword	Description (Default)	Keyword	Description (Default)
USERID	username/password	FULL	import entire file (N)
BUFFER	size of data buffer	FROMUSER	list of owner usernames
FILE	output file (EXPDAT.DMP)	TOUSER	list of usernames
SHOW	just list file contents (N)	TABLES	list of table names
IGNORE	ignore create errors (N)	RECORDLENGTH	length of IO record
GRANTS	import grants (Y)	INCTYPE	incremental import type
INDEXES	import indexes (Y)	COMMIT	commit array insert (N)
ROWS	import data rows (Y)	PARFILE	parameter filename
LOG	log file of screen output		
DESTROY	overwrite tablespace data file (N)		
INDEXFILE	write table/index info to specified file		
FEEDBACK	display progress every x rows(0)		

用于IMP程序的常用关键字在下面描述。

USERID是用户标识/口令，IMP用它来登录到数据库以执行导入操作。你至少必须有CONNECT SESSION特权才能使用IMP。假如你的导出文件是一个完全导出，你必须具有IMP_FULL_DATABASE特权才能导入它。当然，数据库管理员用户可以做任何事情。

提示 你可以把IMP用于使用普通USERID/PASSWORD@REMOTE_NAME约定的远程数据库。

FULL指定是否导入整个数据库，缺省值是N。

提示 当使用一个用FULL=Y导出的导出文件时，你总能选择FULL=N。

BUFFER用来确定在写入数据库以前内存中将要装入多少行。缺省值视操作系统而定。一般说来，缓冲区越大，导入运行的速度就越快（同时使用更多的系统内存）。你可以用如下公式计算装入缓冲区的行数：

行数=缓冲区容量/最大行容量

假如表包含一个LONG列，那么每次仅能有一行装入缓冲区。

FROMUSER定义了导出文件中资源模式的名称。除非你也使用 TOUSER关键字，否则对象将被导入数据库中相同命名的模式中。假如一个在 FROMUSER中命名的模式不存在（并且没有指定TOUSER），那么对象将被导入运行IMP的USERID模式中。

FILE是导出文件的名字（或者是磁带设备的名字）。

TOUSER是数据库中一个已经存在的用户模式的名字，其中数据库对象可以装入该用户模式中。此关键字支持 FROMUSER关键字。一般说来，TOUSER用于把对象导入到与被导出模式不同名称的模式中。

当把SHOW设置为Y时，显示导出文件的内容；没有东西被实际提交给数据库。

TABLES是将被导入的表的列表。假如省略 TABLES的话，所有的表将被导入。

当把IGNORE设置为Y时，如果不能创建一个对象，仍然能够使导出继续进行。假如你正在把行导入一个已经存在的表中时，此关键字非常有用。

提示 假如你打算导入到一个已经存在的含有约束条件的表中，因为数据可能以任何顺序装入表中，所以你应该暂时停用这些约束条件。当备份结束后，你可以重新启用这些约束条件。

RECORDLENGTH用于把导出文件移动到位于另一个操作系统上的 Oracle数据库中。基本上，假如你正在把文件从系统 A移动到系统B，那么你必须知道系统 B上的记录容量。有关于此的信息可以在Oracle提供的操作系统文件中找到。

GRANTS确定权限设置值是否随对象导入。

INCTYPE可以把RESTORE或SYSTEM作为输入。SYSTEM从最后一个增量导出文件中导入该系统的对象（也就是说，仅导入 SYSTEM模式对象）。RESTORE导入自上次导出运行以来发生改变的用户对象。

INDEXES指定用于被导入的表的索引是否也将被导入。

COMMIT指定当导入每个表时IMP是否将生成多个提交。假如 COMMIT=N，那么在装入每张表后仅发生一次提交。COMMIT=Y减少了回滚溢出的可能性。COMMIT=N的优点是如果某个表上的导入失败了，那么将发生一个回滚，它将完全腾空该表。这是很值得做的，因

为它把表返回到导入以前的状态，因此很容易在没有使记录重复的风险下重新开始导入。

ROWS确定是否还装入被导入表的行。假如你想要做的就是表中没有数据的情况下移动一个模式定义的话，那么设置 ROWS=N 非常有用。

就像 PARFILE 用于 EXP 一样，它同样用于 IMP。

LOG 用于记录所有正常情况下发送到终端的输出的文件。提供一个你希望在其中记录输出的标准文件名。使用此选项不会阻止输出在屏幕上显示。

当一个完全导出正在被导入时，DESTROY 是有效的。当创建一个新的表空间时，DESTROY=Y 可以使任何已存在的数据文件被改写。这可用于将一个实例整个替换的情况下。然而，假如你正在相同的机器上创建一个数据库的完全相同的拷贝，应使用 DESTROY=N。因为你不想破坏你的原始数据库！在这种情况下，你应手工创建所有必要的表空间，然后执行导入操作。

INDEXFILE 是一个文件名，在该文件中写入用于在导出文件中创建索引的 SQL 代码。当使用 INDEXFILE 时，不执行对数据库的修改。假如你想改变你的新系统中的初始和下一个区间值，INDEXFILE 就会发挥作用。假如你要人工创建此输出的索引，应确保在 INDEXES=N 时导入以防止它们被自动创建。

就像 EXP 一样，你可以为 IMP 提供你所希望的关键字。假如 IMP 需要的信息多于你所提供的，它会提示你提供信息。要完全交互式地运行，只要在命令行中键入 imp 即可。

你可以执行一个完全导入或部分导入。完全导入在一个目标实例中重新创建一个数据库。显而易见，你的导出文件必须包含你需要导入的所有信息。你总可以导入一个导出文件中的数据子集。在随后的小节中将对这两种不同的方式分别进行讨论。

24.7.1 从逻辑备份完全恢复

完全逻辑恢复一般既可以在同一台机器上，也可以在不同的机器上重新创建一个已存在系统的拷贝。因为导出文件不受平台约束，所以它们是可用于此操作的工具。导出文件还不受版本约束。实际上，当你正在更新 Oracle 版本时，逻辑备份/恢复经常被推荐使用。

要想完成一次成功的恢复，唯一的要求是存在一个数据库实例。所有的表空间、用户、环境资源文件、表、索引等将用 IMP 重新创建。

提示 使用 IMP 创建表空间可以防止一个严重的问题。你可以用相同名称的数据文件（包括它们的路径）重新创建表空间。你可能希望在不同的数据库中有相同的表空间，但是你可能还需要不同的数据文件文件名，把它们存放在不同的目录中，或者使它们具有不同的容量，或一个不同的平台或许根本不能使用相同的文件格式。在这些情况下，你想要在目标系统中人工创建表空间。下面这条 SQL 语句帮助你了解每个表空间的名字以及它们的容量（以字节为单位）：

```
SELECT TABLESPACE_NAME,SUM(BYTES) FROM DBA_DATA_FILES GROUP BY  
TABLESPACE_NAME;
```

如下的 SQL 语句将帮助你确定表空间的容量。它指出了每个表空间中的空间数量（字节），这些表空间还没有被分配给一个段（表、索引等）：

```
SELECT TABLESPACE_NAME,SUM(BYTES) FROM DBA_FREE_SPACE GROUP BY TABLESPACE_NAME;
```

警告 假如你正在同一台机器上把一个实例拷贝到另一个实例，那么你必须在目标实例

中人工创建表空间。假如你不这样做，当IMP试图在目标实例中创建表空间时，它可能会破坏你的原始数据库实例中的数据文件。

一个典型的完全逻辑恢复语法相当简单。下面便是一个例子：

```
imp USERID=SYSTEM/MANAGER FULL=Y IGNORE=Y FILE=<export filename>
```

这里，假设一个实例已经运行并且存在适当的表空间。

24.7.2 使用逻辑备份进行部分恢复

在大多数情况下，你将使用IMP工具把某个对象（或用户模式）集合恢复到你的数据库中。在试图使用IMP恢复部分数据库以前，你应该有合适的表空间、回滚段和用户设置。记住，假如一个表空间不存在或不能容纳被导入的对象的话，那么该对象将被装入用户的缺省表空间中。还有，当导入时，如果当前数据库中没有FROMUSER列举的用户，那么在USERID中指定的用户将用来存储对象。

1. 示例1

回忆一下数据库管理员要使用逻辑备份的情况。在每一个简短的重新叙述后，恢复对象 / 模式需要的IMP命令行将被显示出来。

假定你需要把用户SKLEIN重新命名为SJONES。你已经把原始用户模式导出到一个文件中。新的用户标识SJONES已经被创建，你准备把对象从老模式恢复到新模式中：

```
imp USERID=SYSTEM/MANAGER FILE=SKLEIN2SJONES.DMP FROMUSER=SKLEIN TOUSER=SJONES
```

2. 示例2

上周你被要求从数据库中删除用户TGASPER。在运行了一个FULL=Y导出数据库后（当然在几小时后），你删除了用户TGASPER。然而现在需要查看TGASPER模式中的表。你刚刚重新创建了TGASPER模式并且准备导入该对象：

```
imp USERID=SYSTEM/MANAGER FILE=/dev/rmt0 FROMUSER=TGASPER
```

3. 示例3

一个常见的应用把APAS模式中的表用于它的配置信息。就在上个月，版本2.3被升级到版本3.0。在那次升级中，不再需要表STACONF和USRCONF。作为精明的数据库管理员，你在删除那些表以前导出了它们。版本3.01刚刚推出，你推测它们需要USRCONF表的支持：

```
imp USERID=SYSTEM/MANAGER FILE=table_STACONF.dmp TABLE
```

毫无疑问，你必须处理更复杂的导入 / 导出问题。如果可能的话，在把你的导入用于成品系统以前，总是先在一个测试样本中测试它们——尤其是当你使用IGNORE=Y时。你可以用一个不正确的导入对一个已无法修复的数据库表进行任意地破坏。记住你为Oracle支持程序付了费用。假如你有任何疑问的话，务必使用它解决导入 / 恢复文件问题。

24.8 使用物理恢复

许多数据库管理员认为物理恢复是他们工作中最感到压力和困难的部分。像物理备份一样，大部分的恢复传统上是以操作系统为中心的。Windows NT恢复管理器与恢复管理器(RMAN)都能够替代恢复进程的大量操作系统部分。不管你使用哪一个平台，大部分相同的问题和序列仍然适用。物理恢复包含的内容远超过本书所涉及到的。通常以下四种情况需要执行物理恢复：

你需要在另一个系统上生成当前数据库的一个拷贝（使用相同的操作系统和 Oracle 版本）。这也可以通过使用 FULL=Y 关键字的逻辑备份/恢复来完成。

假如磁盘升级或文件系统分区发生变化，你需要恢复数据库或该数据库的任何一部分。一个或多个数据文件已经遭到破坏或丢失。

你需要把数据库恢复到以前的状态。

前两种情况实际上是同一个过程：你需要从冷备份中恢复丢失的文件。实际上，你正在物理地重新创建该数据库。此论题将在本章的 24.8.1 节中讨论。

当磁盘驱动器失效、数据文件被偶然删除或由于任何原因而使你不能存取数据时，你需要面对恢复丢失的数据文件的问题。因为你不想丢失任何数据，所以你使用完全恢复（complete recovery）。

除了必须采用“前进一步，后退两步”的方法外，把数据恢复到过去的某个时刻也是一种完全恢复形式。这被称为不完全恢复（incomplete recovery）。

提示 假如你正在你的成品系统上工作并且需要执行某种类型的恢复，可能的话，在你考虑恢复数据库以前，你总是应当关闭数据库并且执行一个冷备份。这样，万一某一事物出错，你可以补救。此外，好的经验法则是设法完成一个恢复。假如失败的话，打电话给 Oracle 公司。数据库管理员通过试用恢复技术很快会掌握它。

24.8.1 物理地重建一个数据库

数据库管理员经常需要在另一台机器上创建数据库的一个副本。作为一般维护的一部分，通常要升级数据库磁盘驱动器或重新建立包含数据库文件的文件系统。在任何一种情况下，你都需要对系统“照相”（换句话说，执行一个冷备份）。当系统管理员完成他们的维护工作后，你可以把所有的文件拷贝到另一台机器上或把他们恢复到同一台机器上。

提示 假如物理恢复正用来创建已存在的系统的另一个拷贝，记住即使 SID 改变了，数据库名称也不会改变。参考 Oracle 注释 15390.1 或 Oracle 的技术支持以帮助修改一个已存在数据库的数据库名称。

你需要做的全部事情是把冷备份恢复到原来的机器（当维护结束后）或一个新机器中，然后启动数据库。记住，你的所有数据、重做日志、归档日志和控制文件应当刚好放入与它们原来的存放目录相同的目录位置中。新环境下的 Oracle 不知道自它被关闭后所发生的任何事情。因此，Oracle 希望找到以前的所有组件。

提示 假如你需要改变一个新文件的存放位置，使用 24.8.3 节的“重新命名并移动数据库文件”部分中所描述的 ALTER DATABASE RENAME FILE 语句。假如你需要改变大量数据库文件的存放位置，通常最好的办法是创建一个完成此项工作的脚本或使用一个逻辑备份与恢复。在服务器管理器中一次改变许多数据库文件的存放位置非常单调乏味并且容易出错。

在一个命令行系统中，恢复完全是基于操作系统的。你需要恢复如下的数据库文件：

```
init<instance>.ora。  
config<instance>.ora。  
重做日志。
```

控制文件。

数据文件。

归档重做日志。

你将使用你的机器中可用的任何备份 / 恢复工具恢复这些文件。

假如你在维护期间仅仅丢失了少数文件，只要你没有在冷备份后重新启动数据库，你就可以恢复丢失的文件。在你安置好所有的文件后，通过服务器管理器启动数据库。

在Windows NT环境下，Oracle使用NT恢复管理器进行数据文件的恢复。在运行 NT恢复管理器以前，确保存在合适的 init<instance>.ora文件。记住把它放置在与原来相同的驱动器和目录路径中。

当你启动NT恢复管理器时，你会看到一个类似于图 24-4 的窗口。

要恢复数据库，完成如下的步骤：

1) 选择Restore from full database backup选项。

2) 单击Recover按钮。

3) NT恢复管理器现在需要知道到哪里查找备份文件(该文件由Oracle备份管理器产生)。键入存放备份文件的目录名称，或者选择已经在其中安装了备份磁带的磁带设备。

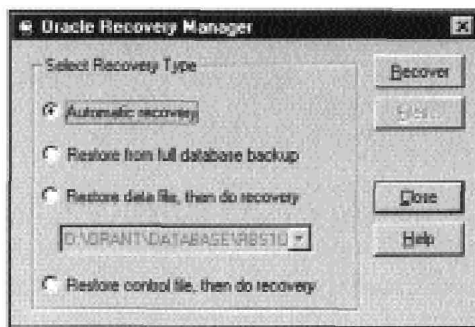


图24-4 Oracle的NT恢复管理器提供了一个方便的图形用户界面用于恢复数据库操作

4) 假如你正在从磁盘中恢复，确保在

Backup域中指示了最新的备份。

5) 单击OK。开始进行恢复。

6) 启动数据库。

提示 假如你正在Windows NT环境下使用Oracle并且发现GUI工具没有提供某个恢复选项，记住你仍然有UNIX数据库管理员可以得到用于恢复的几乎所有命令行工具。

24.8.2 完全恢复

假如一个数据文件丢失的话，完全恢复（complete recovery）在不会丢失任何事务的情况下使你的数据库起死回生。假如一个数据文件丢失或遭到破坏的话，完全恢复使用重做日志文件与归档日志文件来阻止数据的丢失。必须强制性地使控制文件、重做日志和归档日志驻留在与数据文件不同的物理磁盘上。归档日志或重做日志如果和数据文件一起丢了，你就不能够在没有数据丢失的情况下进行恢复（这被称为不完全恢复）。

完全恢复的思想是从一个热备份或冷备份中恢复一个已丢失的数据文件的较老的（并且是有效的）拷贝，然后重新运用自恢复的数据文件得到备份以来发生的所有变化。许多RDBMS（包括Oracle）使用前滚操作提供一个完全恢复，意指“重新运行”重做日志与归档日志中包含的操作。Oracle保存一个称为序列号（sequence number）的主里程表，该序列号记录在数据文件与控制文件中。通过查看控制文件，Oracle便可以知道所有的数据文件已经记录的序列号。假如某个数据文件没有包含当前的序列号，Oracle就知道它不是当前的数据文件并且需要在打开数据库以前得到恢复。

通过查看需要恢复的数据文件中的序列号，Oracle知道它需要从哪里开始重新运行事务。

Oracle期望发现至少一组当前重做日志以及所有的归档日志文件，Oracle在归档日志目录（在init.ora文件中设置）中需要这些文件。因此，你应仅删除在最后一个备份开始以前被记录的归档日志目录中的文件。假如你遗漏了任何一个归档日志或已经丢失了重做日志的所有拷贝，你只能执行一个不完全恢复——你不能够恢复到最后一次提交时的状态。

提示 假如你碰巧将归档日志文件放入几个不同的目录中，当Oracle在缺省目录中（假如你正在使用命令行恢复）不能找到一个指定的归档日志文件时，它会提示你。此时，你应能够提供指向该归档日志文件的完整路径。

当丢失了数据文件时，你需要关闭数据库（如果该数据库还没有关闭的话）。必要时使用SHUTDOWN ABORT关闭它。假如你遇到了硬件故障，你需要在继续工作以前修理该硬件问题。

1. 命令行环境

既然操作系统得到备份并正在运行，那么就检查数据库的损坏情况。当你知道丢失了哪些数据文件后，你（或系统管理员）将把它们恢复到它们原先所在的位置中。即使丢失了某些重做日志或控制文件的话，也不要从备份中恢复任何重做日志或控制文件。

假如你丢失了一个控制文件，你可以用当前控制文件的拷贝来替换它。所有用于你的实例的控制文件在init<sid>.ora文件中列出。

Oracle将使用存放在系统中不同位置上的重做日志的拷贝来替换已丢失的重做日志。在修理了数据文件后，再恢复丢失的重做日志文件。

在继续操作以前，仔细检查自上次备份开始以来你所拥有的全部归档日志。它们应全部存放在归档日志目录中（或至少在系统中的一个或多个其他目录中）。现在准备把老数据文件前滚到最后一次提交时的水平：

- 1) 启动服务器管理器。
- 2) 像平常一样启动数据库。
- 3) 当数据库试图打开时，Oracle会发现一个或多个数据文件含有过时的序列号。Oracle将停下来等待你恢复该数据库。你现在应键入如下语句：

```
RECOVER AUTOMATIC DATABASE ;
```

- 4) 当恢复结束后，过时的数据文件将得到前滚。
- 5) 正常关闭数据库。假如你丢失了一个重做日志文件，你可以用一个来自相同组号的当前且有效的重做日志文件拷贝替换它。通过查询 V\$LOG，你可以确定同一个重做日志组中含有什么文件。

- 6) 执行数据库的冷备份。
- 7) 正常启动数据库。

2. Windows NT环境

此处描述了一个与前面所描述的过程完全相同的恢复过程，但是针对 Windows NT的。尽管你可以使用命令行工具恢复一个 Windows NT环境下的Oracle数据库，然而使用Oracle的图形用户界面工具更容易而且更直观。

你需要鉴别哪一个文件丢失了。当你正在使用 NT恢复管理器完全恢复一个数据库时，拥有一张数据库中所有数据文件的列表对你是有帮助的。只要把一个数据文件添加到数据库中就应该启动NT恢复管理器，然后再关闭它。NT恢复管理器在每次启动以及数据库正在运行时

建立一个数据文件、控制文件和重做日志的列表。通过这样做，你可以通过浏览 NT恢复管理器的数据文件列表来确定哪一个文件丢失或遭到破坏。你仍然需要有在你的基于 NT的Oracle实例中的数据文件的纸张拷贝。

在Windows NT环境中，Oracle的NT恢复管理器几乎管理与完全恢复有关的每件事情（假设全部正常的话）。在开始恢复以前，你应仔细检查自上次备份开始以来你所有的归档日志。这些文件应全部位于 init<instance>.ora或config<instance>.ora文件中所指定的归档日志目录中。假如你把你的数据库备份到磁盘上，就需要保证备份文件处于联机状态。否则你需要手头上有备份磁带。

执行如下这些步骤以恢复你的 Windows NT Oracle数据库；

- 1) 设法通过服务器管理器正常启动 Oracle。这会失败，但这是预料之中的，因为存在遗失或受到破坏的文件。
- 2) 启动Oracle NT恢复管理器。
- 3) 选择Restore data file, then do recovery选项。
- 4) 单击Files按钮。一个类似于图 24-5的窗口显示出来。

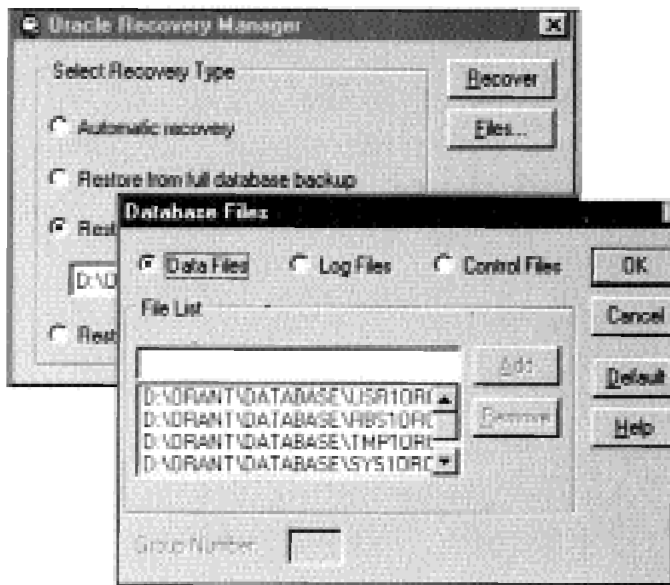


图24-5 选择要使用NT恢复管理器恢复的数据库文件

- 5) 单击Control Files单选钮。
- 6) 通过单击每一个控制文件的文件名和 Remove按钮删除所列举出的任何控制文件。
- 7) 单击Log Files单选钮。
- 8) 通过单击每一个重做日志文件的文件名和 Remove按钮删除所列举出的任何重做日志文件。
- 9) 单击Data Files单选钮。
- 10) 删除所列举出的不需要恢复的全部数据文件。剩下的列举出的文件应是那些遗失或受到破坏的文件。
- 11) 单击OK。

12) 单击 Recover。

13) 选择存放你的数据库备份的地方。选择 Restore from tape 或者 Restore from disk。假如你选择了一个磁盘位置, 确保在 Directory 域中所显示的目录就是你的备份文件所在的地方并且确保 <latest backup> 显示在 Backup 域中。

14) 单击 OK。

15) 当另一个会话打开时, NT 恢复管理器或许会发出一些关于关闭数据库的警告。在弹出的任何对话框中单击 OK。

16) 正常启动数据库。

NT 恢复管理器会使用重做日志与归档日志自动前滚最后一次备份中的数据文件。假如你需要恢复任何遗失的重做日志文件的话, 完成如下的步骤:

1) 关闭数据库。

2) 用来自同一组号的一个文件拷贝替换遗失的重做日志文件。

3) 重新启动数据库。

假如你需要恢复任何遗失的控制文件的话, 完成如下的步骤:

1) 关闭数据库。

2) 用一个当前控制文件的拷贝替换遗失的控制日志文件。

3) 重新启动数据库。

3. 恢复管理器

恢复管理器能够提供一个用于执行物理数据库恢复的完整环境。因为恢复管理器维护一个备份目录, 所以它能够精确地跟踪一个给定的恢复方案需要使用什么样的备份集。恢复管理器能够实际消除曾经在恢复数据库时所做的一些操作系统级的工作。恢复管理器提供了许多恢复选项, 其中包含如下选项:

整个数据库或表空间的时间点恢复。

个别数据文件的恢复。

个别表空间的恢复。

把数据文件恢复到不同位置。

数据库的完全恢复。

就像备份一样, 恢复管理器提供了一大组超出本节范围的恢复选项。请参考 Oracle 的备份与恢复向导得到一个对恢复管理器功能的完整描述。

注意 记住, 要使恢复管理器能够恢复数据库, 它必须在 NOMOUNT 模式下启动。这与大多数其他的恢复系统有所不同, 这些恢复系统使或需要数据库处于完全脱机状态。

4. 示例

在下面的例子中, 你有一个刚刚崩溃且破坏了数据文件的 UNIX 系统。数据库现在当然不能启动, 因为 Oracle 不能使所有的数据文件有效。在本例中, 下述内容是客观的:

BT 的目录已经在另一个 Oracle 8 数据库 TST8 中注册 (TST8 已正确地向 Net 8 注册)。

BT 与在 initBT.ora 文件所指定的目标目录中, 所有新近的归档日志联机运行在 ARCHIVELOG 模式下。

已经使用恢复管理器建立了一个合适的备份周期。

警告 尽管恢复管理器经过非常好地设计以避免进一步地破坏一个已经受到破坏的数据

库，但是在开始任何恢复过程以前，你总是应首先关闭 Oracle，然后执行一个所有 Oracle 组件的完全操作系统级备份。

在下面的示例会话中，你可以看到 BT 数据库正在服务器管理器中被启动。注意 Oracle 已经发现 2 号文件丢失。Oracle 还显示与 2 号文件有关的文件名，然而，恢复管理器仅需要文件号用来恢复正确的数据文件：

```
[tgasper@oreo BT]$ svrmgrl

Oracle Server Manager Release 3.0.5.0.0 - Production

(c) Copyright 1997, Oracle Corporation. All Rights Reserved.

Oracle8 Release 8.0.5.0.0 - Production
PL/SQL Release 8.0.5.0.0 - Production

SVRMGR> connect internal;
Connected.
SVRMGR> startup
ORACLE instance started.
Total System Global Area                4758800 bytes
Fixed Size                               48400 bytes
Variable Size                           4227072 bytes
Database Buffers                        409600 bytes
Redo Buffers                             73728 bytes
Database mounted.
ORA-01157: cannot identify data file 2 - file not found
ORA-01110: data file 2: '/u02/oradata/BT/BT_data1.dbf'
SVRMGR> exit
```

提示 当一个数据库处于 MOUNT 模式时，你可以在 V\$DATABASE 视图中查询 FILE# 与 NAME 域，这样就可以看到数据文件的完整列表以及它们相关的文件号。然后，你可以在操作系统级检查以准确地发现已经丢失的文件以及需要在恢复管理器中得到恢复的文件。

你已经知道接下来需要恢复 2 号文件，下面的示例会话显示了恢复管理器正从备份中恢复丢失文件并使用 RECOVER DATABASE 命令把它完全更新。当从命令行运行恢复管理器时，注意如何使用 RCVCAT 关键字指定目录模式拥有者和数据库：

```
[tgasper@oreo BT]$ rman

Recovery Manager: Release 8.0.5.0.0 - Production

RMAN>

Recovery Manager complete.
[tgasper@oreo BT]$ rman target=sys/change_on_install rcvcat=rman/rman@tst8

Recovery Manager: Release 8.0.5.0.0 - Production

RMAN-06005: connected to target database: BT
RMAN-06008: connected to recovery catalog database

RMAN> run {
2> allocate channel c1 type disk;
3> restore (datafile 2);
4> recover database;
5> release channel c1;
6> }

RMAN-03022: compiling command: allocate
RMAN-03023: executing command: allocate
RMAN-08030: allocated channel: c1
RMAN-08500: channel c1: sid=10 devtype=DISK
```

```
RMAN-03022: compiling command: restore
RMAN-03022: compiling command: IRESTORE
RMAN-03023: executing command: IRESTORE
RMAN-08016: channel c1: starting datafile backupset restore
RMAN-08502: set_count=1 set_stamp=362157796
RMAN-08019: channel c1: restoring datafile 2
RMAN-08509: destination for restore of datafile 2: /u02/oradata/BT/BT_data1.dbf
RMAN-08023: channel c1: restored backup piece 1
RMAN-08511: piece handle=BT.FULL.BTxxxxxx.1.1L
RMAN-08024: channel c1: restore complete
RMAN-03023: executing command: partial resync
RMAN-08003: starting partial resync of recovery catalog
RMAN-08005: partial resync complete

RMAN-03022: compiling command: recover
RMAN-03022: compiling command: recover(1)
RMAN-03022: compiling command: recover(2)

RMAN-03022: compiling command: recover(3)
RMAN-03023: executing command: recover(3)
RMAN-08054: starting media recovery
RMAN-08055: media recovery complete

RMAN-03022: compiling command: recover(4)

RMAN-03022: compiling command: release
RMAN-03023: executing command: release
RMAN-08031: released channel: c1

RMAN> exit
```

在本例子中，你仅恢复一个数据文件，随后恢复该数据库。然而，你可以按需要增加 RESTORE 命令以恢复所有遗失的数据文件。你还可以省略 RECOVER DATABASE 命令并从另一个服务器管理器会话中自己处理最终前滚。

24.8.3 不完全恢复

当你完全恢复一个数据库时，你的目的是要避免丢失任何已提交的事务。要做到这一点，你必须从最后一次备份开始以来的所有的归档日志以及当前重做日志。假如你没有所有的这些文件，你只能执行一个不完全恢复——数据库要丢失一些事务。

提示 在开始一个不完全恢复以前，通常明智的做法是和Oracle技术支持一起检查以确信你真的没有其他的选择。特殊的环境也许能够启用完全恢复，通常情况下这是不太可能的。

一般说来，应避免此类恢复。然而，有很多情况需要你数据库恢复到过去的某个时刻所具有的状态。

1. 示例

假设一家银行由于掩盖金钱贪污而正受到审计员的调查。审计员希望查看掩盖以前的金钱贪污期间的银行数据库。可以这样说，数据库管理员能够在正确设置热备份或冷备份以及适当的归档日志的情况下恢复数据库，然后审计员便可以检查贪污期间的数据库，使犯罪得到曝光。

2. 命令行环境

假如一个归档日志文件遗失，Oracle将停止恢复过程并提示你把遗失的文件放入归档日志

目录中。因为你没有可用的此日志文件，不管什么原因，你需要告诉 Oracle 使用到目前为止已经恢复的日志文件打开数据库。Oracle 不能跳过一个归档日志并用随后的日志继续恢复。清单 24-3 显示了一个用已丢失的数据文件对数据库进行完全恢复的例子。示例系统遗失了 303 号存档日志文件。

清单 24-3 Oracle 服务器管理器归档日志

```
[tgasper@oreo dbs]$ svrmgrl
Oracle Server Manager Release 3.0.5.0.0 - Production
(c) Copyright 1997, Oracle Corporation. All Rights Reserved.
Oracle8 Release 8.0.5.0.0 - Production
PL/SQL Release 8.0.5.0.0 - Production
SVRMGR> connect internal;
Connected.
SVRMGR> startup
ORACLE instance started.
Total System Global Area                2260240 bytes
Fixed Size                              48400 bytes
Variable Size                          1728512 bytes
Database Buffers                       409600 bytes
Redo Buffers                           73728 bytes
Database mounted.
ORA-01113: file 2 needs media recovery
ORA-01110: data file 2: '/u01/oradata/BT/BT_users1.dbf'
SVRMGR> recover automatic database;
ORA-00279: change 68782 generated at 04/07/99 00:47:34 needed for thread 1
ORA-00289: suggestion : /u01/app/oracle/admin/BT/arch/archT0001S0000000303.ARC
ORA-00280: change 68782 for thread 1 is in sequence #303
ORA-00278: log file '/u01/app/oracle/admin/BT/arch/archT0001S0000000303.ARC'
no longer needed for this recovery
ORA-00308: cannot open archive log
'/u01/app/oracle/admin/BT/arch/archT0001S0000000303.ARC'
ORA-27037: unable to obtain file status
Linux Error: 2: No such file or directory
Additional information: 3
Specify log: {<RET>=suggested | filename | AUTO | CANCEL}
```

Oracle 目前正等待你告诉它下一步做什么。因为你的完全恢复失败了，所以你现在必须开始一个不完全恢复。首先，键入 CANCEL 取消当前恢复。然后正常关闭数据库。

表空间 USERS 使用 BT_users1.dbf，由于遗失的归档日志，Oracle 不能完全恢复 BT_users1.dbf。无法把 BT_users1.dbf 文件带回到一致性状态，此一致性状态是相对于数据库剩余部分来说的。你唯一可供选择的安全方法（从数据一致性立场出发）是从你最后一次备份中恢复所有的数据文件并通过 302 号日志应用归档日志。你将丢失发生在 302 号日志后的所有事务。这里是你需要采取的步骤：

- 1) 从你最后一次备份中恢复所有的数据文件。
- 2) 设法正常启动数据库。
- 3) 当 Oracle 需要恢复时，键入此 SQL 语句；
RECOVER AUTOMATIC DATABASE ;
- 4) 当 Oracle 不能找到你遗失的归档日志文件时，它将停止恢复过程。
- 5) 通过键入 CANCEL 取消恢复过程。
- 6) 使用如下这条 SQL 语句执行一个不完全恢复：
RECOVER DATABASE UNTIL CANCEL ;

- 7) 在提示符下用 CANCEL 进行回答。
- 8) 使用如下 SQL 语句强行打开数据库：
`ALTER DATABASE OPEN RESETLOGS;`
- 9) 立刻关闭数据库。
- 10) 执行一个冷备份。
- 11) 重新正常启动数据库。

提示 不必回滚整个数据库，Oracle8的恢复管理器（RMAN）能够对单个表空间执行时间点恢复。就像大多数复杂的恢复操作一样，在使用表空间的时间点恢复以前，你应该同Oracle的技术支持进行协商。

警告 保证只要你用RESETLOGS打开数据库，你就执行一个冷备份。假如你不执行一个冷备份，那么任何归档日志文件就会没用，因为RESETLOGS使当前的归档日志文件与RESETLOGS之前的数据库备份不兼容。

通过大量的工作，你可以把 USERS 表空间中包含的数据恢复到 302 号日志，在没有丢失其他表空间中的数据的情况下把该数据从逻辑上移动到数据库中。但这么做可能引起你的数据的引用完整性问题。

尽管此操作的具体过程超出了本书的范围，然而这里有常用的方案：

- 1) 把你对系统的最后一次备份恢复到一台测试机器上。你需要人工重新创建重做日志。
- 2) 通过 302 号日志把所有的归档日志拷贝到测试机器上。
- 3) 使用 RECOVER DATABASE UNTIL CANCEL 命令恢复数据库。
- 4) 在所有的归档日志上单步执行，直到到达 302 号日志。在 302 号日志之后，取消恢复。
- 5) 使用 RESETLOGS 选项打开数据库。
- 6) 执行 USERS 表空间中所有表的逻辑备份。
- 7) 在你失效的系统中重新创建 USERS 表空间。
- 8) 把在步骤 6 中导出的表从逻辑上恢复到丢失 USERS 表空间的实例中。

在诸如此类的情况下，最好让 Oracle 技术支持参与到该过程中。他们的专业知识将为你提供有关你的特殊恢复状态的所有细节。假如你需要继续进行此操作过程，在尝试任何类型的恢复以前请 Oracle 技术支持出马——不要尝试不完全恢复。

3. 重新命名并移动数据库文件

你经常需要重新命名你的数据库中的文件或把它们从服务器中的一个位置移动到另一个位置。Oracle 使你能够这样做，但是必须遵循指定的步骤。这些步骤中最为重要的是你要在业余时间完成所有的这类工作。你需要关闭数据库以便执行许多这样的操作。在 Oracle 中删除一个重做日志文件、修改控制文件项或执行其他的操作不会物理地删除或重新命名操作系统中的文件。你必须在操作系统层人工执行这些操作。

提示 无论何时，只要超过一个或两个数据库文件被移动，那么使用已经被校正（理想情况是在一个测试系统中检查）过的脚本执行操作系统与 Oracle 任务是明智的。

4. 控制文件

控制文件的位置用 control_files 关键字保存在适当的 init.ora 或 config.ora 文件中。要移动控制文件，应完成如下步骤：

1) 关闭数据库。

2) 把一个控制文件复制或移动到新位置。

3) 编辑适当的init<sid>.ora文件。control_files关键字之后有控制文件列表。你可以改变控制文件的位置或名称、从列表中删除一个控制文件或把一个位置加入一个新的控制文件中。

4) 启动数据库。

5. 重做日志

重做日志必须在SQL*Plus或服务器管理器中改变。从技术上讲，你可以在数据库启动和运行时执行此操作，但是你最好在证实系统中没有活动时做这项操作。采用下面的步骤移动或重新命名重做日志：

1) 关闭数据库。

2) 在EXCLUSIVE模式下启动数据库。如下的示例显示了在服务器管理器中进行此项操作：

```
STARTUP EXCLUSIVE;
```

3) 使用此SQL语句找出你需要改变的文件组号：

```
SELECT * FROM V$LOGFILE;
```

记录下你要处理的组号。

4) 现在，确定哪一个重做日志组是活动的：

```
SELECT * FROM V$LOG;
```

5) 活动的重做日志组用CURRENT标识，显示在步骤4的查询结果的STATUS列中。假如你需要移动的文件当前处于一个活动组中，那么把Oracle转换到下一个重做日志组：

```
ALTER SYSTEM SWITCH LOGFILE;
```

6) 现在删除重做日志组。在本示例中，你正在删除2号日志组。在你的系统中，使用从步骤3中得到的号码。

```
ALTER DATABASE DROP LOGFILE GROUP 2;
```

7) 在你希望的位置用文件创建重做日志文件组。在本示例中，用两个文件重新创建2号组（你在每一个重做日志组中至少有两个文件）。本例中，为每个文件使用1M字节的空间。显而易见，你的组号、实际文件名和文件容量将根据你的要求而变化。

```
ALTER DATABASE ADD LOGFILE GROUP 2
```

```
(' /u03/oradata/test/log2atest.log', '/u04/oradata/test/log2btest.log') size 1M;
```

8) 关闭数据库，然后重新正常启动它。

6. 数据文件

数据文件来回移动相当容易。一般使用服务器管理器和一个操作系统工具实际移动或重新命名数据文件。假如作为物理恢复的一部分重新定位文件，那么只需把数据文件放置在你所希望的地方。下面是改变数据文件位置/名称的步骤：

1) 假如数据库正在运行的话，那么就关闭它。

2) 使用一个操作系统工具把文件移动到一个新的位置或给它重新命名。确保完全了解数据文件曾经在哪儿以及目前在哪儿。

3) 在MOUNT模式下启动数据库。如下的语句显示了服务器管理器中进行此项操作：

STARTUP MOUNT;

4) 对于每一个改动过的文件,在 Oracle中设置新的文件名。如下的示例在内部修改 Oracle以使它将在正确的位置中寻找数据文件。在这里,把 rbs1test.dbf文件从/u01/oradata/test 文件系统移动到/u02/oradata/test文件中。

```
ALTER DATABASE RENAME FILE  
'/u01/oradata/test/rbs1test.dbf' TO  
'/u02/oradata/test/rbs1test.dbf';
```

5) 现在关闭数据库,然后重新启动它。假如该数据库由于遗失了一个数据文件而不能启动,那么你需要找出该文件在哪里并使用 Oracle的RENAME工具以使Oracle发现它所需要的文件。

7. Windows NT环境

不幸的是,Oracle的NT恢复管理器不能完美地处理不完全恢复。由于这个原因,使用 Oracle的NT恢复管理器执行不完全恢复任务最好在 Oracle技术支持的帮助下完成。不完全恢复是精密的过程,因此你不能冒险让 NT恢复管理器发出一条使你的数据库永久不能使用的 SQL语句。

警告 尽管有Oracle专家的帮助,但是在没有对受损数据库进行完全备份以前决不要开始任何恢复过程。只有这样,在恢复进程失败的情况下,你才可以退回并从头开始。

24.9 测试策略

在本章中,已经讨论了大量与备份和恢复 Oracle数据库有关的资料。大部分数据库管理员发现每一个站点都需要独特的备份与恢复操作。因为你严重依赖于你的备份与恢复方案,所以你必须肯定它们确实可靠。一个未经过测试的备份与恢复方案比没有方案还坏——它会给你一种假的安全感。

理想的情况是你有一台与你想要保护的机器完全一样的机器,可以用它来测试备份与恢复方案。通过把成品系统复制到你的测试系统(当然在数据库关闭的情况下),开始你的备份与恢复方案。通过模拟各种失败(切断磁盘驱动器的电源、删除文件、破坏重做日志等)以及从这些失败中进行恢复,你将获得无价的经验,这些经验无法通过读书或参加任何讲座得到。假如在执行这些操作时,能够模拟系统负载,你会获得最好的经验并且保证你的备份与恢复方案使你免遭任何失败。

假如你像大多数数据库管理员一样没有一个用于测试的完整系统,那么就在一台最类似于你需要保护的机器上创建一个实例。最好挑选一台没有 Oracle实例的机器。这将把破坏一个工作中的Oracle数据库的可能性减到最小。如果空间允许的话,尽可能地把成品数据库拷贝到此测试实例中并且尽可能地模仿文件系统/磁盘格式。当模拟失败并设法从失败中恢复时,使用那些与你在成品系统中使用的完全一样的备份与恢复产品。

提示 不管你如何测试你的备份与恢复方案,一定要用文档记录你所采用的确切过程。还要记录你在哪里遇到了困难以及采用了何种解决方案。只要你在成品系统上经历了一次失败,你就应把当时的情况添加到你的失败/恢复练习中。只要该失败曾经发生,它就有可能再次发生。无论何时,只要一名新的数据库管理员加入你的维护人员行列,就让他或她接受系统的事故与恢复训练。

24.10 问题解决策略

1. 出故障的热备份

在数据库的热备份期间，系统发生崩溃，必须重新启动。尽管没有丢失任何数据，但 Oracle 服务器指示系统必须执行介质恢复操作。因为数据文件仍然处在备份模式下，所以 Oracle 认为它们需要得到恢复（这可以通过查询 V\$BACKUP 视图得到确认）。实际上不需要进行恢复，只需为每一个仍然处于备份模式下的文件使用 ALTER DATABASE DATAFILE END BACKUP 语句。关闭数据库，然后正常启动它。

2. 表的导入的约束条件

当把数据导入到一个已存在的表集合时，各种约束条件（例如外关键字关系）使数据不能被导入。

为了随意导入数据，有时需要使用 ALTER TABLE DISABLE CONSTRAINT 语句暂时停用诸如外键或唯一键的各种约束条件。注意当发出 ALTER TABLE ENABLE CONSTRAINT 语句以在导入进程结束后重新启用约束条件时，在实际启用这些约束条件以前，Oracle 首先要确认所有的数据满足这些约束条件。这在一张较大的表上会花费一些时间。

24.11 项目：被破坏的归档日志的不同涵意

如果在试图执行一个完全恢复时，Oracle 提示某个归档日志文件遭到破坏或遗失并且不能用了。那么希望你尽快地执行一个恢复操作。

一般来说，你很难做出选择。你可以执行下面的其中一个操作：

马上停止恢复过程，执行不完全恢复的指令启动该数据库。这将丢失自坏的归档日志创建以来的所有事务。

把数据库恢复到你丢失一个或多个数据文件的初始失败点，然后从数据库中删除丢失或受损的数据文件。你将失去已丢失的数据文件中的任何数据，但是假如你能够从一个逻辑备份的数据文件或另一个数据库系统的拷贝中恢复丢失的信息的话，或许是很有吸引力的。所有剩余的数据就是现存的数据。

假如你选择第二个选项，当数据库处于已安装但没有打开的状态时，你可以使用 ALTER DATABASE DATAFILE OFFLINE 语句将丢失或受损的数据文件脱机，如下所示：

```
SQL> ALTER DATABASE DATAFILE '/u01/oradbs/PRD/PRD_custindx1.dbf' OFFLINE;
```

通过使数据文件脱机，你能够打开数据库并检查受损情况。在打开数据库后，查询 DBA_DATA_FILES 以获得丢失或受损的数据文件的文件号，如下例所示：

```
SQL> select file_id,file_name from dba_data_files where  
2 file_name = '/u01/oradbs/PRD/PRD_custindx1.dbf';
```

```
FILE_ID FILE_NAME
```

```
-----  
3 /u01/oradbs/PRD/PRD_custindx1.dbf
```

你可以在这里看到丢失文件的文件号是3。现在，在下面的例子中，通过查询 DBA_EXTENTS，可以发现哪一个段已经受到失败的影响：

```
SQL> select owner,segment_name,segment_type from dba_extents  
2 where file_id = 3
```

```
OWNER
```

```
SEGMENT_NAME
```

```
SEGMENT_TYPE
```

```

-----
PRDSYS      MCUST_AKEY1      INDEX
PRDSYS      MCUST_AKEY2      INDEX
DBAUSR      AUDIT_PKEY       INDEX
PRDSYS      INVTRY_IDX3      INDEX
PRDSYS      INVTRY_AKEY1     INDEX
BATCHUSR    JOBRESULT_PKEY   INDEX
BATCHUSR    JOBRESULT_IDX1   INDEX

```

这里，你可以看到唯一受到丢失数据文件影响的段是索引——它总能够从原始表中得到重建。实际上，尽管你丢失了信息，然而，你并没有丢失不能重新创建的信息。

执行如下这些步骤重新构造这个系统：

1) 删除前面的查询所显示的索引。在删除以前，你应当确信你拥有稍后用于重建这些索引的DDL代码。

2) 使用ALTER DATABASE DATAFILE OFFLINE DROP语句删除受损的数据文件。

3) 重新创建Oracle中的数据文件。

4) 重新创建丢失的索引对象。

作为一条通用规则，在尝试一种不寻常的恢复方法以前，你应该同 Oracle技术支持商量一下。一般说来，Oracle是一个非常坚固的数据库系统，当你需要从一个受损数据库中可能有的东西时，它会提供帮助的。

即使是对于一个完全丢失的数据文件，你也可以使用表 24-1作为普通Oracle对象的可恢复性特征的准则。

表24-1 丢失文件的可恢复性特征

Oracle对象	可恢复性
表	假如多个区间保存数据，通常可以对那些没有保存在受到数据文件丢失影响的区间中的行进行查询。如果必要应与 Oracle技术支持协商
索引	像以前一样，索引完全是可恢复的。简单地删除数据库中的索引，重新创建受影响的存储区并重新创建该索引
回滚段	尽管启动 Oracle并删除活动回滚段在技术上是可行的，然而你最好还是同 Oracle技术支持商量以确保在你的特定环境下这意味着什么。删除受损的活动回滚段并继续数据库操作非常冒险，如果可能的话应该避免
临时段	它们一般不会引起严重的问题。一般你能够在不引起任何数据完整性问题的前提下重新创建任何丢失的临时表空间数据文件。任何受到影响的事务将被回滚，然后必须重新提交