

第30章 使用导入与导出

本章要点：

理解导出与导入的目的与能力

理解行为

控制与配置导出与导入

导出与导入会话预排

使用SHOW与INDEXFILE选项

便携式表空间

30.1 理解导出与导入的目的与能力

Oracle8.x服务器提供了两个重要的应用工具：导入（Import）与导出（Export）。这两个应用工具对于许多重要的数据库功能非常有用，例如备份数据、在不同模式和数据库之间拷贝对象以及产生对象创建脚本。它们还可以用于将一个版本的 Oracle 移植到另一版本，并可用于将数据库升级到 Oracle8.x。

导出与导入工具提供了范围很广的能力。主要目的是备份并恢复数据及对象。对象和数据以二进制形式存储，只能够被 Oracle 数据库读取。导出与导入可以执行下列任务：

备份与恢复数据库——导出与导入经常作为备份计划的一部分使用。通常，完全备份或热备份用于备份磁盘或计算机故障事件中的整个数据库。但是，如果在整个数据库上仅需要恢复一个表，将不得不在另外一台机器上恢复整个数据库，然后将这表拷贝过来，这会带来难以置信的时间与资源的消耗。导出与导入手段能够帮助你解决这一难题，它为用户提供了导出整个数据库并仅仅导入需要恢复的表的能力。它们还可以作为整个数据库的备份与恢复机制，同基于时间点的恢复（Point-in-time Recovery）一样是一种选择。

在模式之间转移数据和对象——可以使用导出与导入将表、索引、授权、过程、视图，在所有其他的对象类型之间，从一个模式拷贝到另外一个模式中。因为能够指定需要转移的对象，所以这会帮助节省时间与精力。还有，转移数据可以被用作不同 Oracle 数据库之间数据复制的一种形式。

从一个版本将数据库移植到另一个版本——通过使用导出与导入工具，可以升级（或降级）Oracle 数据库的版本。例如，可以导出一个完整的 Oracle7 数据库，然后假设你已经安装了 Oracle8 服务器，那么数据库可以被导入，令数据与应用在 Oracle8 环境中工作。这个过程称为移植。

整理一个表空间或整个数据库的碎片——当对象例如表和索引随着时间的流逝被创建、删除、扩大或缩小时，就会出现磁盘碎片。当拙劣地定义对象存储参数时，也会出现磁盘碎片。通过导出一个表空间，合并空间并重新导入对象，可以整理表空间的磁盘碎片。

生成CREATE脚本——还可以使用导出与导入工具在数据库中全部其他对象之间生成表、分区、视图、授权、索引、约束与表空间的 CREATE脚本。如果一个对象被破坏或删除，对于修改对象并保护对象的结构来说，导出与导入被证明是非常有用的。

在导出中写入多个文件——导出支持写入多个导出文件，导入能够读取多个导出文件。如果在导出时，为导出文件 FILESIZE参数指定了一个值（字节限制），导出将只写入指定数量的字节数据到每个转储文件。在导入时，必须使用导入参数 FILESIZE告诉导入工具你在导出时指定的最大转储文件的大小。

FILESIZE参数有一个最大值，等于64位数据所能存储的最大值。

注意 64位数据存储的最大值依赖于所使用的操作系统（OS），在指定FILESIZE参数之前，应该在操作系统说明文档中确认这个最大值。

可以使用一个后缀K的数字（千字节）指定 FILESIZE参数。例如，FILESIZE=2K等同于 FILESIZE=2048。同样，M表示兆字节（ 1024×1024 ），而G代表十亿字节（ $1024 \times 1024 \times 1024$ ）。B保留作为字节的缩写，最终文件大小数量并没有增加（FILESIZE=20 48b等同于 FILESIZE=2048）。

Oracle8i能够支持精密存取——可以采用精密存取策略导出数据库表。但要注意，要恢复该策略，从含有这些表的导出文件中导入的用户必须拥有一定的权限（尤其是 DBMS_RLS包上的执行权限，以便于恢复表的安全策略）。如果一个没有正确权限的用户试图从一个包含精密访问策略的表的导出文件中导入数据，将会产生一条警告消息。所以，建议出于安全原因，导出/导入这样表的人员应该是数据库系统管理员。

Oracle 8i允许分区级导入——按照下面的方法，可以导入表、分区和子分区：

- 表级导入——从一个导出文件指定的表导入全部数据。
- 分区级导入——只导入来自指定源分区或子分区中的数据。当向一个现有表中加载数据时，必须设置参数 IGNORE=Y。有关参数 IGNORE的信息，参阅本章后面“控制与配置导出与导入”部分参数表中的 IGNORE项。
- Oracle 8i能够导出一部分表——当进行表模式导出时，从一个表中选择记录的子集。查询参数的值是一个串，它在 SQL SELECT 语句中包含 WHERE子句，该子句将应用于 TABLE参数中列出的全部表（或表分区）。

例如，如果SCOTT用户想要导出在部门10的雇员，他可以按照下面所示来做（注意这个例子是基于UNIX的）：

```
exp scott/tiger tables=emp query='"where deptno=10"'
```

注意QUERY参数的值是一个引号括起的串。在字符串的开头和结束使用单引号和双引号是UNIX特定的。请查阅操作系统说明文档了解如何在命令行上传递引号括起字符串的信息。当执行这条命令时，导出建立了一个类似下面的 SELECT语句：

```
SELECT * FROM EMP where deptno=10;
```

QUERY应用于在TABLE参数中列出的全部表（或表分区）。例如，

```
exp scott/tiger tables=emp,dept query='"where deptno=10"'
```

将卸载EMP表和DEPT表中符合查询条件的全部记录。还有，导出执行的 SQL语句类似下列的信息：

```
SELECT * FROM EMP where deptno=10;
SELECT * FROM DEPT where deptno=10;
```

注意 导出文件是二进制格式的文件，只能够被Oracle数据库使用。不能从Oracle数据库导出数据并将数据导入一个非Oracle数据库。同样，不能从一个非Oracle数据库中导入数据。如果想要从其他数据库产品例如微软的ACCESS复制数据到Oracle数据库，应该对一个分隔符文件格式的数据例如 CVS（逗号分割值）使用 SQL*Loader。要使用 Oracle 工具，将数据从 Oracle 数据库转移到一个非 Oracle 数据库，必须手工制作一个分隔符文件，从 PL/SQL 或 SQL*Plus 内部输出。

30.2 理解行为

有三种类型的导出：

FULL（完全）导出——导出数据库中的全部对象、结构和数据。

OWNER（所有者）导出——只导出那些属于特定用户帐户的对象。

TABLE（表）导出——只导出指定的表和分区。

因为许多结构可以被导出，区分使用这三种导出类别中每种不同的导出所得到的结果非常重要。表 30-1 显示了使用每种导出选项导出的结构（按照 Export 导出的顺序）。下面部分说明使用 Oracle8 进行导出与导入，运行 exp 和 imp 可执行程序的语法。

注意 在 Windows NT 和 UNIX 环境中，exp 和 imp 是导出与导入工具的命令。

表30-1 三种导出选项和它们的对象

对象/模式	带有FULL=Y选项	带有OWNER选项	带有TABLE选项
表空间	全部对象定义		
环境资源文件	全部对象		
用户	全部对象定义		
角色	全部对象		
资源开销	全部对象		
回滚段	全部对象定义		
数据库链	全部对象	只有所有者	
序列	全部对象数目	只有所有者	
路径	全部对象别名		
外部函数	全部对象	只有所有者库名称	
对象类型	全部对象定义	只有所有者	
簇	全部对象定义	只有所有者	只有表和所有者
表	全部对象	只有所有者	只有表和所有者
索引	全部对象	只有所有者	只有表和所有者
引用完整性约束	全部对象	只有所有者	只有表和所有者
可传送的动作	全部对象	只有所有者	
同义词	全部对象	只有所有者	只有表和所有者
视图	全部对象	只有所有者	
存储过程	全部对象	只有所有者	
触发器	全部对象	只有所有者	只有表和所有者
快照	全部对象	只有所有者	
快照日志	全部对象	只有所有者	
作业队列	全部对象	只有所有者	

(续)

对象/模式	带有FULL=Y选项	带有OWNER选项	带有TABLE选项
刷新组和孩子	全部对象	只有拥有者	
用户历史表	全部对象		
缺省和系统审计选项	全部对象		

注意 在一个导入过程中，ROWIDS被重新赋值。如果一个表具有带有ROWID信息的列，数据将变为废弃过时的。还有，ROWID在一次导入后变为废弃过时的，在它们能够自动地再次刷新之前，必须使用COMPLETE选项刷新。另外，一些REF属性可能含有ROWID信息，因此在一个导入会话期间，该REF或许会变为过时的。要在导入数据后依据正确的ROWID重新创建REF信息，需要使用ANALYZE TABLE owner.table_name VALIDATE REF UPDATE；命令。

30.3 控制与配置导出与导入

导出与导入工具——带有许多能够被传递的可用参数——是非常灵活的。还有，有两种方法可以运行导出与导入：交互模式和非交互模式。使用交互模式，可以单步执行一系列的提示，为一个导出与导入会话输入基本信息。这种模式缺少灵活性，仅仅提示众多可用参数中的一小部分。

要使用交互模式导出，只需要在命令行输入 exp命令即可。接着被提示输入用户名和口令。

警告 要导出整个数据库，导出所用的用户帐户必须被授予EXP_FULL_DATABASE系统特权。对于拥有数据库系统管理员权限的帐户，例如SYSTEM，这个特权被隐含地授权给该用户。否则导出操作将会失败。

接下来，收到一个提示要求指定数组取出缓存的大小。这是内存缓冲的大小，通过该缓存，记录被导出。这个数字应该大于最大的记录乘以希望在缓存中放置的记录数。这个大小与操作系统有关。

然后收到一个提示信息要求指定导出文件的名称。在缺省情况下，文件名为 expdat.dmp。接下来，一个带有三种选项的小菜单出现了：

如果选择选项1，E（整个数据库），你会被提示为授权（Y/N）、表数据（Y/N）和压缩区间（Y/N）输入一个值。

如果选择选项2，U（用户），你会被提示为授权（Y/N）、表数据（Y/N）和压缩区间（Y/N）以及被导出的用户输入一个值（连续输入，然后在行上输入一个句号结束）。

如果选择选项3，T（表），你会被提示为授权（Y/N）、表数据（Y/N）和压缩区间（Y/N）以及要被导出的表（T）或分区（T:P）输入一个值（连续输入，然后在行上输入一个句号结束）。

要以非交互模式导出，可以在参数行传递所有的命令，或通过一个参数文件传递命令。对所有导出命令可用的参数，在Windows NT中输入exp HELP=Y，如图30-1所示。

在导出会话中可以使用23个参数，可以在命令行或指定的参数文件中指明它们。表30-2描述了所有的导出参数。

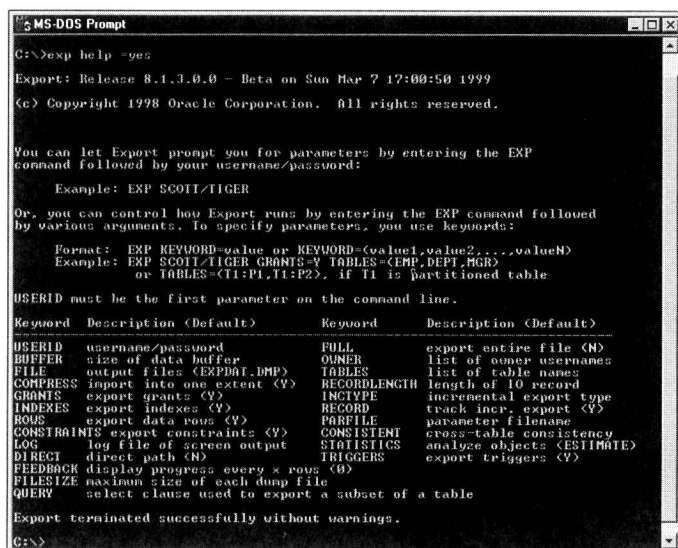


图30-1 在Windows NT环境，“exp HELP=Y”的简单结果

表30-2 导出实用工具的参数描述

参 数	缺 省 值	描 述
BUFFER	与操作系统有关	BUFFER的大小（字节）决定行导出时的内存缓存的大小。应该比最大的记录与想要在缓存中装入的行的数量的乘积大
COMPRESS	Y	如果COMPRESS=Y，INITIAL存储参数将被设置为对象分配的所有区间的总和大小，只有在导入对象时这个改变才发生作用
CONSISTENT	N	选择CONSISTENT=Y以一致状态导出所有的表与索引。这将使导出变慢，因为要使用回滚空间。如果CONSISTENT=N，这是缺省状态，如果在导出过程中，有一条记录发生了改变，数据会成为不一致的
CONSTRAINTS	N	指明是否导出表约束
DIRECT	N	如果DIRECT=Y，Oracle忽略SQL命令处理层，提高导出速度。不幸的是，Oracle8的新的常用对象类型，如LOB，不能被导出
FEEDBACK	0	Oracle为每组记录的插入显示一个句号。组的大小由FEEDBACK定义，例如，如果设置FEEDBACK=1000，每导入1000条记录，显示一个句号。这个参数对跟踪大量导入的进展非常有用
FILE	expdat.dmp	缺省情况下，expdat.dmp（代表导出数据.dump）将是文件名，要使用更有意义的文件名，改变FILE参数
FULL	N	如果FULL=Y，将导出整个数据库，包括表空间定义
FILESIZE		每个卸载文件的最大长度
GRANTS	Y	指明是否导出正被导出对象的全部授权定义
HELP	N	如果指明HELP=Y，不需要其他的参数。将显示一个基本的帮助屏幕
INCTYPE		INCTYPE参数的合法选项是COMPLETE、CUMULATIVE与INCREMENTAL。COMPLETE导出建造一个另两个选项依赖的完全导出，用以重建数据库。CUMULATIVE导出自上次CUMULATIVE或COMPLETE导出发生以来所有改变的表与其他的对象。如果表中有一条记录发生了改变，导出整个表。INCREMENTAL导出上次INCREMENTAL、CUMULATIVE或COMPLETE导出以来发生改变的所有的表与对象

(续)

参 数	缺 省 值	描 述
INDEXES	Y	指明是否导出用户定义索引。系统索引创建时具有约束（主键、唯一键），OID索引自动导出，不管INDEXES参数的值
LOG		LOG参数指明放置导出会话的反馈的文件的名称。除非另外声明，Oracle对文件添加一个.LOG扩展名
PARFILE		代替在命令行输入所有的参数，一些参数或所有的参数可以保留在参数文件中。PARFILE参数指明如果需要，使用哪个文件。这个参数对非交互导入会话非常有用
POINT_IN_	N	对TABLESPACES参数列出的表空间基于时间点的恢复
QUERY		WHERE子句用于导出表的一个子集
RECORD	Y	如果使用具有RECORD=Y的INCTYPE参数，使用导出数据如拥有者、导出类型与导出时间填充 SYS数据字典表 INCEXP、INCFILE与INCVID
RECORD_LENGTH	与操作系统有关	RECORDLENGTH参数只用于当导出到一个与导出发生时具有不同位数机器的时候。在大多数导入会话中，应该使用缺省值
RECOVERY_		RECOVERY_TABLESPACES，与POINT_IN_TIME_RECOVER
TABLESPACES		参数联合使用，指明使用基于时间点的恢复将恢复哪个表空间。这很重要，因为否则导入会恢复导出之后的事务
ROWS	Y	指明是否导出表与对象的数据。如果ROWS=N，只导出对象定义
STATISTICS	ESTIMATE	指明在导入时是否使用 COMPUTE或ESTIMATE分析表与索引统计。注意在导入时只能分析那些早已具有统计信息的对象。如果没有对象需要分析，指明NONE
TABLES		指明一个以逗号分隔的要被导出的表的清单。这个参数应该与FROMUSER参数联合使用，在非 UNIX环境中，如 Windows NT 中，必须使用括号括起表清单
TABLESPACES		使用POINT_IN_TIME_RECOVER参数导出表空间清单
USERID		指明执行导入的用户的用户名与口令。这条命令的格式为：用户名/口令。如果需要，也可以使用NET8的@连接串格式
VOLSIZE		写入每个磁带卷的字节数。VOLSIZE指明每个磁带卷上导出文件的最大字节数，VOLSIZE参数具有的最大值等于以64位存储的最大值

使用交互模式导入，只需在命令行输入 imp，然后会收到用户名与口令的提示。

警告 为使用DBA调用的导出进行导入，导入的用户帐号必须已被赋予 IMP_FULL_DATABASE系统特权。对具有DBA特权的帐户，如SYSTEM，这个特权被隐式地授予该用户，否则，导入会失败。

然后收到一个提示，要求输入要导入的导出文件名。缺省情况下是 expdat.dmp。下面，要求输入数组提取缓冲区大小，这是行将被导出的内存缓存的大小。

接着将被提示是否想列出导出文件的内容清单（是/否）。如果选择是，结果在本章后面的名为“使用SHOW与INDEXFILE选项”一节描述。如果选择否，提问你是否忽略所有的由于对象已存在而产生的创建错误（是/否）。

然后，提问你是否导入授权、表数据与整个导出文件。如果选择否，导入整个导出文件，接收到提示要求输入与对象拥有者对应的用户名，随后会有重复的提示要求输入所有的表与分区。如果保留该行为空，则假设是用户的所有对象。要停止重复提示，在提示符后输入一

个句号。

为在非交互模式中使用导入，可以在命令行传递所有参数，或者通过一个参数文件传递参数。输入“imp HELP=Y”以获得所有导入命令可使用的参数，如图 30-2所示。



图30-2 Windows NT平台上“imp HELP=Y”导出示例

在导入会话中可以使用 27 个参数。可以在命令行指明它们，或在任何指定的参数文件中进行设定。表30-3描述了所有的导入参数。

表30-3 导入实用工具的参数描述

参 数	缺 省 值	描 述
ANALYZE	Y	如果ANALYZE设为Y，导入的表将具有它们的统计分析。注意只有那些在导出时早已具有统计的表才会被计算。缺省情况下表将被ESTIMATED，除非导出执行的是STATISTICS=COMPUTE参数配置
BUFFER	与操作系统有关	BUFFER的大小（字节）决定行导入时的内存缓存的大小。应该比最长的记录与想要在缓存中装入的行的数量的乘积大
CHARSET		CHARSET是一个过时的 Oracle6参数，指明是以 ASCII还是 EBCDIC导出。在 Oracle7与 Oracle8中，这个信息是自动处理的
COMMIT	N	缺省情况下，在每个表、嵌套表与分区之后进行一次提交。如果正在导入一个巨型表，回滚段会变得很大。在装载巨型表时，为提高性能，可以设置COMMIT=Y
DESTROY	N	如果设置DESTROY=Y并且进行一个完全导入，Oracle会覆盖所有现存的数据文件。如果使用裸盘设备作为数据文件，在一个完全导入时，裸盘会被重写，因为 DESTROY=Y并不会阻止重写数据文件！在进行这种导入之前最好进行一次数据库的备份。不要使用这个选项，除非确切知道自己正在做什么
FEEDBACK	0	Oracle为每组记录的插入显示一个句号。组的大小由FEEDBACK定义，例如，如果设置 FEEDBACK=1000，每导入 1000条记录，显示一个句号。这个参数对跟踪大量导入的进展非常有用

(续)

参 数	缺 省 值	描 述
FILE	expdat.dmp	缺省情况下, expdat.dmp (代表导出data.dump) 是要被导入的导入文件名, 如果文件名不是 expdat.dmp, 在FILE参数中指定文件名
FILESIZE		每个转储文件的最大长度——如果从多个文件导入
FROMUSER		指明这个参数将只导入那些由 FROMUSER 用户帐户拥有的对象。
FULL	N	如果 FULL=Y, 将导出整个数据库
GRANTS	Y	指明是否对导出对象创建所有的授权
HELP	N	如果指明 HELP=Y, 不需要其他的参数, 将显示一个基本的帮助屏幕
IGNORE	N	如果 IGNORE=Y, 忽略对象创建错误, 并将记录插入表中。要清楚如果在表中没有唯一性约束, 有可能产生重复的记录。注意仍将报告非对象创建错误, 如操作系统问题
INCTYPE		如果导入一个增长性导出, 表被删除并重建。必须首先从最新的 SYSTEM 导出进行重建 (指明 INCTYPE=SYSTEM)。然后, 导入每个增长性导出 (指明 INCTYPE=RESTORE), 直至需要的改变已应用到数据库为止
INDEXES	Y	指明是否导入用户定义索引。系统索引创建时具有约束 (主键、唯一键), OID 索引不管 INDEXES 参数的值, 自动导入
INDEXFILE		INDEXFILE 参数指明生成 CREAT INDEX 语句的文件名。除非另外指明, Oracle 为该文件附加 .SQL 扩展名。可以在本章稍后找到 INDEXFILE 参数的详细描述
LOG		LOG 参数指明存放导入会话的反馈的文件的名字。除非另外声明, Oracle 对文件附加一个 .LOG 扩展名
PARFILE		代替在命令行输入所有的参数, 一些参数或所有的参数可以保留在参数文件中。PARFILE 参数指明如果需要, 使用哪个文件。这个参数对非交互导入会话非常有用
POINT_IN_TIME	N	对使用 TABLESPACES 参数导出的表空间 RECOVER 执行基于时间点的恢复
RECORDLENGTH	与操作系统有关	RECORDLENGTH 参数只用于当你导出一个与导出发生时具有不同文件位数机器的时候。在大多数导入会话中, 应该使用缺省值
RECALCULATE_STATISTICS	N	重新计算统计值, 用于基于成本的优化
SHOW	N	SHOW 参数在屏幕上显示每条 SQL 语句, 并且不修改数据库。当与 FILE 参数联合使用时, 可以看到并可以修改 SQL 语句。在本章后面的“使用 SHOW 与 INDEXFILE 选项”一节可以找到 SHOW 参数的详细描述
SKIP_UNUSABLE_INDEX	N	SKIP_UNUSABLE_INDEXES 可以延迟索引 INDEXES 的创建直至记录的数据被导入后, 只有那些设置为不使用状态的索引受到影响, 如果 INDEXES=Y (这是缺省值), 其他的索引将被创建
TABLES		指明一个以逗号分隔的要被导入的所有表的清单。这个参数应该与 FROMUSER 参数联合使用, 在非 UNIX 环境中, 如 Windows NT 中, 必须使用括号括起表清单
TOUSER		如果预期要导入到与表的最初拥有者不同的用户帐户中, TOUSER 参数指明将表导入到哪个用户帐户。这个参数需要与 FROMUSER 参数联合使用
TOID_NOVALIDATE		当导入引用一个类型的表, 而在数据库中早已存在这个名字的一个类型时, 导入试图验证事先存在的类型的确是表所使用的类型 (而不是恰巧具有相同名字的不同类型)。要这样做, 导入将类型的唯一标识 (TOID) 与导出文件中存储的标识比较, 如果 TOID

(续)

参 数	缺 省 值	描 述
USERID		不匹配, 将不会导入表记录。在有些情况下, 在指明类型时, 不希望发生这种校验(例如, 如果类型是由插件安装创建的), 可以使用 TOID_NOVALIDATE 参数指明 TOID 比较排除的类型 指明执行导入的用户的用户名与口令。这条命令的格式为: 用户名/口令。如果需要, 也可以使用 NET8 的 @ 连接串格式

30.4 导出与导入会话预排

这一节单步调试一些导入与导出会话的预排。这些预排演示了这两个工具的一些有用的功能, 并指出一些应该避免的缺陷。

30.4.1 当表存在时识别行为

如果试图导入一个数据库中早已存在的表, 会产生一个错误。导入跳过这个表(伴有表上的外键与索引), 继续进行导入处理, 导入所有指定的其他的表与对象, 如图 30-3 所示。

如果想要将数据导入一个早已存在的表中, 使用 IGNORE=Y 参数声明。这会导致数据追加到表中。如果违反约束, 如具有相同主键的重复记录, 不装载那些违反约束的记录。约束的存在对于防止向表中导入重复记录是有帮助的。

例如, 假设 DEMO 用户帐户拥有 EMPLOYEE 表, 在表中拥有记录。尝试启动一个向表中装载重复记录的导入, 图 30-4 显示这种行动的结果。

```

C:\>imp scott/tiger
Import: Release 8.1.3.0.0 - Beta on Sun Mar 7 12:12:44 1999
(c) Copyright 1998 Oracle Corporation. All rights reserved.

Connected to: Oracle8 Enterprise Edition Release 8.1.3.0.0 - Beta
With the Partitioning and Objects options
PL/SQL Release 8.1.3.0.0 - Beta
Import file: EXPDIR.IMP >
Enter insert buffer size (minimum is 8192) 30720
Export file created by EXPORT:V08.01.03 via conventional path
Import done in USASCII character set and NCHAR character set
List contents of import file only (yes/no): no >
Ignore create error due to object existence (yes/no): no >
Import grants (yes/no): yes >
Import table data (yes/no): yes >
Import entire export file (yes/no): no >
Username: scott
Enter table(s) or partition(s): names. Null list means: all tables for user
Enter table(s) or partition(s): name or . if done: emp
Enter table(s) or partition(s): name or . if done:
Importing SCOTT's objects into SCOTT
IMP-00015: following statement failed because the object already exists:
CREATE TABLE 'EMP' ('EMPNO' NUMBER(4, 0), 'ENAME' VARCHAR2(10), 'JOB' VARCHAR2(9), 'MGR' NUMBER(4, 0), 'HIREDATE' DATE, 'SAL' NUMBER(7, 2), 'COMM' NUMBER(2, 2), 'DEPTNO' NUMBER(2, 0) PARTITIONED BY INITIALLY 1 PARTITION 'S 255 LOGGING STORAGE(INITIAL 10240 NEXT 10240 NEXTENTS 1 MAXEXTENTS 121) PCTINCREASE 50 FREELISTS 1 FREELIST GROUPS 1 BUFFER POOL DEFAULT) TABLESPACE 'SYSSEM'.'.
Import terminated successfully with warnings.
C:\>

```

图30-3 导入一个数据库中早已存在的表时产生的错误

```

C:\>imp scott/tiger
Import: Release 8.1.3.0.0 - Beta on Sun Mar 7 12:12:44 1999
(c) Copyright 1998 Oracle Corporation. All rights reserved.

Connected to: Oracle8 Enterprise Edition Release 8.1.3.0.0 - Beta
With the Partitioning and Objects options
PL/SQL Release 8.1.3.0.0 - Beta
Import file: EXPDIR.IMP >
Enter insert buffer size (minimum is 8192) 30720
Export file created by EXPORT:V08.01.03 via conventional path
Import done in USASCII character set and NCHAR character set
List contents of import file only (yes/no): no >
Ignore create error due to object existence (yes/no): no >
Import grants (yes/no): yes >
Import table data (yes/no): yes >
Import entire export file (yes/no): no >
Username: scott
Enter table(s) or partition(s): names. Null list means: all tables for user
Enter table(s) or partition(s): name or . if done: emp
Enter table(s) or partition(s): name or . if done:
Importing SCOTT's objects into SCOTT
IMP-00015: following statement failed because the object already exists:
CREATE TABLE 'EMP' ('EMPNO' NUMBER(4, 0), 'ENAME' VARCHAR2(10), 'JOB' VARCHAR2(9), 'MGR' NUMBER(4, 0), 'HIREDATE' DATE, 'SAL' NUMBER(7, 2), 'COMM' NUMBER(2, 2), 'DEPTNO' NUMBER(2, 0) PARTITIONED BY INITIALLY 1 PARTITION 'S 255 LOGGING STORAGE(INITIAL 10240 NEXT 10240 NEXTENTS 1 MAXEXTENTS 121) PCTINCREASE 50 FREELISTS 1 FREELIST GROUPS 1 BUFFER POOL DEFAULT) TABLESPACE 'SYSSEM'.'.
Import terminated successfully with warnings.
C:\>

```

图30-4 试图向具有唯一性约束的表中导入重复记录示例

30.4.2 重新组织一个碎片表空间

一个碎片表空间具有许多不连续的自由空间块。碎片会导致性能与空间问题。性能受到影响是因为 Oracle 不得不扫描多个对象区间，并有可能跨越多个物理磁盘。当数据进行磁盘碎片整理时，对象可以从多个区间压缩为一个区间，这会减少扫描数据时的内部 Oracle 开销。

相反，一个碎片表空间影响对象存储。具有许多小的自由空间块延伸跨越多个表空间，一些对象可能不会被创建，而如果所有的自由空间是连续的，空间将足够用于创建该对象。通过表空间磁盘碎片整理，重新组织数据以使所有小的自由块形成一个自由块，如图 30-5 所示。

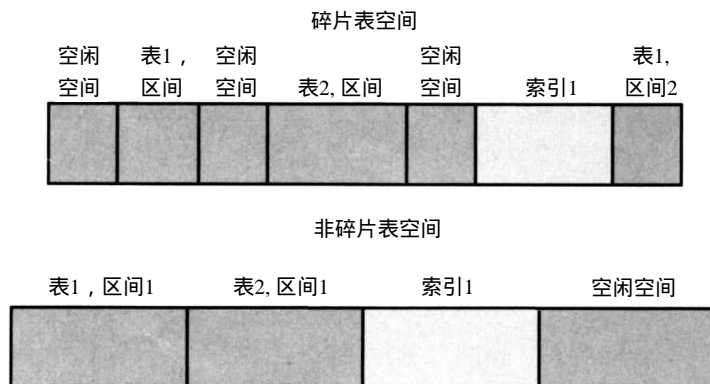


图30-5 碎片空间与非碎片空间示例

当对象如表和索引随时间创建、删除、增大或减小尺寸时，会发生碎片。由于 Oracle 只能在表空间中的一个连续的自由空间内创建一个区间，会开始出现一些小的自由空间。当删除一个对象时，它的自由空间很有可能处于表空间中分散的地点，只能在那个自由空间中创建另一个相等的或小一些的对象。

为检查表空间的碎片，可以检查在指定的表空间中有多少自由空间段，以及它们的大小。这可以通过运行 CHP30_1.SQL 脚本并传递表空间名实现，如程序清单 30-1 所示。可以在 SQL*PLUS、服务器管理器或任何可以连接到 Oracle8 数据库的第三方产品中运行这个脚本。

清单30-1 CHP_30.SQL——决定表空间内自由空间的数量与大小的脚本

```
SQL> START CHP30_1.SQL
SQL> COLUMN FILE_NAME FORMAT A40
SQL> SELECT A.TABLESPACE_NAME, B.FILE_NAME, A.BYTES
2 FROM DBA_FREE_SPACE A, DBA_DATA_FILES B
3 WHERE A.TABLESPACE_NAME = '&tablespace_name' AND
4 A.FILE_ID = B.FILE_ID
5 ORDER BY BYTES DESC
```

当运行这个脚本时，服务器会提示输入表空间名：

Enter value for tablespace_name: USERS

服务器将返回自由空间信息：

TABSPACE_NAME	FILE_NAME	BYTES
USERS	/u02/oradata/TEST/users01.dbf	32768
USERS	/u02/oradata/TEST/users01.dbf	114688
USERS	/u04/oradata/TEST/users02.dbf	122400
USERS	/u04/oradata/TEST/users02.dbf	101908480SQL>

对每一个不同的FILE_NAME, 应该各有一条记录。如果没有, 表空间就是碎片的。在表空间中含有的表在导出与导入过程中会成为用户不可用的。在依赖导入与导入表空间进行表空间碎片整理时, 应该输入命令 ALTER TABLESPACE 表空间名 COALESCE并再次运行 CHP30_1.SQL。如果自由空间的两个大块彼此毗邻, 这条命令使它们组成一个更大的块。如果这条命令没有作用, 必须使用导出与导入以整理表空间碎片。

另一个整理表空间碎片的原因是是否表空间内的一个对象含有多个区间的情况。在大多数情况下, 如果对象具有多于五个区间的情况, 应该加以关注, 因为在这之后性能开始受到显著影响。通过指明 COMPRESS=Y导出表, Oracle计算初始区间的尺寸以将所有的数据放入一个区间, 这也会帮助清除数据库碎片。

运行CHP30_2.SQL以确定哪个对象具有多于五个区间, 如程序清单 30-2所示。

清单30-2 CHP30_2.SQL——列出数据库中具有多于五个区间的所有对象的脚本

```
SQL> START CHP30_2.SQL
SQL> COLUMN SEGMENT_NAME FORMAT A25
SQL> SELECT OWNER, SEGMENT_NAME, EXTENTS
2 FROM DBA_SEGMENTS
3 WHERE EXTENTS > 5 AND
4 OWNER NOT IN ('SYS','SYSTEM')
5 ORDER BY EXTENTS
```

在运行这个脚本之后, 服务器返回具有多于五个区间的所有对象。

OWNER	SEGMENT_NAME	EXTENTS
DEMO	EMPLOYEE	6
DEMO	JOB	9
DEMO	DEPARTMENT	12

在开始进行碎片整理之前, 确信已通知到所有将受影响的人, 因为在这个处理过程中, 表空间中的表将成为不可用的。如果可能, 计划在没有人访问这些表的时刻进行这个处理。

为使用导入与导出以整理一个表空间的碎片, 应遵循以下步骤:

1) 导出表空间包含的所有表, 确保已设置导出的 COMPRESS=Y选项。如果必要, 它会改变表的INITIAL存储参数, 以使每个表适合一个区间。

2) 手工从表空间中删除所有的表。

3) 合并表空间中的自由空间, 这由 ALTER TABLESPACE 表空间名 COALESCE命令完成。所有的自由空间将被合并为一个大块, 或与表空间的数据文件一样多的块。这是由于表空间中不含有对象。

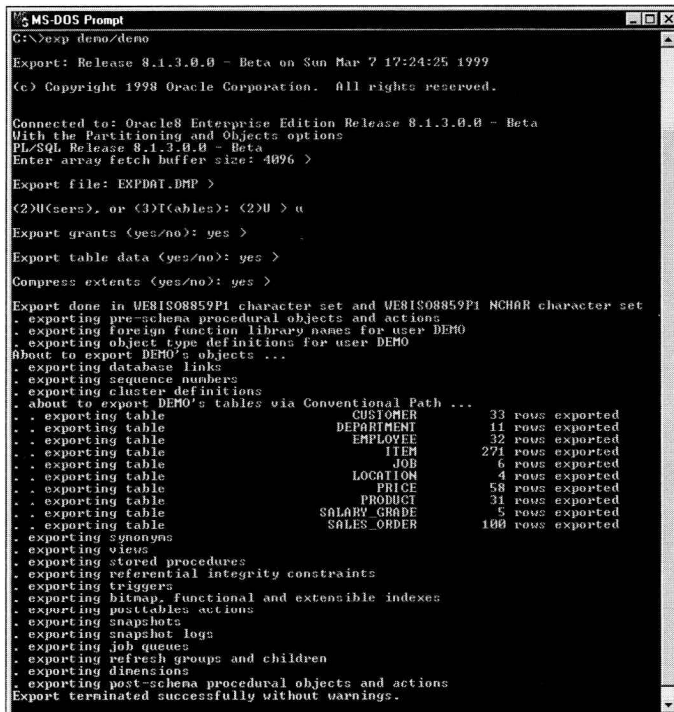
4) 导入表空间包含的所有表。Oracle为每个对象分配合适的空间, 因为在导出时已指明 COMPRESS=Y。完成时, 便具有了一个干净的、没有碎片的表空间了。

警告 如果使用COMPRESS=Y进行导出, 任何被导出的LOB数据将不会被压缩, 它的初始INITIAL与NEXT存储参数不会改变。

前一种方法需要了解表空间中含有的所有对象。在一个更消耗时间的方法中, 可以清除整个数据库的碎片。通过以下方法实现: 导出整个数据库, 使用 CREATE DATABASE命令删除并重建数据库, 然后导入整个导出文件。

30.4.3 在模式间移动数据库对象

许多时候，在模式之间移动对象非常重要。下面就是这种情况，一个开发者想要一套表



```
MS-DOS Prompt
C:\>exp demo/demo

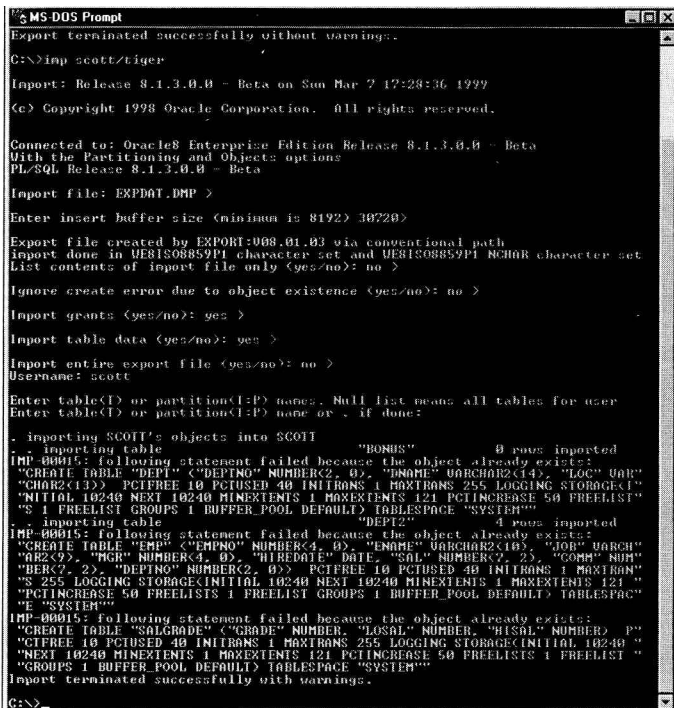
Export: Release 8.1.3.0.0 - Beta on Sun Mar 7 17:24:25 1999
(c) Copyright 1998 Oracle Corporation. All rights reserved.

Connected to: Oracle8 Enterprise Edition Release 8.1.3.0.0 - Beta
With the Partitioning and Objects options
PL/SQL Release 8.1.3.0.0 - Beta
Enter array fetch buffer size: 4096 >

Export file: EXPDAT.DMP >
(2)U(cons), or (3)I(ables): (2)U > u
Export grants (yes/no): yes >
Export table data (yes/no): yes >
Compress extents (yes/no): yes >

Export done in VERIS08859P1 character set and VERIS08859P1 NCHAR character set
. exporting pre-schema procedural objects and actions
. exporting foreign function library names for user DEMO
. exporting object type definitions for user DEMO
About to export DEMO's objects ...
. exporting database links
. exporting sequence numbers
. exporting cluster definitions
. about to export DEMO's tables via Conventional Path ...
. . exporting table
. . DEPARTMENT 33 rows exported
. . exporting table
. . EMPLOYEE 32 rows exported
. . exporting table
. . ITEM 271 rows exported
. . exporting table
. . JOB 6 rows exported
. . exporting table
. . LOCATION 4 rows exported
. . exporting table
. . PRICE 58 rows exported
. . exporting table
. . PRODUCT 31 rows exported
. . exporting table
. . SALARY_GRADE 5 rows exported
. . exporting table
. . SALES_ORDER 100 rows exported
. exporting synonyms
. exporting views
. exporting stored procedures
. exporting referential integrity constraints
. exporting triggers
. exporting bitmap, functional and extensible indexes
. exporting posttables actions
. exporting snapshots
. exporting snapshot logs
. exporting job queues
. exporting refresh groups and children
. exporting dimensions
. exporting post-schema procedural objects and actions
Export terminated successfully without warnings.
```

图30-6 拷贝DEMO模式到QUE模式



```
MS-DOS Prompt
Export terminated successfully without warnings.
C:\>imp scott/tiger

Import: Release 8.1.3.0.0 - Beta on Sun Mar 7 17:28:36 1999
(c) Copyright 1998 Oracle Corporation. All rights reserved.

Connected to: Oracle8 Enterprise Edition Release 8.1.3.0.0 - Beta
With the Partitioning and Objects options
PL/SQL Release 8.1.3.0.0 - Beta
Import file: EXPDAT.DMP >
Enter insert buffer size (minimum is 8192): 30720 >

Export file created by EXPORT=000.01.03 via conventional path
import done in VERIS08859P1 character set and VERIS08859P1 NCHAR character set
List contents of import file only (yes/no): no >
Ignore create error due to object existence (yes/no): no >
Import grants (yes/no): yes >
Import table data (yes/no): yes >
Import entire export file (yes/no): no >
Username: scott

Enter table(s) or partition(s) names. Null list means all tables for user
Enter table(s) or partition(s) name or * if done:

. importing SCOTT's objects into SCOTT
. importing table "BONUS" 0 rows imported
IMP-00015: following statement failed because the object already exists:
"CREATE TABLE "DEPT" ("DEPTNO" NUMBER(2, 0), "DNAME" VARCHAR2(14), "LOC" VARCHAR2(13)) PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 LOGGING STORAGE(INITIAL 10240 NEXT 10240 MINEXTENTS 1 MAXEXTENTS 121 PCTINCREASE 50 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT) TABLESPACE "SYSTEM"
. importing table "DEPT2" 4 rows imported
IMP-00015: following statement failed because the object already exists:
"CREATE TABLE "EMP" ("EMPNO" NUMBER(4, 0), "ENAME" VARCHAR2(16), "JOB" VARCHAR2(9), "MGR" NUMBER(4, 0), "HIREDATE" DATE, "SAL" NUMBER(7, 2), "COMM" NUMBER(7, 2), "DEPTNO" NUMBER(2, 0)) PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 LOGGING STORAGE(INITIAL 10240 NEXT 10240 MINEXTENTS 1 MAXEXTENTS 121 PCTINCREASE 50 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT) TABLESPACE "SYSTEM"
IMP-00015: following statement failed because the object already exists:
"CREATE TABLE "SALGRADE" ("GRADE" NUMBER, "LOCAL" NUMBER, "RISAL" NUMBER, "PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 LOGGING STORAGE(INITIAL 10240 NEXT 10240 MINEXTENTS 1 MAXEXTENTS 121 PCTINCREASE 50 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT) TABLESPACE "SYSTEM"
Import terminated successfully with warnings.
```

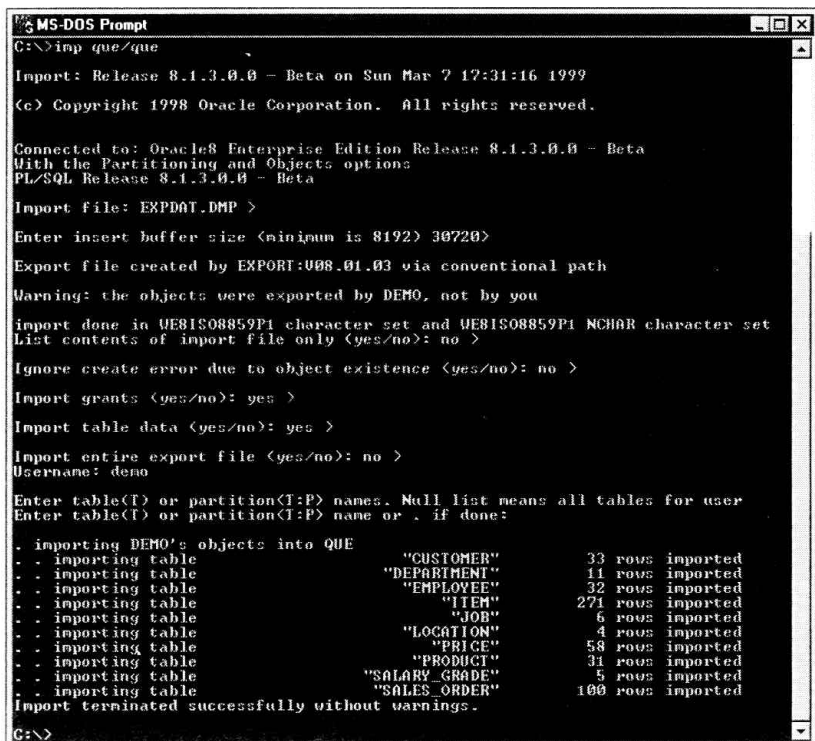
图30-7 导出DEMO用户帐户中的所有对象

用于测试，但不想影响在最初的模式中的数据。这对在实例之间拷贝表也很有用。

首先，导出一个给定用户帐户的所有对象。通过指明导出工具中的 OWNER 参数完成这步。在这个例子中，将拷贝 DEMO 模式到 QUE 模式中，如图 30-6 所示。

如果导入的任何对象与现有的对象冲突，只跳过那些冲突的对象。

接着如果帐户不存在，创建成为新的拥有者的用户帐户。在这一点，可以通过指明导入工具中的 FROMUSER 与 TOUSER 参数导入到新模式中，如图 30-7 所示。



```
MS-DOS Prompt
C:\>imp que/que

Import: Release 8.1.3.0.0 - Beta on Sun Mar 2 17:31:16 1999
(c) Copyright 1998 Oracle Corporation. All rights reserved.

Connected to: Oracle8 Enterprise Edition Release 8.1.3.0.0 - Beta
With the Partitioning and Objects options
PL/SQL Release 8.1.3.0.0 - Beta

Import file: EXPDAT.DMP >
Enter insert buffer size (minimum is 8192) 30720
Export file created by EXPORT:V08.01.03 via conventional path
Warning: the objects were exported by DEMO, not by you
Import done in WE8ISO8859P1 character set and WE8ISO8859P1 NCHAR character set
List contents of import file only (yes/no): no >
Ignore create error due to object existence (yes/no): no >
Import grants (yes/no): yes >
Import table data (yes/no): yes >
Import entire export file (yes/no): no >
Username: demo
Enter table(T) or partition(T:P) names. Null list means all tables for user
Enter table(T) or partition(T:P) name or . if done:
. importing DEMO's objects into QUE
. importing table "CUSTOMER" 33 rows imported
. importing table "DEPARTMENT" 11 rows imported
. importing table "EMPLOYEE" 32 rows imported
. importing table "ITEM" 271 rows imported
. importing table "JOB" 6 rows imported
. importing table "LOCATION" 4 rows imported
. importing table "PRICE" 58 rows imported
. importing table "PRODUCT" 31 rows imported
. importing table "SALARY_GRADE" 5 rows imported
. importing table "SALES_ORDER" 100 rows imported
Import terminated successfully without warnings.
C:\>
```

图30-8 将所有的DEMO用户帐户的对象导入到QUE帐户

提示 如果具有BFILE数据类型，Oracle只存储指向文件自身的指针，真正的数据存储数据库外。如果从一个数据库导出并导入到不同服务器上的另一个数据库中，确信已拷贝所有的外部文件并已将它们放置在相同的路径下，否则，Oracle不能访问有关的BFILE文件。

30.4.4 多个对象与多个对象类型

如果想要将表的一个子集从一个模式移动到另一个模式中，或出于备份目的，在导出时需要在TABLES参数声明中指明对象清单。这必须与导入的 FROMUSER 与 TOUSER 联合使用。

要导出一个表，指明拥有者，随后是一个点与表名。如拥有者 .表名。为导出一个分区，指明拥有者，随后是点与表名，然后是冒号，紧接着是分区名，如拥有者 .表名：分区名。

图30-9显示了导出多个对象与多个对象类型的一个示例：PRICE表、EMP表五个分区中的两个（LOW_SALARY与MEDIUM_SALARY）。

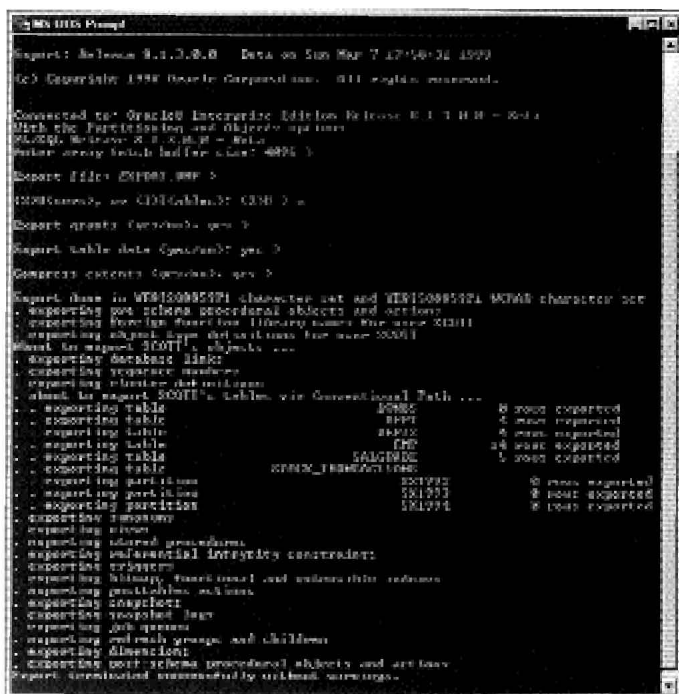


图30-9 导出多个对象与多个对象类型示例

为将多个对象与多个对象类型导入到不同的用户帐户中，导入指明FROMUSER与TOUSER子句。如果想导入到相同的用户中，使用FULL=Y或OWNER参数声明。

30.4.5 分区级导入

分区级导入将源表的一套分区或子分区导入到目标表中。

注意以下要点：

导入总存储与目标表分区模式对应的记录。分区级导入使你可以选择从导出文件的特定分区或子分区检索数据。分区级导入只插入特定的源分区或子分区的记录。如果目标表是分区的，分区级导入拒绝任何比目标表最大分区大的记录。分区级导入可以只以表的模式声明。分区级导出与导入不提供分割一个分区。

30.4.6 当表空间不匹配时识别行为

当导入一个对象时，Oracle试图在与导出相同的表空间内创建这个对象。有时，当在一个数据库上执行一个导出并导入到另一个数据库时，表空间并不总是匹配的。例如，导出数据库中的一个对象存储在 DBA_TOOLS表空间中，而在将导入的数据库中不存在 DBA_TOOLS表空间。

在这种情况下，在导入过程中，导入工具试图在 DBA_TOOLS中创建对象，但失败了，因为在目标数据库中不存在这样的表空间。然后导入试图在该对象拥有者的缺省表空间中创建这个表。如果有足够的空间，并且拥有者具有表空间上适当的配额，对象被导入，否则，产生一个错误，不导入对象。

30.4.7 在表空间之间移动数据库对象

缺省情况下，导入试图在与导出相同的表空间中创建对象。如果用户不具有那个表空间的权限，或者那个表空间不再存在时，Oracle在用户帐户的缺省表空间中创建数据库对象。这些特性可以用于使用导出与导入在表空间之间移动数据库对象。要为 USER_A 将 TABLESPACE_A 的所有对象移动到 TABLESPACE_B，应遵循以下步骤：

1) 为 USER_A 导出 TABLESPACE_A 中的所有对象。

2) 执行 REVOKE UNLIMITED TABLESPACE ON TABLESPACE_A FROM USER_A; 以回收任何授予用户帐户的无限制表空间权限。

3) 执行 ALTER USER USER_A QUOTA 0 ON TABLESPACE_A; 以使 USER_A 帐户不能在 TABLESPACE_A 上创建任何对象。

4) 删除 TABLESPACE_A 中 USER_A 拥有的所有对象。

5) 执行 ALTER USER USER_A DEFAULT TABLESPACE TABLESPACE_B; 以使 TABLESPACE_B 成为 USER_A 用户帐户的缺省表空间。Oracle 试图将对象导入 TABLESPACE_A，因为这些对象是从 TABLESPACE_A 导出的。注意用户不具有 TABLESPACE_A 上的配额。然后将查看用户的缺省表空间。在 Oracle 可以将数据导入 TABLESPACE_B 之前，必须给予 USER_A 用户该表空间上足够大的配额，如以下步骤所示。

6) 执行 ALTER USER USER_A QUOTA UNLIMITED ON TABLESPACE_B;。通过给予 USER_A 一个无限的配额，只要 TABLESPACE_B 足够大，能处理所有被导入的数据库对象，导入就会成功。

7) 导入被导出的数据库对象。缺省情况下，导入工具试图将它们导入到 TABLESPACE_A 中，然而，因为用户不具有这个表空间的配额，所以所有的对象将被创建在 USER_A 的缺省表空间，TABLESPACE_B 中。

前面的步骤显示了怎样使用导入与导出工具，同时还有 SQL 的知识，以对数据进行相对容易的强有力的操作。导入与导出工具最有用的能力之一是 SHOW 与 INDEXFILE 选项的使用，这将在下一节介绍。

30.5 使用 SHOW 与 INDEXFILE 选项

可以使用 SHOW 参数生成用于创建数据库结构与所有对象的全部 SQL 语句。包括创建注释、表空间、用户、权限、授权、角色与它们的分配、限额定义、回滚段、序列、表、约束、索引、包、过程、分区、用户定义数据类型等等。

SHOW 参数的一个最有力的使用是创建可以重建全部或部分数据库的脚本文件。语句按正确的相关性顺序列出——也就是说，在创建索引之前创建表、指向主键的外键在主键创建后创建等等。程序清单 30-3 显示了指明 SHOW=Y 的输出示例的一部分。

清单30-3 CHP30_3.lst 具有 SHOW=Y 导入的部分示例

```
"ALTER SCHEMA = "QUE" "  
"CREATE UNIQUE INDEX "I_PRICE" ON "PRICE" ("PRODUCT_ID" , "START_DATE" ) PC "  
"TFREE 10 INITRANS 2 MAXTRANS 255 STORAGE (INITIAL 10240 NEXT 10240 MINEXTEN "  
"TS 1 MAXEXTENTS 121 PCTINCREASE 50 FREELISTS 1) TABLESPACE "USER_DATA" LOGG "  
"ING "  
"ALTER TABLE "PRICE" ADD CHECK (PRODUCT_ID IS NOT NULL) ENABLE"
```

```

"ALTER TABLE "PRICE" ADD CHECK (START_DATE IS NOT NULL) ENABLE"
"ALTER TABLE "PRICE" ADD CHECK (LIST_PRICE IS NULL OR MIN_PRICE IS NULL OR "
"MIN_PRICE <= LIST_PRICE) ENABLE"
"ALTER TABLE "PRICE" ADD CHECK (END_DATE IS NULL OR START_DATE <= END_DATE)"
" ENABLE"
"ALTER TABLE "PRICE" ADD PRIMARY KEY ("PRODUCT_ID","START_DATE") ENABLE"
"GRANT SELECT ON "PRICE" TO PUBLIC"
"ALTER SCHEMA = "QUE"
"COMMENT ON TABLE "PRICE" IS 'Prices (both standard and minimum) of product"
"s. Database tracks both effective dates and expiration dates for prices.'"
"COMMENT ON COLUMN "PRICE"."PRODUCT_ID" IS 'Product number to which price a"
"pplies. Product name found in table PRICE.'"
"COMMENT ON COLUMN "PRICE"."LIST_PRICE" IS 'Undiscounted price (in U.S.dol"
"lars).'"
"ALTER TABLE "PRICE" ADD FOREIGN KEY ("PRODUCT_ID") REFERENCES "PRODUCT" ("P"
"RODUCT_ID") ENABLE"

```

为从结果产生文件，指明 LOG=文件名参数声明。可以修改这个文件以改变数据库的几乎任一部分。每一行以引号标志开始和结束。确信已在每行的开始和结束串起这些引号。另外，Oracle在输出时不会自动换行，结果是语句的句子和数字有可能被切成两半。为补救它，必须手工连接每条语句的行。前面程序清单中显示的示例程序清单将被整理如程序清单 30-4 所示。

清单30-4 CHP30_4.lst整理SHOW=Y导入文件之后的SQL语句

```

ALTER SCHEMA = "QUE";
CREATE UNIQUE INDEX "I_PRICE" ON "PRICE" ("PRODUCT_ID" , "START_DATE" )
  PCTFREE 10 INITRANS 2 MAXTRANS 255
  STORAGE (INITIAL 10240 NEXT 10240 MINEXTENTS 1 MAXEXTENTS 121
  PCTINCREASE 50 FREELISTS 1)
  TABLESPACE "USER_DATA" LOGGING;
ALTER TABLE "PRICE" ADD CHECK (PRODUCT_ID IS NOT NULL) ENABLE;
ALTER TABLE "PRICE" ADD CHECK (START_DATE IS NOT NULL) ENABLE;
ALTER TABLE "PRICE" ADD CHECK (LIST_PRICE IS NULL OR MIN_PRICE IS NULL
OR MIN_PRICE <= LIST_PRICE) ENABLE;
ALTER TABLE "PRICE" ADD CHECK (END_DATE IS NULL OR START_DATE <= END_DATE)
ENABLE;
ALTER TABLE "PRICE" ADD PRIMARY KEY (PRODUCT_ID,START_DATE) ENABLE;
GRANT SELECT ON "PRICE" TO PUBLIC;
COMMENT ON TABLE "PRICE" IS
  'Prices (both standard and minimum) of products. Database tracks both
  effective dates and expiration dates for prices.';
COMMENT ON COLUMN "PRICE"."PRODUCT_ID" IS 'Product number to which price applies.
  Product name found in table PRICE.';
COMMENT ON COLUMN "PRICE"."LIST_PRICE" IS 'Undiscounted price (in U.S.dollars).';
ALTER TABLE "PRICE" ADD FOREIGN KEY (PRODUCT_ID) REFERENCES PRODUCT (PRODUCT_ID) ENABLE;

```

可以使用INDEXFILE参数生成CREATE INDEX语句。INDEXFILE参数的值指明要被创建的文件的名称。除非另外声明，缺省情况下，Oracle附加一个.SQL扩展名。显示常用的表创建语句，并被注释起来，以便在脚本运行时不被执行。INDEXFILE参数不生成CREATE主键或唯一键子句。程序清单 30-5是具有INDEXFILE=X.LOG声明的输出文件X.LOG的一部分。注意Oracle怎样相应地自动换行，并且不在每行之前和之后增加引号标志。这使索引文件可以不需要进一步修改就能立即使用。

清单30-5 X.LOG 具有INDEXFILE=X.LOG导入创建的X.LOG文件的部分示例

```
REM CREATE TABLE "QUE"."PRICE" ("PRODUCT_ID" NUMBER(6, 0), "LIST_PRICE"  
REM NUMBER(8, 2), "MIN_PRICE" NUMBER(8, 2), "START_DATE" DATE, "END_DATE"  
REM DATE) PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 LOGGING  
REM STORAGE(INITIAL 10240 NEXT 10240 MINEXTENTS 1 MAXEXTENTS 121  
REM PCTINCREASE 50 FREELISTS 1 FREELIST GROUPS 1) TABLESPACE "USER_DATA" ;  
REM ... 58 rows  
CONNECT QUE;  
CREATE UNIQUE INDEX "QUE"."I_PRICE" ON "PRICE" ("PRODUCT_ID" ,  
"START_DATE" ) PCTFREE 10 INITRANS 2 MAXTRANS 255 STORAGE (INITIAL 10240  
NEXT 10240 MINEXTENTS 1 MAXEXTENTS 121 PCTINCREASE 50 FREELISTS 1)  
TABLESPACE "USER_DATA" LOGGING ;
```

30.6 便携式表空间

这个功能使用户可以将 Oracle 数据库的一个子集移动到另一个 Oracle 数据库中。子集看起来像是从初始数据库中拔出来并插到另一个数据库中。也可以克隆一个数据库中的一个表空间并将其插入另一个数据库中，从而，在数据库之间拷贝表空间。使用便携式表空间功能移动数据潜在地比使用导出/导入或卸载/装载要快许多，因为便携式表空间只包括拷贝数据库文件与集成的元数据。在数据仓库环境中，大量数据从最初的 OLTP 数据库流入企业数据仓库并流向数据商业中心，这个功能显示了数据移动的一个更快与更创新的方法。另外，它也可以扩展用于归档数据。