Oracle imp/exp 命令详解

导入/导出是 ORACLE 幸存的最古老的两个命令行工具，其实我从来不认为 Exp/Imp 是一种好的备份方式，正确的说法是 Exp/Imp 只能是一个好的转储工具，特别是在小型数据库的转储，表空间的迁移，表的抽取，检测逻辑和物理冲突等中有不小的功劳。当然，我们也可以把它作为小型数据库的物理备份后的一个逻辑辅助备份，也是不错的建议。对于越来越大的数据库，特别是 TB 级数据库和越来越多数据仓库的出现，EXP/IMP 越来越力不从心了，这个时候，数据库的备份都转向了 RMAN 和第三方工具。下面说明一下 EXP/IMP 的使用。

如何使 exp 的帮助以不同的字符集显示：set nls_lang=simplified chinese_china.zhs16gbk，通过设置环境变量，可以让 exp 的帮助以中文显示，如果 set nls_lang=American_america.字符集，那么帮助就是英文的了

程序代码 EXP 的所有参数（括号中为参数的默认值）：

USERID　　用户名/口令　　如： USERID=duanl/duanl

FULL　　导出整个数据库 (N)

BUFFER　　数据缓冲区的大小

OWNER　　所有者用户名列表,你希望导出哪个用户的对象，就用 owner=username

FILE　　输出文件 (EXPDAT.DMP)

TABLES　　表名列表 ,指定导出的 table 名称，如：TABLES=table1,table2

COMPRESS　　导入一个 extent (Y)

RECORDLENGTH　IO 记录的长度

GRANTS　　导出权限 (Y)

INCTYPE　　增量导出类型

INDEXES　　导出索引 (Y)

RECORD　　跟踪增量导出 (Y)

ROWS　　导出数据行 (Y)

PARFILE　　参数文件名,如果你 exp 的参数很多，可以存成参数文件.

CONSTRAINTS　导出约束 (Y)

CONSISTENT　交叉表一致性

LOG　　屏幕输出的日志文件

STATISTICS　　分析对象 (ESTIMATE)

DIRECT　　直接路径 (N)

TRIGGERS　　导出触发器 (Y)

FEEDBACK　　显示每 x 行 (0) 的进度

FILESIZE　　各转储文件的最大尺寸

QUERY　　选定导出表子集的子句

下列关键字仅用于可传输的表空间

TRANSPORT_TABLESPACE 导出可传输的表空间元数据 (N)

TABLESPACES 将传输的表空间列表

程序代码 IMP 的所有参数（括号中为参数的默认值）：

USERID　用户名/口令

FULL　　导入整个文件 (N)

BUFFER　数据缓冲区大小

FROMUSER　　所有人用户名列表

FILE　　输入文件 (EXPDAT.DMP)

TOUSER　　用户名列表

SHOW　　只列出文件内容 (N)

TABLES 表名列表

IGNORE 忽略创建错误 (N)

RECORDLENGTH IO 记录的长度

GRANTS 导入权限 (Y)

INCTYPE 增量导入类型

INDEXES 导入索引 (Y)

COMMIT 提交数组插入 (N)

ROWS 导入数据行 (Y)

PARFILE 参数文件名

LOG 屏幕输出的日志文件

CONSTRAINTS 导入限制 (Y)

DESTROY 覆盖表空间数据文件 (N)

INDEXFILE 将表/索引信息写入指定的文件

SKIP_UNUSABLE_INDEXES 跳过不可用索引的维护 (N)

ANALYZE 执行转储文件中的 ANALYZE 语句 (Y)

FEEDBACK 显示每 x 行 (0) 的进度

TOID_NOVALIDATE 跳过指定类型 id 的校验

FILESIZE 各转储文件的最大尺寸

RECALCULATE_STATISTICS 重新计算统计值 (N)

下列关键字仅用于可传输的表空间

TRANSPORT_TABLESPACE 导入可传输的表空间元数据 (N)

TABLESPACES 将要传输到数据库的表空间

DATAFILES 将要传输到数据库的数据文件

TTS_OWNERS 拥有可传输表空间集中数据的用户

关于增量参数的说明：exp/imp 的增量并不是真正意义上的增量，所以最好不要使用。

使用方法：

Exp parameter_name=value or Exp parameter_name=(value1,value2……)

只要输入参数 help=y 就可以看到所有帮助.

EXP 常用选项

1.FULL，这个用于导出整个数据库，在 ROWS=N 一起使用时，可以导出整个数据库的结构。例如：

exp userid=test/test file=./db_str.dmp log=./db_str.log full=y rows=n compress=y direct=y

2. OWNER 和 TABLE，这两个选项用于定义 EXP 的对象。OWNER 定义导出指定用户的对象；TABLE 指定 EXP 的 table 名称，例如：

exp userid=test/test file=./db_str.dmp log=./db_str.log owner=duanl

exp userid=test/test file=./db_str.dmp log=./db_str.log table=nc_data,fi_arap

3.BUFFER 和 FEEDBACK，在导出比较多的数据时，我会考虑设置这两个参数。例如：

exp userid=test/test file=yw97_2003.dmp log=yw97_2003_3.log feedback=10000 buffer=100000000 tables=WO4,OK_YT

4.FILE 和 LOG，这两个参数分别指定备份的 DMP 名称和 LOG 名称，包括文件名和目录，例子见上面。

5.COMPRESS 参数不压缩导出数据的内容。用来控制导出对象的 storage 语句如何产生。默认值为 Y，使用默认值，对象的存储语句的 init extent 等于当前导出对象的 extent 的总和。

推荐使用 COMPRESS＝N。

6. FILESIZE 该选项在 8i 中可用。如果导出的 dmp 文件过大时，最好使用 FILESIZE 参数，限制文件大小不要超过 2G。如：

exp userid=duanl/duanl file=f1,f2,f3,f4,f5 filesize=2G owner=scott

这样将创建 f1.dmp, f2.dmp 等一系列文件，每个大小都为 2G，如果导出的总量小于 10G

EXP 不必创建 f5.bmp.

IMP 常用选项

1、FROMUSER 和 TOUSER，使用它们实现将数据从一个 SCHEMA 中导入到另外一个 SCHEMA 中。例如：假设我们做 exp 时导出的为 test 的对象,现在我们想把对象导入用户：

imp userid=test1/test1 file=expdat.dmp fromuser=test1 touser=test1

2、IGNORE、GRANTS 和 INDEXES，其中 IGNORE 参数将忽略表的存在，继续导入，这个对于需要调整表的存储参数时很有用，我们可以先根据实际情况用合理的存储参数建好表，然后

直接导入数据。而 GRANTS 和 INDEXES 则表示是否导入授权和索引，如果想使用新的存储参数重建索引，或者为了加快到入速度，我们可以考虑将 INDEXES 设为 N，而 GRANTS 一般都是 Y

。例如：imp userid=test1/test1 file=expdat.dmp fromuser=test1 touser=test1 indexes=N

表空间传输

表空间传输是 8i 新增加的一种快速在数据库间移动数据的一种办法，是把一个数据库上的格式数据文件附加到另外一个数据库中，而不是把数据导出成 Dmp 文件，这在有些时

候是非常管用的，因为传输表空间移动数据就象复制文件一样快。

关于传输表空间有一些规则，即：

·源数据库和目标数据库必须运行在相同的硬件平台上。

·源数据库与目标数据库必须使用相同的字符集。

·源数据库与目标数据库一定要有相同大小的数据块

·目标数据库不能有与迁移表空间同名的表空间

·SYS 的对象不能迁移

·必须传输自包含的对象集

·有一些对象，如物化视图，基于函数的索引等不能被传输

可以用以下的方法来检测一个表空间或一套表空间是否符合传输标准：

exec sys.dbms_tts.transport_set_check('tablespace_name',true);

select * from sys.transport_set_violation;

如果没有行选择，表示该表空间只包含表数据，并且是自包含的。对于有些非自包含的表空间，如数据表空间和索引表空间，可以一起传输。

以下为简要使用步骤，如果想参考详细使用方法，也可以参考 ORACLE 联机帮助。

1.设置表空间为只读（假定表空间名字为 APP_Data 和 APP_Index）

alter tablespace app_data read only;

alter tablespace app_index read only;

2.发出 EXP 命令

SQL>host exp userid=""sys/password as sysdba""

transport_tablespace=y tablespace=(app_data, app_index)

以上需要注意的是

·为了在 SQL 中执行 EXP，USERID 必须用三个引号，在 UNIX 中也必须注意避免"/"的使用

·在 816 和以后，必须使用 sysdba 才能操作

·这个命令在 SQL 中必须放置在一行（这里是因为显示问题放在了两行）

3.拷贝数据文件到另一个地点，即目标数据库

可以是 cp(unix)或 copy(windows)或通过 ftp 传输文件（一定要在 bin 方式）

4.把本地的表空间设置为读写

5.在目标数据库附加该数据文件

imp file=expdat.dmp userid=""" sys/password as sysdba""" transport_tablespace=y "datafile=(c:\temp\app_data,c:\temp\app_index)"

6.设置目标数据库表空间为读写

alter tablespace app_data read write;

alter tablespace app_index read write;

优化 EXP/IMP 的方法：

当需要 exp/imp 的数据量比较大时，这个过程需要的时间是比较长的，我们可以用一些方法来优化 exp/imp 的操作。

exp:使用直接路径 direct=y

oracle 会避开 sql 语句处理引擎,直接从数据库文件中读取数据,然后写入导出文件.

可以在导出日志中观察到: exp-00067: table xxx will be exported in conventional path

如果没有使用直接路径,必须保证 buffer 参数的值足够大.

有一些参数于 direct=y 不兼容,无法用直接路径导出可移动的 tablespace,或者用 query 参数导出数据库子集.

当导入导出的数据库运行在不同的 os 下时,必须保证 recordlength 参数的值一致.

imp:通过以下几个途径优化

1.避免磁盘排序

将 sort_area_size 设置为一个较大的值,比如 100M

2.避免日志切换等待

增加重做日志组的数量,增大日志文件大小.

3.优化日志缓冲区

比如将 log_buffer 容量扩大 10 倍(最大不要超过 5M)

4.使用阵列插入与提交

commit = y

注意:阵列方式不能处理包含 LOB 和 LONG 类型的表,对于这样的 table,如果使用 commit = y,每插入一行,就会执行一次提交.

5.使用 NOLOGGING 方式减小重做日志大小

在导入时指定参数 indexes=n,只导入数据而忽略 index,在导完数据后在通过脚本创建 index,指定 NOLOGGING 选项

导出/导入与字符集

进行数据的导入导出时，我们要注意关于字符集的问题。在 EXP/IMP 过程中我们需要注意四个字符集的参数：导出端的客户端字符集，导出端数据库字符集，导入端的客户端

字符集，导入端数据库字符集。

我们首先需要查看这四个字符集参数。

查看数据库的字符集的信息：

```
SQL> select * from nls_database_parameters;
PARAMETER                      VALUE
------------------------------ ----------------------------------------------------------------
NLS_LANGUAGE                   AMERICAN
NLS_TERRITORY                  AMERICA
NLS_CURRENCY                   $
NLS_ISO_CURRENCY               AMERICA
NLS_NUMERIC_CHARACTERS         .,
NLS_CHARACTERSET               ZHS16GBK
NLS_CALENDAR                   GREGORIAN
NLS_DATE_FORMAT                DD-MON-RR
NLS_DATE_LANGUAGE              AMERICAN
NLS_SORT                       BINARY
```

NLS_TIME_FORMAT HH.MI.SSXFF AM

NLS_TIMESTAMP_FORMAT DD-MON-RR HH.MI.SSXFF AM

NLS_TIME_TZ_FORMAT HH.MI.SSXFF AM TZH:TZM

NLS_TIMESTAMP_TZ_FORMAT DD-MON-RR HH.MI.SSXFF AM TZH:TZM

NLS_DUAL_CURRENCY $

NLS_COMP BINARY

NLS_NCHAR_CHARACTERSET ZHS16GBK

NLS_RDBMS_VERSION 8.1.7.4.1

NLS_CHARACTERSET：ZHS16GBK 是当前数据库的字符集。

我们再来查看客户端的字符集信息：

客户端字符集的参数 NLS_LANG=_< territory >.

language：指定 oracle 消息使用的语言，日期中日和月的显示。

Territory：指定货币和数字的格式，地区和计算星期及日期的习惯。

Characterset：控制客户端应用程序使用的字符集。通常设置或等于客户端的代码页。或者对于 unicode 应用设为 UTF8。

在 windows 中，查询和修改 NLS_LANG 可在注册表中进行：

HKEY_LOCAL_MACHINE\SOFTWARE\Oracle\HOMExx\

xx 指存在多个 Oracle_HOME 时的系统编号。

在 unix 中：

$ env|grep NLS_LANG

NLS_LANG=simplified chinese_china.ZHS16GBK

修改可用：

$ export NLS_LANG=AMERICAN_AMERICA.UTF8

通常在导出时最好把客户端字符集设置得和数据库端相同。当进行数据导入时，主要有以下两种情况：

(1) 源数据库和目标数据库具有相同的字符集设置。

这时，只需设置导出和导入端的客户端 NLS_LANG 等于数据库字符集即可。

(2) 源数据库和目标数据库字符集不同。

先将导出端客户端的 NLS_LANG 设置成和导出端的数据库字符集一致，导出数据，然后将导入端客户端的 NLS_LANG 设置成和导出端一致，导入数据，这样转换只发生在数据库端

，而且只发生一次。

这种情况下，只有当导入端数据库字符集为导出端数据库字符集的严格超集时，数据才能完全导成功，否则，可能会有数据不一致或乱码出现。

不同版本的 EXP/IMP 问题

一般来说，从低版本导入到高版本问题不大，麻烦的是将高版本的数据导入到低版本中，在 Oracle9i 之前，不同版本 Oracle 之间的 EXP/IMP 可以通过下面的方法来解决：

1、在高版本数据库上运行底版本的 catexp.sql；

2、使用低版本的 EXP 来导出高版本的数据；

3、使用低版本的 IMP 将数据库导入到低版本数据库中；

4、在高版本数据库上重新运行高版本的 catexp.sql 脚本。

但在 9i 中，上面的方法并不能解决问题。如果直接使用低版本 EXP/IMP 会出现如下错误：

EXP-00008: orACLE error %lu encountered

orA-00904: invalid column name

这已经是一个公布的 BUG，需要等到 Oracle10.0 才能解决，BUG 号为 2261722，你可以到 METALINK 上去查看有关此 BUG 的详细信

息。

BUG 归 BUG，我们的工作还是要做，在没有 Oracle 的支持之前，我们就自己解决。在 Oracle9i 中执行下面的 SQL 重建 exu81rls 视图即可。

```
Create or REPLACE view exu81rls
(objown,objnam,policy,polown,polsch,polfun,stmts,chkopt,enabled,spolicy)
AS select u.name, o.name, r.pname, r.pfschma, r.ppname, r.pfname,
decode(bitand(r.stmt_type,1), 0,'', 'Select,')
|| decode(bitand(r.stmt_type,2), 0,'', 'Insert,')
|| decode(bitand(r.stmt_type,4), 0,'', 'Update,')
|| decode(bitand(r.stmt_type,8), 0,'', 'Delete,'),
r.check_opt, r.enable_flag,
DECODE(BITAND(r.stmt_type, 16), 0, 0, 1)
from user$ u, obj$ o, rls$ r
where u.user# = o.owner#
and r.obj# = o.obj#
and (uid = 0 or
uid = o.owner# or
exists ( select * from session_roles where role='Select_CATALOG_ROLE')
)
/
grant select on sys.exu81rls to public;
/
```

可以跨版本的使用 EXP/IMP，但必须正确地使用 EXP 和 IMP 的版本：

1、总是使用 IMP 的版本匹配数据库的版本，如：要导入到 817 中，使用 817 的 IMP 工具。

2、总是使用 EXP 的版本匹配两个数据库中最低的版本，如：从 9201 往 817 中导入，则使用 817 版本的 EXP 工具。

==========

导入注意事项：

(1) 数据库对象已经存在

一般情况，导入数据前应该彻底删除目标数据下的表，序列，函数/过程,触发器等;

数据库对象已经存在，按缺省的 imp 参数，则会导入失败

如果用了参数 ignore=y，会把 exp 文件内的数据内容导入

如果表有唯一关键字的约束条件，不合条件将不被导入

如果表没有唯一关键字的约束条件，将引起记录重复

(2) 数据库对象有主外键约束

不符合主外键约束时，数据会导入失败

解决办法: 先导入主表，再导入依存表

disable 目标导入对象的主外键约束，导入数据后，再 enable 它们

(3) 权限不够

如果要把 A 用户的数据导入 B 用户下, A 用户需要有 imp_full_database 权限

(4) 导入大表(大于 80M) 时，存储分配失败

默认的 EXP 时, compress = Y，也就是把所有的数据压缩在一个数据块上.

导入时，如果不存在连续一个大数据块，则会导入失败.

导出 80M 以上的大表时，记得 compress= N，则不会引起这种错误.

(5) imp 和 exp 使用的字符集不同

如果字符集不同，导入会失败，可以改变 unix 环境变量或者 NT 注册表里 NLS_LANG 相关信息.

导入完成后再改回来.

(6) imp 和 exp 版本不能往上兼容

imp 可以成功导入低版本 exp 生成的文件，不能导入高版本 exp 生成的文件

使用方法：

例题格式及说明：

1.普通数据库全部导出和导入

exp 用户/密码@dbName file=路径.dmp full=y --还有其他的参数,看需要进行填写

$ exp user/pwd file=/dir/xxx.dmp log=xxx.log full=y commit=y ignore=y --全部导出

$ imp user/pwd file=/dir/xxx.dmp log=xxx.log fromuser=dbuser touser=dbuser2 --全部导入

2.指定用户全部导出

/home/oracle/product/9.2.0.4/bin/exp userid=用户/密码 --说明：本地的数据库登入(可以指定其他数据库，则需添加@dbName)

owner=导出的用户名 file=导出路径存放目录.dmp log=导出的日志信息.log --主要：这是不能使用 full=y 或则会出错(默认该用户全导出)

3.文件参数导出

$ exp parfile=username.par // 在参数文件中输入所需的参数

参数文件 username.par 内容

userid=username/userpassword

buffer=8192000

compress=n

grants=y

file=/oracle/test.dmp

full=y

4.制定表导出(分区表导出及条件表导出)

$ exp user/pwd file=/dir/xxx.dmp log=xxx.log tables=table1,table2 --或 tables(table1,table2,…..)

$ exp user/pwd file=/dir/xxx.dmp log=xxx.log tables=(T1: table1,T2: table2,…..) --T1 是分区表

$ exp scott/tiger tables=emp query=\"where job=\'salesman\' and sal\<1600\" file=/directory/scott2.dmp 或根据参数文件进行导出

5.导入（一张或多张表）

$ imp user/pwd file=/dir/xxx.dmp log=xxx.log tables=(table1,table2) fromuser=dbuser

touser=dbuser2 commit=y ignore=y

$ imp user/pwd file=/dir/xxx.dmp log=xxx.log fromuser=dbuser touser=dbuser2

commit=y ignore=y

6.只导出数据对象不导出数据

$ exp user/pwd file=/dir/xxx.dmp log=xxx.log owner=user rows=n --rows=n/y 说明是否导出数据行

7.分割多个文件导出和导入

$ exp user/pwd file=1.dmp,2.dmp,3.dmp,… filesize=1000m log=xxx.log full=y

$ imp user/pwd file=1.dmp,2.dmp,3.dmp,… filesize=1000m tables=xxx fromuser=dbuser

touser=dbuser2 commit=y ignore=y

8.增量导出和导入

a.完全增量导出（inctype=complete） // 备份整个数据库

$ exp user/pwd file=/dir/xxx.dmp log=xxx.log inctype=complete

b.增量型增量导出 导出上一次备份后改变的数据（inctype=incremental）。

$ exp user/pwd file=/dir/xxx.dmp log=xxx.log inctype=incremental

c.累计型增量导出（Cumulative） 只导出自上次"完全"导出之后数据库中变化的信息。

$ exp user/pwd file=/dir/xxx.dmp log=xxx.log inctype=cumulative

d.增量导入:

$ imp usr/pwd FULL=y inctype=system/restore/inctype --(SYSTEM: 导入系统对象,RESTORE: 导入所有用户对象)

9.使用 sysdba 进行导出和导入

1. 命令行方式:

A: Windows 平台:

C:\> exp as sysdba"

10.表空间传输（建议:10g 以上使用,但我试了在 9i 没有找到相对应的检查表空是否传输的语句,10g 支持跨平台的表空间传输）

注意：

I.索引在待传输表空间集中而表却不在。(注意，如果表在待传输表空间集中，而索引不在并不违反自包含原则，当然如果你坚持这样传输的话，会造成目标库中该表索引丢失)。

2.分区表中只有部分分区在待传输表空间集(对于分区表，要么全部包含在待传输表空间集中，要么全不包含)。

3.待传输表空间中，对于引用完整性约束，如果约束指向的表不在待传输表空间集，则违反自包含约束；但如果不传输该约束，则与约束指向无关。

4.对于包含 LOB 列的表，如果表在待传输表空间集中，而 Lob 列不在，也是违反自包含原则的。

a.查看表空间包含那些 XML 文件

```
select distinct p.tablespace_name
from dba_tablespaces p, dba_xml_tables x, dba_users u, all_all_tables t
where t.table_name = x.table_name
and t.tablespace_name = p.tablespace_name
and x.owner = u.username
```

b.检测一个表空间是否符合传输标准的方法：

```
SQL > exec sys.dbms_tts.transport_set_check('tablespace_name',true);
SQL > select * from sys.transport_set_violations;
```

c.简要使用步骤

1.设置表空间为只读（假定表空间名字为 APP_Data 和 APP_Index）

```
SQL > alter tablespace app_data read only;
SQL > alter tablespace app_index read only;
```

2.发出 EXP 命令

```
SQL> host exp userid="'sys/password as sysdba"' transport_tablespace=y
tablespaces=(app_data, app_index)
```

以上需要注意的是:（或则参考我自己写的 表空间导入和导出例题）

·为了在 SQL 中执行 EXP，USERID 必须用三个引号，在 UNIX 中也必须注意避免"/"的使用

·在 816 和以后，必须使用 sysdba 才能操作

·这个命令在 SQL 中必须放置在一行（这里是因为显示问题放在了两行）

3.拷贝.dbf 数据文件（以及.dmp 文件）到另一个地点，即目标数据库可以是 cp(unix)或 copy(windows)或通过 ftp 传输文件（一定要在 bin 方式）

4.把本地的表空间设置为读写

$ alter tablespace app_data read write;

$ alter tablespace app_index read write;

5.在目标数据库附加该数据文件 (直接指定数据文件名)

(表空间不能存在，必须建立相应用户名或者用 fromuser/touser)

$ imp file=expdat.dmp userid="""sys/password as sysdba"""

transport_tablespace=y datafiles=("c:\app_data.dbf,c:\app_index.dbf")

tablespaces=app_data,app_index tts_owners=hr,oe

6.设置目标数据库表空间为读写

$ alter tablespace app_data read write;

$ alter tablespace app_index read write;

11.优化 IMP/EXP 的速度(修改参数配置文件)

EXP:

加大 large_pool_size，可以提高 exp 的速度

采用直接路径的方式(direct=y)，数据不需要经过内存进行整合和检查.

设置较大的 buffer，如果导出大对象，小 buffer 会失败。

export 文件不在 ORACLE 使用的驱动器上,不要 export 到 NFS 文件系统

UNIX 环境：用管道模式直接导入导出来提高 imp/exp 的性能

IMP:

建立一个 indexfile，在数据 import 完成后在建立索引

将 import 文件放在不同的驱动器上

增加 DB_BLOCK_BUFFERS

增加 LOG_BUFFER

用非归档方式运行 ORACLE：ALTER DATABASE NOARCHIVELOG;

建立大的表空间和回滚段，OFFLINE 其他回滚段，回滚段的大小为最大表的 1/2

使用 COMMIT=N

使用 ANALYZE=N

单用户模式导入

UNIX 环境：用管道模式直接导入导出来提高 imp/exp 的性能

12.通过 unix/Linux PIPE 管道加快 exp/imp 速度

步骤如下：

通过管道导出数据:

1.通过 mknod -p 建立管道

$ mknod /home/exppipe p // 在目录/home 下建立一个管道 exppipe 注意参数 p

2.通过 exp 和 gzip 导出数据到建立的管道并压缩

$ exp test/test file=/home/exppipe & gzip < /home/exppipe > exp.dmp.gz

$ exp test/test tables=bitmap file=/home/newsys/test.pipe &

gzip < /home/newsys/test.pipe > bitmap.dmp.gz

3.导出成功完成之后删除建立的管道

$ rm -rf /home/exppipe

4.shell 脚本可以这样写(我只是写主要的)

unix 下：

mkfifo /home/exp.pipe

chmod a+rw exp.pipe

compress < exp.pipe > exp.dmp.Z &

su -u oracle -c "exp userid=ll/ll file=/home/exp.pipe full=y buffer=20000000"

rm exp.pipe

linux 下：

mknod /home/exppipe p

$ imp test/test file=/home/exppipe fromuser=test touser=macro &

gunzip < exp.dmp.gz > /home/exppipe

$ rm –fr /home/exppipe

实例：

Oracle 的导入实用程序(Import utility)允许从数据库提取数据，并且将数据写入操作系统文件。imp 使用的基本格式：imp[username[/password[@service]]]，以下例举 imp 常用用法。

1. 获取帮助

imp help=y

2. 导入一个完整数据库

imp system/manager file=bible_db log=dible_db full=y ignore=y

3. 导入一个或一组指定用户所属的全部表、索引和其他对象

imp system/manager file=seapark log=seapark fromuser=seapark imp

system/manager file=seapark log=seapark fromuser=(seapark,amy,amyc,harold)

4. 将一个用户所属的数据导入另一个用户

imp system/manager file=tank log=tank fromuser=seapark touser=seapark_copy

imp system/manager file=tank log=tank fromuser=(seapark,amy)

touser=(seapark1, amy1)

5. 导入一个表

imp system/manager file=tank log=tank fromuser=seapark TABLES=(a,b)

6. 从多个文件导入

imp system/manager file=(paycheck_1,paycheck_2,paycheck_3,paycheck_4)

log=paycheck, filesize=1G full=y

7. 使用参数文件

imp system/manager parfile=bible_tables.par

bible_tables.par 参数文件：

#Import the sample tables used for the Oracle8i Database Administrator's

Bible. fromuser=seapark touser=seapark_copy file=seapark log=seapark_import

8. 增量导入

imp system./manager inctype= RECTORE FULL=Y FILE=A

Oracle imp/exp

C:Documents and Settingsadministrator>exp help=y

Export: Release 9.2.0.1.0 - Production on 星期三 7月 28 17:04:43 2004

Copyright (c) 1982, 2002, Oracle Corporation. All rights reserved.

通过输入 EXP 命令和用户名/口令，您可以

后接用户名/口令的命令:

例程: EXP SCOTT/TIGER

或者，您也可以通过输入跟有各种参数的 EXP 命令来控制"导出"

按照不同参数。要指定参数，您可以使用关键字:

格式: EXP KEYWORD=value 或 KEYWORD=(value1,value2,...,valueN)

例程: EXP SCOTT/TIGER GRANTS=Y TABLES=(EMP,DEPT,MGR)

或 TABLES=(T1: P1,T1: P2)，如果 T1 是分区表

USERID 必须是命令行中的第一个参数。

关键字 说明(默认) 关键字 说明(默认)

USERID 用户名/口令 FULL 导出整个文件 (N)

BUFFER 数据缓冲区大小 OWNER 所有者用户名列表

FILE 输出文件 (EXPDAT.DMP) TABLES 表名称列表

COMPRESS 导入到一个区 (Y) RECORDLENGTH IO 记录的长度

GRANTS 导出权限 (Y) INCTYPE 增量导出类型

INDEXES 导出索引 (Y) RECORD 跟踪增量导出 (Y)

DIRECT 直接路径 (N) TRIGGERS 导出触发器 (Y)

LOG 屏幕输出的日志文件 STATISTICS 分析对象 (ESTIMATE)

ROWS 导出数据行 (Y) PARFILE 参数文件名

CONSISTENT 交叉表的一致性 (N) CONSTRAINTS 导出的约束条件 (Y)

OBJECT_CONSISTENT 只在对象导出期间设置为读的事务处理 (N)

FEEDBACK 每 x 行的显示进度 (0)

FILESIZE 每个转储文件的最大大小

FLASHBACK_SCN 用于将会话快照设置回以前状态的 SCN

FLASHBACK_TIME 用于获取最接近指定时间的 SCN 的时间

QUERY 用于导出表的子集的 select 子句

RESUMABLE 遇到与空格相关的错误时挂起 (N)

RESUMABLE_NAME 用于标识可恢复语句的文本字符串

RESUMABLE_TIMEOUT RESUMABLE 的等待时间

TTS_FULL_CHECK 对 TTS 执行完整的或部分相关性检查

TABLESPACES 要导出的表空间列表

TRANSPORT_TABLESPACE 导出可传输的表空间元数据 (N)

TEMPLATE 调用 iAS 模式导出的模板名

在没有警告的情况下成功终止导出。

==========

C:Documents and Settingsadministrator>imp help=y

Import: Release 9.2.0.1.0 - Production on 星期三 7月 28 17:06:54 2004

Copyright (c) 1982, 2002, Oracle Corporation. All rights reserved.

可以通过输入 IMP 命令和您的用户名/口令

后接用户名/口令的命令:

例程: IMP SCOTT/TIGER

或者，可以通过输入 IMP 命令和各种参数来控制“导入”

按照不同参数。要指定参数，您可以使用关键字:

格式: IMP KEYWORD=value 或 KEYWORD=(value1,value2,...,vlaueN)

例程: IMP SCOTT/TIGER IGNORE=Y TABLES=(EMP,DEPT) FULL=N

或 TABLES=(T1: P1,T1: P2)，如果 T1 是分区表

USERID 必须是命令行中的第一个参数。

关键字 说明（默认） 关键字 说明（默认）

USERID 用户名/口令 FULL 导入整个文件 (N)

BUFFER 数据缓冲区大小 FROMUSER 所有人用户名列表

FILE 输入文件 (EXPDAT.DMP) TOUSER 用户名列表

SHOW 只列出文件内容 (N) TABLES 表名列表

IGNORE 忽略创建错误 (N) RECORDLENGTH IO 记录的长度

GRANTS 导入权限 (Y) INCTYPE 增量导入类型

INDEXES 导入索引 (Y) COMMIT 提交数组插入 (N)

ROWS 导入数据行 (Y) PARFILE 参数文件名

LOG 屏幕输出的日志文件 CONSTRAINTS 导入限制 (Y)

DESTROY 覆盖表空间数据文件 (N)

INDEXFILE 将表/索引信息写入指定的文件

SKIP_UNUSABLE_INDEXES 跳过不可用索引的维护 (N)

FEEDBACK 每 x 行显示进度 (0)

TOID_NOVALIDATE 跳过指定类型 ID 的验证

FILESIZE 每个转储文件的最大大小

STATISTICS 始终导入预计算的统计信息

RESUMABLE 在遇到有关空间的错误时挂起 (N)

RESUMABLE_NAME 用来标识可恢复语句的文本字符串

RESUMABLE_TIMEOUT RESUMABLE 的等待时间

COMPILE 编译过程, 程序包和函数 (Y)

STREAMS_CONFIGURATION 导入 Streams 的一般元数据 (Y)

STREAMS_INSTANITATION 导入 Streams 的实例化元数据 (N)

下列关键字仅用于可传输的表空间

TRANSPORT_TABLESPACE 导入可传输的表空间元数据 (N)

TABLESPACES 将要传输到数据库的表空间

DATAFILES 将要传输到数据库的数据文件

TTS_OWNERS 拥有可传输表空间集中数据的用户

成功终止导入，但出现警告。

oracle 的 imp 和 exp 的一些用法- -

Oracle8i/9i EXP/IMP 使用经验

一、8i EXP 常用选项

1、FULL，这个用于导出整个数据库，在 ROWS=N 一起使用时，可以导出整个数据库的结构。例如：

exp sysfile=./db_str.dmp log=./db_str.log full=y rows=n compress=y direct=y

2、BUFFER 和 FEEDBACK，在导出比较多的数据时，我会考虑设置这两个参数。例如：

exp newfile=yw97_2003.dmp log=yw97_2003_3.log feedback=10000 buffer=100000000 tables=WO4,OK_YT

3、FILL 和 LOG，这两个参数分别指定备份的 DMP 名称和 LOG 名称，包括文件名和目录，例子见上面。

需要说明的是，EXP 可以直接备份到磁带中，即使用 FILE=/dev/rmt0(磁带设备名)，但是一般我们都不这么做，原因有二：一、这样做的速度会慢很多，二、现在一般都是使用磁带库的，不建议直接对磁带进行操作。至于没有使用磁带库的朋友可以考虑和 UNIX 的 TAR 结合使用。

如果你真想使用 EXP 直接到磁带，你可以参考 Metalink 文章“EXPORTING TO TAPE ON UNIX SYSTEMS”（文档号：30428.1），该文中有详细解释。

4、COMPRESS 参数将在导出的同时合并碎块，尽量把数据压缩到 initial 的 EXTENT 里，默认是 N，一般建议使用。DIRECT 参数将告诉 EXP 直接读取数据，而不像传统的 EXP 那样，使用 SELECT 来读取表中的数据，这样就减少了 SQL 语句处理过程。一般也建议使用。不过有些情况下 DIRECT 参数是无法使用的。

5、如何使用 SYSDBA 执行 EXP/IMP？

这是一个很现实的问题，有时候我们需要使用 SYSDBA 来执行 EXP/IMP，如进行传输表空间的 EXP/IMP，以及在 9i 下用 SYS 用户来执行 EXP/IMP 时，都需要使用 SYSDBA 才可。我们可以使用下面方式连入 EXP/IMP：

exp '"sys/sys as sysdba"' file=1.dmp tables=gototop.t rows=n

6、QUERY 参数后面跟的是 where 条件，值得注意的是，整个 where 子句需要使用""括起来，where 子句的写法和 SELECT 中相同。

exp gototop/gototop file=1.dmp log=1.log tables=cyx.t query="where c1=20 and c2=gototop"

如果是 windows 平台，则使用下面的格式：

exp c/c@ncn file=c.dmp log=c.log tables=t query="""where id=1 and name='gototop'"""

二、8i IMP 常用选项

1、FROMUSER 和 TOUSER，使用它们实现将数据从一个 SCHEMA 中导入到另外一个 SCHEMA 中。

2、IGNORE、GRANTS 和 INDEXES，其中 IGNORE 参数将忽略表的存在，继续导入，这个对于需要调整表的存储参数时很有用，我们可以先根据实际情况用合理的存储参数建好表，然后直接导入数据。而 GRANTS 和 INDEXES 则表示是否导入授权和索引，如果想使用新的存储参数重建索引，或者为了加快到入速度，我们可以考虑将 INDEXES 设为 N，而 GRANTS 一般都是 Y。

另外一个 EXP/IMP 都有的参数是 PARFILE，它是用来定义 EXP/IMP 的参数文件，也就是说，上面的参数都可以写在一个参数文件中，但我们一般很少使用。

三、Oracle9i EXP 功能描述

Oracle9i EXP 在原有的基础上新增了部分新的参数，按功能主要分为以下几个部分：

1、OBJECT_CONSISTENT - 用于设置 EXP 对象为只读以保持对象的一致性。默认是 N。

2、FLASHBACK_SCN 和 FLASHBACK_TIME - 用于支持 FLASHBACK 功能而新增。

3、RESUMABLE、RESUMABLE_NAME 和 RESUMABLE_TIMEOUT - 用于支持 RESUMABLE 空间分配而新增。

4、TTS_FULL_CHECK - 用于在传输表空间时使用依赖性检查。

5、TEMPLATE - 用于支持 iAS。

6、TABLESPACES - 设置表空间导出模式。个人觉得对于一般用户而言，这个才是新增参数中最实用的一个，可以让用户在原来的 FULL、OWNER、TABLES 的基础上多了一种选择，使得 EXP 更加灵活。

四、不同版本的 EXP/IMP 问题？

一般来说，从低版本导入到高版本问题不大，麻烦的是将高版本的数据导入到低版本中，在 Oracle9i 之前，不同版本 Oracle 之间的 EXP/IMP 可以通过下面的方法来解决：

1、在高版本数据库上运行底版本的 catexp.sql；

2、使用低版本的 EXP 来导出高版本的数据；

3、使用低版本的 IMP 将数据库导入到底版本数据库中；

4、在高版本数据库上重新运行高版本的 catexp.sql 脚本。

但在 9i 中，上面的方法并不能解决问题。如果直接使用底版本 EXP/IMP 会出现如下错误：

EXP-00008: ORACLE error %lu encountered

ORA-00904: invalid column name

这已经是一个公布的 BUG，需要等到 Oracle10.0 才能解决，BUG 号为 2261，你可以到 METALINK 上去查看有关此 BUG 的详细信息。

BUG 归 BUG，我们的工作还是要做，在没有 Oracle 的支持之前，我们就自己解决。在 Oracle9i 中执行下面的 SQL 重建 exu81rls 视图即可。

```
CREATE OR REPLACE view exu81rls
(objown,objnam,policy,polown,polsch,polfun,stmts,chkopt,enabled,spolicy)
AS select u.name, o.name, r.pname, r.pfschma, r.ppname, r.pfname,
decode(bitand(r.stmt_type,1), 0,'', 'SELECT,')
|| decode(bitand(r.stmt_type,2), 0,'', 'INSERT,')
|| decode(bitand(r.stmt_type,4), 0,'', 'UPDATE,')
|| decode(bitand(r.stmt_type,8), 0,'', 'DELETE,'),
r.check_opt, r.enable_flag,
DECODE(BITAND(r.stmt_type, 16), 0, 0, 1)
from user$ u, obj$ o, rls$ r
where u.user# = o.owner#
and r.obj# = o.obj#
and (uid = 0 or
uid = o.owner# or
exists ( select * from session_roles where role='SELECT_CATALOG_ROLE')
)
/
grant select on sys.exu81rls to public;
/
```