2Gb or Not 2Gb - File limits in Oracle

翻译：Kamus（Seraphim）
校正：Bloomit
邮件：kamus@itpub.net
日期：2004-1

经常会听说导入导出的时候，备份恢复的时候，SQL*Loader 导入数据的时候，文件超出了 2G 大小，结果导致错误。
本人文科毕业，什么二进制，十六进制，数据结构，操作系统等等的一概没有学过，所以对此问题一直都只有一个模糊的认识，今天在 metalink 上面闲逛，忽然发现了这篇文章，兴之所致，决定好好看看，看完以后，感觉对于此问题明白了很多，于是就顺便翻译成了中文，希望对大家也有所帮助。
对于本文中提及的所有其它 Notes 和 Bugs 也做了尽可能的整理（还未翻译，以后如果有时间再慢慢作），只是文中提到的一些 Bug 由于本人 Metalink 帐号的限制亦或是 Bug 的不公开性，并不能看到详细描述，对于这样的 Bug 就无法再作整理。

原文出处：http://metalink.oracle.com
Doc ID: Note:62427.1 原文创建日期：1998-9-2 原文最后更新日期：2003-8-6

介绍

本文阐述了“2Gb”问题，解释了为什么 2Gb 会是一个这样充满魔力的数字，并且如果你想在 Oracle 的应用中使用超过 2Gb 大小的文件，那么这篇文章也告诉了一些你应该知道的事情。

本文以 Unix 操作系统为基础，因为大部分的 2Gb 问题都发生在 Unix 上面，当然文中也提到了一些对于其它非 Unix 操作系统的相关资料，在本文的最后一节列出了各个操作系统自己的限制。

本文主题包括以下几点：

- 为什么 2Gb 是一个特殊的数字？
- 怎样使用超过 2Gb（2Gb+）的文件？
- 导出（Export）和 2Gb
- SQL*Loader 和 2Gb
- Oracle 和其它 2Gb 问题
- 不同操作系统上的“大文件”(Large Files)

为什么 2Gb 是一个特殊的数字？

很多当前正在使用的 CPU 和 API 都使用了 32 位（bit）的字长，而正是这个字长对于很多操作产生了影响。

众多的场合下文件操作的标准 API 在对文件大小和文件中当前位置的处理都使用了有符号

32 位字（32-bit signed word）。一个有符号 32 位字以最高位来表示正负，所以只剩下 31 位来存储真正的数值，而 16 进制中存储在 31 位中的最大正值就是 0x7FFFFFFF，也就是 10 进制中的+2147483647，这正是一个临近 2Gb 的值。

2Gb 或者更大的文件一般被称为“大文件”，当你在 32 位环境中使用 2147483647 甚至更大的数字时，你就很可能会碰到一些问题。为了解决这些问题，最新的操作系统已经重新定义了一系列完全利用 64 位寻址方式来操作文件大小和偏移量的系统函数。最新的 Oracle 发行版也已经使用了这些新的接口，但是如果在你决定要使用“大文件”以前，你仍然有不少的问题需要考虑。

另外一个特殊的数字是 4Gb。也就是作为无符号字（unaigned value）的十六进制数字 0xFFFFFFFF（十进制是 4294967295），这是一个略小于 4Gb 的值。将该值加一将使低 4 位字节成为 0x00000000，同时产生'1'进位。这个进位在 32 位运算中将会失去。所以 4Gb 也是一个可能会产生问题的特殊数字。本文中对这个问题也有所提及。

对于使用 Oracle 这意味着什么？

32bit 的问题在不少方面都影响到了 Oracle，为了使用“大文件”，你需要满足以下条件：

1. 一个支持 2Gb+文件的操作系统或者裸设备（Raw devices）
2. 一个具有支持存取 2Gb+文件的 API 的操作系统
3. 一个使用了这些 API 的 Oracle 版本

今天大多数的平台都已经支持了大文件，并且对于这些文件有 64bit 的 API，Oracle7.3 及以后版本一般已经使用了这些 API，但是根据不同的平台，不同的操作系统以及不同的 Oracle 版本仍然有很多不一样的情况。在一些场合下默认就是支持“大文件”的，但是另外一些场合却可能必须要打一些补丁。

一直到写这篇文章的时候，Oracle 里面还有一些工具是没有更新到使用这些新的 API 的，比如众所周知的 EXPORT 和 SQL*LOADER，不过仍然再次强调一下，由于平台和操作系统的不同，情况还是不一样的。

为什么要使用 2Gb+的文件？

在这一节中我们试着总结一下对于 Oracle 的数据文件使用大文件和设备（"large" files / devices）的优点以及缺点。

使用大于 2Gb 文件的优点：

- 在大多数平台上 Oracle7 支持最多 1022 个数据文件。如果文件小于 2G，那么也就是限制了数据库的大小只能是小于 2044Gb。当然这对于支持了更多数据文件的 Oracle8 来说已经不再是个问题（Oracle8 支持每个表空间中包含最多 1022 个数据文件）。
- 在现实情况中 Oracle7 的最大数据库尺寸会比 2044Gb 小，因为一般数据文件都存放在单独的表空间中，而很多数据文件就可能远远小于 2Gb。使用大文件可以让数据库超越 2044Gb 的这个限制。
- 使用大文件意味着对于较小的数据库只需要管理较少的文件。

- 需要较少的文件处理资源。

使用大于 2Gb 文件的缺点：

- 恢复的单位更大了。一个 2Gb 文件的备份和还原，根据备份媒体和磁盘速度的差异，会耗费 15 分钟到 1 个小时的时间，那么一个 8Gb 的文件就要花 4 倍这样的时间。
- 备份和恢复的并行操作新能将会收到影响。
- 会碰到一些平台特有的限制，比如说在超过 2Gb 以上异步 I/O 的操作就可能会变成线性操作。
- 处理 2Gb 以上的文件可能会需要补丁或者一些特殊的配置。相对于小文件来说也会有更大的风险性。比如在一些 AIX 的发行版上，超过 2Gb，异步 I/O 就会使用线性操作。

使用大于 2Gb 文件的要点：

- 跟操作系统提供商确认，大文件是否被支持，同时要如何去配置。
- 跟操作系统提供商确认，真正的最大文件限制是多少？
- 询问 Oracle 技术支持，确定对于你现有的平台，操作系统版本，Oracle 版本，是否需要什么补丁或者还有什么限制？
- 记住，如果你真的考虑对于操作系统或者 Oracle 要打一些补丁的话，那么就再检查一遍上面提到的这些问题。
- 确认对于所有要使用大文件读取的用户来说，操作系统的限制已经正确设定。
- 确认所有的备份脚本都能够处理大文件。
- 注意，对于使用超过 2Gb 的数据文件，对于最大文件大小还有一个限制。这个限制依赖于你的系统平台以及 Oracle 初始化参数 DB_BLOCK_SIZE。在大多数平台上（包括 Unix, NT, VMS）文件大小的限制在 4194302*DB_BLOCK_SIZE 这么大。

[NOTE:112011.1]文档描述了改变文件大小中存在的问题，特别是超过了 2Gb 的时候。

一般需要注意的要点：

需要小心设置文件的自动扩展。明智的作法是在不使用“大文件”的场合下，将自动扩展的数据文件最大尺寸限制在 2Gb 以下。另外注意由于[BUG:568232]将有可能定义一个超过 Oracle 处理极限的 MAXSIZE 数值，这在 resize 之后将会引发一个内部错误（错误显示为 ORA-600 [3292]）

在很多平台上，Oracle 数据文件的头部都包含着附加的数据块，所以创建一个 2Gb 的数据文件实际上需要比 2Gb 更多的磁盘空间。在 UNIX 平台上数据文件头部的附加数据块大小通常等于 DB_BLOCK_SIZE 的大小，但是在裸设备上可能需要占用更多一些的空间。

2Gb 相关的 Oracle 错误

当 2Gb 限制到达的时候可能会发生下面这些错误，这些错误的产生没有特定的顺序。

ORA-01119 Error in creating datafile xxxx
ORA-27044 unable to write header block of file
SVR4 Error: 22: Invalid argument
ORA-19502 write error on file 'filename', blockno x (blocksize=nn)
ORA-27070 skgfdisp: async read/write failed
ORA-02237 invalid file size

KCF:write/open error dba=xxxxxx block=xxxx online=xxxx file=xxxxxxxx file limit exceed.
Unix error 27, EFBIG

导出（Export）和 2Gb

2Gb 导出文件的大小

当编写大部分版本的 Export 时，在创建导出文件上都是使用了默认的文件操作 API。这就意味着在很多平台上根本就没有可能导出 2Gb 或者大于 2Gb 的文件系统文件（file system file）。
但是仍然有一些可选项可以用于在 Export 时解决 2Gb 的限制：

- ✓ 将大于 2Gb 的文件导出到裸设备上基本上是没有问题的，当然这首先要求裸设备的大小必须能够容纳整个导出文件。
- ✓ 导出到一个允许压缩或者切割的命名管道中（适用 Unix 平台）。
  参看“在 Unix 平台上导出大于 2Gb 文件的快速参考”一文 [NOTE:30528.1]。
- ✓ 导出到磁带（适用大多数平台）
  参看“在 Unix 系统中导出到磁带”一文[NOTE:30428.1]。（这篇文章同时页详细描述了如何导出到 Unix 管道和远程 shell 中）
- ✓ Oracle8i 允许导出到多个小文件中，以替代单一的大文件。

其它的 2Gb 导出问题

Oracle 允许区（extent）的尺寸最大为 2Gb。但是不幸的是，在大多数的 Oracle 发行版中 Export 都存在这样一个问题，当你 Export 一个大文件，并且指定了 COMPRESS=Y，那么就有可能在导出文件的 NEXT 存储子句中包含了一个大于 2Gb 的值。这样将会导致 Import 失败，即使是在 Import 时候指定了 IGNORE=Y。Oracle 已经在在[BUG:708790]中报告了这个问题，并且在[NOTE:62436.1]中提出了警告。

当 Export 碰到 2Gb 限制的时候，会报类似下面的错误：

```
. . exporting table                        BIGEXPORT
EXP-00015: error on row 10660 of table BIGEXPORT,
     column MYCOL, datatype 96
EXP-00002: error in writing to export file
EXP-00002: error in writing to export file
EXP-00000: Export terminated unsuccessfully
```

在[BUG:185855]中提到了第二个问题，这个问题指出一个全库导出产生的 CREATE TABLESPACE 命令将在文件大小上使用 BYTES 为单位，如果文件大小超过 2Gb，那么在导入的时候就会产生一个 ORA-2237 错误。这个问题可以通过在导入之前先以 M 为单位而不是 BYTES 为单位来创建表空间这样的方法来解决。[BUG:490837]也指出了相类似的问题。

导出到磁带

导出的时候 VOLSIZE 参数限制在 4Gb 以下，在有些平台上可能只有 2Gb。
在 Oracle8i 中已经修正了这个问题。[BUG:490190]中对此问题有所描述。

SQL*Loader 和 2Gb

在 SQL*Loader 试图打开一个超过 2Gb 的文件时，将会报以下错误：

SQL*Loader-500: Unable to open file (bigfile.dat)
SVR4 Error: 79: Value too large for defined data type

在[NOTE:30528.1]中的例子可以稍作修改以使在 SQL*Loader 中使用大的输入文件。
Oracle 8.0.6 在 SQL*Loader 中已经对 discard file 和 log file 实现了大文件支持，但是对于输入的 data file 在各个平台上仍然时不一样的。[BUG:948460]中记录了输入文件大小限制的详细信息。[BUG:749600]则记录了最大的 discard file 文件大小。

Oracle 和其它的 2Gb 问题

这个章节列举了其它各色 2Gb 问题。

- Oracle 8.0.5 版本以后在大部分的平台上 Oracle 都提供了 64 位的版本。从 8.0.5 的 README 文件中可以看到相应的介绍－[NOTE:62252.1]
- DBV（数据库验证程序）可能无法扫描超过 2Gb 的数据文件，并会报 DBV-100 错误。在[BUG:710888]中报告了此错误。
- 如果要在 Oracle 中创建大于 2Gb 的文件， SQL 命令行的"DATAFILE ... SIZE xxxxxx" 子句部分必须以 M 或者 K 作单位来指定，否则将会报"ORA-02237: invalid file size"错误。在[BUG:185855]中报告了此错误。
- 在 Oracle 7.3.4 发行版以前表空间的限额不能超过 2Gb。比如：
  ALTER USER <username> QUOTA 2500M ON <tablespacename>
  这样将会报" ORA-2187: invalid quota specification."错误。
  在[BUG:425831]中报告了此错误。解决方法是如果一个用户需要超过 2Gb 的限额，那么就给他赋予 UNLIMITED TABLESPACE 权限。
- 如果 spool 的输出文件达到了 2Gb，那么会出现错误。比如：SQLPLUS 的命令 spool。
- 在 Oracle 工具中的一些 CORE 函数不支持大文件。[BUG:749600]中报告了此错误，在 Oracle 8.0.6 和 8.1.6 版本中已经修正了。但是要注意在 Oracle 8.1.5 和别的任何补丁中都没有修改这个错误。另外即使已经有修正，但是仍然还会有大文件限制因为不是所有的代码都使用了这些 CORE 函数。
  注意：[BUG:749600]虽然阐明了 CORE 函数，但是代码的某些部分仍然有问题。比如：SQL*Loader 中输入文件的读取就没有使用 CORE。
- UTL_FILE 包使用了上述的 CORE 函数，所以在没有修正的 Oracle 版本中仍然有 2Gb 限制。<Package:UTL_FILE>是一个允许在 PL/SQL 中进行文件存取的 PL/SQL 包。

特定平台中的大文件

下面是一些特定平台中关于大文件支持的参考资料。虽然我们已经努力使这些文章的资料始终保持更新，但是仍然建议在存取大文件时对每一个操作要小心谨慎地测试。

平台	参考
AIX (RS6000 / SP)	[NOTE:60888.1]
HP	[NOTE:62407.1]
Digital Unix	[NOTE:62426.1]
Sequent PTX	[NOTE:62415.1]

Sun Solaris	[NOTE:62409.1]
Windows NT	FAT 文件系统支持最大 4Gb 的文件 NTFS 文件系统理论上支持最大 16Tb 的文件 1．在 NT 的 Oracle8 上使用大文件之前请先参考[NOTE:67421.1] 2．Oracle8.1.6 的 DBVERIFY 程序有问题（参考[BUG:1372172]） 3．在 8.1.6 / 8.1.7 中自动扩展到 4Gb 时会出现问题导致数据库崩溃。（参考[BUG:1668488]）

附录：

[BUG:1372172]

DBV FAILS AGAINST FILES GREATER THAN 2GB WITH DBV-102

***** 08/04/00 04:28 am *****
Problem Description

When DBV is run against datafiles files greater than 2Gb on Windows NT or
Windows 2000 it fails with :
DBV-00102 File I/O error on FILE (<filename>) during end read operation (-2)
.
This error is returned immediately.
.
Diagnostic Analysis

I have tested files up to 2047Mb and DBV works correctly.
Files from 2048Mb fail with this error.
.
Workaround

Customers can use DBV Version 8.1.5 against later 8.1.x database files but
this is not usually recommended.
.
Reproducibility

Problem does not reproduce with DBV versions 8.0.5, 8.0.6 or 8.1.5
Problem reproduced on 8.1.6 and 8.1.7 beta drop #6
.
***** 08/04/00 04:45 am *** (CHG: Sta->10)**
***** 08/04/00 04:45 am *****
***** 08/04/00 07:18 am *** (CHG: Sta->16)**
***** 08/04/00 07:19 am *** (CHG: Sta->11)**

***** 08/04/00 07:19 am *****
Screened
***** 08/04/00 07:19 am *****
***** 08/04/00 07:44 am *** (CHG: Asg->NEW OWNER)**
***** 08/09/00 12:16 pm *** (CHG: Asg->NEW OWNER)**
***** 08/09/00 09:52 pm *** (CHG: Asg->NEW OWNER)**
***** 08/19/00 11:10 am *** (CHG: Asg->NEW OWNER)**
***** 08/30/00 10:24 am *** (CHG: Asg->NEW OWNER)**
***** 08/30/00 10:24 am *****
***** 08/31/00 12:41 pm *** (CHG: Asg->NEW OWNER)**
***** 09/27/00 04:13 pm *** (CHG: Asg->NEW OWNER)**
***** 10/02/00 01:38 am *****
.
Possible workarounds are:
Use DBV from a different Oracle release
(This requires the ORACLE_HOME for the alternative release to be installed as the DBV executable cannot be run stand alone)
Use RMAN for backup or copy of files
(The CHECK LOGICAL clause checks logical block integrity)
***** 10/02/00 02:17 am *** ESCALATED**
***** 10/02/00 02:18 am *****
STESC - Escalating for Silver customer: PSR (UK)
Tracking TAR: 2426694.1
***** 10/12/00 01:24 pm *** (CHG: Sta->35)**
***** 10/12/00 01:24 pm *** (CHG: Confirmed Flag->Y)**
***** 10/12/00 01:24 pm *** (CHG: Fixed->8.1.7)**
***** 10/12/00 01:24 pm *****
A fix has been made in 8.1.7 and 8.2 and should be available in to next release. The Q/A, filer needs to verify the fix on the actual test case it should have.
***** 10/16/00 01:52 am *** ESCALATION -> CLOSED**
***** 11/30/00 07:38 am *** (CHG: Sta->90)**
***** 11/30/00 07:39 am *****
Verified as fixed in 8.1.7 Release Candidate

[BUG:1668488]

UNABLE TO RECOVER DATAFILE THAT HAS AUTO EXTENDED ONTO A 4GB BOUNDARY

***** 03/02/01 08:56 am *****
Problem Description

==========

When a datafile has been configured such that it will autoextend onto a
4GB boundary, any attempt to autoextend the file causes the database to
crash (when in noarchive log mode). Checkpoint writer (CKPT) and database
writer (DBWR) report :

.

```
KCF: write/open error block=0x32002 online=1
        file=5 E:\ORACLE\ORADATA\V817\XXX.DBF
        error=27069 txt: 'OSD-04026: Invalid parameter passed. (OS 204802)'
```

.

```
ksedmp: internal or fatal error
ORA-01242: data file suffered media failure: database in NOARCHIVELOG mode
ORA-01114: IO error writing block to file 5 (block # 204802)
ORA-01110: data file 5: 'E:\ORACLE\ORADATA\V817\XXX.DBF'
ORA-27069: skgfdisp: attempt to do I/O beyond the range of the file
OSD-04026: Invalid parameter passed. (OS 204802)
```

.

The database can be reopened and appears to work without error. If the
database was in archive log mode, the file will be marked offline recover
and access to the database continues. The following error appears in the
alert.log :

.

```
KCF: write/open error block=0x32002 online=1
        file=5 E:\ORACLE\ORADATA\V817\XXX.DBF
        error=27069 txt: 'OSD-04026: Invalid parameter passed. (OS 204802)'
Automatic datafile offline due to write error on
file 5: E:\ORACLE\ORADATA\V817\XXX.DBF
```

.

Any attempt to online / recover this file results in :

.

```
ORA-00283: recovery session canceled due to errors
ORA-01115: IO error reading block from file 5 (block # 204802)
ORA-01110: data file 5: 'E:\ORACLE\ORADATA\V817\XXX.DBF'
ORA-27069: skgfdisp: attempt to do I/O beyond the range of the file
OSD-04026: Invalid parameter passed. (OS 204802)
```

.

The only way a customer can recover this file is to restore the whole
database and roll forward to a point before the file autoextended, because
we replay the autoextend on recover. In all cases the file will already
be at the new autoextended size.

.

The session that causes the autoextension receives ORA-3113.

.

Diagnostic Analysis

==========
Many different configurations of autoextend have been tested, including creating objects, allocating extents and performing DML so that the file autoextends onto the 4Gb boundary, all cause the same problem.
.
Resizing files below, on and above the 4GB boundary works correctly.
.
Workaround
==========
None if it has already happened and the customer is in archive log mode.
Restart the database if in noarchive log mode.
Avoid autoextend where files can autoextend onto 4GB boundary.
.
Related Bugs
==========
BUG:1612154
.
Reproducibility
==========
Does not reproduce on 8.1.7 Sun Solaris.
Reproduces on 8.1.6.3 and 8.1.7 Windows NT / 2000.
No testing has been performed at the 8GB, 12Gb or 16GB boundaries.
.
Testcase
========
Running the following script reproduces the problem :
.

```
create tablespace XXX
datafile 'E:\ORACLE\ORADATA\V817\XXX.DBF' size 3072M
autoextend on next 1024M
default storage (initial 10M next 10M);
```

.

```
create table ddd1 (col1 number) tablespace xxx storage (initial 1600M);
create table ddd2 (col1 number) tablespace xxx storage (initial 1600M);
```

.

```
alter system checkpoint;
```

.
***** 03/04/01 10:37 pm *** (CHG: Asg->NEW OWNER)**
***** 03/22/01 07:31 pm *****
***** 04/11/01 02:27 pm *** (CHG: DevPri->4)**
***** 04/11/01 02:29 pm *** (CHG: DevPri->2)**
***** 04/20/01 05:36 am *****
I have added customer (Oracle Consultant - tar 1336962.996) to this bug.
***** 04/23/01 04:25 pm *** (CHG: Asg->NEW OWNER)**

***** 04/23/01 04:25 pm *****
I'm able to reproduce this bug against 8.1.7
***** 04/27/01 05:10 pm *****
update: While debugging I found the value of last block is 8192,
it should be 524289 (4G/8K + 1).
After reproducing the problem, the size of the file xxx.dbf
becoming 8192 bytes. The resize function is not working properly.
***** 05/03/01 10:42 am *****
The above update is incorrect. please ignore. resize function seems
to be working fine. new observation and work around is:
if the new filesize after extended is not exactly equal to 4G, it works
fine. so this is some boundary issue.
***** 05/03/01 11:05 am *****
My customer (already linked to bug) is experiencing this on
Platform: Windows 2000
Version: 8.1.6
***** 05/03/01 11:10 am *****
The above workaround didnt help your customer?
***** 05/03/01 05:49 pm *****
***** 05/04/01 11:57 am *****
Workaround : calculate storage option to make extended file size to be
atleast 1M morethan 4G.
***** 05/09/01 03:51 pm *****
fixed and running regression tests...
***** 05/15/01 02:34 pm *** (CHG: Confirmed Flag->Y)**
***** 05/15/01 02:34 pm *****
adding testcase, so this delay
***** 05/16/01 11:33 pm *****
***** 05/17/01 04:32 am *****
***** 05/18/01 12:31 pm *** (CHG: Sta->80)**
***** 05/18/01 12:31 pm *** (CHG: Fixed->9.0.2)**
***** 05/18/01 12:31 pm *****
Rediscovery Information:
Resize a datafile file to 4GB, 8G, 12G, 16G, etc.
]] After resizing a datafile to 4G, alter system checkpoint
]] was failing with ORA-27069
***** 05/21/01 08:45 am *****
***** 05/23/01 06:41 pm *****
***** 05/25/01 05:53 pm *****
***** 06/28/01 10:08 am *****
.
***** 04/01/02 10:49 am *****
***** 04/02/02 03:28 pm *****
***** 10/02/02 10:08 am *****

***** 11/16/02 01:54 pm *****
***** 11/22/02 04:19 pm *** (CHG: Sta->11)**
***** 11/22/02 04:19 pm *****
***** 12/06/02 09:54 am *****
This fix is in 8.1.7.4.1 and 9.0.1.4.0 but not in 9.2.0.2.1
***** 12/06/02 09:55 am *** ESCALATED**
***** 12/06/02 10:47 am *****
***** 12/20/02 11:41 am *****
***** 12/23/02 11:46 am *** (CHG: Sta->80)**
***** 12/23/02 11:46 am *** (CHG: Fixed->10.0)**
***** 12/23/02 11:46 am *****
***** 12/27/02 01:03 am *** ESCALATION -> CLOSED**
***** 01/08/03 07:16 pm *****
***** 02/26/03 11:41 am *****
***** 02/26/03 11:43 am *****
***** 02/27/03 06:53 am *****
***** 03/18/03 08:43 pm *****
***** 03/18/03 08:44 pm *****
***** 03/19/03 12:30 am *****
Verified fix in 8.1.7.4.1 / 9.0.1.4.0 / 9.2.0.3.0 for Windows.
***** 07/18/03 10:14 am *****
***** 07/18/03 10:16 am *****
***** 09/02/03 12:58 pm *****
***** 09/02/03 01:01 pm *****

[BUG:708790]

CT IS RUNNING INTO BUG NUMBER 617486

***** 07/30/98 06:27 pm *****
ct is running into bug number 617486
and needs a special image of imp so that they can run
an import of their production database.
they are down because they do not have a backup of the database
the contact is
***** 07/30/98 08:32 pm *** (CHG: Sta->30)**
***** 07/30/98 10:28 pm *****
***** 08/06/98 05:35 pm *****
***** 08/08/98 02:50 am *** (CHG: Asg->NEW OWNER)**
***** 08/08/98 02:50 am *****
***** 08/08/98 07:08 am *****
***** 08/08/98 08:39 am *****

***** 08/08/98 01:05 pm *****
***** 08/09/98 02:15 am *****
***** 08/10/98 05:33 am *****
***** 08/11/98 09:09 pm *****
***** 08/13/98 10:51 pm *****
***** 08/13/98 11:11 pm *****
***** 08/13/98 11:11 pm *** (CHG: Confirmed Flag->Y)**
***** 08/13/98 11:29 pm *****
]] Import was giving error ORA-2219 when processing some DDL statements
***** 08/14/98 12:56 am *** (CHG: Sta->80)**
***** 08/14/98 12:56 am *** (CHG: Fixed->8.1.4)**
***** 08/14/98 12:56 am *****
***** 09/10/98 04:53 am *****
***** 09/10/98 06:17 pm *****
***** 09/12/98 03:05 am *** (CHG: Fixed->8.0.6)**
***** 09/12/98 03:05 am *****
***** 11/30/98 06:36 pm *****
***** 12/29/98 04:09 pm *** (CHG: Fixed->7.3.4.4)**
***** 12/29/98 04:09 pm *****
***** 01/24/99 10:29 pm *****
***** 01/27/99 10:06 pm *****
***** 03/31/99 03:31 am *****
***** 04/05/99 09:54 pm *** (CHG: Fixed->7.3.3.7)**
***** 04/05/99 09:54 pm *****

[NOTE:112011.1]

ALERT: RESIZE or AUTOEXTEND can "Over-size" Datafiles and Corrupt the Dictionary

*** This alert was updated on 09-Mar-2001 to add details of Bug:1646512

Oracle can allow OVERSIZED Datafiles in the Database
~~~~~~
This alert covers: [BUG:568232] [BUG:925105] [BUG:813983] and [BUG:1646512]

All SQL in this note should be run when connected as the SYS user.
eg: connect internal, connect / as sysdba or connect sys/sys_password
Versions Affected
~~~~~~~~~~~~~~~~~~
The problems described here affect many Oracle releases thus:
7.1 to 7.3.4.5 inclusive
8.0 to 8.0.6.3 inclusive

8.1.5 to 8.1.7.3 inclusive

The individual problems are addressed in various releases.
Individual fix versions are described below in the "Patches" section.

Platforms Affected
~~~~~~~~~~~~~~~~~~

GENERIC - these problems can affect all platforms
except OS/390 which does NOT support datafile resizing.

Description
~~~~~~~~~~~

There are three bugs covered by this alert but all four can result in
the same form of dictionary corruption.

The underlying problem is that an Oracle datafile can have, at most,
4194303 Oracle datablocks. But, certain operations allow this value to
be exceeded resulting in corruption to the data dictionary and
subsequent ORA-600 errors.

Likelihood of Occurrence
~~~~~~~~~~~~~~~~~~~~~~~~

An 'oversized' file is one with more than 4194303 Oracle data blocks.
As the DB_BLOCK_SIZE is set at database creation time the actual
maximum file size for a given database (barring any port specific
limits, notably 2Gb) can be found using the statement:

```
SELECT to_char(4194303*value,'999,999,999,999')||' bytes' MAX_FILE_SIZE
  FROM v$parameter WHERE name='db_block_size';
```

The following operations can cause a file to contain too many Oracle
blocks:

Bug:813983
~~~~~~~~~~

Resizing a datafile allows you to resize to a size larger
than Oracle should allow.
Eg: ALTER DATABASE DATAFILE 'xxxx' RESIZE xxxM;

Bug:568232
~~~~~~~~~~

It is possible to set AUTOEXTEND on a datafile with a MAXSIZE
~~~~~~~~~~

which exceeds the maximum allowable size.
Eg: ALTER DATABASE DATAFILE 'xxxx' AUTOEXTEND ON MAXSIZE xxxM;

Bug:925105
~~~~~~~~~~
It is possible to issue an 'ADD DATAFILE' command without
specifying a size (eg: on a RAW device or if a file already
exists with the name being added).
On some platforms this sets the file size to a default value of
4 billion blocks which is above the maximum 4194303 blocks.
Eg: ALTER TABLESPACE test ADD DATAFILE '/dev/rdsk/dummy';
(the lack of a SIZE on this command can cause the problem)

Bug:1646512
~~~~~~~~~~
It is possible to issue an 'ADD DATAFILE' command with a SIZE
larger than should be allowed. ie: Above 4194303 DB blocks.
Eg: ALTER TABLESPACE ts ADD DATAFILE '/dev/rdsk/ts' size 1073741824 M;
This only occurs for very specific sizes - most invalid sizes will
raise an error.

Workaround
~~~~~~~~~~
The workaround for all of the above problems is not to use the commands
described in this alert with sizes above the maximum for your database
DB_BLOCK_SIZE.

As sizes are often specified in "K" or "M" never try to use file sizes
greater than the values given by the following select:

```
SELECT to_char(4194303*value,'999,999,999,999') MAX_BYTES,
       to_char(trunc(4194303*value/1024),'999,999,999')||' Kb' MAX_KB,
       to_char(trunc(4194303*value/1024/1024),'999,999')||' Mb' MAX_MB
  FROM v$parameter WHERE name='db_block_size';
```

For convenience the table below shows the maximum sizes for common
DB_BLOCK_SIZES:

DB_BLOCK_SIZE	Max Mb value to use in any command
2048	8191 M
4096	16383 M
8192	32767 M
~~~~~~~~~~

16384 65535 M

Note: For a 2K (2048 byte) DB_BLOCK_SIZE an 8Gb datafile is TOO LARGE.
An 8Gb file would be 8192Mb which is more than 4194303 DB blocks.

Possible Symptoms

The symptoms of an oversized datafile include any of the following internal errors:

ORA-600 [25012]
ORA-600 [3292]
ORA-600 [4375]
ORA-600 [2847]
ORA-600 [KCFNEW_1]

Errors typically occur during:

ALTER DATABASE DATAFILE '...' RESIZE ...M;
or
DROP TABLESPACE ...;
or
When a user session tries to use space beyond the 4194303 block mark in the file.

Note that the above ORA-600 errors do not mean you have hit one of these bugs as there are other possible causes for these errors.

Checking for Problem Files

The following statements will show if you have a problem due to one of the above bugs:

1. Check for Oversized files:

```
SELECT f.ts#, f.file#, f.status$, f.blocks, v.name
   FROM file$ f, v$datafile v
      WHERE f.blocks > 4194303
    AND f.file#=v.file#
;
```

If this shows any rows then go to the section below entitled "What to do if I have an oversized file". If no rows are returned

go the step 2.

2. Check for files that could extend too far:

```
SELECT x.file#, x.maxextend , v.name
   FROM filext$ x, v$datafile v
  WHERE x.maxextend > 4194303
    and v.file#=x.file#
;
```

If this reports ORA-942 then you have no files which can over-extend.
(If filext$ does not exist you have no files with AUTOEXTEND set)

If this reports "no rows selected" then you have no files which can
over-extend.

If there are rows returned then reset the maximum file size for the
file.
Eg: ALTER DATABASE DATAFILE 'filename' AUTOEXTEND ON MAXSIZE xxxM;
where 'xxxM' is less than the maximum allowed (see the
select in "Workaround" above).

3. Check for zero length files:

```
SELECT f.ts#, f.file#, f.status$, f.blocks, v.name
   FROM file$ f, v$datafile v
      WHERE f.blocks = 0
    AND f.file#=v.file#
;
```

If this shows any rows then go to the section below entitled
"What to do if I have an oversized file".

What to do if I have an 'oversized' file
~~~~~~

Once a file has been 'over-sized' in the database then there are three
possible options as described below. In ALL cases it is advisable to take
a full backup of your current situation before proceeding.

a. If there have been no ORA-600 errors and no space has been
allocated in the illegal part of the file then you should
be able to drop the tablespace including its existing contents.
~~~~~~

ie: If the select below returns NO ROWS then you can attempt
to drop the tablespace including contents:

SELECT * FROM uet$ WHERE block#+length-1 > 4194303;

NOTE-A: You should extract any required data from the tablespace
before you drop it.

b. The second option is to recover the entire database to a point in
time BEFORE the problem was introduced. For this you need to
examine the alert log and find the earliest time when any file
extended / resized / got added to the database which was oversized.
Perform point in time recovery to a time before any file became
oversized.

c. Consult Oracle Support Services with all the information
collected so far and ask them to refer to this alert.

Patches
~~~~~~~

Please contact Oracle Support to find out if a patch is available for
your platform / version or consider upgrading to 8.1.7 if available.
The individual bugs are fixed in releases as below:

Bug:813983 8.1.6.0 onwards
Bug:568232 8.0.5.0 onwards (including 8.0.6.X and 8.1.X)
Bug:925105 8.1.7.0 onwards
Bug:1646512 8.1.7.4.

None of the bugs are fixed in any 7.3 patch set.

References
~~~~~~~~~~

Manual RESIZE allows more than 4 million DB Blocks	[BUG:813983]
Autoextend MAXSIZE can be set above 4 million DB blocks	[BUG:568232]
ADD DATAFILE with no SIZE can add a bad dictionary entry	[BUG:925105]
ADD DATAFILE allows SIZE larger than should be possible	[BUG:1646512]
2Gb or not 2Gb (2Gb related issues)	[NOTE:62427.1]

[NOTE:30428.1]

Exporting to Tape on Unix System
INTRODUCTION

The following article discusses the various ways you can perform an export
to tape on unix systems. This includes exporting directly to tape as well
as exporting via a unix named pipe. Also, a discussion on calculating how
big a size we can expect the resulting export file to be.

EXPORTING TO TAPE

It is possible on most unix platforms, to perform a database export
directly to a magnetic tape device. This is not a preferred practice,
since it introduces extra points of failure into your export strategy,
but it may be necessary for the following reasons :

1. Lack of disk space - There is not enough disk space to perform
   the export to disk.

2. Resultant export file will be greater than 2 Gigabytes -
   On some platforms / Oracle versions there is a restriction of 2G
   on the size of an export dump file.

Anyone who chooses to use this method to perform an export should test it
thoroughly before implementing it on their systems.

NOTES

In this article all of the examples are using the command line method of
export, rather than the interactive method. Exactly the same principles
are involved in using the interactive export, except export cannot
be run as a background job (Using '&'.). This means that if
you want to run export to tape via named pipes using the interactive
method, you will need to use two windows.

This article also only deals with parameters that directly affect exporting
to tape. Other command line parameters are referenced as <other options>.
To find out what the command line arguments are for your version of export
type 'exp help=y' as a valid oracle user.

Please note that some of the methods used in this article use unix concepts
e.g. named pipes, and their use may vary on different unix ports.
If you have a problem with any of the unix commands in this article please
check the syntax against the vendors manual pages, and then finally with

the vendor.

CALCULATING THE SIZE OF AN EXPORT FILE

If the site is unsure how large a resultant export file will be, they
can use the following commands to calculate its size :

1. Create a unix named pipe :

```
os> mknod /tmp/exp_pipe p
```

2. Start the export in the background, specifying the named pipe as
the output file :

```
os> exp file=/tmp/exp_pipe <other options> &
```

3. dd in from the named pipe, out to /dev/null in 1K blocks :

```
os> dd if=/tmp/exp_pipe of=/dev/null bs=1024
```

This will return the size of the export file in 1K blocks as follows:

```
<no. of 1K blocks>+0 records in
<no. of 1K blocks>+0 records out
```

EXPORTING DIRECTLY TO TAPE

Once you have decided that you need to perform the export directly to tape,
you will need to change the syntax of your export statement so that
export knows the name of the tape device and how much data can be written
to that tape device.

There are only 2 parameters that you need to change to do this :

1. FILE - The name of the tape device you are exporting to
e.g. /dev/rmt/0 .

2. VOLSIZE - The amount of data that can be written to one tape.

If the entire export file will fit onto a single tape, then a volsize of
0 (variable length) can be used i.e. exp <other options> volsize=0.

If the resultant export file is larger than a single tape, then volsize
needs to be set accordingly e.g. exp <other options> volsize=<size>M.
Please note, there is a 4 Gigabyte limit to the volsize parameter.

EXAMPLE OF EXPORTING DIRECTLY TO TAPE

A full database export to a QIC 150 (150 Megabytes capacity) to a
tape device "/dev/rmt/0m",where the resultant file is 200M would be
exported using the following syntax :

```
os> exp userid=system/manager full=y file=/dev/rmt/0M volsize=145M
```

The volsize is set to 145M, even though the tape is a QIC 150 because
there may not be exactly 150M of tape on the volume.
Once the export utility has written <volsize> to the tape, it will
prompt for the next tape:
"Please mount the next volume and hit <ret> when you are done."

To import from the resultant export on tape, the following commands
would be used :

```
os> imp userid=system/manager full=y file=/dev/rmt/0M volsize=145M
```

EXPORTING TO TAPE VIA UNIX NAMED PIPES

On some Oracle platforms/versions there may be problems with exporting
directly to tape using the Oracle export utility, you may need to perform
the export to tape via unix named pipes.

A unix named pipe is a FIFO special file, created using the unix mknod
command.The syntax of the mknod command may change from port to port so
please check the manual pages on your system.

Please note that exporting via a named pipe (to tape or disk), is slower
than using export directly. This is because you are limited by the size
of a unix named pipe, usually 8K.

The following commands should be used to export to tape via a named pipe:

1. Create a unix named pipe :

```
os> mknod /tmp/exp_pipe p
```

2. dd in from the named pipe, out to the tape device, in the background :

```
os> dd if=/tmp/exp_pipe of=<tape device> &
```

3. Start the export, specifying the named pipe as the output file:

```
os> exp file=/tmp/exp_pipe <other options>
```

To import from the resultant export on tape, the following commands would be used :

1. Create a unix named pipe :

```
os> mknod /tmp/imp_pipe p
```

2. dd in from the tape device, out to the named pipe, in the background :

```
os> dd if=<tape device> of=/tmp/imp_pipe &
```

3. Start the import, specifying the named pipe as the input file :

```
os> imp file=/tmp/imp_pipe <other options>
```

CREATING A COMPRESSED EXPORT FILE

If you have calculated the size of the file your export will produce and it is too large to fit onto disk, you may want to consider producing a compressed export file as an alternative to exporting directly to tape. Again this method should be thoroughly tested before being implemented.

1. Create a unix named pipe :

```
os> mknod /tmp/exp_pipe p
```

2. Start compress in the background reading in from the named pipe, writing out to 'export.dmp.Z' :

```
os> compress < /tmp/exp_pipe > export.dmp.Z &
```

3. Start the export, specifying the named pipe as the output file :

```
os> exp file=/tmp/exp_pipe <other options>
```

To import from the resultant compressed export file, the following commands would be used :

1. Create a unix named pipe :

```
os> mknod /tmp/imp_pipe p
```

2. Start uncompress in the background reading from 'export.dmp.Z', writing out to the named pipe :

```
os> uncompress < export.dmp.Z > /tmp/imp_pipe
```

3. Start the import, specifying the named pipe as the input file :

```
os> imp file=/tmp/imp_pipe <other options>
```

EXPORTING TO A REMOTE TAPE DEVICE

Finally, you may want to perform an export directly to tape but you do not have a local tape drive on your machine. There is, however a tape drive on another machine on the network, that you have remote shell (rsh) access on.

The following commands allow you to perform the export to the remote machine, either to file or to a tape device :

1. Create a unix named pipe :

```
os> mknod /tmp/exp_pipe p
```

2. dd in from the named pipe, and out to the remote tape device via a remote shell :

```
os> dd if=/tmp/exp_pipe | rsh <hostname> dd of=<file or device> &
```

3. Start the export, specifying named pipe as the output file :

```
os> exp file=/tmp/exp_pipe <other options>
```

To import from the resultant export on tape, the following commands would be used :

1. Create a unix named pipe :

os> mknod /tmp/imp_pipe p

2. Start a remote shell, in the background that dd's from the remote tape, piping to the local named pipe :

os> rsh <hostname> dd if=<file or device> | dd if=/tmp/imp_pipe &

3. Start the import, specifying the named pipe as the input file :

os> imp file=/tmp/imp_pipe <other options>

** Note: On some platforms 'rsh' may need to be run with a '-n' or
similar option to ensure terminal messages are written
to /dev/null and avoid potential problems with timing.

eg: $ rsh -n <hostname> dd if=/tmp/expdat.dmp | dd of=/tmp/imp_pipe &
$ imp un/pw file=/tmp/imp_pipe <options>

[NOTE:62436.1]

Alert: 7.0 To 8.0.6 Export With Compress=Y May Create a Bad Export File

Versions Affected
~~~~~~~~~~~~~~~~~~

This problem affects all versions of export shipped with Oracle releases:
7.0.12 to 7.3.4.3 inclusive
8.0.3 to 8.0.5 inclusive

This problem is fixed in Oracle releases:
7.3.4.4 , 8.0.4.4, 8.0.5.2 and 8.0.6.0

Platforms Affected
~~~~~~~~~~~~~~~~~~~~

This problem is GENERIC.

Description
~~~~~~~~~~~~

If an export is performed using COMPRESS=Y then it is possible for the storage clause in the export file to contain invalid values for INITIAL
~~~~~~~~~~~~

and / or NEXT. This can prevent IMPORT from importing particular tables.

Likelihood of Occurrence
~~~~~~~~~~~~~~~~~~~~~~~~

This problem can only occur if export is run with the COMPRESS option set to YES (which is the default). Objects likely to be affected can be located using the following query:

```
SELECT OWNER, SEGMENT_NAME
   FROM DBA_SEGMENTS
  WHERE NEXT_EXTENT >= (2*1024*1024*1024)
;
```

Possible Symptoms
~~~~~~~~~~~~~~~~~

There is no error during the export process but when an attempt is made to import error such as this are reported:

```
IMP-00017: following statement failed with ORACLE error 2219:
CREATE TABLE x ...
IMP-00003: ORACLE error 2219 encountered
ORA-02219: invalid NEXT storage option value
```

Note: Import will raise an error even if the IGNORE=Y option is set.

Workaround
~~~~~~~~~~

There are 2 workarounds which can be used at the time that EXPORT is run:

1. Export with COMPRESS=N. This has to be chosen explicitly as the default option is COMPRESS=Y

or

2. For tables listed by the query above change the NEXT storage option to a value less than 2Gb prior to running export.
   Eg: ALTER TABLE x STORAGE (NEXT 2000M PCTINCREASE 0);

   Setting PCTINCREASE to zero prevents the NEXT value from growing above 2Gb at a later point in time.

If you have an export file which has a problem STORAGE clause in it then the standard version of IMPORT will not be able to import the affected table/s. You should either:

1. Where possible generate a new export file using one of the above
~~~~~~~~~~

workarounds.
or
2. Use a version of IMPORT which can ignore the ORA-2219 error
(see "Versions Affected" above for versions of IMPORT which can
ignore this error).
or
3. Contact Oracle Support for assistance.

Patches
~~~~~~~
Please contact Oracle Worldwide Support to find out if a patch is
available for your platform / version.

References
~~~~~~~~~~
2Gb or not 2Gb - file limits and Oracle [NOTE:62427.1]

[NOTE:62252.1]

[8.0.5] (22) 64-Bit Issues

Oracle now provides support for 64-bit computers. This support eliminates the 2GB addressing limitation of 32-bit releases. Therefore, SGAs (System Global Aresas) and PGAs (Program Global Areas) are limited only by the physical memory on the computer system.
64-bit addressing can improve performance by allowing very large buffer caches to be configured, therby reducing I/Os. Further, by eliminating the limit on the maximum size of the shared pool, 64-bit addressing can allow more users to be logged into Oracle.

In 8.0.x, all Oracle databases on a hardware cluster that are linked in Parallel Server mode must match the word size of the GMS executable. Therefore, they must all run a 32 or 64 bit executable. Mixing word sizes of parallel server executables, even across different databases, will not work in 8.0.x. This restriction, which may be relaxed in 8.1, does not apply to Oracle executables that are not linked in Parallel Server modes.

Each platform which can support 64-bits will receive two CDs, one for 32-bits, and one for 64-bits.

[NOTE:60888.1]

2Gb Filesize Limits for AIX

** This is a quick reference summary note for 2Gb issues on AIX **

AIX Limits

~~~~~~~~~~

```
  ********************************************************************
  * PLEASE NOTE THAT THIS ARTICLE ONLY APPLIES TO 4.2.1 & HIGHER     *
  *   SINCE IBM DID NOT SUPPORT GREATER THAN 2GIG FILES PRIOR TO      *
  *    AIX 4.2.1 UNLESS THESE FILES WERE ON RAW LOGICAL VOLUMES       *
  ********************************************************************
```

AIX 4.2.1 and 4.3 allows datafiles up to 32Gb.
However Oracle may require patches in order to create files > 2Gb.
If you have problems creating files above a given size then set the
ulimit for fsize to -1 in /etc/security/limits and reboot.
This sets the hard filesize limit within AIX which are sometimes
defaulted to 1Gb.

Oracle Generic Limits

~~~~~~~~~~~~~~~~~~~~~

See [NOTE:62427.1] for generic 2Gb information.
IMPORTANT: There is a generic limit on the maximum Oracle datafile
size of 4million Oracle data blocks. The exact limit is
4194303 * DB_BLOCK_SIZE. Some Oracle operations may try
to exceed this as alerted in [NOTE:112011.1].

Oracle on AIX Limits

~~~~~~~~~~~~~~~~~~~~

Release	AIX	Platform	Required Patch	FS/RAW	AIO >2Gb
8.1.7.x	>=4.3.2	RS/6000 & SP	none 8.1.7 can write EXPORT files > 2Gb *5	both	yes
8.1.6.x	>=4.3.2	RS/6000 & SP	none	both	yes
8.1.5.0	>=4.2.1	RS/6000 & SP	none	both	yes
8.0.6.x	>=4.2.1	RS/6000 & SP	none	both	yes
8.0.6.x	4.1.x	RS/6000 & SP	none	no	no
8.0.5.x	>=4.2.1	RS/6000 & SP	In makefile *4	both	yes
~~~~~~~~~~~~~~~~~~~~

8.0.5.x	4.1.x	RS/6000 & SP	none	no	no
8.0.4.x	>=4.2.1	RS/6000 & SP	653747 *2	both	yes
8.0.4.x	4.1.x	RS/6000 & SP	none	no	no
8.0.3.2	>=4.2.1	RS/6000 & SP	650952 *3	both	yes
8.0.3.2	4.1.x	RS/6000 & SP	none	no	no
7.3.4.x *1	4.1.x	RS/6000	none	raw	no
7.3.4.x *1	>=4.2.1	RS/6000	617038	both	yes
7.3.4.x *1	4.1.x	SP	none	raw	no
7.3.4.x *1	>=4.2.1	SP	601604	both	yes
7.3.3.x	4.1.x	RS/6000	522160	raw	no
7.3.3.x	>=4.2.1	RS/6000	558715	both	yes
7.3.3.x	4.1.x	SP	>= 7.3.3.3	raw	no
7.3.3.x	>=4.2.1	SP	561541	both	yes
7.3.2.3	4.1.x	RS/6000	537084	raw	no
7.3.2.3	>=4.2.1	RS/6000	559497	both	yes
7.3.2.3	4.1.x	SP	461324	raw	no
7.3.2.3	>=4.2.1	SP	561543	both	yes

NOTE: The mentioned 2G limit (lifted up by the patchsets) is still
enforced for the Oracle Utilities like Export, Import and SQL*Loader.
These utilities do NOT create files >2G after applying the patches.

*5 = If exporting from a 64bit Oracle server then the client needs
the patch for Bug 2298968 applying. This patch is for a memory
leak which can cause EXP to crash well before 2Gb of output
has been written

*4 = In 8.0.5 the 2Gb patch is shipped with Oracle and can
be enabled / disabled using the commands:

```
cd $ORACLE_HOME/rdbms/lib
make -f ins_rdbms.mk 2g_on      # or 2g_off as appropriate
make -f ins_rdbms.mk ioracle
```

*3 = obsoletes fir for [BUG:637360] that does not work on AIX 4.3.0

*2 = obsoletes fix for [BUG:640396] that does not work on AIX 4.3.0

*1 = Please note that the greater than 2Gig patch is supplied with the
7.3.4.x patchset release (PSR), under a directory called

large_file_support.

- It needs to be applied after the PSR has been applied.
- If you applied it in 7.3.4.2 (for example) it still needs to be re-applied in 7.3.4.3

In 7.3.4.4 there is a problem with the supplied patch and so on this release the fix for [BUG:937731] needs to be applied instead.

Patch Sets:
For 7.3.3.6+, 7.3.4.2+, 8.0.4.1+, and 8.0.[5,6].*, the latest >2G patch is be shipped with the patch set and bundled release.
Customers will still need to apply the >2G patch AFTER applying the patch set to get full large file support on AIX 4.2.1.

[NOTE:62407.1]

Filesize Limits for HPUX

Summary File Limits for Oracle on HP
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

HPUX Unix Limits
~~~~~~~~~~~~~~~~~~

Max file system size:

<= HP-UX 10.10	4Gb
>= HP-UX 10.20	128Gb
>= HP-UX 11.00	1Tb

Max OS file size:

<= HP-UX 10.10	2Gb
>= HP-UX 10.20	128Gb
>= HP-UX 11.00	1Tb

Ulimit for filesize:

Shell	Query/Set	Defaulted In
ksh / sh	ulimit	/etc/profile
csh	limit [-h] filesize	/etc/csh.login

The 'largefiles' option is required to support large files on a file system.
The following example shows how to convert a no-large-files filesystem to a large-files filesystem:

/usr/sbin/fsadm -F vxfs -o largefiles /oracle
or
/usr/sbin/fsadm -F vxfs -o largefiles {special device file}

Oracle Generic Limits
~~~~~~~~~~~~~~~~~~~~~

See [NOTE:62427.1] for generic 2Gb information.
IMPORTANT: There is a generic limit on the maximum Oracle datafile
size of 4million Oracle data blocks. The exact limit is
4194303 * DB_BLOCK_SIZE. Some Oracle operations may try
to exceed this as alerted in [NOTE:112011.1].
Hence if the DB_BLOCK_SIZE=2048 no datafile can be 8Gb
or larger.

Oracle on HPUX Limits
~~~~~~~~~~~~~~~~~~~~~

If you have problems with files >2GB see the following matrix:
Limits apply to HPUX 10.20 and 11.0 except where noted otherwise.

Release	Max Datafile Size File System	Raw Device	Async I/O	Export/ Import**	SQL*Loader (see **)
8.1.7.4 64bit	64Gb	64Gb	raw only	>4Gb (See*3) ?	
8.1.7.4 32bit	64Gb	64Gb	raw only	>4Gb	>2Gb
8.1.6.x	64Gb	64Gb	raw only	>2Gb	>2Gb
8.0.6.x	64Gb	64Gb	raw only	See*1	See*2
8.0.5.x	64Gb	64Gb	raw only	See*1	<2Gb
8.0.4.x	64Gb	64Gb	raw only	SIL	HPUX10.20 > 2Gb HPUX-11 <2Gb
8.0.3.x	<2Gb	<2Gb	raw only	2Gb	2Gb
7.3.4.x	64Gb	64Gb	raw only	SIL	2Gb
>=7.3.3.4.1	64Gb	64Gb	raw only	SIL	2Gb
7.3.2.3	<2Gb	<2Gb	raw only	2Gb	2Gb
7.1.6	<2Gb	<2Gb	raw only	2Gb	2Gb

SIL = System Imposed Limit
Always test on the actual system to ensure large files can be
read / written.

*3 = 8.1.7.4 export against a 64bit Oracle database suffers
from a client side memory leak which can cause the export

process to terminate well before 2Gb of output has been written. Apply the patch for bug 2298968 to the client.

*2 = HPUX 10.20 requires a patch for > 2Gb SQL Loader files:
8.0.6 - Get the patch for Bug:1344224
HPUX 11 can SQLLoader can read >2Gb files as standard.

*1 = HPUX 10.20 requires a patch to create >2Gb export files:
8.0.5.0 to 8.0.5.2 inclusive - Get the patch for Bug:872947
8.0.6.0 to 8.0.6.1 inclusive - Get the patch for Bug:1330994
HPUX 11 can export >2Gb on these RDBMS releases.

** Exp/Imp/Loader
The above matrix indicates the maximum file size which can be directly handled by exp/imp/loader. However, it is possible to read/write files >2Gb using the OS commands summarised in [NOTE:30528.1]

[NOTE:62426.1]

Filesize Limits for Tru64 (Digital Unix)

Summary File Limits for Oracle on Tru64 (Digital Unix)
~~~~~~

Tru64 Unix Limits
~~~~~~~~~~~~~~~~~~

Max file system size

UFS	4.0d and 4.0e support a file system of up to 128Gb.
AdvFS	49T (Hardware limit)

Max OS file size

UFS	A single file can span the whole file system. ie the file limit is 128Gb.(4.0d/4.0e)
AdvFS	16Tb

Ulimit:

Shell	Command	Default
csh	limit (filesize figure)	/etc/csh.login
sh	ulimit (limit in 512 byte blocks)	/etc/profile

Oracle Generic Limits
~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~

See [NOTE:62427.1] for generic 2Gb information.
IMPORTANT: There is a generic limit on the maximum Oracle datafile
size of 4million Oracle data blocks. The exact limit is
4194303 * DB_BLOCK_SIZE. Some Oracle operations may try
to exceed this as alerted in [NOTE:112011.1].

Oracle on Tru64 Limits
~~~~~~~~~~~~~~~~~~~~~~

If you have problems with files >2GB see the following matrix:

Oracle	File System	RAW Imp	AIO Loader	Exp/SQL* DBV	Patch required
8.1.6.x	128Gb	128Gb	#1	>2Gb	>2Gb >2Gb
8.1.5.x	128Gb	128Gb	#1	>2Gb	>2Gb -
8.0.6.x	128Gb	128Gb	#1	>2Gb	>2Gb -
8.0.5.x	128Gb	128Gb	#1	>2Gb	>2Gb -
8.0.4.2+	128Gb	128Gb	#1	-	- -
8.0.4.0	128Gb	128Gb	#1	#3	#3 -
8.0.3	128Gb	128Gb	#1	#3	#3 - Bug:524095 for > 4Gb
7.3.4	128Gb	128Gb	#1	#3	#3 -
7.3.3	128Gb	128Gb	#1	#3	#3 -
7.3.2.3	128Gb	128Gb	#1	#3	#3 -
7.2.3	128Gb	128Gb	#1	#3	#3 - Bug:351462 required

- = Not confirmed by support

#1 Async IO is supported to raw devices only.
See [NOTE:50548.1] for details.

#3 Export and SQL*Loader do not officially support files > 2Gb until
8.0.4.2 or later. However, as this is a 64bit platform these tools should
work with large files in all releases except where a generic issue causes
a problem. It is also possible to read/write files >2Gb using the OS
commands summarised in [NOTE:30528.1]

Important
=========
The 128Gb limit for Oracle7 and Oracle8 is dependant on your
Oracle block size. 128Gb is the maximum ONLY if you use a
~~~~~~~~~~~~~~~~~~~~~~

32k block size.

As the block size reduces, the maximum file size also reduces.
For an 8k block size, the maximum filesize drops to 32Gb.
See [NOTE:112011.1] for scenarios where Oracle may allow
you to create a file larger than should be allowed.

[NOTE:62415.1]

PTX: Filesize Limits for IBM NumaQ / PTX

Summary File Limits for Oracle on IBM/NumaQ (formerly Sequent PTX)
~~~~~~

PTX Unix Limits
~~~~~~~~~~~~~~~~~

Max file system size
< PTX 4.2 1Tb (EFS), 2Gb (UFS)
>= PTX 4.2 1Tb (EFS or UFS)

Max OS file size
< PTX 4.2 2Gb
>= PTX 4.2 1Tb

Ulimit:
Kernel maximums configured by the parameters:
SFSZLIM max. file size soft limit (bytes)
HFSZLIMmax. file size hard limit (bytes)

To Display limits:
kernel /etc/sysdef
csh limit (filesize figure)
sh ulimit (limit in 512 byte blocks)

Note: Any filesystem to be used for >2Gb files must be mounted
with largefile support enabled.

Oracle Generic Limits
~~~~~~~~~~~~~~~~~~~~~

See [NOTE:62427.1] for generic 2Gb information.
IMPORTANT: There is a generic limit on the maximum Oracle datafile
size of 4million Oracle data blocks. The exact limit is
~~~~~~~~~~~~~~~~~~~~~

4194303 * DB_BLOCK_SIZE. Some Oracle operations may try
to exceed this as alerted in [NOTE:112011.1].
Hence if the DB_BLOCK_SIZE=2048 no datafile can be 8Gb
or larger.

Oracle on PTX Limits
~~~~~~~~~~~~~~~~~~~~
If you have problems with files >2GB see the following matrix:

Oracle	PTX System	File <-- #2 -->	RAW Imp**	AIO see **	Exp/Loader Patch required	
8.1.7	4.4.2+	64Gb	64Gb	#1	>2Gb	?
8.1.6	4.4.2+	64Gb	64Gb	#1	?	?
8.1.5	4.4.2+	64Gb	64Gb	#1	?	?
8.0.6	4.4.2+	64Gb	64Gb	#1	?	?
8.0.4	4.4.2+	64Gb	64Gb	#1	?	?
8.0.3	4.4.2+	64Gb	64Gb	#1	?	<2Gb
7.3.4	4.2+	64Gb	64Gb	#1	?	?
7.3.3.2+	4.2+	64Gb	64Gb	#1	?	?
7.3.3.0/1	4.2+	64Gb	64Gb	#1	<2Gb	?
7.3.2.3	4.2	64Gb	64Gb	#1	?	? ptx4.2_perf_patch
7.3.2.3	4.3	64Gb	64Gb	#1	?	? ptx4.3_numa_patch
7.3.2.3	4.4	64Gb	64Gb	#1	?	? ptx4.4_numa_patch
<7.3.2.3	any	<2Gb	<2Gb	<2Gb	<2Gb	<2Gb
any	<4.2	<2Gb	<2Gb	<2Gb	<2Gb	<2Gb

? = Unconfirmed.
>2Gb = Tested and confirmed to work for > 2Gb
<2Gb = Tested and confirmed not to work for 2Gb or higher

#1 Async IO is supported to files > 2Gb but ensure NABUF and MAXAIO
are set to AT LEAST:
NABUF kernel parameter >= 2048*(number of instances).
MAXAIO kernel parameter >= 2048.

#2 Maximum file size of 64Gb requires a 16k block size

** Exp/Imp/Loader The above matrix indicates the maximum file size
which can be directly handled by exp/imp/loader.
However, it is possible to read/write files >2Gb
~~~~~~~~~~~~~~~~~~~~

using the OS commands summarised in [NOTE:30528.1]

[NOTE:62409.1]

SOLARIS: Filesize Limits For Oracle RDBMS 32-bit On Solaris (SPARC) Servers

Summary File Limits For Oracle RDBMS 32-bit On Solaris (SPARC) Servers
==========

OS Limits
~~~~~~~~~

Release	Max file-system size	Max OS File size
< Solaris 2.6	1Tb (UFS)	2Gb
>= Solaris 2.6	1Tb (40 bits)	1Tb

PLEASE NOTE: In order to use large files on file systems the
file system must be mounted with the 'largefiles' option

Ulimit

Shell	Query/Set	Defaulted In
ksh / sh	ulimit	/etc/profile
csh	limit [-h] filesize	

==========
Oracle Limits
~~~~~~~~~~~~~

IMPORTANT: There is a generic limit on the maximum Oracle datafile
size of 4million Oracle data blocks. The exact limit is
4194303 * DB_BLOCK_SIZE. Some Oracle operations may try
to exceed this as alerted in [NOTE:112011.1].

See [NOTE:62427.1] for generic 2Gb information.

The matrix below shows where Oracle can work with files >=2Gb in size
on Solaris.

~~~~~~~~~~~~~~~~~~~~~~~~
SOLARIS 2.6 / 2.7 / 2.8
~~~~~~~~~~~~~~~~~~~~~~~~

Oracle	File System	Raw	Exp/ Imp**	Loader see **	UTL_FILE	Notes (see below)

```
==========================================================
  8.1.6.0         >4Gb   >=2Gb   >=2Gb    >=2Gb    >=2Gb
  8.1.5.0 32Bit >4Gb   >=2Gb    <2Gb    <2Gb     <2Gb
  8.1.5.0 64Bit >4Gb   >=2Gb    <2Gb     ?        ?        Exp    is    a    32bit
executable
  8.0.6.0         >4Gb   >=2Gb    <2Gb#A   #9      >2Gb
  8.0.5.2         >4Gb   >=2Gb    <2Gb    <2Gb     <2Gb
  8.0.5.1        <4Gb#6 >=2Gb    <2Gb    <2Gb     <2Gb       #6 #8
  8.0.5.0         <2Gb   >=2Gb    <2Gb    <2Gb     <2Gb       #4 #8
>=8.0.4.2        <4Gb#6 >=2Gb    <2Gb    <2Gb     <2Gb       #6 #7 #8
< 8.0.4.2         <2Gb   >=2Gb    <2Gb    <2Gb     <2Gb       #4 #5 #8
  8.0.3.x         <2Gb   >=2Gb    <2Gb    <2Gb     <2Gb
  7.3.4.x         <2Gb   >=2Gb    <2Gb    <2Gb     <2Gb       #1 #2
  7.3.3.x         <2Gb   >=2Gb    <2Gb    <2Gb     <2Gb       #1 #2

~~~~~~~~~~~~~~
SOLARIS 2.5.1     NOTE: Solaris 2.5 does not allow large file system files.
~~~~~~~~~~~~~~
              File            Exp/     Loader
  Oracle    System    Raw     Imp**  see **    UTL_FILE    Notes

==========================================================
  8.0.5.x   <2Gb   >=2Gb    <2Gb     <2Gb     <2Gb
  8.0.4.x   <2Gb   >=2Gb    <2Gb     <2Gb     <2Gb
  8.0.3.x   <2Gb   >=2Gb    <2Gb     <2Gb    <2Gb
  7.3.4.x   <2Gb   >=2Gb    <2Gb     <2Gb    <2Gb
  7.3.3.x   <2Gb   >=2Gb    <2Gb     <2Gb    <2Gb       #1
  7.3.2.3   <2Gb   >=2Gb  <2Gb     <2Gb    <2Gb       #1

~~~~~~~~~~~~
SOLARIS 2.4 NOTE: Solaris 2.4 does not allow large file system files.
~~~~~~~~~~~~
              File            Exp/
  Oracle    System    Raw     Imp     Loader    UTL_FILE    Notes

==========================================================
  7.3.4.x   <2Gb   >=2Gb    <2Gb     <2Gb    <2Gb
  7.3.3.x   <2Gb   >=2Gb  <2Gb     <2Gb    <2Gb       #3
  7.3.2.3   <2Gb   >=2Gb    <2Gb     <2Gb    <2Gb       #3

< 7.3.2.3   <2Gb    <2Gb   <2Gb    <2Gb    <2Gb       Any Solaris release
==========================================================
NOTES:
~~~~~~
```

#1: [BUG:411881] needed for Asynchronous I/O for files > 2Gb
#2: [BUG:675792] indicates problems with filesystem files > 2Gb , especially
if RESIZE is used on datafiles.
#3: [BUG:441394] needed or corruption can occur
#4: [BUG:594800] describes problems with file system files > 2Gb
#5: [BUG:568779] describes problems with file system files > 2Gb
#6: [BUG:715574] Problems with file system files > 4Gb
#7: [BUG:754069] Resizing an OS file to >= 2Gb can cause DBWR to crash
#8: [BUG:774252] ORA-27072 may occur reading multiple blocks spanning
the 2Gb boundary.
#9: SQL*Loader in 8.0.6 can write > 2Gb discard and log files
but cannot read >= 2Gb input files.
#A: Although [BUG:749600] implies export supports >2Gb
on 8.0.6 generically the Solaris version of export
uses the creat() system call which does not allow the
.dmp to be a large file.
** Exp/Imp/Loader The above matrix indicates the maximum file size
which can be directly handled by exp/imp/loader.
However, it is possible to read/write files >2Gb
using the OS commands summarised in [NOTE:30528.1]
==========

[NOTE:67421.1]

ALERT: Do NOT use files >= 4Gb on NT in Oracle8

Bug:711563 Problems with Datafiles >= 4Gb on Oracle8 on NT Platforms
~~~~~~~~~~

Versions Affected
~~~~~~~~~~~~~~~~~
The problems described here can affect all releases of Oracle8 on NT only.

Platforms Affected
~~~~~~~~~~~~~~~~~~
The problems affect Oracle8 releases on all NT platforms
(ie: Both Intel and Alpha NT platforms are affected)

Description
~~~~~~~~~~~
Releases of Oracle8 on NT platforms can exhibit serious problems when
using files larger than 4Gb in size. Some versions of Oracle8 will not
allow such files to be created whilst others can give problems if a file
is resized beyond 4Gb.

Likelihood of Occurrence

It is highly likely that problems will be encountered if an attempt is made to create or resize a datafile to larger than 4Gb. One should also avoid migrating an Oracle7 database to Oracle8 if it contains files which are 4Gb or larger.
NOTE: Resizing includes AUTOEXTEND of a file beyond 4Gb.

Possible Symptoms

There are several possible symptoms from this problem including:

a. ORA-1122: database file n failed verification check
   ORA-1110: data file n: '.....'
   ORA-1200: actual file size of n is smaller than correct size

b. ORA-1237 cannot extend datafile %s
   ORA-1110 data file n: '.....'

c. If a file is resized above 4Gb it is possible for the file to be truncated to a much smaller size such that data could be lost from the file.

Workaround

The only safe workaround at present is to ensure that no file is created which is >= 4Gb in size. This is best achieved by making sure all datafiles are 4095Mb or less and additionally setting MAXSIZE to 4095Mb for files that are configured to AUTOEXTEND.
NOTE: Attempting to create a file of EXACTLY 4Gb can cause problems.

It is advisable to read the article below which describes potential issues using files of 2Gb or larger in size as this may influence the maximum datafile size you choose to use.

Patches

A fix is now available in 8.0.4.3.5 and 8.0.5.1.0a

References

2Gb or not 2Gb - File limits in Oracle [NOTE:62427.1]
Base bug reporting this problem [BUG:711563]