**处理 Oracle 块损坏 (Doc ID 1526911.1)**

**适用于:**

Oracle Database - Enterprise Edition - 版本 7.0.16.0 到 11.2.0.2.0 [发行版 7.0 到 11.2]

本文档所含信息适用于所有平台

**用途**

 处理 Oracle7/8/8i/9i/10g/11g/12c 中的 Oracle 块损坏

**详细信息**

**文档内容**

[简介](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INTRODUCTION)

[概述处理坏块的步骤](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#OVERVIEW)

[由于 NOLOGGING 或 UNRECOVERABLE 操作导致的坏块](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#NOLOGGING)

[(1) 确定坏块问题的范围](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#step1)

[(2) 更换或拆下可疑硬件](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#step2)

[(3) 每次坏块需记录的信息](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#step3)

[(4) 受影响的对象有哪些？](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#objects)

*Options for various Segment Types:*

[*CACHE*](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#CACHE) [*CLUSTER*](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#CLUSTER) [*INDEX PARTITION*](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDPART) [*INDEX*](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#IND) [*LOBINDEX*](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#LOBINDEX) [*LOBSEGMENT*](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#LOBSEGMENT) [*ROLLBACK*](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#ROLLBACK)

[*TABLE PARTITION*](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#TABLEPART) [*TABLE*](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#TABLE) [*TEMPORARY*](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#TEMPORARY) [*IOT（索引组织表）*](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#IOT) [*TYPE2 UNDO*](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#TYPE2UNDO) [*其他一些段类型*](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#OTHER) [*No Segment*](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#NOROWS)

[(5) 选择恢复选项](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#4)

[(5A) 完全恢复](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#4A)

[块级恢复](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#BLOCKRECOVER) ,

[数据文件恢复](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#DATAFILERECOVER) ,

[数据库恢复](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#DATABASERECOVER) ,

[完全恢复后](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#aftercomplete)

[(5B) 重建索引](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#step4b)

[(5C) 抢救表中数据](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#step4c)

[从坏块表中提取坏块周围数据的方法](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#EXT1)

[从包含损坏的 LOBSEGMENT 块的表中提取数据的方法](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#EXT2)

[从坏块本身提取数据](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#EXT3)

[(5D) 忽略坏块](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#step4d)

[针对忽略坏块的警告](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#WARNINGS)

[(5E) 最后的选项](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#step4e)

**简介**

本文章讨论如何处理 Oracle 数据文件中的一个或多个坏块，并介绍了处理这些坏块的主要方法。在采取任何措施之前，请先阅读完整篇文章。

本文档未介绍内存坏块问题（通常为 ORA-600 [17xxx] 类型错误）。

**注意:**如果在启动时出现 ORA-1578 问题，请与当地支持中心联系，以获得参考建议。

本文章介绍了许多类型的错误，很多其他地方也可能引用到本文章。重要的是，您需要知道关于每个坏块的以下信息：

* 包含坏块的文件的绝对文件编号 (FILE NUMBER)。

本文中称为“&AFN”。

* 包含坏块的文件的名称。

本文中称为“&FILENAME”。

如果您知道文件编号，但不知道文件名，则可以使用 V$DATAFILE 来获取文件名：

SELECT name FROM v$datafile WHERE file#=&AFN;

如果文件号未显示在 Oracle8i 的 V$DATAFILE 中，且 &AFN 大于 DB\_FILES 参数值，则该文件可能是临时文件。在这种情况下，可以使用以下查询找到文件名：

SELECT name FROM v$tempfile WHERE file#=(&AFN - &DB\_FILES\_value);

* 文件中坏块的块编号。

本文中称为“&BL”。

* 包含受影响块的表空间编号和名称。

本文中称为“&TSN”（表空间编号）和“&TABLESPACE\_NAME”。

如果您不知道这些信息，请使用以下查询找到它们：

SELECT ts# "TSN" FROM v$datafile WHERE file#=&AFN;

SELECT tablespace\_name FROM dba\_data\_files WHERE file\_id=&AFN;

* 表空间中的坏块大小。

本文中称为“&TS\_BLOCK\_SIZE”。

SELECT block\_size FROM dba\_tablespaces

WHERE tablespace\_name =

(SELECT tablespace\_name FROM dba\_data\_files WHERE file\_id=&AFN);

对于 Oracle 7、8.0 和 8.1，数据库中所有表空间使用相同的块大小。

对于这些版本，使用命令"SHOW PARAMETER DB\_BLOCK\_SIZE" 返回的值作为 “ &TS\_BLOCK\_SIZE”

例如：对于 ORA-1578 错误：

ORA-01578: ORACLE data block corrupted (file # 7, block # 12698)

ORA-01110: data file 22: '/oracle1/oradata/V816/oradata/V816/users01.dbf'

那么:

&AFN      为 "22"     (从错误 ORA-1110 部分获得)

&RFN      为 "7"      (从错误 ORA-1578 的"file #"部分获得)

&BL       为 "12698"  (从错误 ORA-1578 的"block #"部分获得)

&FILENAME 为 '/oracle1/oradata/V816/oradata/V816/users01.dbf'

&TSN 及其他信息可以从上面提到的 SQL 语句获得

   对于其他错误（ORA-600、ORA-1498 等），上述值应由 Oracle Support 提供给您，或从涵盖相关错误的文章中获取。

   对于某些错误，如 ORA-1410“invalid ROWID（无效 ROWID）”、ORA-12899“value too large for column（列值过大）”等，未给出损坏的文件/块的详细信息。对于此类情况， [Note:869305.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=869305.1) 可以帮助您定位损坏的行。

[{go to content}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)

**概述处理坏块的步骤**

有多种原因可能导致坏块，包括：

* 坏的 IO 硬件/固件
* OS 问题
* Oracle 问题
* 对于执行过“UNRECOVERABLE”或“NOLOGGING”操作的数据库进行恢复

（在这种情况下可能产生 ORA-1578 错误 - 如下方所示）

产生 Oracle 错误的时间点可能要比最初发生任何块损坏的时间点晚得多。

在遇到坏块时，我们通常无从了解根本原因，并且在大多数情况下，当下最迫切的需求是重新启动数据库并使其运行起来，正因如此，本文将介绍用于解决坏块问题的步骤，如下所列：

1. 确定坏块问题的范围，并确定这些问题是持久性问题还是暂时性问题。

如果问题涉及范围很大，或错误不稳定，则关键在于先识别原因（检查硬件等）。这点很重要，因为如果是底层硬件出现错误，恢复系统便毫无意义。

1. 更换或拆下任何有问题的或可疑的硬件。
2. 确定哪些数据库对象受到影响。
3. 选择最合适的数据库恢复/数据抢救选项。

对于上述所有步骤，最好应收集证据并详细记录所采取的措施。本文中的“证据>>”标签列出了应收集的信息，以帮助您识别问题的根本原因。

**由于 NOLOGGING 或 UNRECOVERABLE 操作导致的坏块**

如果对某个对象执行了 NOLOGGING（或 UNRECOVERABLE）操作，随后又恢复了包含该对象的数据文件，则受到 NOLOGGING 操作影响的数据块将被标记为“坏块”，当您访问该数据块时将显示 ORA-1578 错误。

从Oracle8i开始报错ORA-26040("ORA-26040: Data block was loaded using the NOLOGGING option")，此时原因一目了然，而较早版本中则没有附件这条错误消息。如果坏块是由于对执行过 NOLOGGING 操作的数据文件进行恢复而产生的，则可以使用本文中从  [Section 3 "Information to Record for Each Corruption"](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#step3) 开始之后介绍的内容，但请注意以下问题：

1. 恢复操作无法找回受 NOLOGGING 操作影响的数据
2. 块内的数据无法抢救
3. 解决方法请参考[Note:794505.1](https://support.oracle.com/DocumentDisplay?id=794505.1)

(1) 确定坏块问题的范围

请参考[Note 836658.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=836658.1)查找坏块的范围。使用 RMAN或者DBVERIFY 扫描受影响的文件（以及一切重要的文件）也是不错的办法，这样可以检查是否有其他坏块，从而确定问题的范围。有关使用 DBVERIFY 的详细信息，请参阅 [Note:35512.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=35512.1)

每次发生坏块错误时，都应记下完整的错误消息，并查看该实例的告警日志和跟踪文件，以了解任何相关的错误。首先进行这些步骤非常重要，这可以评估该损坏是单个块，还是由于 UNRECOVERABLE 操作产生的错误，抑或是更严重的问题。

一旦确定了损坏的文件/块组合列表，即可使用以下步骤来帮助确定应采取何种措施。

**证据:**

* 完整记录初始错误，以及发生错误的应用程序的详细信息。
* 及时地保存从告警日志中首次 (FIRST) 记录到问题前数小时到当前时间点所提取的内容。
* 保存告警日志中提到的任何跟踪文件。
* 记录最近遇到的任何 OS 问题。
* 记录是否正在使用任何特殊功能，例如：ASYNC IO、快速写入磁盘选项等。
* 记录当前的备份位置（日期、类型等）
* 记录数据库是否处于 ARCHIVELOG 模式，

例如：在SQL\*Plus（或 Server Manager）中运行“ARCHIVE LOG LIST”

**(2) 更换或拆下可疑硬件**

大多数坏块问题是由故障硬件导致的。

如果出现硬件错误或可疑组件，最好进行修复，或者在执行恢复操作之前，确保在单独的磁盘子系统上有足够的可用空间用于恢复。

您可以使用以下步骤移动数据文件：

* 确保要迁移的文件已离线或数据库实例处于 MOUNT 状态（未打开）
* 将该数据文件物理还原（或复制）到新位置

例如：/newlocation/myfile.dbf

* 将该文件的新位置告知 Oracle。

例如：ALTER DATABASE RENAME FILE '/oldlocation/myfile.dbf' TO '/newlocation/myfile.dbf';

    （请注意，您不能对临时文件进行重命名，而应删除临时文件并在新位置重新创建）

**重要信息:** 如果存在多个错误（不是由于 NOLOGGING操作导致的）

            或 受影响文件所在的 OS 层面出现错误

            或 错误是暂时性的且游离不定，

    那么，如果不解决底层问题或准备另外的磁盘空间，那么进行任何操作都是毫无意义的。

    与硬件供应商联系，以全面检查系统，并联系 Oracle Support，告知所有错误详情。

**请注意:** 如果硬件检查失败，即表明存在硬件问题，但硬件检查成功通过却并不能证明没有硬件相关问题 — 硬件测试报告成功但确实存在底层错误，这种情况也是经常发生。

 如果使用了任何特殊 IO 选项，例如 direct IO、async IO 或类似的选项，最好将其禁用，以消除这些选项成为潜在问题原因的可能性。

**(3) 每次坏块需记录的信息**

在决定如何恢复之前，最好先确定哪些对象受到了影响，因为坏块可能发生在那些容易被重新创建的对象中。

例如：对于只有 5 行数据的表中发生的坏块，删除并重新创建表可能要比执行恢复快得多。

对于每个坏块，请收集下表中的信息。

进行此操作的步骤如下所述。

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **针对每次坏块需记录的信息** |  |  |  |  |  |  |  |  |
| **初始错误** | **绝对文件号**  **&AFN** | **相关文件号**  **&RFN** | **块编号**  **&BL** | **表空间** | **段类型** | **段**  **所有者.名称** | **相关对象** | **恢复选项** |

 下列说明将有助于您针对每个坏块填写此表。

1. **"初始错误"**最初报告的错误。例如：ORA-1578/ORA-1110、ORA-600 和所有参数等。
2. **"绝对文件号", "相关文件号"和"块号"**文件号和块号应该已在错误中显示，或由 Oracle Support 提供，或在指引您参考本文章的其他文章的步骤中提供。

**在 Oracle8/8i/9i/10g 中:** 绝对文件号和相关文件号通常是一样的，但也可能不同（尤其是在数据库是由 Oracle7 迁移而来的情况下）。请务必获得正确的 &AFN 和 &RFN 编号，否则您可能最终抢救的是错误的对象！！

ORA-1578 报告相关文件号，绝对文件号在伴随的 ORA-1110 错误中显示。对于 ORA-600 错误，您应该会被告知绝对文件号。

下列查询将显示数据库中数据文件的绝对和相关文件号：

SELECT tablespace\_name, file\_id "AFN", relative\_fno "RFN" FROM dba\_data\_files;[Insert code here. Use 'Paste from Word' to retain layout.]

**在 Oracle8i/9i/10g 中:**

除了上述关于 Oracle8 的说明外，从 Oracle8i 开始将拥有临时文件。下列查询将显示数据库中临时文件的绝对和相关文件号：

SELECT tablespace\_name, file\_id+value "AFN", relative\_fno "RFN" FROM dba\_temp\_files, v$parameter WHERE name='db\_files';

**在 Oracle7 中:** “绝对文件号”和“相关文件号”使用相同的文件号

1. **"段类型", "所有者", "名称"和"表空间"**

在给定坏块的绝对文件号“&AFN”和块编号“&B”的情况下，下列查询将显示对象的段类型、所有者和名称，数据库必须打开才能使用此查询：

SELECT \* FROM dba\_extents

WHERE file\_id = &AFN

and &BL between block\_id AND block\_id + blocks - 1;

1. 如果坏块位于临时文件中，则上述查询将不会返回任何数据。

对于临时文件，“段类型”应为“TEMPORARY”。

如果上述查询未返回行，也可能是因为坏块是本地管理表空间 (Locally Managed Tablespace, LMT) 中的段头。当坏块为 LMT 中的段头块时，上述查询将在 alert.log 中生成一个坏块消息，但查询不会失败。在这种情况下，请使用以下查询：

SELECT owner, segment\_name, segment\_type, partition\_name

FROM dba\_segments

WHERE header\_file = &AFN

and header\_block = &BL;

[{go to content}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)

(4) Which Object is affected and 可能的恢复选项:

相关对象和能够使用的恢复选项取决于 SEGMENT\_TYPE。对于各种最常见的段类型，其他查询和可能的恢复选项如下所列。

[CACHE](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#CACHE)

[CLUSTER](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#CLUSTER)

[INDEX PARTITION](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDPART)

[INDEX](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#IND)

[LOBINDEX](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#LOBINDEX)

[LOBSEGMENT](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#LOBSEGMENT)

[ROLLBACK](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#ROLLBACK)

[TABLE PARTITION](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#TABPART)

[TABLE](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#TABLE)

[TEMPORARY](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#TEMPORARY)

IOT

[TYPE2 UNDO](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#TYPE2UNDO)

[Some other Segment Type](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#OTHER)

["no rows" from the query](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#NOROWS)

***CACHE***

    如果段类型为 CACHE，请再次检查您是否输入了正确的 SQL语句和参数。

    **解决方法：**[修复数据块采用块级恢复方法](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#BLOCKRECOVER)

    [{Continue}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#EVIDENCES)  [{Back to Segment List}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#SEGMENTLIST)

***CLUSTER***

    如果段类型为 CLUSTER，则应确定它包含哪些表。

例如：  SELECT owner, table\_name fROM dba\_tables WHERE owner='&OWNER' AND cluster\_name='&SEGMENT\_NAME';

**解决方法：**

[修复数据块采用块级恢复方法](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#BLOCKRECOVER)

[{Collect TABLE information}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#TABLE)  [{Back to Segment List}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#SEGMENTLIST)

***INDEX PARTITION***

    如果段类型为 INDEX PARTITION，请记录名称和所有者，然后确定哪些分区受到影响：

    SELECT partition\_name

    FROM dba\_extents

    WHERE file\_id = &AFN AND &BL BETWEEN block\_id AND block\_id + blocks - 1

**解决方法：**

[修复数据块采用块级恢复方法](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#BLOCKRECOVER)

      或者

      使用下列语句可以重建索引分区：

        ALTER INDEX xxx REBUILD PARTITION ppp;

      （请注意下方“重建索引”中所述的 REBUILD 选项）

[{Continue}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#EVIDENCES)  [{Back to Segment List}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#SEGMENTLIST)

***INDEX***

  确定索引位于哪个表中：

    SELECT table\_owner, table\_name

    FROM dba\_indexes

    WHERE owner='&OWNER' AND index\_name='&SEGMENT\_NAME';

 CONSTRAINT\_TYPE 的可能值包括：

    -  P    索引支持主键约束。

    -  U    索引支持唯一约束。

    如果索引支持主键约束（类型“P”），则确认主键是否被任何外键约束引用：

    SELECT owner, constraint\_name, constraint\_type, table\_name

    FROM dba\_constraints

    WHERE r\_owner='&TABLE\_OWNER' AND r\_constraint\_name='&INDEX\_NAME';

**解决方法：**

[修复数据块采用块级恢复方法](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#BLOCKRECOVER)

     或  重建索引（任何相关联的约束会随之禁用/启用）

     （请注意下方"[重建索引](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383" \l "step4b)"中所述的 REBUILD 选项）。如果对象owner是sys用户，那么请联系oracle技术支持。

    [{Continue}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#EVIDENCES)  [{Back to Segment List}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#SEGMENTLIST)

***ROLLBACK***

如果段类型为 ROLLBACK，请联系 Oracle Support，因为 ROLLBACK 段坏块需要特殊处理。

**解决方法：**

[修复数据块采用块级恢复方法](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#BLOCKRECOVER)

[{Continue}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#EVIDENCE)  [{Back to Segment List}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#SEGMENTLIST)

***TYPE2 UNDO***

TYPE2 UNDO 是系统管理的 undo 段，它是 rollback 段的一种特殊形式。这些段的坏块需要特殊处理。

解决方法：

[修复数据块采用块级恢复方法](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#BLOCKRECOVER)

[{Continue}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#EVIDENCE)  [{Back to Segment List}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#SEGMENTLIST)

***TABLE PARTITION***

如果段类型为 TABLE PARTITION，请记录名称和所有者，然后确定哪些分区受到影响：

SELECT partition\_name

FROM dba\_extents

WHERE file\_id = &AFN

AND &BL BETWEEN block\_id AND block\_id + blocks - 1;

然后按照处理 [TABLE](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#TABLE) 段的步骤继续下面的操作。

解决方法：

[修复数据块采用块级恢复方法](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#BLOCKRECOVER)

或者

如果所有坏块均位于同一个分区，则此时可以采取的一个做法是用一个空表 EXCHANGE 坏块所在的分区，这可以让应用程序继续运行（无法访问坏块所在的分区中的数据），然后可以从之前的空表中提取任何未损坏的数据。

有关其他选项，请参见下面的 [TABLE](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#TABLE) 选项。

[{Continue}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#EVIDENCE)  [{Back to Segment List}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#SEGMENTLIST)

***TABLE***

如果所有者为“SYS”，则联系 Oracle Support 并上传所有详细信息。

可能需要恢复数据库。

对于非字典 TABLE 或 TABLE PARTITION，确定表中存在哪些索引：

      例如： SELECT owner, index\_name, index\_type

          FROM dba\_indexes

          WHERE table\_owner='&OWNER'

            AND table\_name='&SEGMENT\_NAME';

并确定表中是否存在任何主键：

      例如：SELECT owner, constraint\_name, constraint\_type, table\_name

         FROM dba\_constraints

         WHERE owner='&OWNER'

           AND table\_name='&SEGMENT\_NAME'

           AND constraint\_type='P';

如果存在主键，则确认它是否被任何外键约束引用：

      例如：

        SELECT owner, constraint\_name, constraint\_type, table\_name

        FROM dba\_constraints

        WHERE r\_owner='&OWNER'

          AND r\_constraint\_name='&CONSTRAINT\_NAME';

解决方法：

[修复数据块采用块级恢复方法](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#BLOCKRECOVER)

或者

如果所有者为“SYS”，则联系 Oracle Support 并上传所有详细信息。

      对于非字典表，可能的选项包括：

        恢复

        或  抢救表（或分区）中的数据

           然后 重新创建表（或分区）

        或  忽略坏块

        （例如：使用 DBMS\_REPAIR 标记需要跳过的问题块）

    [{Continue}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#EVIDENCE)  [{Back to Segment List}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#SEGMENTLIST)

***IOT（索引组织表）***

IOT 表中的坏块应按照表或分区表中的处理方式来处理。

唯一的例外是如果 PK 损坏。

IOT 表的 PK 就是表本身，它不能被删除和重新创建。

**解决方法：**

[修复数据块采用块级恢复方法](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#BLOCKRECOVER)

或者

如果所有者为“SYS”，则联系 Oracle Support 并上传所有详细信息。

  对于非字典表，可能的选项包括：

    恢复

    或  抢救表（或分区）中的数据

    然后 重新创建表（或分区）

    或  忽略坏块

    （DBMS\_REPAIR 不适用于 IOT）

[{Continue}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#EVIDENCE)  [{Back to Segment List}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#SEGMENTLIST)

***LOBINDEX***

确定 LOB 属于哪个表：

 SELECT table\_name, column\_name

 fROM dba\_lobs

 wHERE owner='&OWNER'

 AND index\_name='&SEGMENT\_NAME';

不可以重建 LOB 索引，因此您必须将该问题作为受影响的表中 LOB 列上的坏块来处理。

使用 ”TABLE“ 部分中的 SQL 语句获取包含损坏的 LOB 索引的表的索引和约束信息，然后返回此处。

   解决方法**：**

[**修复数据块采用块级恢复方法**](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#BLOCKRECOVER)

**或者**

移动LOG 段

alter table &table\_owner.&table\_with\_lob move LOB (&&lob\_column) store as (tablespace &tablespace\_name);

  如果所有者为“SYS”，则联系 Oracle Support 并上传所有详细信息。

    [{Continue}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#EVIDENCE)  [{Back to Segment List}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#SEGMENTLIST)

***LOBSEGMENT***

确定 LOB 属于哪个表：

例如：

SELECT table\_name, column\_name

  FROM dba\_lobs

   WHERE owner='&OWNER'

   AND segment\_name='&SEGMENT\_NAME';

对于非字典表 ...

使用 ”TABLE“ 部分中的 SQL 语句获取包含坏块的 LOB 数据的表的索引和约束信息，然后返回此处查找具体受影响的行的详细信息。

要查找引用损坏的 LOB 块的具体行可能比较困难，因为报告的错误中不会显示表中的哪一行数据包含损坏的 LOB 数据。

Typically one can refer to application logs or any SQL\_TRACE or [10046](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=21154.1) trace of a session hitting the error (if available) or see if having

event "1578 trace name errorstack level 3"

 set in the session helps identify the current SQL/binds/row.

例如：

ALTER SYSTEM SET EVENTS '1578 trace name errorstack level 3';

 然后等待应用程序触发该错误，并查找跟踪文件。

 如果没有任何线索，您可以构建 PLSQL 块，逐行扫描问题表以提取 LOB 列数据，扫描将一直循环进行，直至发生错误。此方法可能需要一段时间，但它应该可以找到引用了损坏的 LOB 块的数据行的主键或 ROWID。

例如：

set serverout on

exec dbms\_output.enable(100000);

declare

 error\_1578 exception;

 pragma exception\_init(error\_1578,-1578);

 n number;

 cnt number:=0;

 badcnt number:=0;

begin

  for cursor\_lob in

        (select rowid r, &LOB\_COLUMN\_NAME L from &OWNER..&TABLE\_NAME)

  loop

    begin

      n:=dbms\_lob.instr(cursor\_lob.L,hextoraw('AA25889911'),1,999999) ;

    exception

     when error\_1578 then

       dbms\_output.put\_line('Got ORA-1578 reading LOB at '||cursor\_lob.R);

       badcnt:=badcnt+1;

    end;

    cnt:=cnt+1;

  end loop;

  dbms\_output.put\_line('Scanned '||cnt||' rows - saw '||badcnt||' errors');

end;

/

 another script more generic:

set serverout on

exec dbms\_output.enable(100000);

declare

  pag    number;

  len    number;

  c      varchar2(10);

  charpp number := 8132/2;

begin

  for r in (select rowid rid, dbms\_lob.getlength (<your\_clob\_column>) len

            from   <your\_table\_with\_clcob\_column>) loop

    if r.len is not null then

      for page in 0..r.len/charpp loop

        begin

          select dbms\_lob.substr (<your\_clob\_column>, 1, 1+ (page \* charpp))

          into   c

          from   <your\_table\_with\_clcob\_column>

          where  rowid = r.rid;

        exception

          when others then

            dbms\_output.put\_line ('Error on rowid ' ||R.rid||' page '||page);

            dbms\_output.put\_line (sqlerrm);

        end;

      end loop;

    end if;

  end loop;

end;

/

**解决方法：**

可能需要恢复数据库，如果所有者为“SYS”，则联系 Oracle Support 并上传所有详细信息。

对于非字典表，可能的选项包括：

恢复

或者用empty\_clob/empty\_blob更新lob列，避免在出现ORA-1578；这样可以清理表里面的lob列：

SQL> set concat off

SQL> update &table\_owner.&table\_with\_lob

        set &lob\_column = empty\_blob()

     where rowid in ('list the identified rowids from the table from the plsql above');

       将损坏的lob设置为empty lob后，之前的数据块还会放回这个lob的freelist。这些空间以后可能会被重用，

       当这些数据块再次被使用时，会出现ORA-1578 错误，因此在empty lob后，再将lob 段移动到新的段：

alter table &table\_owner.&table\_with\_lob move LOB (&&lob\_column) store as (tablespace &tablespace\_name);

或  抢救表（及其 LOB 列）中的数据

然后 重新创建表

或  忽略坏块

(不可以在 LOB 段上使用 DBMS\_REPAIR）

[{Continue}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#EVIDENCE)  [{Back to Segment List}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#SEGMENTLIST)

***TEMPORARY***

 如果段类型为 TEMPORARY，则坏块不会影响永久对象。检查发生问题的表空间是否正在被用作 TEMPORARY 表空间：

SELECT count(\*) FROM dba\_users WHERE temporary\_tablespace='&TABLESPACE\_NAME';

**解决方法：**

参考[Note 1332088.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=1332088.1)

通常情况下，不需要进行任何还原，但如果磁盘可能有问题，且表空间包含有用数据，则最好对数据库中受影响的文件进行恢复。

[{Continue}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#EVIDENCE)  [{Back to Segment List}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#SEGMENTLIST)

***其他一些段类型***

如果返回的段类型未包含在上述类型中，则请联系 Oracle Support 并提供迄今为止收集的所有信息，以获得相关建议。

[{Continue}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#EVIDENCE)  [{Back to Segment List}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#SEGMENTLIST)

***“无返回行”***

如果没有包含坏块的 extent，则首先再次检查查询中使用的参数。如果您确定文件号和块编号是正确的，且不属于 DBA\_EXTENTS 中的某个对象，则执行以下操作：

* 再次检查相关文件是否为临时文件。

请注意，临时文件的文件号取决于数据库初始化参数 DB\_FILES，因此对该参数的任何更改都会改变错误中报告的绝对文件号。

* DBA\_EXTENTS 不包含本地管理表空间中用于本地空间管理的块。
* 如果您在数据库运行查询语句的时间点与出错的时间点不相同，那么问题对象可能已经被删除，因此针对 DBA\_EXTENTS 的查询可能不会显示任何行。
* 如果您正在调查的错误由 DBVERIFY 报告，则 DBV 将检查所有块，而不管它们是否属于某个对象。因此，坏块可能存在于数据文件中，但却未被任何对象使用。

**选项：**

未使用的 Oracle 块上的错误可以忽略，因为如果需要使用该块，Oracle 会创建新的块映像（格式化），因此，该块上的任何问题将永不会被读取。

如果您怀疑该块可能是空间管理块，则可以使用 DBMS\_SPACE\_ADMIN 包来帮助您进行检查：

exec DBMS\_SPACE\_ADMIN.TABLESPACE\_VERIFY('&TABLESPACE\_NAME');

 以上命令会将不一致写入跟踪文件，但如果遇到致命的坏块，它将报告如下错误：

ORA-03216: Tablespace/Segment Verification cannot proceed

[{Continue}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#EVIDENCE)  [{Back to Segment List}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#SEGMENTLIST)

[{go to content}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)

**证据**

对于每个坏块，如果需要尝试并确定实际坏块原因，则收集如下物理证据也是一个比较好的方法：

1. 坏块及位于其任意一侧的块的操作系统 HEX 转储。

在 UNIX 上：

dd if=&FILENAME bs=&TS\_BLOCK\_SIZE skip=&BL-1 count=3 of=BL.dd

   ^^^^^^^^      ^^^^^^^^^^^^^^         ^^^

例如：对于 BL=1224：

dd if=ts11.dbf bs=4k skip=1223 count=3 of=1223\_1225.dd

在 VMS 上：

DUMP/BLOCKS=(start:XXXX,end:YYYY)/out=dump.out &FILENAME

其中 XXXX=操作系统块编号（512 字节块中）

要计算此值，用报告的块编号乘以“&TS\_BLOCK\_SIZE/512”。

1. 处于 ARCHIVELOG 模式时，复制出错时间前后的归档日志文件的安全副本，最好包括报告错误前数小时的日志文件。并且，保存问题数据文件在出错前的所有副本，因为之前的数据文件映像以及 redo 记录有助于找出错误原因。

（DBV 通常可用于检查问题是否存在于文件的备份副本中）。理想的情况是获得没有报告坏块的数据文件备份映像，以及从该时间点开始到首次报告坏块时间之后不久的时段内的所有 redo 记录。

1. 获得问题块的 Oracle 转储：

ALTER SYSTEM DUMP DATAFILE '&FILENAME' BLOCK &BL;

（DUMP将生成到 USER\_DUMP\_DEST 下的跟踪文件）。

[{Continue}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#step4)  [{Back to Segment List}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#SEGMENTLIST)

[{go to content}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)

**(5) 选择恢复选项**

现在，最佳的恢复选项取决于受影响的对象。前面[第 (3)](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#step3) 部分中的说明应该已经重点介绍了针对每个受影响对象的主要可用选项。选择的实际恢复方法可能包含以下一种或多种混合方法：

**是否需要进行任何恢复操作？**

表空间中，或位于不再属于任何数据库对象的块中，则无需进行任何操作，尽管将问题表空间重定位到其他存储设备中可能较为明智。

请参阅[警告](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#warnings)

**可以使用完全恢复吗？**

要选用完全恢复，必须满足如下条件：

* 数据库处于 ARCHIVELOG 模式

（“ARCHIVE LOG LIST”命令显示 Archivelog 模式）

* 拥有受影响文件的完好备份。请注意，在某些情况下，坏块可能已经存在，但在很长一段时间内未被发现。如果最近的数据文件备份仍包含坏块，那么只要您拥有所 有必需的归档日志，就可以尝试使用更早的备份。

（通常可以使用 DBV START= / END= 选项来检查位于某个备份文件的恢复副本中的特定块是否损坏）

* 从备份时间开始到当前时间点的所有归档日志均可用
* 当前的在线日志均可用且完好无缺
* 错误不是由运行 [NOLOGGING](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#NOLOGGING) 操作之后执行的恢复所导致的

如果满足上述条件，完全恢复通常是首选方法

**\*但请注意\***

1. 如果事务回滚已发现坏块位于对象上，而非 rollback 段本身，则 undo 操作可能已被放弃。在这种情况下，可能需要在恢复完成后重建索引/检查数据完整性。

1. 如果要恢复的文件包含自上次备份以来执行的 [NOLOGGING](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#NOLOGGING) 操作的数据，在使用了数据文件或数据库恢复的情况下，这些块将被标记为“坏块”。在某些情况下，这会使情况更加糟糕。

如果执行数据库恢复后坏块仍然存在，则表示所有备份都包含坏块，底层错误仍存在，或问题通过 redo 重现。在这些情况下，需要选择其他一些恢复选项。

 请参阅 ["(4A) 完全恢复"](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#step4a) ，以了解完全恢复步骤。

[{go to content}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)

**如果不需要从对象本身提取任何数据，能否删除或重新创建该对象？**

您可以删除对象或从脚本/最近导出的副本重新创建对象。一旦删除一个对象后，该对象中的块将被标记为“空闲”，并且该块在被分配到新对象时将被重新格式化。明智的做法是，对表进行重命名，而不是删除，除非您完全确定不再需要其中的数据。

对于表分区，只需要删除受影响的分区。

例如： ALTER TABLE ... DROP PARTITION ...

如果坏块影响到分区段头，或者包含分区头的文件处于离线状态，则 DROP PARTITION 可能会失败。在这种情况下，首先将其更换为具有相同定义的表，之后仍然可以删除该分区。

例如： ALTER TABLE .. EXCHANGE PARTITION .. WITH TABLE ..;

最常见的可重建对象为索引。始终在处理表中的索引问题之前处理表坏块。

有关详细信息，请参阅"[(4B) 重建索引](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#step4b)" 。

对于任何段，如果您拥有坏块的绝对文件号和块号，则可使用以下快速提取对象 DDL 的方法：

set long 64000

select dbms\_metadata.get\_ddl(segment\_type, segment\_name, owner)

FROM dba\_extents

WHERE file\_id=&AFN

AND &BL BETWEEN block\_id AND block\_id + blocks -1;

[{go to content}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)

**是否需要在重新创建对象之前抢救数据？**

如果问题位于定期更新的关键应用表上，则可能需要尽可能多地抢救表中数据，然后重新创建该表。

有关详细信息，请参阅"[(5C) 抢救表中数据](https://support.oracle.com/epmos/faces/step4c)" 。

[{go to content}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)

**当前忽略坏块是否可取？**

在某些情况下，最直接的选项可能就是忽略坏块，并阻止应用程序对它进行访问。

有关详细信息，请参阅 "[(5D) 忽略坏块](https://support.oracle.com/epmos/faces/step4d)"。

[{go to content}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)

**最后的选项**

下列选项是否可行？

将数据库或表空间恢复到较早的时间点（通过时间点恢复）

或还原出现坏块前的冷备份

或使用现有导出文件

有关详细信息，请参阅"[(5E) 最后的选项](https://support.oracle.com/epmos/faces/step4e)"。

[{go to content}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)

**(5A) 完全恢复**

如果数据库处于 ARCHIVELOG 模式下，且您拥有受影响文件的完好备份，则恢复通常为首选方法。

这不保证可以解决问题，但的确可以有效的解决大部分坏块问题。如果恢复再次引发问题，则返回到以上选项列表并选择其他方法。

如果使用的是 Oracle9i（或更高版本），则可以使用 RMAN BLOCKRECOVER 命令执行块级恢复。

如果使用的是较早版本的 Oracle，则可以执行数据文件恢复（数据库其他部分可以继续运行），或数据库恢复（需要关闭数据库）。

如果使用的是 Oracle 11g（或更高版本），则可以使用“Data Recovery Advisor（数据恢复指导）”.

***块级恢复***

自 Oracle9i 版本起，RMAN 允许恢复单个块，同时数据库的其他部分（包括数据文件中的其他块）仍可以进行正常访问。请注意，块级恢复只能将块完全恢复到当前时间点。

例如：

实际情况是，文件 6 的块 30 上发生 ORA-1578 错误，可能是由于介质问题导致的坏块，且您拥有该文件的完好冷备份映像，并已还原到“.../RESTORE/filename.dbf”。

假设所有归档日志均存在（位于默认位置），则可以通过 RMAN 使用以下命令序列执行块级恢复：

rman nocatalog

  connect target

   catalog datafilecopy '.../RESTORE/filename.dbf';

      run {blockrecover datafile 6 block 30;}

 此操作将使用注册的数据文件备份映像和任何需要的归档日志来执行块恢复，仅将有问题的块恢复到当前时间点。

有关 RMAN BLOCKRECOVER 命令和限制的所有详细信息，请参阅文档 [Note 144911.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=144911.1)。

[{go to content}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)

***数据文件恢复***

数据文件恢复包括下列步骤。如果有多个文件，则针对每个文件重复执行这些步骤，或参阅下面的“数据库恢复”。当数据库处于 OPEN 或 MOUNTED 状态时，均可使用这些步骤。

使受影响的数据文件离线

例如： ALTER DATABASE DATAFILE 'name\_of\_file' OFFLINE;

将文件复制到安全位置（以防备份损坏）

将文件的最新备份还原到完好的磁盘上

使用 DBVERIFY 检查还原的文件是否有坏块

    有关使用 DBVERIFY 的详细信息，请参阅 [Note:35512.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=35512.1)

假设还原的文件完好，则将数据文件重命名并保存到新位置（如果不是原来的位置）

例如： ALTER DATABASE RENAME FILE 'old\_name' TO 'new\_name';

 恢复数据文件

例如： RECOVER DATAFILE 'name\_of\_file';

 使数据文件上线

例如： ALTER DATABASE DATAFILE 'name\_of\_file' ONLINE;

[{Continue}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#EVIDENCE)

[{go to content}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)

***数据库恢复***

数据库恢复通常包含以下步骤：

关闭数据库（使用选项 immediate 或 abort）

将待恢复的所有文件的当前副本复制到安全位置

将备份文件还原到完好的磁盘上

请勿还原控制文件或在线 REDO 日志文件

使用 DBVERIFY 检查还原的文件。有关使用 DBVERIFY 的详细信息，请参阅 [Note:35512.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=35512.1)

启动数据库到MOUNT状态（startup mount）

 对任何需要重新定位的数据文件进行重命名

例如： ALTER DATABASE RENAME FILE 'old\_name' TO 'new\_name';

确保所有必需的文件在线

例如： ALTER DATABASE DATAFILE 'name\_of\_file' ONLINE;

 恢复数据库

例如： RECOVER DATABASE

 打开数据库

例如： ALTER DATABASE OPEN;

[{go to content}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)

***完全恢复后***

一旦执行了完全恢复，最好在允许使用之前先检查数据库：

针对每个问题对象运行：

"ANALYZE <table\_name> VALIDATE STRUCTURE CASCADE"

检查表/索引是否存在不匹配。

如果有任何 undo 操作曾被放弃，此命令可能会显示不匹配，此时需要重建索引。

在应用程序级别检查表中数据的逻辑完整性。

[{go to content}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)

**(5B) 重建索引**

损坏对象为用户索引时，如果底层表没有损坏，则可以删除并重建该索引。

如果底层表也已经损坏，则应在重建任何索引之前先解决该表的坏块。

如果收集的信息表示索引有从属外键约束，则需要执行以下操作：

* 对于每个外键

ALTER TABLE <table> DISABLE CONSTRAINT <pk\_constraint>;

* 使用以下命令重建主键

ALTER TABLE <table> DISABLE CONSTRAINT <pk\_constraint>;

DROP INDEX <index\_name>;

CREATE INDEX <index\_name> .. with appropriate storage clause

ALTER TABLE <table> ENABLE CONSTRAINT <pk\_constraint>;

* 启用外键约束

ALTER TABLE <child\_table> ENABLE CONSTRAINT <fk\_constraint>;

对于索引分区，可以执行以下命令：

ALTER INDEX ... REBUILD PARTITION ...;

**注意：**

1. 不要使用“ALTER INDEX ..  REBUILD”命令重建损坏的非分区索引，这一点非常重要，因为此操作通常会尝试从包含坏块的现有索引段中构建新索引。

"ALTER INDEX ... REBUILD ONLINE" and "ALTER INDEX ... REBUILD PARTITION ..."

不会从旧索引段中构建新索引，因此可以使用。

1. 如果新索引包含的列为现有索引的子集，则 Create INDEX 可以使用现有索引中的数据。因此，如果您有两个损坏的索引，应在重建之前将两个都删除。
2. 重建索引时，请确保使用正确的存储选项。

[{go to content}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)

**(4C) 抢救表中数据**

如果损坏的对象为 TABLE 或 CLUSTER 或 LOBSEGMENT，则您必须明白，坏块内的数据已经丢失。

部分数据可能可以从块的 HEX 转储中，或从索引涵盖的列中抢救回来。

重要信息：

由于可能需要从索引中抢救坏块中的数据，因此最好不要删除任何现有索引，直至所有需要的数据提取完成。

从包含坏块的表中提取数据有多种方法。选择最恰当的方法，详细信息如下所述。这些方法的目的是从可访问的表块中提取尽可能多的数据。通常，将损坏的表重命名是一个比较好的方法，这样就可以使用正确的名称创建新对象。

例如： RENAME <emp> TO <emp\_corrupt>;

***从坏块表中提取坏块周围数据的方法***

1. 从 Oracle 7.2 开始（包括 Oracle 8.0、8.1 和 9i），可以跳过表中的坏块。

这是到目前为止最简单的提取表数据的方法，此方法在以下文档中做了讨论：

Extracting data using [DBMS\_REPAIR.SKIP\_CORRUPT\_BLOCKS](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=556733.1) or Event 10231  [Note:33405.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=33405.1)

如果坏块位于 IOT overflow 段，则应使用相同的方法，不同的是使用 Event 10233 和全索引扫描。

请注意，此方法只适用于块的“包装”已被标记为“坏块”的情况。例如：如果块报告 ORA-1578 错误。

如果问题为 ORA-600 或其他非ORA-1578 错误，则通常可以使用 DBMS\_REPAIR 将表中坏块标记为“软坏块”。这样在您访问该数据块时，系统将显示 ORA-1578 错误，从而可以使用 DBMS\_REPAIR.SKIP\_CORRUPT\_BLOCKS。

注意：被“FIX\_CORRUPT\_BLOCKS”程序标记为“坏块”的块在任何还原/恢复操作之后还将被标记为“坏块”.

有关使用 DBMS\_REPAIR 进行此操作的全部详细信息，请参阅相关文档，但概括起来说，步骤如下：

- 使用 DBMS\_REPAIR.ADMIN\_TABLES 创建管理表

- 使用 DBMS\_REPAIR.CHECK\_OBJECT 找到问题块

- 在损坏问题块之前将其中所有完好的数据导出。

- 使用 DBMS\_REPAIR.FIX\_CORRUPT\_BLOCKS 将找到的问题块标记为“坏块”，然后它们就会显示 ORA-1578

- 如果需要，使用 DBMS\_REPAIR.SKIP\_CORRUPT\_BLOCKS 跳过表中的坏块。

1. 从 Oracle 7.1 开始，可以使用 ROWID 范围扫描。

此功能的语法较为复杂，但可以使用 ROWID 提示选择坏块周围的数据。

由于 Oracle7 和 Oracle8 中的 ROWID 格式有所不同，故有两篇文章分别介绍此功能：

Using ROWID Range Scans to extract data in Oracle8 and higher [Note:61685.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=61685.1)

Using ROWID Range Scans to extract data in Oracle7 [Note:34371.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=34371.1)

1. 如果存在主键，则可以通过此索引选择表数据。

也可以通过任何其他索引选择一些数据。

此方法较慢，花费时间较长，通常只有 Oracle 7.0 版本才使用。此方法在 [Note:34371.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=34371.1) 中进行了介绍（此外，还介绍了 ROWID 范围扫描）。

1. 有多种抢救程序/PLSQL 脚本可用于抢救表中的数据。与上述方法相比，这些方法在设置和使用方面需要花费更长的时间，但常常能够处理除 ORA-1578 之外的各类坏块。

由于这些方法通常需要由技术支持人员给予大量的亲身指导，因此客户可能看不到关于这些方法的部分文章。

要使用以下程序，需要使用 Pro\*C，且需要了解如何构建 Pro\*C 可执行文件：

SALVAGE.PC for Oracle8.1 [Note:97357.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=97357.1)

SALVAGE.PC for Oracle7  [Note:2077307.6](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=2077307.6)

要使用以下程序，需要手动交互：

SALVAGE.SQL for Oracle7/8 [Note:2064553.4](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=2064553.4)

SALVAGE.SQL for Oracle7/8 Note:28308.1

The following is only possible in Oracle8i and 9i. The aim is to mark the block as corrupt and then use the SKIP\_CORRUPT table attribute to extract the table data:

Use DBMS\_REPAIR to mark the block corrupt [Note:68013.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=68013.1)

For corruption in a LONG column: Recreating a Table with a corruption in a LONG Note:876493.1

[{go to content}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)

***从包含损坏的 LOBSEGMENT 块的表中提取数据的方法***

在 LOB 段上不可以使用 DBMS\_REPAIR。

如果坏块 LOB 块未被表中的任何行引用，则应该可以使用 CREATE TABLE as SELECT (CTAS) 来按选择创建表，或按原样导出/删除/导入该表。

如果坏块 LOB 块被某个行引用，则应该可以使用不包括问题行的 WHERE 谓词进行选择或导出。

**警告：**

可以将问题行的 LOB 列值更新为 NULL，从而使 SELECT 操作不再返回ORA-1578 错误 \*但是\* 坏块将等待被重新使用，随着对行中的 LOB 列进行 INSERT 或 UPDATE 操作，当有问题的块被重新使用时，最后还是会报ORA-1578错误，那时的情况比已知行出现坏块更糟糕。

因此，只有您打算立刻重新创建表，才应该将 LOB 列设为 NULL。

***从坏块本身提取数据***

由于坏块本身已经“损坏”，则从该块中提取的任何数据都应被视为可疑数据。从坏块本身获取数据行的主要方法包括：

* 对于 TABLE 的块，Oracle Support 可以使用一款尝试解释块内容的工具。

"Convert HEX or BLOCKDUMP to Readable form" Note:47419.1

* 使用表中现有索引，利用落在坏块内的ROWID 来提取索引所涵盖的列数据，上文提到的 ROWID 范围扫描文章在接近结束时对此内容有所介绍：

  对于 Oracle8/8i，请参阅 [Note:61685.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=61685.1)

  对于 Oracle7，请参阅 [Note:34371.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=34371.1)

* 在 redo 流上可以使用 LogMiner 来查找向问题块加载数据的初始插入/更新操作。此处的主要因素是数据实际被放入问题块的时间。

例如，行 2 可能在昨天已插入，而行 1 可能在 5 年前已插入。

[{go to content}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)

**(5D) 忽略坏块**

出错时可以忽略坏块并接受报告的错误，或在应用程序级别阻止对出问题的块行进行访问。

例如：如果问题块/行位于子表中，则可以在应用程序级别阻止对父表中对应行的访问，从而子行就永不会被访问。（但要注意级联类约束）

这样做可能不利于批量访问数据的报告和其他任务，因此，为了阻止块在被访问时报错，前面 4C 中所述的 DBMS\_REPAIR 选项也不失为一个可取的方法。使用这种方法标记并跳过坏块提供了一种短期的解决方案，从而在计划停机时可以尝试进行完全数据抢救和/或恢复，或留出更多时 间在第二个（克隆）数据库上尝试其他恢复选项。但请注意，使用 [DBMS\_REPAIR.FIX\_CORRUPT\_BLOCKS](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=556733.1) 标记块坏块将导致标记的块在恢复后还是“坏块”。

忽略坏块对于快速老化且即将被清除的数据而言是比较好的选择（例如，在按日期分区的表中，较老的分区将在某时间点被删除）。

***忽略 LOB 段上的坏块***

在应用程序级别，可以忽略损坏的 LOB 列，直到可以重新构建该表。

确保不出现上述“警告”中的情形的一种方法是，确保应用程序只能通过表上的包含WHERE 谓词的视图来访分表中的数据。

例如：假设表 MYTAB（a number primary key，b clob）有一行或多行指向损坏的 LOB 数据。

ALTER TABLE MYTAB ADD ( BAD VARCHAR2(1) );

CREATE VIEW MYVIEW AS SELECT a,b FROM MYTAB WHERE BAD is null;

对任何问题行设置 BAD='Y'

如果只通过 MYVIEW 访问 MYTAB，该行将永不可见，因此也无法更新，从而实现了坏块条目隔离，直到问题解决。

很明显，此示例更多的是一个设计时解决方案，但某些应用程序可能已有类似机制，且可能只通过某个视图（或通过 RLS 策略）访问数据，从而提供某些选项来隐藏问题行。

[{go to content}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)

***针对忽略坏块的警告***

虽然可以忽略坏块，但需要注意的是，坏块在运行 DBVERIFY、RMAN 备份时仍然会以警告/错误等形式出现。

请务必仔细记录您将在这些工具中看到的任何坏块，尤其是您期望在使用 RMAN 时跳过的任何块（例如，设置了 MAX\_CORRUPT），并确保在清除坏块后移除任何对错误的“接受”选项。

例如：假设坏块已处理为忽略坏块，并在应用程序级别跳过问题行。

RMAN 可能被配置为在备份时接受坏块。

然后在稍后的表重组期间重新创建表。

如果 RMAN 配置未及时更新以反映目前已无任何错误，则 RMAN 可能会忽略稍后出现的某些其他坏块。

此外，还有重要的一点需要注意，忽略 table 段中的坏块可能导致查询返回不一致的结果。

例如：设置了 SKIP\_CORRUPT 的表可能出现不同的结果，具体取决于是使用了了索引扫描还是表访问。

其他报告可能只是报错。

请注意，如果忽略坏块但使用 [DBMS\_REPAIR.FIX\_CORRUPT\_BLOCKS](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=556733.1) 标记，系统会向坏块中写入 redo 信息，这可能会限制后续的恢复选项。

[**{go to content}**](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)

**(5E) 最后的选项**

如果你有 standby 环境（物理或逻辑），请首先对其进行检查。

无论问题发生在何种类型的块上，均可使用一种可能的选项，即将数据库或问题表空间恢复到出现坏块之前的某个时间点。此选项的困难之处在于，并不总能知道问题首次出现的时间。

DBVERIFY 通常可用于检查还原的文件是否存在坏块。

有关使用 DBVERIFY 的详细信息，请参阅[Note:35512.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=35512.1) 。尤其是，START= / END= DBV 选项可用于在还原的备份映像上快速进行首次测试，以检查问题块本身是否出错。

本部分列出了一些可用于进行恢复操作的最终选项。

如果您看到这里，则必定发生了以下一种或多种情况：

* 您丢失了非常重要的数据文件（或数据文件出现坏块），而没有问题文件的正常备份（无坏块）
* 既不处于 ARCHIVELOG 模式，也没有自文件创建以来的全部归档日志
* 完全恢复后仍重复出现问题

最后的机会：

请注意，如果您丢失了数据文件的所有副本，但仍具有自文件创建以来的全部归档日志，则仍有可能恢复该文件。

例如：

ALTER DATABASE CREATE DATAFILE '....' [as '...'] ;

RECOVER DATAFILE '....'

ALTER DATABASE DATAFILE '....' ONLINE;

如果您遇到这种情况，请在继续下面的操作之前先尝试使用这些步骤来恢复数据文件。

如果您到达这一步，就说明没有其他办法可以将文件恢复到当前时间点。此时最好关闭实例，并对当前数据库进行备份，以便在选用的措施失败后仍然能够回退到当前时间点。（例如：如果发现备份坏块）。

可用的一些选项概述如下：

恢复到早期的冷备份

- 例如：如果处于 NOARCHIVELOG 模式

从冷备份建立克隆数据库，并提取（导出）问题表，或传输问题表空间。

使用基于时间点的恢复将数据库恢复到一致的时间点

* 需要完好备份和任何所需的归档日志
* 必须还原所有文件且将整个数据库前滚到恰当的时间点。
* 可以在克隆数据库中执行基于时间点的恢复，然后将问题表空间传输到问题数据库，或将问题表利用导出/导入工具从克隆数据库导入到问题数据库。

表空间基于时间点的恢复

- 可以仅对受影响的表空间执行基于时间点的恢复。许多文档均对表空间基于时间点的恢复做了介绍，如 [Note:223543.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1526911.1&id=223543.1).

从逻辑导出/副本重新创建数据库

- 需要具有完好的数据库逻辑备份

- 注意：要使用此选项，必须重新创建数据库。

- 与其他选项一样，可以在克隆数据库中进行重新创建，只为获得问题表的完好映像。

如果已具有完好备份，使用 DB\_BLOCK\_CHECKING=TRUE 进行前滚将有助于找到首次出错的时间点。在调查恢复选项时，通常不需要关闭问题数据库。

例如：可以只将系统表空间和问题表空间数据文件还原到完全不同的位置和/或机器，作为不同的实例，以便于调查可以前滚到多久以前的时间点等。

自 Oracle9i 起，您还可以使用“试验恢复”选项来让自己摆脱一边研究选项一边必须不断还原备份的情形。

[{go to content}](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1526911.1&_adf.ctrl-state=ahlj0rovk_72&_afrLoop=349398812554383#INDEX)