Oracle学习笔记—存储结构

|  |  |
| --- | --- |
|  | 🢂 内容概览 |
|  | Why：此文档用来做什么？它存在的意义是什么？为解决什么问题？   |  | | --- | |  |   What：当前包含了那些内容？   |  | | --- | |  |   How：此文档应如何参考？   |  | | --- | |  |   Who：此文档适用于那些人员阅读参考？   |  | | --- | |  |   Summary：摘要   |  | | --- | |  |   Reference：参考文献   |  | | --- | | [Oracle bigfile大文件表空间](http://blog.chinaunix.net/uid-20779720-id-3078273.html)  [ORA-1653 oracle单个数据文件最大限制](http://blog.itpub.net/26972107/viewspace-774143/) | |

目录

[1 存储结构描述 4](#_Toc461100786)

[1.1 表空间（tablespace） 4](#_Toc461100787)

[1.2 段（segment） 5](#_Toc461100788)

[1.2.1 示例：查询数据库现有的段类型 5](#_Toc461100789)

[1.2.2 示例：精确定位段所在的位置 6](#_Toc461100790)

[1.3 块（block） 7](#_Toc461100791)

[1.4 区间（extent） 7](#_Toc461100792)

[1.5 Oracle块&操作系统块 7](#_Toc461100793)

[1.6 数据库文件 8](#_Toc461100794)

[1.6.1 控制文件 8](#_Toc461100795)

[1.6.2 联机重做日志文件 8](#_Toc461100796)

[1.6.3 数据文件 8](#_Toc461100797)

[1.6.4 示例：查看数据库文件信息 8](#_Toc461100798)

[2 文件存储技术 9](#_Toc461100799)

[2.1 文件系统 9](#_Toc461100800)

[2.2 裸设备（不推荐） 9](#_Toc461100801)

[2.3 ASM（推荐） 9](#_Toc461100802)

[2.3.1 逻辑卷 9](#_Toc461100803)

[3 表空间管理 9](#_Toc461100804)

[3.1 OEM管理方式 10](#_Toc461100805)

[4 使用表空间 10](#_Toc461100806)

[4.1 修改表空间状态 10](#_Toc461100807)

[4.2 查询表空间状态 11](#_Toc461100808)

[4.3 删除表空间 11](#_Toc461100809)

[4.4 扩展表空间 11](#_Toc461100810)

[4.5 移动数据文件 11](#_Toc461100811)

[4.6 显示表空间信息 12](#_Toc461100812)

[4.7 其他表空间 12](#_Toc461100813)

[5 相关SQL整理 12](#_Toc461100814)

# 存储结构描述

|  |
| --- |
|  |

1. **Oracle分为实例和数据库两个大的部分：**所有数据的而处理都在内存（实例）中执行，但数据存储发生在磁盘的数据库中；
2. **Oracle的存储结构分为物理结构和逻辑结构两部分：**物理结构描述数据在磁盘上是怎么存放的；逻辑结构用来给用户提供无差别的数据存储管理；从逻辑上讲，数据存储在段中（通常是数据表）；从物理上讲，数据存储在数据文件中；
3. Oracle数据库将逻辑存储从物理存储中抽象出来，从而实现数据操作和具体物理存储结构的解耦，用户只用关心逻辑结构，无需关心具体的物理存储结构，从而降低操作复杂性，增加灵活性；
4. Oracle中逻辑结构包括表空间、段、区和块；数据库由表空间构成，表空间由段构成，段又由区构成，而区又是由块构成的一种结构，块可以提高数据库效率；

## 表空间（tablespace）

1. 表空间是数据库的逻辑组成部分，作为物理存储与逻辑存储的中转点。从物理上讲，数据库数据存放在数据文件中；从逻辑上讲，数据库则是存放在表空间中的，表空间由一个或多个数据文件组成；
2. 表空间用于从逻辑上组织数据库的数据。数据库逻辑上是由一个或多个表空间组成。控制数据库占用的磁盘空间；dba可以将不同数据类型部署到不同的位置，这样有利于提高i/o性能，同时利于备份和恢复等管理操作；
3. 一个表空间可以关联一个或者多个数据文件，作为数据存放的物理结构；
4. 同时，表空间中又可以创建多个段，用来表示不同的数据对象；
5. 表空间的划分比较灵活，可以根据不同的业务划分，也可以按照存放数据的不同划分为临时表空间，索引表空间，数据表空间等；
6. 建表时，默认放在system表空间，system表空间为系统表空间；
7. 通常一个小数据文件大小有限制，故一般创建表空间会提前规划好数据文件的大小，并随着数据量的增大，不断追加数据文件；
8. 表空间是逻辑概念，它对应的数据文件既可以放在普通的文件系统中，也可以存放在Oracle ASM中；一般使用HA或RAC方案部署，必须存放在ASM上。
9. 表空间存储3类对象：永久对象，临时对象，撤销对象；

## 段（segment）

1. 表示需要存放在表空间中的数据对象，比如普通数据表，索引段，分区表的一个分区等；
2. 一个段仅存在一个表空间中，可以映射到所属表空间的多个数据文件中；故段的大小并不直接依赖数据文件的大小，也无须关心要存放在哪个数据文件中；
3. 段属于模式对象，故一个端需要由模式名和段名共同确定。

注意：存储过程、视图、序列等可编程模式对象不是段，不存储数据，而是存放到数据字典中；

### 示例：查询数据库现有的段类型

|  |
| --- |
| *--查看数据库的段类型*  select segment\_type , count(1) from dba\_segments group by segment\_type; |

结果如下：

|  |
| --- |
|  |

主要段类型如下：

|  |  |
| --- | --- |
| 类型 | 描述 |
| TABLE | 包含数据行的堆结构表，即普通表； |
| INDEX | 索引是键值的有序表，对应的指针ROWID指向每行的物理位置，ROWID指定了行位于哪个数据文件的哪个Oracle块中，并指定行在块中的编号； |
| TYPE2 UNDO | 撤销段，存储数据更改前的版本，为回滚、读取一致性、隔离提供能力； |
| ROLLBACK | 回滚段，9i以后不推荐在普通行使用；使用基于撤销段的自动撤销管理； |
| TABLE PARTITION | 表分区，分区表的一个表分区即为一个段；一个分区表的分区可以位于不同的表空间中； |
| INDEX PARTITION | 索引分区，即分区索引的一个索引分区即为一个段； |
| LOBSEGMENT | 当表中的字段类型为LOB时，LOB为一个段，表中存放的是指向段的地址； |
| LOBINDEX | LOB字段本身可以建立索引，索引单独为一个段； |
| LOB PARTITION | LOB字段也可以进行分区； |
| CLUSTER | 与分区相反，可以将多个表放入一个段； |
| NESTED TABLE | 嵌套表，表中的列类型本身又具有列，存放在自己的段中； |

### 示例：精确定位段所在的位置

1. 查询数据表对应的区间信息；

|  |
| --- |
| *--查询区间信息*  select tablespace\_name , file\_id , extent\_id , block\_id , blocks , bytes  from dba\_extents  where owner = upper('scott') and segment\_name = upper('emp'); |

结果如下：

|  |
| --- |
|  |

1. 为表创建新区间，再次查询区间信息；

|  |
| --- |
| *--创建新区间*  alter table scott.emp allocate extent;  *--再次查询区间信息*  select tablespace\_name , file\_id , extent\_id , block\_id , blocks , bytes  from dba\_extents  where owner = upper('scott') and segment\_name = upper('emp'); |

结果如下：

|  |
| --- |
|  |

1. 查询区间所在的数据文件；

|  |
| --- |
| *--查询区间所在的数据文件*  select tablespace\_name , file\_name from dba\_data\_files where file\_id = 4; |

结果如下：

|  |
| --- |
|  |

**注意：**Oracle创建区间并不会仅在一个数据文件中连续创建区间；故如果需要在指定数据文件创建区间，需要用到如下语句：

|  |
| --- |
| alter table 模式名.表名 allocate extent storage ( datafile '数据文件名'); |

1. 查询段所在的表空间；

|  |
| --- |
| *--查询段所在的表空间*  select tablespace\_name from dba\_segments where segment\_name = upper('emp') and owner = upper('scott'); |

## 块（block）

1. 数据库I/O的基本单元，一个数据文件由一群连续编号的块组成，每个块都由数据文件中的编号唯一标识；
2. 一个表空间中的所有块大小都是相同的，Oracle11g默认块大小为8K；
3. 一个块可以存放一行或者多行记录，数据的读写操作都是以块为基本单位的；
4. 当会话需要读取一行数据时，会把整个块读入数据库缓冲区中；反之，当一行数据发生变动时，DBWn也会将整个块回写到源数据文件中，覆盖以前的块；
5. 块的大小由DB\_BLOCK\_SIZE参数设置，一般可以设置为2K~32K；由于DB\_BLOCK\_SIZE参数用来设置SYSTEM表空间的数据文件格式，所以一个数据库的块大小一旦确认就不能更改，只能重新建库；

## 区间（extent）

1. 如果在管理表空间时每次只管理一个块，比较费力，故抽象出区间，用来描述一个数据文件中一组连续编号的块；
2. 一个段可以包含多个区间，这些区间可以分布在表空间的一个或者多个数据文件中；区间根据所属的段从0开始连续编号；
3. 一个区间由一组连续编号的块组成。每个块都包含标题区域和数据区域；

## Oracle块&操作系统块

1. 从物理结构看，数据库文件由许多操作系统块构成。数据文件和操作系统块的结构完全取决于文件系统；
2. 操作系统块是文件系统的I/O单元；一个进程可能只想从磁盘读取一个字节，但I/O系统必须读取至少一个操作系统块；操作系统块通常可以设置；
3. 一个Oracle块的大小应该设置为一个操作系统块的一个或多个倍数；要避免操作系统块大小大于Oracle块大小；

## 数据库文件

数据库包含三类文件：**控制文件**，**联机重做日志文件**，**数据文件**；

### 控制文件

### 联机重做日志文件

### 数据文件

1. 数据存储在数据文件中；
2. 用户永远看不到物理数据文件，只能看到逻辑段；
3. 数据文件分为3类：**普通数据文件**，**大数据文件**，**临时文件**；

|  |  |  |  |
| --- | --- | --- | --- |
| **文件类型** | **用途** | **数量限制** | **大小限制** |
| 普通数据文件 | 作为表空间数据的物理存储结构 | 1个表空间最多1022个； | 单个数据文件可以包含4M个blocks，故取决于block的大小，默认block大小为8K，则最大存放32G; |
| 大数据文件 | 作为表空间数据的物理存储结构 | 1个表空间对应1个； | 使用32位标识block的编号，故最多支持4G个block，默认block大小为8K，则最大存放32T；  注意：需要考虑文件系统本身支持的最大文件大小； |
| 临时文件 | 作为临时表空间数据的物理存储结构，数据不记录日志，不做备份恢复； | 1个表空间最多1022个； | 单个数据文件可以包含4M个blocks，故取决于block的大小，默认block大小为8K，则最大存放32G; |

### 示例：查看数据库文件信息

|  |
| --- |
| */\*\**  *了解数据库的数据存储结构*  *\*\*/*  *--确定控制文件的名称和大小*  select name, ( block\_size \* file\_size\_blks/1024/1024 ) MB from v$controlfile;  *--确定联机重做日志文件的名称和大小*  select member , bytes/1024/1024 MB from v$log join v$logfile using (group#);  *--确定数据文件和临时文件的名称和大小*  select name , bytes/1024/1024 MB from v$datafile  union all  select name , bytes/1024/1024 MB from v$tempfile; |

# 文件存储技术

## 文件系统

即普通的操作系统上的文件系统；适合单实例Oracle使用；

## 裸设备（不推荐）

早期用来实现并行服务器的方式，已被Oracle RAC替代；

## ASM（推荐）

1. ASM（Automatic Storage Management，自动存储管理），可以理解为Oracle数据库自己的文件系统（逻辑卷管理器），用于实现HA或者RAC时使用，性能高于普通文件系统，适合Oracle集群使用；
2. ASM可以用来保存数据文库文件（控制文件，联机重做日志文件，数据文件），存储备份文件，归档重做日志文件，Data Pump文件等。但不能存放Oracle主目录的二进制文件以及日志文件等，这些文件只能存放在常规文件系统上；

### 逻辑卷

1. 逻辑卷是对物理磁盘的一层抽象；一个物理卷是一个完整的磁盘。而逻辑卷的底层存储可能不仅有一个物理卷，可以在逻辑卷上创建文件系统；
2. 一个逻辑卷可以比组成它的任何物理卷大的多；
3. 可以利用逻辑卷的数据条带化和镜像特性，充分利用多个物理卷产生的性能和安全潜力；
4. 数据条带化：将数据分配到多个物理卷上，提升I/O读写性能；
5. 镜像：将相同的数据同时写入多个物理卷，提升数据安全性；

# 表空间管理

Oracle表空间的管理完全可以通过OEM进行，直接登录OEM执行可视化的表空间管理操作，简单方便；但为了体现专业素质，也可以通过管理命令完成；

## OEM管理方式

进入OEM界面，选择Server🡪Storage🡪Tablespaces，即可进入表空间管理界面；

|  |
| --- |
|  |

### 示例：查看表空间使用情况

|  |
| --- |
| *--查询表空间的使用情况*  select t.tablespace\_name as name,  d.allocated as allocated,  u.used,  f.free,  t.status,  d.cnt,  contents,  t.extent\_management as extman,  t.segment\_space\_management as segman  from dba\_tablespaces t,  ( select sum(bytes)/1024/1024 as allocated,  count(file\_id) as cnt  from dba\_data\_files  where tablespace\_name='EXAMPLE'  ) d,  ( select sum(bytes)/1024/1024 as free  from dba\_free\_space  where tablespace\_name='EXAMPLE'  ) f,  ( select sum(bytes)/1024/1024 as used  from dba\_segments  where tablespace\_name='EXAMPLE'  ) u  where t.tablespace\_name='EXAMPLE'; |

结果如下：

|  |
| --- |
|  |

## 表空间分类

|  |  |
| --- | --- |
| **类型** | **描述** |
| 小文件表空间 | 1. 使用smallfile创建的表空间； 2. 可以关联多个数据文件，文件上限1022个； 3. 使用V$DATAFILE或DBA\_DATA\_FILES查看； |
| 大文件表空间 | 1. 使用bigfile创建表空间； 2. 仅可以关联一个大数据文件； 3. 使用V$DATAFILE或DBA\_DATA\_FILES查看； |
| 临时表空间 | 1. 用于存放临时数据的表空间； 2. 可以关联多个临时文件，文件上限1022个； 3. 使用V$TEMPFILE或DBA\_TEMP\_FILES查看； |

## 创建表空间

### 示例：创建表空间

|  |
| --- |
| *--创建普通表空间*  create smallfile tablespace tbs\_cime\_data  datafile 'D:/developer/oracle/oradata/cime/tbs\_cime\_data\_01.dbf'  size 100m autoextend on next 10m maxsize 10G,  'D:/developer/oracle/oradata/cime/tbs\_cime\_data\_02.dbf'  size 100m autoextend on next 10m maxsize 10G  logging  extent management local  segment space management auto  default nocompress; |

**说明：**

1. 第1行：指定表空间的名称，以及表空间为小文件表空间，如果是大文件表空间，则可以使用bigfile替换字段，默认为smallfile；
2. 第2~5行：指定表空间对应的数据文件名称、初始大小、自动扩展、每次扩展增量、文件能够达到的最大大小；默认情况下不会启用自动扩展；
3. 第6行：为表空间中所有操作生成重做日志，默认操作；索引表空间可以考虑不生成重做日志；
4. 第7行：表空间使用本地管理区间（位图），默认操作，不推荐使用字典管理；
5. 第8行：表空间中的段将使用位图跟踪块的使用情况，默认操作；
6. 第9行：表空间中的段不压缩，默认操作；

### 示例：创建大数据文件表空间

|  |
| --- |
| *--创建大数据文件表空间*  create bigfile tablespace tbs\_big  datafile 'D:/developer/oracle/oradata/cime/tbs\_big.dbf'  size 100m autoextend on next 10m maxsize 10G  logging  extent management local  segment space management auto  default nocompress; |

### 示例：创建临时表空间

|  |
| --- |
| *--创建临时表空间*  create temporary tablespace tbs\_cime\_temp  tempfile 'D:/developer/oracle/oradata/cime/tbs\_cime\_temp\_01.dbf'  size 100m autoextend on next 10m maxsize 10G  extent management local; |

## 重命名表空间

|  |
| --- |
| **alter tablespace 旧名称 rename to 新名称；** |

注：重命名表空间，并不会重命名相应的数据文件名称，需要专门修改数据文件名称；

## 使用表空间

### 创建用户时指定默认表空间

|  |
| --- |
| *--创建用户关联表空间*  create user cime identified by cime  default tablespace tbs\_cime\_data  temporary tablespace tbs\_cime\_temp;  *--对用户授权*  grant create session,connect,dba to cime; |

### 创建数据表时指定表空间

创建数据表，并放入指定的表空间；

|  |
| --- |
| **建表语句 TABLESPACE 表空间名；** |

## 修改表空间状态

当建立表空间时，表空间处于联机状态（online），此时该表空间是可以访问的，并且该表空间是可以读写的，即可以查询该表空间的数据，而且还可以在表空间执行各种语句。但是在进行系统维护或是数据维护时，可能需要改变表空间的状态。一般情况下，由特权或是dba来操作； 语法如下：

|  |
| --- |
| alter tablespace 表空间名 状态名 [ normal | immediate | temporary ] |

修改方式说明：

1） normal：默认，正常下线数据库；

2） immediate：直接下线数据库，不考虑缓冲区数据是否落盘；可能造成数据损坏；

3） temporary：强制设置检查点，对损坏文件立刻脱机；

### 使表空间脱机

|  |
| --- |
| **ALTER TABLESPACE 表空间名OFFLINE;** |

### 使表空间联机

|  |
| --- |
| **ALTER TABLESPACE 表空间名 ONLINE;** |

### 只读表空间

表空间上不可以执行DML语句，只能查询；

|  |
| --- |
| **ALTER TABLESPACE 表空间名 READ ONLY;** |

### 恢复读写

|  |
| --- |
| **ALTER TABLESPACE 表空间名 READ WRITE;** |

## 查询表空间状态

### 查询表空间中有那些表；

|  |
| --- |
| **SELECT \* FROM dba\_tables WHERE TABLESPACE=’表空间名’;** |

### 查看数据表属于哪个表空间；

|  |
| --- |
| **SELECT tablespace\_name,table\_name FROM dba\_tables WHERE table\_name=’表名’;** |

## 删除表空间

|  |
| --- |
| **DROP TABLESPACE 表空间名 [INCLUDING CONTENTS AND DATAFILES];** |

## 扩展表空间

表空间是由数据文件组成的，表空间的大小实际上就是数据文件相加后的大小。若要扩展表空间的大小可以用如下方式修改：

**方式一**：增加数据文件

|  |
| --- |
| **ALTER TABLESPACE 表空间名 ADD DATAFILE ‘文件路径’ SIZE 文件大小;** |

**方式二**：增加数据文件大小

|  |
| --- |
| **ALTER DATABASE DATAFIEL ‘文件路径’ RESIZE 文件大小;** |

**方式三**：设置文件自动增长

|  |
| --- |
| **ALTER DATABASE DATAFILE ‘文件路径’ AUTOEXTEND ON NEXT 文件大小增量 MAXSIZE 500;** |

**注**：在Oracle中，文件的大小不能超过500M;

## 移动数据文件

如果数据文件所在的磁盘损坏时，该数据文件将不能再使用，为了能够重新使用，需要将这些文件的副本移动到其他磁盘，然后恢复；步骤如下：

1）确定数据文件所在的表空间

|  |
| --- |
| **SELECT tablespace\_name FROM dba\_data\_files WHERE file\_name=’文件路径’;** |

2）使表空间脱机

确保数据文件的一致性，将表空间转变为offline的状态；

|  |
| --- |
| **ALTER TABLESPACE表空间名 OFFLINE;** |

3）使用命令移动数据文件到指定的目标位置

|  |
| --- |
| **HOST MOVE 要移动的文件路径 文件路径;** |

4）移动数据文件

|  |
| --- |
| **ALTER TABLESPACE 表空间名 RENAME DATAFILE ‘移动前文件路径’ TO ‘移动后文件路径’** |

5）使表空间联机

|  |
| --- |
| **ALTER TABLESPACE 表空间名 ONLINE;** |

## 显示表空间信息

* 查询数据字典视图dba\_tablespaces，显示表空间信息：

|  |
| --- |
| **SELECT tablespace\_name FROM dba\_tablesapces;** |

显示表空间所包含的数据文件

* 数据字典视图dba\_data\_files

|  |
| --- |
| SELECT file\_name,bytes FROM dba\_data\_files WHERE tablespace\_name=’表空间名’; |

## 其他表空间

* 索引表空间
* undo表空间
* 临时表空间
* 非标准块表空间

# 空间分配过程

1. 将空间分配给表空间：通过控制数据文件大小；
2. 将空间分配给段：通过分配区间完成；
3. 将空间分配给行：通过维护那些跟踪每个块中的空闲空间量的位图来完成；

# 相关字典

## dba\_segments

## dba\_extents

## v$tablespace和dba\_tablespaces

## v$datafile和dba\_data\_files

## v$tempfile和dba\_temp\_files

## v$controlfile

# 问题思考

## V$视图和dba视图内容为什么不同？

1. V$视图数据源自于控制文件，dba视图数据源自于数据字典，故二者数据可能存在差别；
2. V$视图可以在数据加载模式下使用，dba视图不可以；

# 相关SQL

|  |
| --- |
| */\*\**  *使用system登录*  *\*\*/*  *--查看数据库的段类型*  select segment\_type , count(1) from dba\_segments group by segment\_type;  */\*\**  *精确定位段所在的位置*  *\*\*/*  *--查询区间信息*  select tablespace\_name , file\_id , extent\_id , block\_id , blocks , bytes  from dba\_extents  where owner = upper('scott') and segment\_name = upper('emp');    *--创建新区间*  alter table scott.emp allocate extent;  *--再次查询区间信息*  select tablespace\_name , file\_id , extent\_id , block\_id , blocks , bytes  from dba\_extents  where owner = upper('scott') and segment\_name = upper('emp');  *--查询区间所在的数据文件*  select tablespace\_name , file\_name from dba\_data\_files where file\_id = 4;  *--查询段所在的表空间*  select tablespace\_name from dba\_segments where segment\_name = upper('emp') and owner = upper('scott');  */\*\**  *了解数据库的数据存储结构*  *\*\*/*  *--确定控制文件的名称和大小*  select name, ( block\_size \* file\_size\_blks/1024/1024 ) MB from v$controlfile;  *--确定联机重做日志文件的名称和大小*  select member , bytes/1024/1024 MB from v$log join v$logfile using (group#);  *--确定数据文件和临时文件的名称和大小*  select name , bytes/1024/1024 MB from v$datafile  union all  select name , bytes/1024/1024 MB from v$tempfile;  */\*\**  *查看表空间使用情况*  *\*\*/*  *--查询表空间的使用情况*  select t.tablespace\_name as name,  d.allocated as allocated,  u.used,  f.free,  t.status,  d.cnt,  contents,  t.extent\_management as extman,  t.segment\_space\_management as segman  from dba\_tablespaces t,  ( select sum(bytes)/1024/1024 as allocated,  count(file\_id) as cnt  from dba\_data\_files  where tablespace\_name='EXAMPLE'  ) d,  ( select sum(bytes)/1024/1024 as free  from dba\_free\_space  where tablespace\_name='EXAMPLE'  ) f,  ( select sum(bytes)/1024/1024 as used  from dba\_segments  where tablespace\_name='EXAMPLE'  ) u  where t.tablespace\_name='EXAMPLE';      */\*\**  *创建表空间示例*  *\*\*/*  *--创建普通表空间*  create smallfile tablespace tbs\_cime\_data  datafile 'D:/developer/oracle/oradata/cime/tbs\_cime\_data\_01.dbf'  size 100m autoextend on next 10m maxsize 10G,  'D:/developer/oracle/oradata/cime/tbs\_cime\_data\_02.dbf'  size 100m autoextend on next 10m maxsize 10G  logging  extent management local  segment space management auto  default nocompress;  *--创建临时表空间*  create temporary tablespace tbs\_cime\_temp  tempfile 'D:/developer/oracle/oradata/cime/tbs\_cime\_temp\_01.dbf'  size 100m autoextend on next 10m maxsize 10G  extent management local;  *--创建用户关联表空间*  create user cime identified by cime  default tablespace tbs\_cime\_data  temporary tablespace tbs\_cime\_temp;  *--对用户授权*  grant create session,connect,dba to cime; |