### Oracle 找出需要回收高水位的表

2009-10-21 18:49:31|  分类： [系统管理技术](http://junfengwang060905.blog.163.com/blog/#m=0&t=1&c=fks_087064092094086065092084084095093082083065087085085067)|字号 订阅

第一种方法：

前提要首先分析表，日期越近，越精确。

select ds.owner, ds.segment\_name,round(ds.bytes/1024/1024,2) MB,dt.num\_rows,dt.last\_analyzed

from dba\_segments ds,dba\_tables dt

where dt.table\_name=ds.segment\_name

and dt.owner=ds.owner

and dt.owner ='用户'

and round(bytes/1024/1024,0)\*2000>num\_rows

order by round(bytes/1024/1024,2);

这个sql列出了：所有者，段名，行数，分析时间

主要是加了一个条件（ round(bytes/1024/1024,0)\*1000>num\_rows），用表大小（MB）乘以2000大于行数，基于的原理就是一般情况下一个1M的表大小在1万行左右，这里我把它看成1M的表2000行（行数已经很少了）。

所以在这种条件下得到得表，很大程度上是需要回收高水位线的（当然你也可以把2000改成的更小些，但这个值在我接触的数据库中已经是很少的了），

当然你也要核实一下，看表里是否有大的类型字段，如blong等。

第二种方法：

首先还是分析表，以得到最准确的统计，

select  round((1-a.used/b.num\_total)\*100,0)  percent from

(SELECT COUNT (DISTINCT SUBSTR(rowid,1,15)) Used FROM 表名) a,

(select blocks num\_total from dba\_tables where table\_name='表名 and owner='用户名') b;

通过dba\_tables里的blocks字段和有数据的字段做对比，在用1减，求出没有数据的块的百分比（一般超过20%就需要回收了）从而判定是否需要回收高水位线

这种方法很准确，也是我现在一直用的方法，上面的方法是我最早开始使用的，随着不断地学习和积累找到了更准确的判断方法，在此希望大家不断学习积累，

这样我们才能不断地进步，不断地提高。

如何回收高水位线

如果表的高水位线比较高，或者表经历了大数据量的产生操作，经常会使用ALTER TABLE MOVE的方法来减少表占用的空间。

　　不过今天才发现以前对MOVE的了解一直存在偏差。

　　看一个简单的例子：

　　两张表几乎一样大，而且都是空表。但是通过MOVE之后，得到的结果却完全不同。

　　SQL> SELECT SEGMENT\_NAME, BYTES/1024/1024/1024 G

　　2 FROM USER\_SEGMENTS

　　3 WHERE SEGMENT\_NAME LIKE 'T\_BIG\_TABLE%';

　　SEGMENT\_NAME G

　　------------------------------ ----------

　　T\_BIG\_TABLE 1

　　T\_BIG\_TABLE2 1.3125

　　SQL > ALTER TABLE T\_BIG\_TABLE MOVE;

　　表已更改。

　　SQL> ALTER TABLE T\_BIG\_TABLE2 MOVE;

　　表已更改。

　　SQL> SELECT SEGMENT\_NAME, BYTES/1024/1024/1024 G

　　2 FROM USER\_SEGMENTS

　　3 WHERE SEGMENT\_NAME LIKE 'T\_BIG\_TABLE%';

　　SEGMENT\_NAME G

　　------------------------------ ----------

　　T\_BIG\_TABLE 1

　　T\_BIG\_TABLE2 .000061035

　　SQL > SELECT COUNT(\*) FROM T\_BIG\_TABLE;

　　COUNT(\*)

　　----------

　　0

　　SQL > SELECT COUNT(\*) FROM T\_BIG\_TABLE2;

　　COUNT(\*)

　　----------

　　0

造成这个现象的原因是由于T\_BIT\_TABLE表只包含的一个初始EXTENT，而这个EXTENT的大小是1G，而T\_BIT\_TABLE2则包含很多的64K大小的EXTENT。

　　SQL> SET LONG 10000

　　SQL> SELECT DBMS\_METADATA.GET\_DDL('TABLE', TABLE\_NAME)

　　2 FROM USER\_TABLES

　　3 WHERE TABLE\_NAME LIKE 'T\_BIG\_TABLE%';

　　DBMS\_METADATA.GET\_DDL('TABLE',TABLE\_NAME)

　　--------------------------------------------------------------------------------

　　CREATE TABLE "YANGTK"."T\_BIG\_TABLE2"

　　( "ID" NUMBER,

　　"NAME" VARCHAR2(30)

　　) PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 NOCOMPRESS LOGGING

　　STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

　　PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER\_POOL DEFAULT)

　　TABLESPACE "LOB\_SPACE"

　　CREATE TABLE "YANGTK"."T\_BIG\_TABLE"

　　( "ID" NUMBER,

　　"NAME" VARCHAR2(30)

　　) PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 NOCOMPRESS LOGGING

　　STORAGE(INITIAL 1073741824 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

　　PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER\_POOL DEFAULT)

　　TABLESPACE "LOB\_SPACE"

　　也就是说MOVE操作会根据原表的INITIAL大小为新表建立第1个EXTENT。MOVE不会自动减少表的初始扩展的大小。

　　因此对于表T\_BIG\_TABLE这种情况，执行MOVE的时候应该指定新的存储参数：

　　SQL> ALTER TABLE T\_BIG\_TABLE MOVE STORAGE (INITIAL 1M);

　　表已更改。

　　SQL> SELECT SEGMENT\_NAME, BYTES/1024/1024/1024 G

　　2 FROM USER\_SEGMENTS

　　3 WHERE SEGMENT\_NAME LIKE 'T\_BIG\_TABLE%';

　　SEGMENT\_NAME G

　　------------------------------ ----------

　　T\_BIG\_TABLE .000976563

　　T\_BIG\_TABLE2 .000061035