**Oracle Lob类型相关参数以及性能影响**

Oracle Lob类型是Oracle数据库提供的一种将非标准格式文件或者信息保存在数据库存储系统的一种列类型。相对于传统的关系型数据，Lob类型的数据覆盖类型更加多样，同样也就面对更多的存储、检索和更新方面的性能问题。

目前，Oracle显示支持的Lob类型子类包括四个，分别为CLOB、BLOB、NLOB和BFILE。针对不同的数据类型，我们选择不同的Lob子类型进行匹配。在四种类型中，前三种是真实将数据保存在数据库中，而BFILE的真实数据则是保存在数据库外。

本篇将针对Lob类型在存储上使用的一些特殊属性，进行简单的介绍说明。同时推荐一些常用的Lob存储优化手段。

**1、LobSegment和LogIndex**

包括Lob类型列的数据表在段结构上存在一些特殊性。最直接的表现就是Lob数据段和Lob索引段的额外建立。

SQL> select \* from v$version;

BANNER

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

Oracle Database 10g Enterprise Edition Release 10.2.0.1.0 - Prod

PL/SQL Release 10.2.0.1.0 - Production

CORE     10.2.0.1.0       Production

TNS for 32-bit Windows: Version 10.2.0.1.0 - Production

NLSRTL Version 10.2.0.1.0 – Production

当我们建立一个数据表T，其中包括lob类型列时。

SQL> create table t (id number, cl clob);

Table created

SQL> select segment\_name, segment\_type, tablespace\_name from user\_segments;

SEGMENT\_NAME                   SEGMENT\_TYPE       TABLESPACE\_NAME

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

SYS\_LOB0000056106C00002$$      LOBSEGMENT         USERS

T                   TABLE              USERS

SYS\_IL0000056106C00002$$       LOBINDEX           USERS

9 rows selected

当数据表中有一个lob类型列时，Oracle会自动多创建出两个列，一个是保存lob数据的lob segment，另一个是对这个lob列生成的索引。注意，这个索引是由Oracle自动创建和管理，与lob列共生不能被单独删除。

此处存在一些经验实践项目，如果lob列对应的段和索引很大时，处于平衡IO的目的，可以考虑将lob列段和索引与数据表分开，单独放置在其他表空间上。进而实现放置在不同数据磁盘的目的。这部分实验可以参见笔者之前文章《Oracle Lob类型存储浅析》（<http://space.itpub.net/17203031/viewspace-708336）>。

注意，这里我们的Lob数据段和索引段是要放置在同一个表空间中的。这个与我们通常的经验相左。在Oracle 8i中，用户是可以指定lob index的创建和管理的。但是从9i开始，这种控制权就被Oracle内部所回收，如果我们试图使用8i时代的DDL语句进行指定，将lob数据和索引分离保存，从前端上执行语句是不会有错误报出的。但是最终结果上没有任何变化。相当于强制要求lob data segment和lob index segment存放在同一个表空间上。

**2、Lob数据表存储参数**

Lob列数据和一般数据存在很大的差异，所以在存储方面也有很多独特的特性存在。下面分别介绍一些与Lob存储相关的重要参数类型和含义。

CREATE TABLE "SCOTT"."T"

   ( "ID" NUMBER,

       "CL" CLOB

   )

  TABLESPACE "USERS"

 LOB ("CL") STORE AS (

  TABLESPACE "USERS" ENABLE STORAGE IN ROW CHUNK 8192 PCTVERSION 10

  NOCACHE LOGGING

  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

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

上面是将数据表T的创建DDL语句抽取出的代码片段。其中标注红色的部分，明显是针对Lob类型列CL进行的独特store参数特性。其中包括enable storeage in row、chunk、pctversion/retention、cache和logging。这些参数的设置与Oracle Lob类对象存储和性能息息相关。

        Chunk——lob对象存储最小单元

通常情况下，普通数据存储是以字节byte为单位写入到数据块datablock中。而lob类型存在一些特殊。LobSegment作为Lob数据的主体，是以chunk作为最小的分配单元。一个chunk的大小是数据块大小db\_block\_size的整数倍。最大为32K。

一个chunk只能保存一个lob对象的数据。如果出现填不满的情况，空闲空间是被置空的。

SQL> select table\_name, column\_name, chunk from user\_lobs;

TABLE\_NAME COLUMN\_NAM      CHUNK

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

T          CL               8192

如果一个chunk是8K的空间，对一个lob只有1K的大小，那么Oracle为这个lob只会分配一个chunk，而且这个chunk是为了这个lob唯一独占的。剩余的7K空间是剩下留空的。

同数据块大小db\_block\_size确定之后不能修改有差别，chunk是数据lob段级别的参数。从上面的定义语句可以看出，即使在一个数据表中有多个lob列，也是可以分别制定不同的chunk大小的。所以说，chunk大小的选择要针对lob的实际而“度身定制”！

一旦确定了chunk的大小，就不能在数据表中进行修改了。

那么为lob选择chunk大小是依据什么准则呢？根据Oracle推荐，如果lob列保存的数据是小于32K，那么设置的chunk大小建议为lob的60%比较合适。如果lob列保存的数据是大于32K，那么建议设置chunk的大小为经常进行update lob变化的大小。

        In-row storage and out-of-row storage

Oracle的确为了lob列值过大的情况，准备了专门lobsegment空间。但是实际上，Oracle是可以选择lob数据的保存方式的。Enable storage in row就意味着是否可以in row存储功能，提供将lob数据保存在数据段，和其他列保存在一起的可能。

当我们开启了in row storage，Oracle就会根据处理lob的大小选择是否将其保存在数据行中，而不是专门的lobsegment。当lob大小小于4000 bytes，Oracle会将它保存在数据表段中，和同行的其他列保存在一起。否则，才会保存在专门的lobsegment中。如果lob数据是保存在lobsegment里，那么数据行对应的位置上只会有一个很小的指向信息，记录对应的lobsegment位置。

SQL> select table\_name, column\_name, in\_row from user\_lobs;

TABLE\_NAME COLUMN\_NAM IN\_ROW

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

T          CL         YES

对lob store in row的情况而言，数据行的行迁移（row migration）和行链接（row chaining）是一直存在的。而且随着数据块大小与lob分配不匹配，这两种现象会越来越严重。对lob store in row的数据来说，是不保存在chunk单元上的。

如果lob数据保存在其他的数据段上，那么意味着我们每次读取一行全部内容，都需要读多个数据段的内容。这个显然不是我们期望的结果。所以，在一般情况下，我们都会选择enable storage in row来优化存储。

考虑使用disable storage in row的可能性不是没有，如果我们通常浏览数据是不包括lob类型或者很少包括lob类型，可以优先选择使用lob类型。

        Cache/NoCache

普通数据表也是有cache参数选项，那么参数意味着数据块读入到buffer cache中的管理策略：是一直驻留（Keep Pool）、常规驻留（Default Pool）还是用后清除（Recycle Pool）。

对Lob数据而言，cache和nocache的含义不同，意味着在使用访问lob类型数据的时候，是否通过buffer cache进行读写。如果是nocache选项，则意味着Oracle在操作该Lob列的时候会绕开Buffer Cache，进行direct write/read操作。

SQL> select table\_name, column\_name, cache from user\_lobs;

TABLE\_NAME COLUMN\_NAM CACHE

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

T          CL         NO

设置这个参数的背景是lob类型数据在buffer cache中的管理问题。有时候由于体积等原因，buffer cache中的lob类型数据是会带来很多的性能问题，特别是将其flush出buffer cache的过程。

通常情况下，我们还是希望设置为cache的，用来优化性能。直接对文件的direct write/read通常不会有很好的性能。

注意：cache/nocache参数是针对在lobsegment中以chunk保存的数据而言的，对能够in-row storage的lob而言，这个是不起作用的。因为in-row是以data block为基本操作单位。

        Lob类型数据的一致读consisten read

多版本一致读、当前读是Oracle数据库具有的独特属性，也是其最重要的特性之一。借助undo表空间的前镜像数据保存，Oracle Server Process可以访问到一些特定时间点（SCN）的数据，作为一致性读取、免于脏数据。

但对于Lob类型而言，一致读问题同样存在。Oracle需要一种保留Lob数据镜像的机制，保存一系列old version。目前，Oracle提供了两种维持机制来进行控制：基于时间的版本保留retention和基于空间的版本保留pctversion。

SQL> select table\_name, column\_name, pctversion, retention from user\_lobs;

TABLE\_NAME COLUMN\_NAM PCTVERSION  RETENTION

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

T          CL                           900

Retention是表示采用基于时间版本保留策略。简单的说，就是尽量保证保留一个时间段内的数据lob版本不会清除掉。在数据库版本的兼容性版本设置在9.2.0.0以上，并且undo\_management参数值为true时，lob是默认直接使用retetion设置的。同时，使用的参数值与系统参数undo\_retention相同。

SQL> show parameter comp

NAME                                 TYPE        VALUE

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

compatible           string      10.2.0.1.0

SQL> show parameter undo

NAME               TYPE        VALUE

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

undo\_management             string      AUTO

undo\_retention              integer     900

undo\_tablespace             string      UNDOTBS1

参数pctversion只对于logsegment中使用chunk保存的数据其效果。当pctversion=0的时候，表示旧版本数据是可以被其他事务产生的版本占用。如果设置为100，就表示；旧版本数据永远都不会被覆写使用。

SQL> alter table t move lob(cl) store as t\_segment (pctversion 10 disable storage in row);

Table altered

SQL> select table\_name, column\_name, pctversion, retention, in\_row from user\_lobs;

TABLE\_NAME COLUMN\_NAM PCTVERSION  RETENTION IN\_ROW

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

T          CL        10            NO

        Logging/NoLogging

设置logging属性表示进行Lob数据变化的时候，要将变化的信息记录在redo log里。而相反nologging表示数据不会记录在redo log里面。

注意几个问题：当Lob数据设置为cache的时候，自动就是logging属性。如果设置为nologging，只是lobsegment部分的数据变化不会写redo log，不会影响到其他的in row和column列的数据redo记录工作。

**3、结论**

Lob类型是Oracle的一种较为特殊的数据类型。和其他数据相比具有存储、使用上的特殊性。