

Rollback segment management in Oracle database

霍宇红 (北方交通大学网络管理研究中心 100044)

酒正超 (中国信达资产管理公司)

摘要: 回退段是 Oracle 数据库中的一个重要资源, 本文通过对回退段管理的分析, 论述了 Oracle 回退段的概念、用法和规划, 同时提出了数据库优化的一些方法。

关键词: Oracle 回退段 数据库 事务 数据字典

1 引言

每个 Oracle 数据库都有一个或多个回退段, 回退段是数据库的一部分, 用来记录当前环境中一个事务的活动。它记录事务中修改数据之前的值, 允许在特定的条件下回退修改或取消操作。回退段用来保证读一致性、回退事务和恢复数据库。回退段管理一直是 Oracle 数据库管理的一个难题, 本文通过实例介绍 Oracle 回退段的概念、用法和规划及问题的解决。

2 关于回退段

2.1 事务的概念

Oracle 将一组顺序执行的 SQL 语句作为一个逻辑的工作单元, 称之为事务。事务是对数据库修改或对行加锁的工作单元。一个事务是以例程开始或上一个事务结束时作为开始的, 而事务的结束可以是事

务执行提交, 如显式或隐式提交命令发出, 也可以是事务异常中断时结束, 如: 执行 ROLLBACK 语句、用户请求中断、用户程序异常退出数据库、处理器失败、磁盘失败等。但在缺省情况下, 语句失败 Oracle 只回退该语句, 而不是整个事务。每个事务分配一个唯一标识, 只能与一个回退段相连。

2.2 回退段组成

回退段用于存放修改前的数据, 每一个正在进行的事务都要使用回退段。

数据库中的每个回退段都有一个事务表, 它包含了正在使用该回退段的事务信息。这张表是由与该回退段相连的所有事务和为这些事务执行的每一次修改提供的回退入口 (rollback entries) 的列表组成。一个回退入口包括块信息 (与需要修改数据相联系的文件名和块标号)

和在事务中修改前的数据。回退入口用来执行回退事务和为查询创建读一致性结果。同一事务的回退入口被链接在一起, 以便事务要回退时容易找到。所以, 一个事务只能使用一个回退段来存放它的回退信息, 而一个回退段可以存放多个事务的回退信息。

回退段不能由数据库用户或管理员读写, 它们只能通过 Oracle 来进行操作。它们属于 SYS 用户, 而不论是谁创建的。

2.3 回退段的作用

回退段的主要作用是提供读一致性、回退事务和数据库恢复。

(1) 回退事务: 当事务修改表中数据的时候, 该数据修改前的值会存放在回退段中, 当用户回退事务 (ROLLBACK) 时, Oracle 将会利用回退段中修改前的数据把改变的数据恢复到原来的值, 在操作错误时

事务将做回退处理。

对每个事务而言, 每次新的修改都链接在前一次修改之上。如果事务需要回退, 在链上的这些修改操作按顺序恢复修改前的数据。

(2) 数据库恢复: 回退段的信息也保存在重做日志文件中, 对于在例程失败时仍未提交的活动事务, 回退记录是很重要的。如果发生例程失败, 回退段信息, 包括活动事务的回退入口, 会作为例程或介质恢复的一部分自动恢复。在失败时未提交或回退的事务, Oracle 将在下次打开数据库时完成恢复后才能利用回退段将其回退。

(3) 保证读一致性: 当一个会话正在修改数据时, 其他的会话将看不到该会话未提交的修改。而且, 当一个语句正在执行时, 该语句将看不到从该语句开始执行后的未提交的修改 (语句级读一致性), 当



Oracle 执行 SELECT 语句时, Oracle 依照当前的系统改变号 (SYSTEM CHANGE NUMBER-SCN) 来保证任何前于当前 SCN 的未提交的改变不被该语句处理。例如: 当一个长时间的查询正在执行时, 若其他会话改变了它要查询的某个数据块时, Oracle 将利用回退段的回退入口来构造一个读一致性视图, 即一个在期望时间点上的数据集合, 来保证数据的读一致性。

Oracle 一般提供 SQL 语句级 (SQL STATEMENT LEVEL) 的读一致性, 可以用以下语句来实现事务级的读一致性。

```
SET TRANSACTION READ ONLY;
```

或:

```
SET TRANSACTION SERIALIZABLE;
```

以上两个语句都将在事务开始后提供读一致性。需要注意的是, 使用第二个语句对数据库的并发性和性能将带来影响。

2.4 回退段的种类

Oracle 数据库的回退段有下列几种类型:

(1) SYSTEM 回退段: 当数据库创建后, 将自动创建一个称为 SYSTEM 的初始回退段, 该回退段只用于存放系统表空间中对象修改前的值。SYSTEM 回退段建立在 SYSTEM 表空间中, 并且使用与该表空间相关的缺省存储参数。SYSTEM 回退段不能被删除。例程在得到它需要的其他回退段之前总是已经获得了 SYSTEM 回退段, 如果有多个

回退段, Oracle 尽量把 SYSTEM 回退段给特殊的系统事务来使用, 用户事务使用另外的回退段。

(2) 非 SYSTEM 回退段: 拥有多个表空间的数据库至少应该有一个非 SYSTEM 回退段, 用于存放非系统表空间中对象修改前的值。非 SYSTEM 回退段又分为公共回退段和私有回退段, 私有回退段必须在数据库的参数文件的 ROLLBACK_SEGMENTS 参数中列出, 以便例程启动时能够识别, 自动使其联机 (ONLINE)。公共回退段, 形成可以被需要附加回退段的任何例程请求和使用的回退段池。公共回退段一般在 OPS (Oracle Parallel Server—Oracle 并行服务器) 中出现, 将在例程启动时自动联机。

(3) 延迟回退段: 当表空间变为脱机 (OFFLINE) 时, 事务不能立即被回退, 这时由系统自动创建该回退段。延迟回退段包含不能由表空间使用的回退入口, 所以, 当表空间变回联机 (ONLINE) 时由系统自动删除, 用于存放表空间脱机时产生的回退信息, 它总是创建在 SYSTEM 表空间中, 由系统自动建立。

3 回退段的使用与管理

3.1 回退段使用原理

当用户的事务开始时, Oracle 自动把事务分配到下一个可用的回退段, 在事务第一次发出 DML 或 DDL 语句时分配。只读事务 (事务中仅包含查询) 不会分配回退段, 不论该事务是否用 SET TRANSACTION READ ONLY

语句开始。

可把事务明确地分配到一个指定的回退段中, 允许用户选择一个大小合适的回退段, 以满足事务的特征。在事务执行期间, 与事务相联系的用户进程仅将回退信息写入分配的回退段, 当事务被提交后, 回退信息被释放, 但并不马上被破坏掉。为保证回退数据的视图在足够长的时间内是可用的, 回退段的各个区是按顺序逐个写的。

每个回退段可以处理一个例程中的多个事务。

(1) 回退段的分配: 当事务开始时, Oracle 将根据 LRU 算法为该事务分配回退段, 并将拥有最少事务的回退段分配给该事务。事务可以用以下语句申请指定的回退段:

```
SET TRANSACTION USE ROLLBACK SEGMENT rollback_segment;
```

事务将以顺序、循环的方式使用回退段的区 (EXTENTS), 当前区写满后将移到下一个区。几个事务可以写在回退段的同一个区, 但每个回退段的块只能包含一个事务的信息。

例如, 两个事务使用同一个回退段, 该回退段有四个区, 则其分配情况如下:

- 事务在进行中, 它们正在使用回退段的第三个区;
- 当两个事务产生更多的回退信息, 它们将继续使用第三个区;
- 当第三个区写满后, 事务将写到第四个区, 当事务开始写到一个新的区时, 称为翻转 (WRAP);

· 当第四个区写满时, 如果第一个区是空闲或非活动, 即使用该区的所有事务已提交或回退完成, 事务将循环使用第一个区。

(2) 回退段的扩展: 当前回退段区的所有块用完而事务还需要更多的回退空间时, 回退段的指针将移到下一个区。当最后一个区用完, 指针将移到第一个区的前面。回退段指针移到下一个区的前提是下一个区没有活动的事务, 同时指针不能跨区。当下一个区正在使用时, 事务将为回退段分配一个新的区, 这种分配称为回退段的扩展 (EXTEND)。回退段将一直扩展到该回退段区的个数到达回退段的参数 MAXEXTENTS 的值为止。

(3) 回退段的回收和 OPTIMAL 参数: OPTIMAL 称为优化参数, 它指明回退段空闲时收缩到的位置, 即系统将回收超出 OPTIMAL 所规定的那部分空闲的回退段空间。指明回退段的 OPTIMAL 参数可以减少回退段空间的浪费, 也避免了回退段的空间频繁扩展。

对长时间运行的更新操作需增加 MINEXTENTS 的值, 增加区的大小, 并设置一个高的 OPTIMAL 值, 以避免空间过度分配。对长时间运行的查询, 也应保证更新事务使用的回退段有较大的 OPTIMAL, 避免产生“快照过旧” (SNAPSHOT TOO OLD) 错误。短时间运行的更新事务和查询事务所使用的回退段就必须设置较小的 OPTIMAL 来加速回退段快存。

3.2 回退段管理

3.2.1 创建回退段

回退段只能由 DBA 管理, 所以只有拥有 CREATE ROLLBACK SEGMENT 权限的 DBA 才能创建回退段。

一个例题试图得到的回退段个数由 CEIL (TRANSACTIONS/ TRANSACTION_PER_ROLLBACK_SEGMENT) 的计算得出。

回退段的创建步骤如下:

(1) 在数据库中创建一个专门存放回退段的表空间;

(2) 在第一步创建的表空间中创建新的回退段;

(3) 设置回退段为联机状态。当回退段创建后, 回退段是脱机的, 不能被数据库使用, 为了使回退段被事务利用, 必须将回退段联机。可以用以下命令使回退段联机。

```
ALTER ROLLBACK SEGMENT rollback_segment ONLINE;
```

例如:

```
ALTER ROLLBACK SEGMENT rbs01 ONLINE;
```

在使用中应注意, 应防止包含 ONLINE 回退段的表空间被置为脱机。

(4) 如果是私有回退段, 为了使回退段在数据库启动时自动联机, 要在数据库的参数文件中列出回退段的名称。例如, 在参数文件中加入以下参数:

```
ROLLBACK_SEGMENT=(rbs01,rbs02)
```

3.2.2 修改回退段

(1) 修改存储参数, 可以使

ALTER ROLLBACK SEGMENT 命令修改回退段的存储参数 (包括 OPTIMAL, MAXEXTENTS),

(2) 回收回退段的空间。如果指定了回退段的 OPTIMAL 参数, Oracle 将自动回收回退段到 OPTIMAL 指定的位置, 但数据库管理员也可以手动回收回退段的空间。

(3) 回退段脱机, 为了达到以下两个目的需要将回退段脱机:

- 阻止新的事务使用该回退段。
- 该回退段需要删除。

3.2.3 回退段的状态

回退段有几种状态, 它决定了回退段能否被事务使用。其各种状态如下:

· OFFLINE: 未被任何数据库例程占用。

· ONLINE: 已经被一个例程占用, 可以包含活动事务的数据。

· NEEDS RECOVERY: 包含未提交事务的数据, 该事务由于错误不能回退。

· PARTLY AVAILABLE: 包含悬挂事务的数据, 即没有结果的分布事务。

· INVALID: 已经删除了的回退段, 分配给该回退段的空间将在以后被一个新创建的回退段使用。

PARTLY AVAILABLE 和 NEEDS RECOVERY 回退段的状态很相像, 都包含着未解决事务的数据, 但是这两种状态也有一定的差别。PARTLY AVAILABLE 回退段是处于悬挂状态的分布事务使用的, 没有结果的原因是由于网络故障所致。NEEDS RECOVERY 回退

段是正被一个事务使用的回退段, 但是由于本地介质故障或者是由于事务本身毁坏所致。

Oracle 系统或数据库管理员可以使 PARTLY AVAILABLE 回退段转为联机状态, 而 NEEDS RECOVERY 回退段在它转为联机前必须先脱机。

数据库管理员可以删除一个 NEEDS RECOVERY 回退段, 但是, PARTLY AVAILABLE 回退段不能被删除, 首先必须由 RECO (恢复进程, 用于解决 Oracle 分布式事务处理) 后台进程自动地, 或者由数据库管理员手工地解决悬而未决的分布式事务, 即该回退段由联机转为脱机状态后才能被删除。数据字典 DBA_ROLLBACK_SEGS 可以反映出每个回退段的状态和信息。

3.2.4 删除回退段

当回退段不再需要或要重建以改变 INITIAL, NEXT 或 MINEXTENTS 参数时, 可以将其删除。要删除回退段, 必须使该回退段脱机。

语法:

```
DROP ROLLBACK SEGMENT rollback_segment;
```

例如:

```
DROP ROLLBACK SEGMENT rbs01;
```

3.3 回退段的数量规划

在数据库的实际应用中, 对于联机事务处理系统, 存在大量的小事务处理, 一般建议建立数量多的小回退段, 至少保证每四个事务一个回退段, 每个回退段不要超过十

个事务。而对于以查询为主的数据仓库的应用系统, 一般建议建立数量少而容量大的回退段, 确保每个事务分配一个回退段, 用于大批量的数据处理。回退段有最优大小, 它是在创建或改变时指定, 一旦活动事务数降低, 回退段将自动缩回最优大小。

4 结束语

综上所述, 能否实现对回退段的正确管理对 Oracle 数据库至关重要。为实现 Oracle 数据库适用于多种应用的、高效的、可靠的、安全的数据管理, 包括大负荷的联机事务处理系统以及查询为主的数据仓库应用系统, 同时, 为了实现事务处理的高性能, 就需要对回退段进行合理地设计、使用和规划。 ■

【参考文献】

- 1 [美] Douglas Scherer, William Gaynor Jr, Arlene Valentinsen, Xerxes Cursetjee 著, 京京工作室译, Oracle8i 数据库开发与技巧, 机械工业出版社, 2000.6.
- 2 刘辰, 高月秋等, 数据库系统-管理与应用, 人民邮电出版社, 1999.3.
- 3 李昭原, 数据库技术新进展, 清华大学出版社, 1996.5.
- 4 [美] Eyal Aronoff, Kevin Loney, Noorali Sonawalla 著, 李逸波, 王华驹, 马赛红, 曲宁等译, Oracle8 性能优化和管理手册, 机械工业出版社, 2000.1.