УДК 681.3

ПОВЫШЕНИЕ ЭФФЕКТИВНОСТИ РАСПРЕДЕЛЕННЫХ СИСТЕМ ТРАНЗАКЦИОННОЙ ОБРАБОТКИ ИНФОРМАЦИИ

С.Б.САМЕДОВ
Бакинский Государственный Университет
samir.samedov@gmail.com

В данной работе рассмотрен алгоритм ARIES восстановления узлов в распределенной базе данных, предложен метод увеличения производительности системы при расположении UNDO табличного пространства в ОЗУ. Приведены результаты эксперимента, а также график работы системы при расположении UNDO как в оперативной памяти, так и на жестком диске.

Ключевые слова: распределенная база данных, UNDO табличное пространство, ОЗУ.

На сегодняшний день широкое применение получили распределенные системы (РС) обработки информации - множество территориально удаленных друг от друга узлов, объединенных системой передачи данных и взаимодействующих посредством обмена сообщениями.

В особенности актуальны РС, где каждый узел представляет собой полноценную локальную систему управления базой данных (СУБД), такие РС называют системами распределенной базы данных (РБД). В РБД узлы связаны между собой коммуникационной сетью и взаимодействуют таким образом, что пользователь любого из них может получить доступ к любым данным в сети так, как будто они находятся на его собственном узле [1].

Одним из основных требований к РС, в том числе к РБД, является обеспечение отказоустойчивости, т.е. способность системы не терять свою работоспособность независимо от отказов в системе.

Во время отказов, приводящие к потере данных в узлах РБД, используются механизмы восстановления СУБД. Восстановление системы - это возвращение в определенное прошлое состояние системы, которое было правильным, после некоторого сбоя, которого привело систему к неправильному состоянию. Правильное состояние узла системы - это когда данный узел полностью работоспособный и данные в данном узле

непротиворечивы.

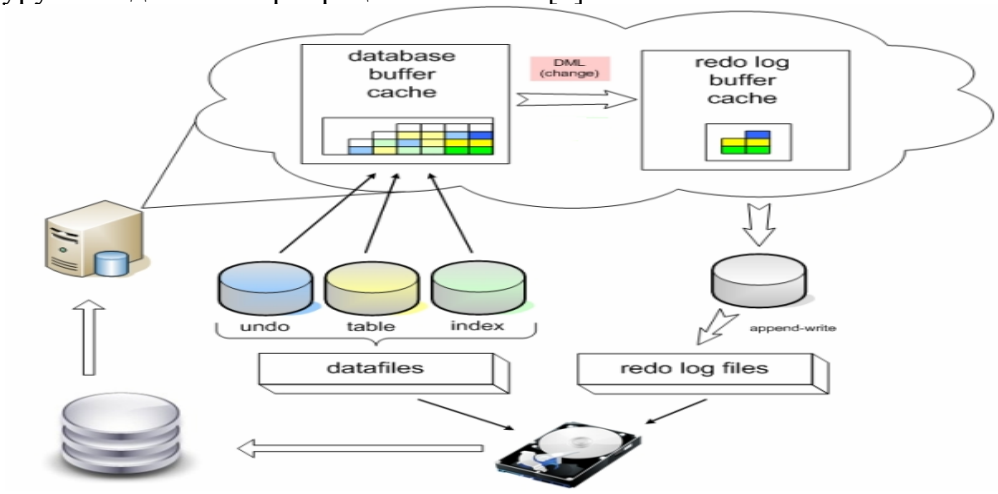
На сегодняшний день используется наиболее популярный алгоритм восстановления и изоляции на основе семантики (ARIES) [1], но разрабатываются различные алгоритмы, которые показывают более лучшие результаты производительности в сравнение с ARIES в различных ситуациях. К примеру, можно привести алгоритм HARBOR [2], использующийся в хранилищах данных.

Перед тем как рассмотрим механизм работы ARIES, приведем основные понятия и современную архитектуру базы данных.

Транзакция — это логическая единица работы; это одна целая операция, состоящая из одной или нескольких операций, в результате чего должны выполняться все операции, иначе отменяются все операции. Транзакция обладает свойством АСИД (атомарность, согласованность, изолированность, долговечность). Транзакция начинается с выполнения операции BEGIN TRANSACTION и заканчивается операцией COMMIT или ROLLBACK. COMMIT – зафиксировать результат транзакции, ROLLBACK – откатить результат транзакции.

Во время работы базы данных имеется точка фиксации SCN (system change number) – это нумерованное состояние, в котором база данных находится в непротиворечивом состоянии. Оператор COMMIT во время своего выполнения как раз и назначает новую точку сохранения.

Для примера современной архитектуры базы данных приведем архитектуру базы данных корпорации ORACLE [3].



Алгоритм ARIES состоит из этапов:

- Формирование журнальных файлов и UNDO сегментов.
- Накат. Начиная с позиции журнала, в котором она находилась во время аварийного останова.
- Откат. Отмена результатов внесения изменений теми транзакциями, фиксация которых не была выполнена.

При работе системы заносятся все SQL операторы, приводящие к изменению данных или к структуре базы данных, в журнальный лог буффер располагающийся в оперативном запоминающемся устройстве (ОЗУ). Содержимое лог буффера сбрасывается на диск в журнальные файлы:

- раз в три секунды;
- при фиксации транзакции;
- при заполнении буфера на треть или, когда в нем оказывается 1 Мбайт данных журнала повторного выполнения.

При этом если SQL приводит к изменению данных, то эти данные первоначально переносятся в undo сегменты. Undo сегменты находятся в табличном пространстве UNDO, которое, в свою очередь, располагается на физическом носителе.

При восстановлении системы определяется наименьший SCN в файлах базы данных и все данные, располагающиеся выше данного SCN, считываются с журнальных файлов и начинают применяться все изменения. При этом если на транзакцию не была послана команда фиксации или отката, то применяется команда отката.

Рассмотрим подробно, что происходит до и после команд фиксаций [4]. Перед выполнением операций COMMIT или ROLLBACK уже были выполнены следующие операции:

- сгенерированы записи сегмента отката;
- сгенерированы измененные блоки данных;
- помещены в буфер журнала повторного выполнения данные повторного выполнения для перечисленных выше изменений;
- в зависимости от размера трех предыдущих фрагментов данных и прошедшего времени, часть их может уже быть сброшена на диск;
- установлены все необходимые блокировки.

При выполнении фиксации осталось сделать следующее:

- Сгенерировать номер системного изменения SCN для транзакции.
- Надо записать на диск все оставшиеся записи из буфера журнала повторного выполнения, а также записать в активные файлы журнала повторного выполнения номер SCN. Именно этот шаг и является фактической фиксацией. Если этот шаг выполнен, транзакция зафиксирована.
- Все блокировки, удерживаемые транзакцией, снимаются, и все сеансы, ожидавшие в очередях снятия этих блокировок, могут продолжить работу.
- Многие измененные транзакцией блоки данных будут повторно обработаны и "очищены" в быстром режиме, если они еще находятся в буферном кэше.

При выполнении же оператора ROLLBACK происходит:

- Отменяются все сделанные изменения. Это достигается путем считы-

вания данных из сегмента отката и выполнения обратной операции. Если строка была вставлена, при откате она будет удалена. Если строка изменена, при откате будут восстановлены прежние значения столбцов. Если строка была удалена, при откате она будет снова вставлена.

• Снимаются все блокировки, удерживаемые сеансом, и все сеансы, ждавшие в очереди снятия этих блокировок, могут продолжить работу.

Таким образом, видно, что операция ROLLBACK очень дорогостоящая, так как при этом приходится считывать данные сегмента отката, которые располагаются в UNDO табличном пространстве на физическом носителе.

На сегодняшний день цена на ОЗУ уже не столь велика, как была раньше. Использование ОЗУ до 64G в серверах средних компаний это стандартный факт. В среднем 8G-16G табличного пространства удовлетворяет потребности базы данных Oracle для средних компаний. В связи с этим для увеличения производительности работы базы данных предложен вариант полного расположения UNDO табличного пространства в ОЗУ.

Для данного эксперемента использовались:

- операционная система Microsoft Windows XP, Service Pack 2
- утилита SoftPerfect RAM Disk Version 3.0.2, позволяющая резервировать место в ОЗУ для использования его как обычного жесткого диска, при этом для работающих приложений это выглядит, как виртуальный диск
- база данных Oracle: Oracle Database 10g Enterprise Edition Release 10.2.0.1.0 - Prod

1. Создаем еще одно табличное пространство [5], расположив его на виртуальном диске V:

```
CREATE UNDO TABLESPACE UNDO_TEST_ON_RAM
DATAFILE  'V:\UNDO_TEST_ON_RAM01.dbf'
SIZE 350M AUTOEXTEND OFF
RETENTION NOGUARANTEE
BLOCKSIZE 8K
FLASHBACK ON;
```

2. Создаем тестовую таблицу:

```
CREATE TABLE SCOTT.UNDO_RAM_TEST
(
 ID          INTEGER,
 uname       VARCHAR2(50),
 username    VARCHAR2(50),
 ucomment    VARCHAR2(200)
)
LOGGING
```

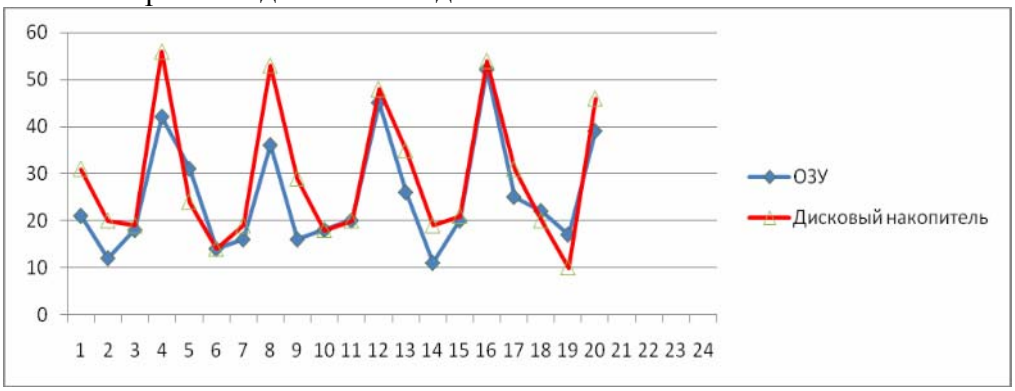
NOCACHE
NOPARALLEL
NOMONITORING;

3. Запускаем скрипт, который заполняет тестовую таблицу UNDO_RAM_TEST данными. Кол-во записей в таблице равно 266430.
4. Запускаем несколько раз скрипт, который три раза изменяет данные и далее запускает команду отката ROLLBACK, замеряем.
5. Перенастраиваем базу данных на использование undo табличного пространства, на ранее созданное UNDO_TEST_ON_RAM.
   SQL>ALTER SYSTEM SET UNDO_TABLESPACE = UNDO_TEST_ON_RAM ;
6. Повторно запускаем несколько раз скрипт, который три раза подряд изменяет данные и далее запускает команду отката ROLLBACK, замеряем.

Таблица результата эксперемента:

Используя UNDO в ОЗУ длительность (в секундах)	*Test №*	*Используя дисковый носитель Длительность (в секундах)*
21	1_1	31
12	1_2	20
18	1_3	19
42	1_4	56
31	2_1	24
14	2_2	14
16	2_3	19
36	2_4	53
16	3_1	29
18	3_2	18
20	3_3	20
45	3_4	48
26	4_1	35
11	4_2	19
20	4_3	21
52	4_4	54
25	5_1	31
22	5_2	20
17	5_3	10
39	5_4	46

Используя данные, получаем график работы. Надо отметить в этом графике, что чем ниже располагается линия, тем более производительней отработал данный метод.



В приведенных графиках, показана разница между выполнением одного и тоже скрипта “Дисковый накопитель”- при undo располагающем на физическом диске, “ОЗУ” – при undo располагающем в ОЗУ.

Таким образом, получается, что возможно достичь выигрыш производительности при расположении UNDO табличного пространства в ОЗУ. Надо отметить, что данный выигрыш в случае с COMMIT не наблюдается, так как при этой операции UNDO не используется.

Но при этом возникает другого рода проблемы, так как расположенная любая информация в ОЗУ, при обычном отключении питания полностью теряется. Поэтому в этом случае происходит полная потеря UNDO табличного пространства и при этом возникают проблемы восстановления.

Всю необходимую информацию для восстановления у Oracle имеется в журнальных файлах. Поэтому в случае потери UNDO табличного пространства, нужно следовать стандартным методам восстановления базы данных при потере UNDO табличного пространства [6].

ЛИТЕРАТУРА

1. Дейт К.Дж. Введение в системы баз данных. М.: Вильяме, 2005, 1328 с.
2. Edmond Lau, Samuel Madden. An Integrated Approach to Recovery and High Availability in an Updatable, Distributed Data Warehouse // Proceedings of the 32nd international conference on Very large database VLDB, September 12-15, Seoul, Korea, 2006, p. 703-714.
3. Дмитрий Вихарев. Интернет ресурс: Oracle RAC. Общее описание. http://habrahabr.ru/blogs/oracle/72122/, 2009, 23 с.
4. Thomas Kyte. Expert Oracle Database Architecture 9i and 10g Programming Techniques and Solutions. United States of America, 2005, 768 p.
5. Sam R.Alapati. Expert Oracle Database 10g Administration. United States of America, 2005, 1304 p.
6. Oracle Literature. Backup and Recovery Basics 10*g* Release 2 (10.2). Part No. B14192-03. United States of America, 2005, 220 p.

İNFORMASİYANIN TRANZAKT İŞLƏNMƏSİNİN PAYLANMIŞ SİSTEMİNİN EFFEKTİVLİYİNİN ARTIRILMASI

S.B.SƏMƏDOV

XÜLASƏ

Bu məqalədə paylanmış verilənlər bazasında ARIES bərpa alqoritminin UNDO tablespace-ni, RAM-da yerləşdirmək metodunu tətbiq etməklə sistemin məhsuldarlığını artırmaq olar. Sınaq nəticələri, eləcə də iş qrafiki UNDO həm RAM-da, həm də diskdə yerləşdiyi zaman göstərilib.

Açar sözlər: paylanmış verilənlər bazası, UNDO tablespace, RAM.

IMPROVING THE EFFICIENCY OF TRANSACTION PROCESSING OF DISTRIBUTED SYSTEMS

S.B.SAMADOV

SUMMARY

The paper examines the ARIES recovery algorithm for nodes in a distributed database, and provides a method for increasing the productivity of the system at the location of UNDO tablespace in RAM. The article elucidates the results of the experiment and provides the graph of UNDO location in RAM and hard disk space.

Key words: distributed database, UNDO tablespace, RAM.

Поступила в редакцию 06.12.2010 г.
Принято к печати 10.03.2011 г.