УДК 004.75
Самедов С.Б.
Бакинский Государственный Университет, Баку, Азербайджан
samir.samedov@gmail.com

ПОВЫШЕНИЕ ЭФФЕКТИВНОСТИ РАСПРЕДЕЛЕННЫХ СИСТЕМ ПРИ ИСПОЛЬЗОВАНИИ ФРАГМЕНТАЦИИ ДАННЫХ

В работе предложен алгоритм фрагментации данных в распределенных базах данных, увеличивающий эффективность системы. Рассмотрены современные технологии корпорации Oracle для создания фрагментации в распределенных базах данных. Проведен эксперимент по влиянию на эффективность распределенной системы при использовании фрагментации.

Ключевые слова: *распределенная система, распределенная база данных, фрагментация, эффективность системы, транзакция.*

1. Введение

С каждым днем увеличивается количество пользователей, работающих с распределенными системами (РС), а также возрастают объемы информации, обрабатываемые в РС. Это требует соответствующего повышения быстродействия РС для обеспечения приемлемого времени реакции на пользовательский запрос. РС – это система, состоящая из различных узлов, взаимодействующих между собой в сети посредством сообщений [1]. Большинство систем работает с данными, которые им приходится хранить в базах данных (БД). Распределенные базы данных (РБД) – это распределенные системы, в качестве узлов которых выступают базы данных. РБД являются одной из актуальных областей на сегодняшний день.

РБД должны удовлетворять двенадцати фундаментальным правилам: локальная независимость, отсутствие зависимости от центрального узла, непрерывное функционирование, независимость от расположения, независимость от фрагментации, независимость от репликации, обработка распределенных запросов, управление распределенными транзакциями, аппаратная независимость, независимость от операционной системы, независимость от сети, независимость от типа БД.

Рассмотрим более подробно фундаментальное правило – независимость от фрагментации. Фрагментация в РС – это разделение информации на несколько порций и распределение этих порций на различные узлы. В этом случае данные могут храниться в том узле, где они чаще всего используются, что позволяет достичь локализации большинства операций и уменьшения сетевого трафика. Благодаря этому получается выигрыш в эффективности всей РБД. Таким образом, фрагментация является необходимостью в системах, где присутствует большой сетевой трафик обмена информацией между узлами, при том что узлы располагаются на удаленных расстояниях между собой и, как следствие этого, между узлами слабая скорость обмена информацией.

Также необходимо учитывать то, что для пользователей системы все должно быть прозрачно, то есть для пользователей при работе с РС, в которой применяется или не применяется фрагментация, никакой разницы в функциональности не должно быть. На сегодняшний день имеется много продуктов, поддерживающих фрагментацию. Одним из них является продукт Oracle 10g корпорации Oracle, являющейся одним из лидеров в разработке программного обеспечения для РС.

При использовании фрагментации встречаются различные виды проблем:

- как определить фрагментируемые объекты;
- где размещать информацию о критериях разделения фрагментируемых объектов;

- как происходит восстановление фрагментируемых объектов;
- как будет проходить транзакция, включающая в себя различные куски информации фрагментируемого объекта, расположенного на различных узлах.

2. Постановка задачи

Рассмотрены современные технологии корпорации Oracle для создания фрагментации в РБД. Главной целью данной работы являются нахождение решения проблем, возникающих при фрагментации объекта в РБД, разработка алгоритма, увеличивающего эффективность прохождения транзакций над фрагментируемыми объектами в РБД, и проведение эксперимента по влиянию на эффективность работы РБД при настроенной фрагментации объектов.

3. Методы решения

- *Как определить фрагментируемые объекты*
  Схематически размещение объекта в РБД выглядит следующим образом:

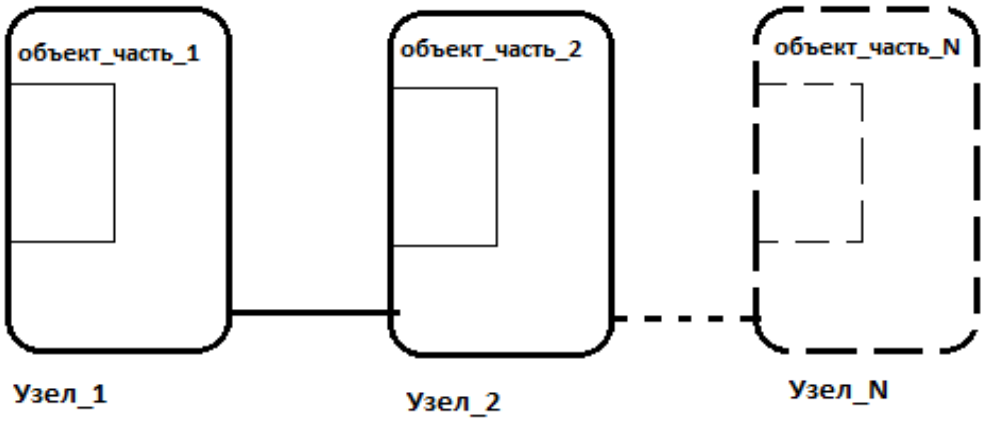

Рис.1. Фрагментация объекта в РБД

К примеру, рассмотрим объект, содержащий в себе информацию о сотрудниках компании. В обычной базе данных нефрагментированный данный объект определяется с помощью структурного языка запросов Oracle SQL [2].

```
CREATE TABLE EMP
(id       integer,
 name     varchar2(25),
 sername  varchar2(25),
 middle_name varchar2(25),
 birthday date,
 departament_id int,
 salary float,
work_city varchar2(25),
 PRIMARY KEY(id)).
```

Так как фрагментация объекта подразумевает деление объекта на куски и расположение их на различных узлах, нужно определиться с критерием, исходя из которого нужно разделить объект на куски. К примеру, вышеуказанную таблицу EMP разделим по узлам, исходя из условия принадлежности сотрудника тому или иному городу, в котором он работает. Таким образом, верхнее определение объекта примет следующий вид:

```
CREATE TABLE EMP
(id       integer,
 name     varchar2(25),
 sername  varchar2(25),
 middle_name varchar2(25),
 birthday date,
 departament_id int,
 salary float,
 work_city varchar2(25),
 PRIMARY KEY(id)
TABLE EMP fragmented:
work_city AT NODE ‘NODE_1’ where work_city =’Baku’
UNION
work_city AT NODE ‘NODE_2’ where work_city =’Sumgait’
……
work_city AT NODE ‘NODE_N’ where work_city =’NNN’;
).
```

- *Где размещать информацию о критериях разделения фрагментируемых объектов*

В зависимости от связей между узлами РС делятся на централизованные и децентрализованные. Централизованные системы – это системы, где связь между узлами осуществляется с помощью одного центрального узла. В децентрализованных системах связь между узлами осуществляется напрямую между собой.

Использование централизованной системы нежелательно, ведь при этом центральный узел становится узким местом системы. Нужно продумать все аспекты: мощность центрального узла, сетевую пропускную способность, так как весь обмен происходит через центральный узел. И, конечно же, система становится уязвимой к отказу, потому что при выходе из строя центрального узла выйдет из строя вся РС.

В случае РБД все узлы также равноправны и независимы, а расположенные на них базы являются равноправными поставщиками данных в общее пространство данных. База данных на каждом из узлов самодостаточна – она включает полный собственный словарь данных и полностью защищена от несанкционированного доступа.

Таким образом, так как в системе должна отсутствовать зависимость от центрального узла, то определять объект нужно на каждом узле РБД.

- *Как происходит восстановление фрагментируемых объектов*

В случае отказа какого-либо узла информация с фрагментируемого объекта остается доступной, за исключением того куска информации, который располагался на отказавшемся узле. При запросе пользователем информации, которая доступна, то есть располагается на работающих узлах, ему возвращается запрашиваемая информация и пользователь продолжает работу. В случае же если хотя бы часть информации недоступна, то пользователю возвращается ошибка о невозможности прочтения информации.

Так как РБД состоит из БД, которые равноправны и независимы, а расположенные в них базы данных являются равноправными системами управления базами данных (СУБД), то при восстановлении фрагментируемых объектов используются стандартные механизмы восстановления данных, применяемые в СУБД [3].

- *Как будет проходить транзакция, включающая в себя различные куски информации фрагментируемого объекта, расположенного на различных узлах*

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.

В РБД транзакции могут обрабатывать как объекты, находящиеся на одном узле, так и объекты, располагающиеся на различных узлах. Транзакции, выполняемые на двух или более узлах, называются распределенными транзакциями. РБД должна управлять распределенной транзакцией на внутреннем уровне. Для пользователя она действует как локальная транзакция. В приложении управление распределенной транзакцией во многом похоже на управление локальной. В конце транзакции приложение также запрашивает ее фиксацию или откат.

РБД должна управлять процессом фиксации, чтобы свести к минимуму риск сбоя сети, в результате которого одни узлы могут фиксировать транзакцию, тогда как другие будут выполнять ее откат. Выход из положения заключается в двухфазном процессе фиксации (фаза подготовки и фаза фиксации), который называется двухфазной фиксацией (2PC) [4]. В этом сценарии за координацию распределенной транзакции на узлах отвечают диспетчеры транзакций, предусмотренные в каждом узле.

Предложен алгоритм – выполнение распределенных транзакций над фрагментированными объектами в РБД.

4. Алгоритм

Шаг 1. В каждом узле РБД создаются таблицы для каждого фрагментируемого объекта системы. Эти таблицы содержат всю необходимую информацию для данных объектов. К примеру, определение фрагментированной таблицы EMP было приведено выше.

В каждом узле имеется свой словарь данных, в нем описаны все объекты узла, своего рода справочник всех объектов. В справочниках узлов предложено создание дополнительной таблицы – таблицы условий фрагментируемых объектов. Данная таблица будет содержать информацию о фрагментируемых объектах, условия фрагментации объектов и соответственно номера узлов, в которых расположена информация, удовлетворяющая данным условиям.

Таблица 1

Таблица условий фрагментируемых объектов

Название колонки	*Предназначение*
name	название объекта
conditions	условия фрагментации
node	номер узла, где расположена информация

К примеру, объект EMP представляется в таблице условий фрагментируемых объектов в следующем виде:

Таблица 2

Определение объекта EMP в таблице условий фрагментируемых объектов

Name	Condition	Node
emp	work_city=’Baku’	Node_1
emp	work_city=’Sumgait’	Node_2
...	...	...
emp	work_city=’XXXX’	Node_N

Шаг 2. В РБД имеется множество диспетчеров транзакции, которые располагаются на узлах РС. На одном из узлов инициализируется транзакция, начинается процесс анализа транзакции. В зависимости от включаемой информации в запросе методом сравнения с колонкой conditions таблицы условий фрагментируемых объектов выясняется подмножество узлов, которые будут участвовать в распределенной транзакции.

Шаг 3. Узел, инициирующий транзакцию, посылает узлам, участвовавшим в распределенной транзакции, команду на выполнение.

К примеру, запрос:

update emp set salary=salary*1.2 where city in (‘Baku’, ‘Sumgait’, ‘Gandja’, ‘Sheki’, Shamaxa’, ‘Salyan’) and departament_id=1.

Каждый узел после получения запроса анализирует полученный запрос. В случае, если запрос включает в себя условие, связанное с фрагментированным объектом, то с помощью таблицы условий фрагментируемых объектов берутся соответствующие условия фрагментации с колонки conditions таблицы условий фрагментируемых объектов для данного узла. Далее условие в запросе, которое включает в себя условие, связанное с фрагментированным объектом, заменяется этим условием в обрабатываемом запросе. Таким образом, вышеуказанный запрос в Node_2 примет следующий вид:

update emp set salary=salary*1.2 where city = ‘Sumgait’ and departament_id=1.

Благодаря этому увеличивается скорость обработки запроса, так как не приходится проверять условия, которые заведомо возвращают FALSE. Покажем это в случае вышеуказанного запроса: количество всех строк таблицы emp на узле_N равняется M. Условие column_name in (expression_1, expression_2, …. , expression_N), если для данного узла сравнение справедливо при последнем expression_N, то узлу_N придется проделать N*M операций сравнения. Но после замены условия с таблицы условий фрагментации объектов узлу придется сделать всего лишь *M* операций сравнения.

После идет непосредственное выполнение данного запроса. В случае положительного, результата узел посылает сообщение инициирующему узлу результат выполнения и при этом фиксация не проделывается Commit. В случае же отрицательного результата узел посылает сообщение инициирующему узлу результат выполнения и проделывается откат транзакции Rollback.

Шаг 4. В случае получения инициирующим узлом транзакции положительных ответов от всех узлов, участвовавших в распределенной транзакции, узел посылает сообщения фиксации Commit этим узлам.

Шаг 5. В случае неполучения инициирующим узлом транзакции положительных ответов от всех узлов, участвовавших в распределенной транзакции, узел посылает сообщения отката Rollback этим узлам.

Благодаря данному алгоритму достигается увеличение скорости прохождения транзакции в РБД над фрагментированными объектами.

5. Экспериментальная часть

Рассмотрим фрагментацию данных, реализуемую корпорацией Oracle, являющейся одним из лидеров в разработке программного обеспечения для РС. Поставлена цель – выяснить, насколько влияет фрагментация данных на эффективность работы РБД при настроенной на ней фрагментации объекта. Рассмотрим РБД, состоящую из трех узлов, на которых установлен Oracle Database 10g Enterprise Edition Release 10.2.0.1.0 [5, 6].

Между узлами установлена беспроводная сеть, скорость которой равняется 11 Мбит/сек.

Таблица 3

Проведение операций вставки при использовании и неиспользовании фрагментации объекта

Эксперимент. №	Кол-во операций	Состояние объекта	Время, потраченное на исполнение (в секундах)
1_1	10,000	Объект фрагментирован и располагается на узлах Node_1, Node_2 и Node_3	40
1_2	10,000	Объект не фрагментирован и располагается весь на втором узле Node_2	55
2_1	20,000	Объект фрагментирован и располагается на узлах Node_1, Node_2 и Node_3	78
2_2	20,000	Объект не фрагментирован и располагается весь на втором узле Node_2	112
3_1	30,000	Объект фрагментирован и располагается на узлах Node_1, Node_2 и Node_3	124
3_2	30,000	Объект не фрагментирован и располагается весь на втором узле Node_2	171
4_1	40,000	Объект фрагментирован и располагается на узлах Node_1, Node_2 и Node_3	163
4_2	40,000	Объект не фрагментирован и располагается весь на втором узле Node_2	221
5_1	50,000	Объект фрагментирован и располагается на узлах Node_1, Node_2 и Node_3	211
5_2	50,000	Объект не фрагментирован и располагается весь на втором узле Node_2	277

На рис. 2 построен график работы, основывающийся на данных из таблицы 3.

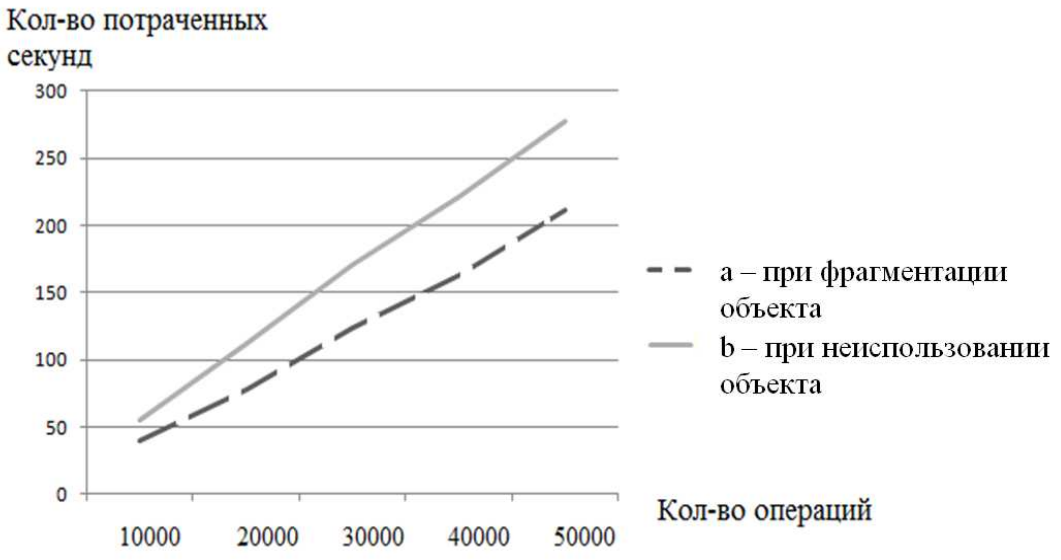

Рис. 2. График прохождения операций вставки на узле Node_1 при использовании и неиспользовании фрагментации объекта

Получается, что при использовании фрагментации объекта достигается увеличение эффективности работы РБД.

6. Заключение

Таким образом, в данной работе предложен алгоритм, обеспечивающий увеличение эффективности при выполнении распределенных транзакций в РБД над фрагментированными объектами. Включив возможность фрагментации объектов в РБД, со слабой связью между узлами, получаем повышение эффективности в РБД за счет локализации операций при работе с фрагментируемыми данными.

Литература

1. Дейт К.Дж. Введение в системы баз данных. М.: Вильяме, 2005, 1328 с.
2. Oracle Literature. Oracle Database SQL Language Reference 11*g* Release 1 (11.1). Part No. B28286-06. United States of America, 2010, 1446 p.
3. Oracle Literature. Backup and Recovery User's Guide 11*g* Release 1 (11.1). Part No. B28270-03. United States of America, 2008, 598 p.
4. Philip A. Bernstein, Vassos Hadzilacos, Nathan Goodman. Concurrency control and recovery in database systems. Menlo Park, California, 1987, 370 p.
5. Thomas Kyte. Expert Oracle Database Architecture 9i and 10g Programming Techniques and Solutions. United States of America, 2005, 768 p.
6. Sam R.Alapati. Expert Oracle Database 10g Administration. United States of America, 2005, 1304 p.

UOT 004.75

Səmədov Samir B.
Bakı Dövlət Universiteti, Bakı, Azərbaycan
samir.samedov@gmail.com

Verilənlərin fraqmentasiyasından istifadə edilməklə paylanmış sistemlərin səmərəliliyinin artırılması

Məqalədə paylanmış verilənlər bazasının səmərəliliyinin artırılması üçün verilənlərin fraqmentasiyası alqoritmi təklif olunub. Paylanmış verilənlər bazasında verinlərin fraqmentasiyası üçün müasir Oracle texnologiyaları təhlil edilib. Verilənlərin fraqmentasiyasının paylanmış sistemlərin səmərəliliyinə təsiri haqqında sınaq keçirilib.

Açar sözlər: *paylanmış sistem, paylanmış verilənlər bazası, fraqmentasiya, sistemin səmərəliliyi, tranzaksiya.*

Samir B.Samadov
Baku State University, Baku, Azerbaijan
samir.samedov@gmail.com

Distributed system efficiency increase by data fragmentation.

In this paper an algorithm for data fragmentation is proposed to increase the efficiency of distributed databases. The modern technology of Oracle is analyzed to create a data fragmentation in the distributed databases. An experiment is implemented in order to influence the efficiency of the distributed database by using data fragmentation.

Keywords: *distributed system, distributed database, fragmentation, system efficiency, transaction.*