УДК 004.65

ОСОБЕННОСТИ ОПЕРАЦИЙ ОБЪЕДИНЕНИЯ В GOOGLE CLOUD SPANNER

Лазурик В.М., Лофицкий Э.А.
Харьковский национальный университет имени В.Н. Каразина

Работа посвящена рассмотрению возможностей использования NewSQL системы управления базами данных GOOGLE CLOUD SPANNER. Определены достоинства и недостатки системы. Особое внимание уделено исследованию операций объединения с использованием Interleaved таблиц и индексов. Проведено несколько нагрузочных тестов с использованием кластера с 5-ю узлами SPANNER в регионе europe-west1.
Ключевые слова: Google Cloud Spanner, JOIN операции, тестирование, кластер.

Постановка проблемы. В последнее время в информационных технологиях часто используется термин Big Data (большие данные). Big data – обозначение структурированных и неструктурированных данных огромных объёмов, и различных инструментов, подходов и методов их обработки для конкретных задач и целей [1]. Термин Big Data ввёл редактор журнала Nature Клиффорд Линч ещё в 2008 году в спецвыпуске, посвящённом взрывному росту мировых объёмов информации. Для больших данных выделяют традиционные определяющие характеристики, выработанные Meta Group в 2001 году, которые называются «Три V»: Volume – величина физического объёма, Velocity – скорость прироста и необходимости быстрой обработки данных для получения результатов, Variety – возможность одновременно обрабатывать различные типы данных. В качестве базового принципа обработки больших данных наиболее часто указывают горизонтальную масштабируемость, обеспечивающую обработ-

ку данных, распределённых на сотни и тысячи вычислительных узлов, без деградации производительности. Большинство аналитиков для этих целей рассматривают технологии NoSQL, MapReduce, Hadoop. Но в [2] для обработки больших данных включают также технологии Business Intelligence и реляционные системы управления базами данных с поддержкой языка SQL.

Работа посвящена рассмотрению возможностей NewSQL хранилища данных Google Cloud Spanner. Проанализированы несомненные достоинства, отмечены недостатки. Особое внимание уделено анализу производительности запросов с объединением (JOIN).

Анализ последних исследований и публикаций. **Bigtable** – высокопроизводительная база данных, реализующая колоночную схему хранения и построенная на основе GFS и некоторых других внутренних продуктах Google. Google File System (GFS) – распределенная файловая система, созданная компанией Google в 2000 году для своих внутренних потребностей. GFS – кластерная система, оптимизированная для центрального хранилища данных Google и нужд поискового механизма, обладающая повышенной защитой от сбоев. Система предназначена для взаимодействия между вычислительными системами, а не между пользователем и вычислительной системой. Как и GFS, Bigtable – проприетарная система, внутреннее устройство которой, тем не менее, было подробно описано инженерами Google в research paper [3].

Bigtable – хорошо масштабирующееся хранилище данных, рассчитанное на хранение петабайтов информации и работающее на commodity-серверах. Bigtable работает на production-серверах с 2005 года. В разное время в BigTable хранили данные web-индексов, сервисов Google Analytics, Google Earth, Google Finance [3]. До появления Spanner и других более узкоспециализированных внутренних хранилищ данных в Google, Bigtable «работал» с клиентами с крайне разнообразными требованиями как по объему данных (от URL до снимков спутника), так и по задержке в ответе (от пакетных режимов до режимов близких к реальному времени). В свою очередь клиенты определяли схему хранимых структурированных/полуструктурированных данных. Bigtable представляет собой разреженную распределенную многомерную отсортированную карту, которая проиндексирована по ключу строки (row key), ключу столбца (column key) и временной метке (timestamp). На основе лексикографического анализа, поддерживаемого Bigtable, row key объединяются в диапазоны строк, которые в терминах Bigtable называются Tablet. Tablet может динамически партицироваться и является ключевым элементом балансировки нагрузки. Column key, в свою очередь, объединяются в семейства столбцов (column family). Последние представляют собой простейшую единицу доступа к данным. Timestamp позволяет добавить в Bigtable версионность данных, а также реализовать append-only модель хранения данных.

Bigtable обладает рядом ограничений: не поддерживаются транзакции на уровне нескольких строк (только на уровне одной строки); определение способа хранения данных – RAM или диск – ответственность клиента. Bigtable не пытается определить наилучший способ хранения динамически.

Bigtable, без сомнения, была самой большим по объему хранимых данных NoSQL базой в мире. Более того, концепции, лежащие в основе Bigtable и описанные в [3], легли в основу многих популярных сегодня NoSQL-решений, в т.ч. HBase – NoSQL БД, входящая в экосистему Hadoop.

MapReduce – это модель распределенной обработки данных, предложенная компанией Google для обработки больших объёмов данных на компьютерных кластерах.

Как известно парадигму MapReduce предложила компания Google в 2004 году в своей статье MapReduce: Simplified Data Processing on Large Clusters. Поскольку предложенная статья содержала описание парадигмы, но реализация отсутствовала – несколько программистов из Yahoo предложили свою реализацию в рамках работ над web-краулером nutch. Так появился Hadoop.

Hadoop – первоначально инструмент для хранения данных и запуска MapReduce-задач. Сейчас Hadoop представляет собой большой стек технологий, так или иначе связанных с обработкой больших данных (не только при помощи MapReduce) [4]. Основными (core) компонентами Hadoop являются:

- Hadoop Distributed File System (HDFS) – распределённая файловая система, позволяющая хранить информацию практически неограниченного объёма.
- Hadoop YARN – фреймворк для управления ресурсами кластера и менеджмента задач, в том числе включает фреймворк MapReduce.
- Hbase – колоночная база данных, реализующая парадигму Bigtable.

Hbase представляет из себя попытку объединения удобства пакетной обработки и удобства обновления и произвольного доступа. Это распределенная, колоночно-ориентированная, мультиверсионная база типа «ключ-значение». Данные организованы в таблицы, проиндексированные первичным ключом, который в Hbase называется RowKey. Для каждого RowKey ключа может храниться неограниченный набор атрибутов (или колонок). Колонки организованны в группы колонок, называемые Column Family. Как правило в одну Column Family объединяют колонки, для которых одинаковы паттерн использования и хранения. Для каждого аттрибута может храниться несколько различных версий. Разные версии имеют разный timestamp. Записи физически хранятся в отсортированном по RowKey порядке. При этом данные, соответствующие разным Column Family хранятся отдельно, что позволяет при необходимости читать данные только из нужного семейства колонок. Модель данных Hbase можно представить как соответствие ключ значение: <table, RowKey, Column Family, Column, timestamp> -> Value:

Hbase поддерживает 4 основные операции: Put – добавить новую запись; Get– получить данные по определенному RowKey; Scan– читать записи по очереди; Delete – пометить определенную версию к удалению. Hbase сложна в администрировании и использовании.

Spanner – географически распределенная высокомасштабируемая мультиверсионная база данных с поддержкой распределенных транзакций [5]. Хранилище было разработана инженерами Google для внутренних сервисов корпорации. Research paper [6], описывающий базовые принципы и архитектуру Spanner, был представлен на научной конференции 10th USENIX Symposium on Operating Systems Design and Implementation в 2012 году. Spanner является эволюционным развитием NoSQL-предшественника – Google Bigtable. Сам же с Spanner относят к семейству NewSQL решений. В research paper [6] заявляется, что дизайн Spanner позволяет системе масштабироваться на миллионы вычислительных узлов через сотни дата-центров и работать с триллионами строк данных.

Данные в Spanner хранятся в полуреляционных таблицах, имеющих схему. Все данные имеют версию – временную метку (timestamp) подтверждения записи этих данных (commit). Spanner имеет SQL подобный язык запросов, возможность конфигурировать количество реплик и политики Garbage Collector, ответственного за удаление записей со «старыми» временными метками. Кроме наличия NoSQL возможностей, Spanner обладает и рядом сложно реализуемых в распределенных системах свойств [5]:

· поддержка распределенных транзакций;

· глобальная согласованность операций чтения между географически распределенными ДЦ, т.о. данные, которые возвращают операции чтения из разных ДЦ, всегда согласованны и непротиворечивы.

· неблокирующее чтение данных «из прошлого» (in past);

· отсутствие блокировок для read-only транзакций;

· атомарное изменение схемы таблиц данных;

· синхронная репликация;

· автоматическая обработка отказов как вычислительных узлов, так и ДЦ;

· автоматическая миграция данных как между вычислительными узлами, так и между ДЦ.

Цель статьи: описать возможности релиационной базы данных при работе с BigData на примере Google Cloud Spanner.

Объекты для хранения данных в Cloud Spanner

Для работы с данными в Спаннере существует два типа объектов: ключ таблицы и индексы. Назначение данных типов схоже с аналогами в традиционных реляционных базах. Пример определения таблицы:

CREATE TABLE Persons (
PersonId INT64 NOT NULL;
FirstName STRING (32);
LastName STRING (32), PRIMARY KEY (PersonId)).

Распределение данных в Cloud Spanner по кластеру основано на PRIMARY KEY, поэтому при выборе типа и генератора первичного ключа нужно выбирать правильный подход, дабы бы избежать концентрации схожих данных на одном узле кластера. Временные метки и монотонно увеличивающиеся числа – это неправильный выбор, т.к. нет случайного распределения данных по кластеру, и сохраняет приходящие новые данные на одном узле. В то время как случайная информация (например, имя пользователя) или UUID (universally unique identifier) являются более правильным выбором, т.к. увеличивают дисперсию разброса информации по кластеру.

Внешний ключ (foreign key) – понятие теории реляционных баз данных, относящееся к ограничениям целостности базы данных. Неформально выражаясь, внешний ключ представляет собой подмножество атрибутов некоторой переменной отношения R2, значения которых должны совпадать со значениями некоторого потенциального ключа некоторой переменной отношения R1. Spanner не предоставляет возможности определить отношение FOREIGN KEY между таблицами. Тем не менее, у Spanner есть концепция, которая несколько похожа на FOREIGN KEY – Interleaved таблицы

Interleaved таблицы в Cloud Spanner

Если таблица определена следующим образом:

CREATE TABLE Tasks (
PersonId INT64 NOT NULL;
TasksId INT64 NOT NULL;
Description STRING (MAX) NOT NULL;
IsCompleted BOOL), PRIMARY KEY (PersonId, TasksId) INTERLEAVE IN Persons.

В результате строки таблицы Tasks будут физически совмещены/чередоваться с соответствующими строками родительской таблицы Persons. Для правильной работы совмещенных таблиц, в «дочерней» таблице должен быть PRIMARY KEY, который сначала содержит первичные ключи родителя, при этом имена столбцов должны совпадать (рис. 1).

Person(1)	Tom	Miller	
Task(1,1)			Clean up flat
Task(1,2)			Do homework
Person(2)	Kyle	Snee	
Task(2,1)			Prepare for trip
Task(2,2)			Buy toothpaste
Task(2,3)			Call to sister

Рис. 1. Физическое расположение строк Person и дочерней таблицы Task в Interleaved таблицах

Вставка в дочернюю таблицу возможна только в том случае, если у родителя есть соответствующая строка (это ближайшая конструкция к ограничению FOREIGN KEY, которое поддерживает Spanner). Строки из родительской таблицы могут быть удалены только в том случае, если в дочерней таблице нет соответствующих строк. ON DELETE CASCADE – опция, позволяющая удалять любые связанные данные из дочерней таблицы в автоматическом режиме. Отношения родителей и дочерних таблиц могут идти на произвольную глубину, но при этом структура с глубиной в 1000 и более отношений, может значительно уменьшить производительность и читабельность такой структуры вызывает сомнения. Не рекомендуется “раздувать” размер данных, которые «принадлежат» одному родительскому PRIMARY KEY более 2 GiB.

Хотя между родительскими и дочерними строками существует логическая взаимосвязь, основной целью чередования является ускорение некоторых запросов, особенно JOIN операций. Это напрямую изменяет способ вывода данных на диск

и гарантирует, что весь JOIN запрос может быть выполнен без доступа к каждому узлу кластера. В этом смысле эта модель на самом деле намного ближе к концепции Cassandra, чем к ограниченным возможностям FOREIGN KEY реляционной базы данных, т.к. FOREIGN KEY не позволяет гарантировать выполнение запроса на одном узле).

Нагрузочные тесты в Google Cloud Spanner на производительность объединений

Было проведено несколько нагрузочных тестов в Google Cloud Spanner на производительность объединений (JOIN). Данные тесты ограничены пропускной способностью тестовой инфраструктуры и не являются пределами того, что может достичь Spanner. Кроме того, дальнейшая настройка клиента (объединение пулов) может улучшить наблюдаемую производительность. Для тестирования был использован кластер с 5 узлами Spanner в регионе europe-west1 и 2 вычислительных экземпляра n1-highcpu-4 в разных зонах того же региона для запуска тестового кода.

Чтение без объединения. В этом случае отдельные записи Persons считывались по PRIMARY KEY, чтобы установить базовую производительность. Наблюдалось около 11 000 транзакций в секунду (условно ограниченные узлами тестирования) со средней задержкой около 30 мс.

Чтение с помощью объединений с использованием Interleaved таблицы. В этом случае считались все Tasks (не отдельные элементы, а все данные заказа), принадлежащие одному человеку (таблица Persons) по идентификатору PersonId. Был использован следующий запрос, адаптированный под Interleaved таблицы:

SELECT p.PersonId, p.FirstName, p.LastName, t.TasksId, t.Description, t.IsCompleted

FROM Persons AS p

INNER JOIN Tasks AS t ON p.PersonId = t.PersonId

WHERE p.PersonId = @PersonId.

В синтаксисе Spanner @PersonId — это переменная запроса. Наблюдаемые цифры очень близки к тому, что было получено для простого сценария чтения: 11 000 транзакций в секунду, средняя задержка около 30 мс.

Чтение с помощью объединений с использованием таблицы без Interleaved и без индексов. Был повторен тот же запрос соединения, что и раньше, но с использованием non-Interleaved таблиц.

CREATE TABLE Tasks (PersonId INT64 NOT NULL, TasksId INT64 NOT NULL,

Description String(MAX) NOT NULL, IsCompleted Bool), PRIMARY KEY (PersonId, TasksId).

Снижение производительности было довольно значительным: 12 транзакций в секунду, средняя латентность около 17 секунд. Производительность соединения зависит от объема данных в разных таблицах.

Чтение с помощью объединений с использованием таблицы без Interleaved, с использованием явного индекса в столбце объединения. Использовалась таблица non-Interleaved с помещенным индексом в столбец, к которому делается JOIN. Эти модификации так же потребовали изменения самого запроса:

SELECT p.PersonId, p.FirstName, p.LastName, t.TasksId, t.Description, t.IsCompleted

FROM Persons AS p

INNER JOIN Tasks @ {FORCE_INDEX=order_by_Persons} AS t ON p.PersonId = t.PersonId

WHERE p.PersonId = @PersonId

Tasks @ {FORCE_INDEX=order_by_Persons} – выражение, заставляющее делать выборку с использованием индексов. Если опустить использование явного FORCE_INDEX, Spanner игнорирует индекс в соединении, и результаты будут аналогичны тем, которые были получены без использования индексов. Директива FORCE_INDEX приводит к значительному повышению производительности. Хотя индексирование все же несколько менее эффективно, чем использование INTERLEAVE, индекс может обеспечить очень хорошие уровни производительности: 9 000 транзакций в секунду, средняя латентность около 35.

Результаты работы и выводы. Рассмотрена проблема Больших данных. Исследованы программные средства, позволяющие работать с большими данными. Проанализированы возможности использования NewSQL системы управления базами данных GOOGLE CLOUD SPANNER. Уделено внимание операциям объединения с использованием Interleaved таблиц и индексов. Проведено несколько нагрузочных тестов с использованием кластера с 5-ю узлами SPANNER в регионеі europe-west1.

Cloud Spanner является новой генерацией облачных СУБД, объединяющей в себе твердые ACID гарантии и возможность легкого горизонтального масштабирования. Cloud Spanner дает возможность пользоваться распределенной СУБД, сохраняя при этом доступность данных и сводя согласованность данных до максимума. Тем не менее, это облачное решение имеет и минусы. Spanner не совместим с какой-либо другой СУБД, он имеет полностью собственный API, который нужно изучить и под который адаптировать код. Кроме того, нет функционирующего драйвера для работы с API базы данных, такой как JDBC. Отсутствие инструментария (например, «консоли» SQL для изучения данных и экспериментов) делает повседневное использование неприятным. Кроме того, если было принято решение использовать Spanner, происходит полное “запирание” проекта в Google Cloud – не остается возможности запустить эту базу данных нигде, кроме инфраструктуры Google, и код перестанет быть переносимым.

Список литературы:

1. Большие данные: Материал из Википедии – свободной энциклопедии [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/.
2. James Manyika et al. Big data: The next frontier for innovation, competition, and productivity. McKinsey Global Institute, June, 2011.
3. Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A.Wallach, et al. Bigtable: A Distributed Storage System for Structured Data. Proceedings of OSDI, 2006.

4. Александр Петров. Big Data от А до Я. Hadoop, Hbase [Электронный ресурс]. – Режим доступа: https://habr.com/company/dca/blog/267361/.
5. Dmitry Petukhov. Spanner. NewSQL хранилище от Google [Электронный ресурс]. – Режим доступа: https://www.codeinstinct.pro/2013/12/spanner.html.
6. James C. Corbett, Jeffrey Dean, Michael Epstein, Andrew Fikes, Christopher Frost, J.J. Furman, et al. Spanner: Google's Globally-Distributed Database. Proceedings of OSDI, 2012.
7.

Лазурик В.М., Лофіцкий Е.О.
Харківський національний університет імені В.Н. Каразіна

ОСОБЛИВОСТІ ОПЕРАЦІЙ ОБ'ЄДНАННЯ В GOOGLE CLOUD SPANNER

Анотація
Робота присвячена розгляду можливостей застосування NewSQL системи керування базами даних GOOGLE CLOUD SPANNER. Визначені досягнення та недоліки системи. Особливу увагу приділено дослідженню операцій об'єднання з застосуванням Interleaved таблиць та індексів. Проведено декілька тестів з навантаженням з використанням кластеру с 5 вузлами SPANNER в регіоні europe-west1.
Ключові слова: Google Cloud Spanner, JOIN операції, тестування, кластер.

Lazurik V.M., Lofitsky E.A.
V.N. Karazin Kharkiv National University

FEATURES OF GOOGLE CLOUD SPANNER INTEGRATION OPERATIONS

Summary
The work is devoted to consideration of possibilities of application of NewSQL of system of management of databases Google Cloud Spanner. Defined achievements and disadvantages of the system. Particular attention is paid to the study of union operations using Interleaved tables and indexes. Several load tests have been performed using a cluster with 5 spanner nodes in the europe-west1 region.
Keywords: Google Cloud Spanner, JOIN operations, testing, cluster.