СОВРЕМЕННЫЕ ПОДХОДЫ К РАЗРАБОТКЕ ГЕТЕРОГЕННЫХ ХРАНИЛИЩ ДАННЫХ

Саклаков В. М.,
Научный руководитель – к. т. н. Иванов М. А.
Томский политехнический университет, Институт кибернетики
romanov_ky@mail.ru

Долгое время количество данных в системах управления базами данных (СУБД) не превышало некоторой критической отметки. В тот период времени SQL (structured query language) являлся наиболее значимым инструментом формирования и обработки данных в реляционных базах данных (РБД). Данный язык, как и подход в целом, развивался с 1970-х годов и на данный момент обладает рядом значимых свойств:

1. Простота переноса текстов SQL-запросов в различные СУБД позволяют не зависеть от какой-либо из них.
2. SQL достаточно сложный язык и требует квалификации программиста, а не пользователя.
3. Существование стандартов не помешало разработчикам создавать специфические диалекты для различных СУБД.
4. SQL не полностью соответствует реляционной модели данных.

Однако в последние годы количество данных, которые необходимо создавать, хранить и обрабатывать достигло значительных объемов. При этом характерной особенностью становится не только рост объема данных, но и увеличение их неоднородности, а также скорости поступления в хранилища. Все более актуально становится проблема масштабируемости и доступности данных. В связи с этим происходит постепенный отход от традиционных моделей реляционных СУБД, в рамках которых возможности решения выше обозначенных задач были ограничены [1]. Развивается методология Больших Данных (Big Data) (см. рисунок 1). По сути происходит размежевание последней и традиционной аналитики (бизнес-, научной и др.) в соответствии с целями, задачами и возможностями [2]. При этом в их основе лежит потребность в наблюдении за реальными физическими, а в след за ними и экономическими процессами, а также интерпретация полученной информации.

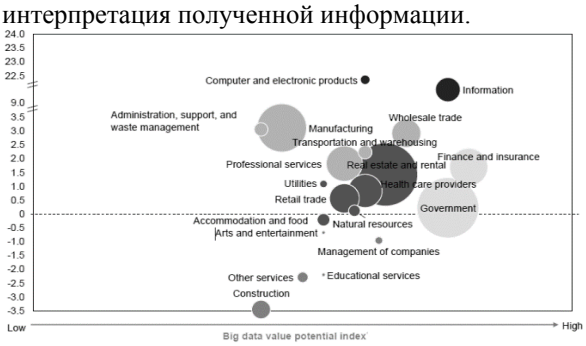

Рис. 1 – Потенциал роста отраслей экономики США, основанный на Больших Данных, % [3]

Целью настоящей работы является анализ существующих подходов к разработке гетерогенных хранилищ данных. Выделим и опишем основные категории решений к хранению и обработке Больших Данных:

1. **NoSQL** (Not only SQL). Рассмотрим его основные преимущества: (а) в отличие от РБД нет строгих требований к согласованности данных, что обеспечивает отсутствие излишнего усложнения. (б) высокая, в сравнении с РБД, пропускная способность данных. (в) неограниченное горизонтальное масштабирование. (г) согласно теореме CAP NoSQL отдает приоритет доступности данных и устойчивости к разделению, но не их согласованности [1].

NoSQL предназначен для работы с web-приложениями. Несмотря на указанное в пункте (г) свойство он может делать выбор в пользу свойств согласованности или доступности при обработке отдельных запросов [4]. Это повышает гибкость системы при работе с разнородной информацией.

Дадим классификацию NoSQL хранилищ:

- *Ключ-значение*. Примерами служат Redis, Riak, SQL Data Services и др. Основной упор в данном случае делается на производительность. Информация о структуре значений не сохраняется.
- *Документированные*. Например, MongoBD, CouchBD, Couchbase и др. Абстракция «документ» формируется путем объединения множества пар ключ-значение. Данный тип хранилищ хорошо подходит для работы с глубоко-вложенной, сложной информацией. Может взаимодействовать с другими приложениями посредством JavaScript.
- *Колоночные*. Например, BigTable, Cassandra и др. Данный тип хранилищ имеет схожесть с реляционными СУБД. В основе модели лежит колонка. Она объединяется в семейства с определенным набором свойств.
- *Хранилища типа граф*. Примерами могут служить OrientDB, Neo4J. Данные хранилища более всего подходят для работы со сложно-связанными данными, представленными в виде графов с соответствующими вершинами, ребрами и свойствами. С их помощью можно осуществлять моделирование данных и классификацию информации.

2. **MapReduce**. Вычислительная модель от Google, на основе которого были созданы несколько основополагающих проектов для отрасли Больших данных, а именно:

- *Hadoop*. Ключевым отличием данного проекта является последовательная обновление

всего набора данных при их обработке, в отличие от произвольного обновления в моделях, основанных на NoSQL. Так последние решают задачу увеличения производительности путем горизонтального масштабирования. То есть, чтобы прочесть 1 Тб информации при средней производительности одного жесткого диска ~100Мб/сек максимально быстро – необходимо считывать ее с большого количества дисков. Таким образом, принцип последовательной обновления при обработке данных позволяет добиться значимого прироста производительности. Именно он лежит в основе архитектуры Hadoop.

Для хранения и организации данных используется Hadoop Distributed File System (HDFS). Она предназначена для хранения больших неизменяемых файлов на кластере недорогих машин. Надежность гарантируется системой репликации данных. Структуры файловой системы проходит процедуру оптимизации для работы с большими файлами. Основная задача HDFS: при поточном доступе к данным обеспечить максимальную производительность.

• *Disco*. Особенностью данного подхода является высокая заявленная отказоустойчивость оборудования, использующего данное ПО.

Данный проект, как и Hadoop осуществляет распределенную обработку больших пакетов данных. При этом большую роль играют задержки, т. е. отказ от работы в режиме реального времени. Данный фактор делает невозможным реализацию подходов этих проектов в целом спектре задач, требующим работы в режиме реального времени.

Обобщая вышесказанное стоит заметить, что технология MapReduce является более эффективным по сравнению с традиционными реляционными СУБД, однако у неё есть недостатки, в числе которых отсутствие поддержки блокировки записей. Поэтому в ряде случаев их невозможно применять. Например, в финансовых системах [5] или системах учета движения товарно-материальных ценностей.

3. **NewSQL**. Основным отличием данного подхода является совмещение преимуществ NoSQL – производительность и масштабируемость – и транзакционных требований классических баз данных (ACID). Потребность в таких системах возникла в следствие необходимости создания приложений для бизнеса, контроля финансовых потоков и т. д. Основным требованием к ним была согласованность данных, которую не могла обеспечить NoSQL-система [6]. В основе высокой производительности лежат возможности горизонтального масштабирования и оптимизация всех узлов. Распределим NewSQL по категориям:

• *Системы с новыми архитектурами*. Например, VoltDB – хорошо справляется с большим количеством «узких» и коротких транзакций; Clustrix – сервис или готовые аппаратные решения, совместимые с MySQL.

• *Подсистема хранения данных для MySQL*. Данная СУБД подключает подсистемы хранения данных. Они производят операции над данными в таблицах не задействуя ядро. Примерами могут служить NDB, TokuDB.

• *Решения для масштабирования на основе традиционных систем*. Основана на использовании промежуточного программного обеспечения для «шардинга» и репликации между серверами. Уровень производительности ниже, чем у проектов, основанных на новой архитектуре. При этом сохраняется совместимость с существующими инструментами и приложениями. Пример: DBShards, Data Traffic Manager.

Заключение

В настоящей работе был произведен анализ архитектурных особенностей различных подходов к обработке и хранению больших данных. Каждый из них является самостоятельным инструментом с ограниченным кругом возможностей и не претендует на абсолютную универсальность.

Стоит помнить и о требованиях к обработке больших данных. Одним из главных является сокращение времени на их обработку и анализ конечным пользователем вследствие их очень быстрого обновления. Поэтому Big Data являются нишевым инструментом для организаций, которые могут их обрабатывать и интерпретировать.

В следующей работе будет произведен сравнительный анализ систем хранения и обработки больших данных. За основу будут взяты три основных параметра – производительность, масштабируемость и эластичность [7].

Литература

1. Клеменков П.А., Кузнецов С.Д. Большие данные: современные подходы к хранению и обработке. // Труды института системного программирования РАН. Том: 23. 2012. с. 143-158

2. Тиндал С. Большие данные: все, что вам необходимо знать // PCWeek №25 (810) 2 окт. 2012

3. Григорьев Ю. А. Анализ свойств баз данных NoSQL // 2013. Информатика и системы управления. №2 (36) с. 3-13

4. Big data: The next frontier for innovation, competition, and productivity. McKinsey Global Institute. June 2011

5. Григорьев Ю. А., Плутенко А. Д. Оценка времени соединения таблиц в базе данных NoSQL по технологии MapReduce. // Информатика и системы управления. №1 (39). 2014. с. 3-14

6. Кузнецов С.Д., Посконин А.В. Распределенные горизонтально масштабируемые решения для управления данными // Труды института системного программирования РАН. Том: 24. 2013. с. 327-358

7. Чернов А. В., Дицков А. В. Сравнительный анализ облачных хранилищ не реляционного типа для больших объемов данных. // Труды Ростовского государственного университета путей сообщения. №4. 2014. с. 121-130