УДК 004.9

ИНСТРУМЕНТ HADOOP И ОПТИМИЗАЦИЯ ХРАНИЛИЩА ДАННЫХ

Картанова А.Дж., Абдрасакова А.Б., Иманбеков Т.И.
Кыргызский государственный университет строительства,
транспорта и архитектуры им. Н.Исанова

Хранилища данных превратились в крупнейшие базы данных организации. Хранилища данных растут в зависимости от количества пользователей, объема хранимых данных, количества источников данных и сложности отчетов и аналитических требований. С ростом хранилищ данных возникают проблемы с производительностью и параллелизмом, с затратами на хранение и обработку данных, которые могут стать неприемлемыми. В таких случаях организациям необходимо оптимизировать свои хранилища данных, а выбор методов и технологий оптимизации является актуальной проблемой.

Предложены подходы применения оптимальной технологии хранения данных Hadoop, которую используют в хранилищах данных для снижения затрат на хранение и обработку, а также для повышения эффективности отчетности и анализа.

***Ключевые слова*:** хранилища данных, оптимизация, большие данные, технология, затраты.

HADOOP КУРАЛЫ ЖАНА МААЛЫМАТ КАМПАЛАРЫН ОПТИМИЗАЦИЯЛОО

Картанова А.Дж., Абдрасакова А.Б., Иманбеков Т.И.
Н.Исанов атындагы Кыргыз мамлекеттик курулуш,
транспорт жана архитектура университети

Ишкананын берилиштер кампалары эң ири берилиштер базалары болуп өстү. Берилиштер кампалары колдонуучулардын санына, сакталган маалыматтардын көлөмүнө, маалымат булактарынын санына жана

отчеттордун татаалдыгына жана аналитикалык талаптарга жараша өсөт. Берилиштер кампаларынын өсүшү менен, натыйжалуулук жана параллелизм маселелери, сактоо жана иштетүү чыгымдары кабыл алынгыс болуп калышы мүмкүн. Мындай учурларда компаниялар маалымат кампаларын оптималдаштырышы керек жана оптималдаштыруу ыкмаларын жана технологияларын тандоо актуалдуу маселе болуп саналат.

Берилиштер кампаларында сактоо жана иштетүү чыгымдарын азайтуу, ошондой эле отчеттуулуктун жана талдоонун натыйжалуулугун жогорулатуу максатында Hadoop оптималдуу технологиясын маалыматтарды сактоого колдонууга ыкмалар сунушталат.

***Баштапкы сөздөр*:** берилиштер кампалары, оптимизациялоо, чоң маалыматтар, технология, чыгымдар.

HADOOP TOOL AND OPTIMIZATION DATA WAREHOUSE

Kartanova A.Dzh., Abdrasakova F.B., Imanbekov T.I.
Kyrgyz state construction, transport and architecture
university named of N.Isanov

Data warehouses have grown to be the largest databases in the organization. Data warehouses grow based on the number of users, the amount of data stored, the number of data sources, and the complexity of the reports and analytical requirements. As data warehouses grow, there are performance and concurrency issues, storage and processing costs that can become unacceptable. In such cases, organizations need to optimize their data warehouses, and the choice of optimization methods and technologies is a pressing issue.

Approaches are proposed for the use of the optimal Hadoop data storage technology, which is used in data warehouses to reduce storage and processing costs, as well as to improve the efficiency of reporting and analysis.

***Key words*:** data warehouses, optimization, big data, technology, costs.

Введение. В прошлом, было время, когда все хранилища данных были небольшими, а теперь большинство хранилищ данных превратились в крупнейшие базы данных организации. Хранилища данных растут, и будут расти в будущем не только в отношении объема хранимых данных,

но и в нескольких измерениях: по числу пользователей, по числу запросов, по числу аналитических сессий и отчетов и т.п.

Большинство хранилищ данных изначально были разработаны для поддержки стратегического и тактического управления. В настоящее время оперативному руководству, персоналу и даже внешним сторонам необходим оперативный доступ к хранимым данным. Кроме того, им могут потребоваться данные, которых еще нет в хранилище данных. Для новых форм аналитики и отчетности классических операционных данных уже не всегда достаточно. Внутренние данные должны быть дополнены внешними данными. Например, внутренние данные о продажах клиентов необходимо дополнить социально-демографическими данными, чтобы лучше понять модели покупок клиентов. Большие данные здесь также играют важную роль. Организации хотят проанализировать, что клиенты «говорят» о своих продуктах в социальных сетях. Они хотят объединить данные социальных сетей с внутренними данными.

До сих пор пользователи хранилищ данных были довольны, когда данные, используемые для их отчетов, были недельными. Сегодня пользователи больше не принимают задержку данных в одну неделю, им нужен один час, а может быть, даже пять секунд или меньше. Особенно группы пользователей, такие как операционное руководство и внешние стороны, хотят иметь представление о самой текущей ситуации - вчерашние данные бесполезны. Для сокращения задержек данных необходимо увеличить размер хранилища данных, поскольку необходимо хранить более подробные данные и большее количество моментов времени для этих значений данных. Независимо от того, являются ли большие данные структурированными, неструктурированными, мультиструктурированными или частично структурированными, это всегда огромный объем данных.

Приведем статистику по большим данным. В сентябре 2020 г. агентство ResearchAndMarkets опубликовало отчет о глобальном рынке

аналитики больших данных, где установлено, что мировой рынок аналитики больших данных оценивается в $41,85 млрд по итогам 2019 года и по прогнозам аналитиков, он вырастет до $115,13 млрд, при средней динамике в 11,9% в течение прогнозируемого периода с 2020 по 2028 год.

Аналитика больших данных становится одной из самых востребованных задач в современном бизнесе. По оценкам компании Frost & Sullivan в 2021 году общий объем мирового рынка аналитики больших данных увеличится по сравнению с показателем 2016 года более чем в 2,5 раза и составит $67,2 млрд, при ежегодных темпах роста на уровне 35,9% [1].

Методы и материалы. В общем, доступные методы оптимизации варьируются от второстепенных методов, таких как оптимизация прямого доступа к данным, вплоть до методов, с помощью которых реконструируется вся среда. Методы оптимизации можно классифицировать следующим образом:

Архитектурная оптимизация: Хранилища данных можно оптимизировать, внося улучшения на архитектурном уровне. Например, промежуточная область удаляется, чтобы снизить уровень задержки данных, или данные выгружаются в хранилище архивных данных.

Оптимизация инструмента: Хранилище данных можно улучшить, настроив и оптимизировав отдельные инструменты. Например, производительность запросов на серверах баз данных можно настроить, сохраняя больше данных в памяти или добавляя дополнительные индексы. Кроме того, серверные машины можно настроить для оптимизации обработки. Эти формы оптимизации обычно выполняются администраторами баз данных и разработчиками.

Оптимизация дизайна: Оптимизация дизайна подразумевает внесение изменений в исходный дизайн. Например, нормализованные структуры данных в витрине данных заменяются звездообразными

схемами или стиль копирования данных изменяется с пакетной передачи на передачу в реальном времени.

Замена технологии: Методика оптимизации также может заключаться в развертывании новой технологии. Например, для повышения производительности или масштабируемости может быть приобретен более специализированный сервер базы данных, более продвинутый инструмент ETL или аналитический инструмент.

В последние годы стало доступно множество новых мощных технологий хранилищ данных. В частности, для поддержки больших данных стали доступны многочисленные новые системы. Все они предназначены для хранения и управления огромными объемами данных при относительно низких затратах. Из всех этих технологий Hadoop является самой популярной.

Hadoop – это свободно распространяемый набор утилит, библиотек и Фреймворк для разработки и выполнения распределённых программ, работающих на кластерах из сотен и тысяч узлов.

Эта основополагающая технология хранения и обработки больших данных (Big Data) является проектом верхнего уровня фонда Apache Software Foundation и предназначен для приложений, в которых необходимо обрабатывать, хранить и управлять непрерывным потоком новых, входящих данных, он отлично справляется с обработкой сложных форм аналитики быстро и относительно недорого.

Hadoop был разработан для поддержки приложений с потоком кликов или управляемые датчиками, которые постоянно генерируют огромное количество записей в секунду, затем все эти данные необходимо сохранить для использования в будущем, что в итоге приведет к хранению огромного объема данных. Обработка, хранение и все вытекающие действия с такими объемами данных и есть Big Data. Hadoop — технология работы с BigData.

Что особенного в Hadoop, так это его относительно невысокая цена. Затраты на хранение данных и лицензию низкие по сравнению с традиционными серверами баз данных. Именно эта комбинация недорогого и объемного хранилища данных делает эту технологию подходящей для хранилищ данных.

Конечно за последние несколько лет было внедрено много новых технологий хранения данных. Все они были разработаны для хранения и управления огромными объемами данных при относительно низких затратах. Из всех этих технологий Hadoop, без сомнения, является самой популярной.

Изначально проект Hadoop разработан на Java в рамках вычислительной парадигмы MapReduce, когда приложение разделяется на большое количество одинаковых элементарных заданий, которые выполняются на распределенных компьютерах (узлах) кластера и сводятся в единый результат [2].

Проект Hadoop состоит из основных 4-х модулей:

Hadoop Common – набор инфраструктурных программных библиотек и утилит, которые используются в других решениях и родственных проектах, в частности, для управления распределенными файлами и создания необходимой инфраструктуры [2];

HDFS – распределённая файловая система, Hadoop Distributed File System – технология хранения файлов на различных серверах данных (узлах, DataNodes), адреса которых находятся на специальном сервере имен (мастере, NameNode) [3]. За счет дублирования (репликации) информационных блоков, HDFS обеспечивает надежное хранение файлов больших размеров, поблочно распределённых между узлами вычислительного кластера [2];

YARN – система планирования заданий и управления кластером (Yet Another Resource Negotiator), которую также называют MapReduce 2.0 (MRv2) – набор системных программ (демонов), обеспечивающих

совместное использование, масштабирование и надежность работы распределенных приложений [4]. Фактически, YARN является интерфейсом между аппаратными ресурсами кластера и приложениями, использующих его мощности для вычислений и обработки данных [2];

Hadoop MapReduce – платформа программирования и выполнения распределённых MapReduce-вычислений, с использованием большого количества компьютеров (узлов, nodes), образующих кластер.

Результаты исследования. Предположим, что архитектура информационной системы поддержки принятия решений (СППР) напоминает цепочку или сеть баз данных; см. рисунок 1. Данные передаются через эту сеть баз данных с помощью средств интеграции. В большинстве случаев все базы данных реализованы с использованием баз данных SQL.

Существует, как минимум четыре различных подхода к использованию Hadoop в СППР, чтобы сделать эту систему более дешевой, масштабируемой или более эффективной.

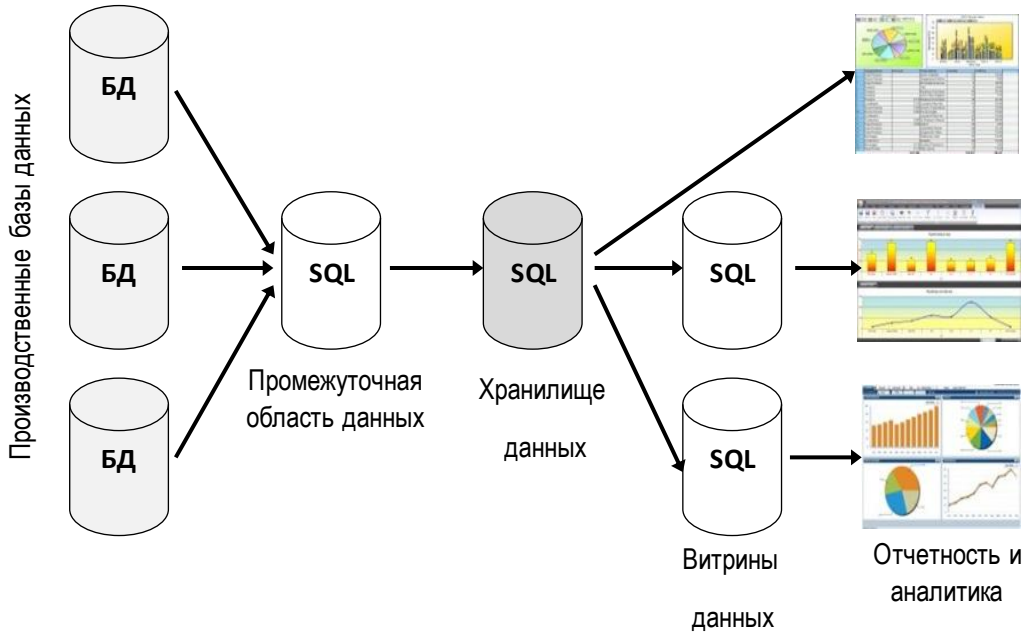

Рис.1. Классическая архитектура СППР

Приведем эти подходы использования Hadoop для оптимизации хранилища данных:

- Использование Hadoop для хранения холодных данных;
- Использование Hadoop в качестве промежуточной области;
- Использование Hadoop в качестве дополнительной базы данных хранилища данных;
- Использование Hadoop в процессе ETL (Extract-Transformation-Load/извлечение-трансформация-загрузка).

Существует много причин, в связи с которыми запросы к хранилищам данных и витринам данных становятся медленными. Одна из причин может заключаться в том, что таблицы, к которым осуществляется доступ, стали огромными. Представьте, что в настоящее время в хранилище данных хранятся данные за 10 лет. Возможно, не все эти данные используются часто. Во многих ситуациях, чем старше становятся данные, тем меньше они используются. Это позволяет классифицировать данные как холодные, теплые или горячие. Горячие данные используются почти каждый день, а холодные - изредка. Хранение холодных данных в хранилище данных замедляет выполнение большинства запросов и является дорогостоящим, поскольку хранится в дорогостоящей системе хранения.

В этой ситуации может быть полезно хранить холодные данные вне базы данных SQL и в Hadoop HDFS. Эта форма хранения дешевле, и данные по-прежнему доступны. В этом случае архитектура СППР показана, на рисунке 2. Здесь технически хранилище данных состоит из двух баз данных, одна из которых содержит теплые и горячие данные и реализована с классической базой данных SQL, а другая хранит холодные данные и использует Hadoop. Черная стрелка на рисунке 2 указывает на то, что периодически данные, которые перестали работать, перемещаются из базы данных SQL в хранилище данных Hadoop. Витрины данных заполнены данными, поступающими из обоих хранилищ данных, и, при необходимости, отчеты могут получить доступ к обоим хранилищам данных.

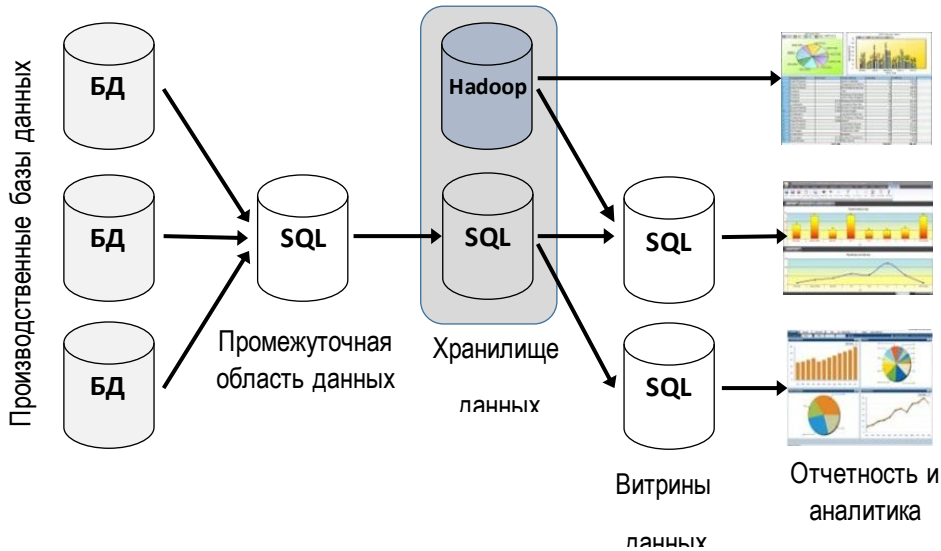

Рис. 2. Архитектура СППР, где холодные данные хранятся с помощью Hadoop, а теплые и горячие данные - с помощью сервера базы данных SQL

Рабочая нагрузка промежуточной области обычно проста: новые данные поступают и сохраняются, затем сохраненные данные преобразуются и копируются в базу данных хранилища данных. Большинство промежуточных областей разрабатываются с использованием технологии SQL. В конце концов, данные, которые были скопированы в хранилище данных, могут быть удалены из промежуточной области. Обычно отчеты на промежуточной площадке не выполняются.

Рабочая нагрузка промежуточной зоны направлена на Hadoop, см. рис. 3. Hadoop может быстро выгружать большие объемы данных и отлично извлекать данные с использованием пакетно-ориентированного подхода. Если данные должны быть сильно преобразованы перед копированием в хранилище данных, вся эта обработка может быть выполнена заданиями MapReduce.

Развертывание Hadoop в качестве промежуточной области (вместо базы данных SQL) снижает затраты на хранение данных и позволяет хранить и обрабатывать больше данных. В этом случае хранилище данных Hadoop не содержит таблиц, которые не хранятся в база данных SQL. Он содержит только таблицы, которые также доступны в базе данных

SQL, и то только в том случае, если в этих таблицах можно идентифицировать холодные данные.

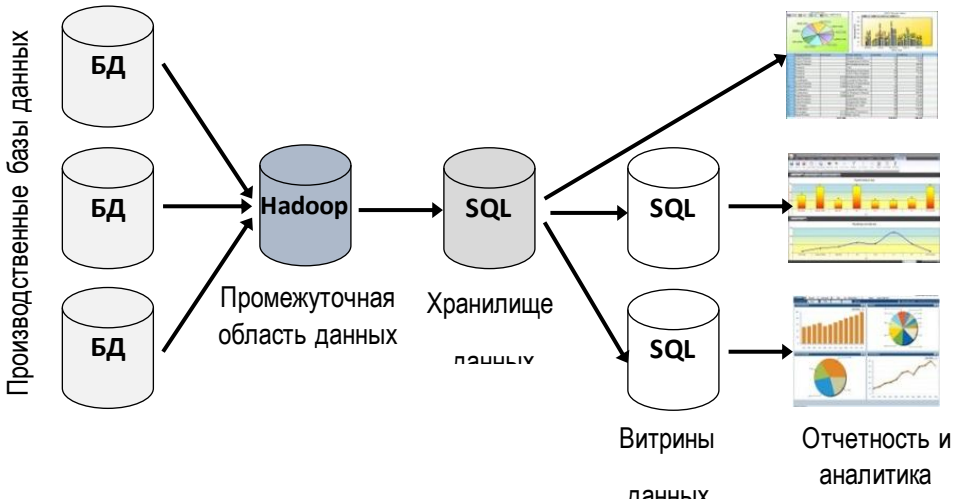

Рис.3.Использование Hadoop для создания промежуточной области

При третьем подходе два хранилища данных содержат разные наборы таблиц (и, следовательно, разные данные), см. рисунок 4. В этом решении данные копируются прямо из производственных систем в Hadoop, некоторые отчеты извлекают данные из этого хранилища данных Hadoop, а данные извлекаются для витрин данных для поддержки определенных форм отчетности и анализа. При таком подходе хранилище данных состоит из двух хранилищ данных: более классической базы данных и хранилища данных Hadoop.

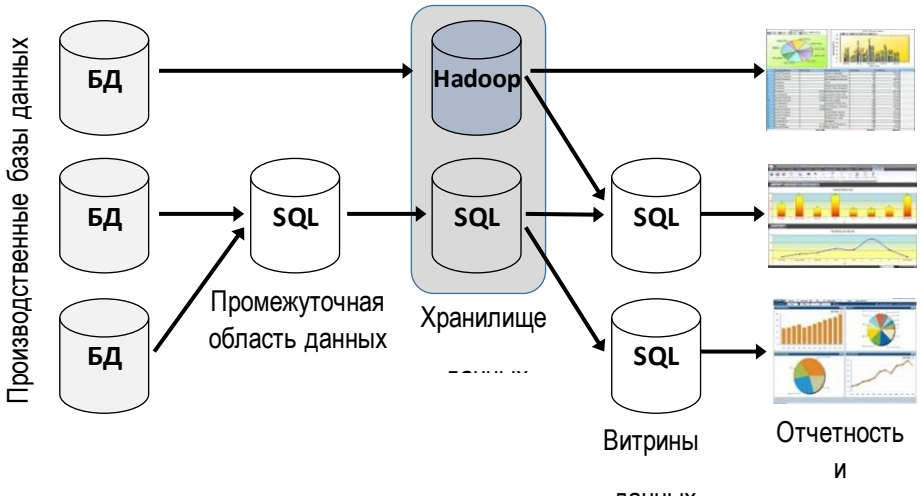

Рис.4.Использование Hadoop как дополнительное хранилище данных

Эта архитектура может быть полезна, если некоторые операционные системы генерируют огромные объемы новых данных, а база данных SQL, используемая для создания промежуточной области, не может справиться с этой тяжелой рабочей нагрузкой. Это может произойти, например, на заводе, где в машинах установлено множество датчиков или приложение, которое отслеживает все сообщения в социальных сетях, которые касаются организации, ее продуктов и брендов. В случае если новые данные вообще не хранятся в производственной системе их можно хранить прямо в Hadoop.

Hadoop используется в этой архитектуре для области приложения, в которой он выделяется: большая база данных, большая рабочая нагрузка входящих данных и периодические отчеты для извлечения данных.

Чистый объем данных в конкретном хранилище данных может оказаться слишком большим для базы данных SQL. Это может стать слишком дорогостоящим, а запросы - слишком медленными. В этой ситуации может быть принято решение агрегировать данные до того, как они будут сохранены в базе данных хранилища данных; см. рисунок 5. Это уменьшает размер базы данных и ускоряет запросы. Фактически, помимо агрегирования данных, могут применяться и другие формы обработки.

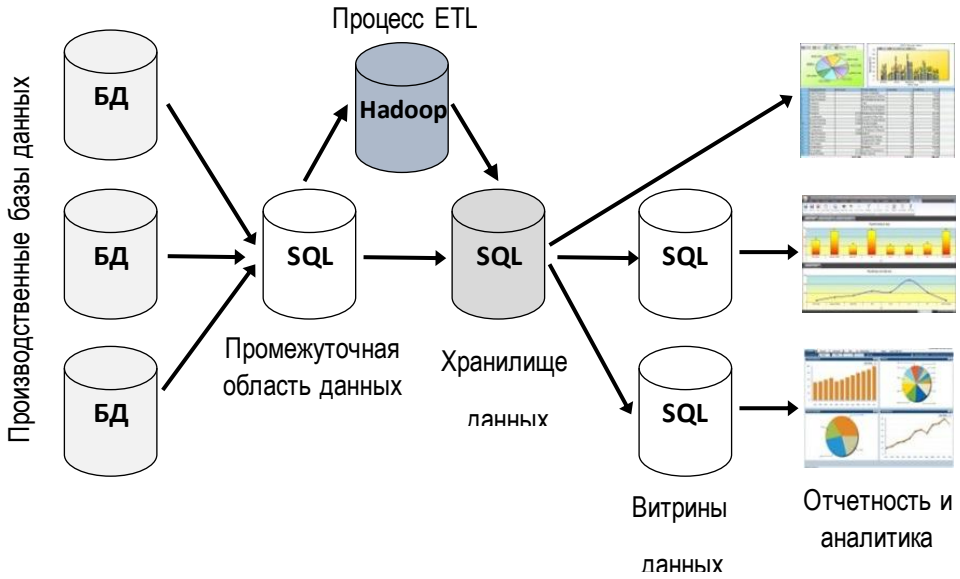

Рис. 5. Использование Hadoop для ускорения процесса ETL

Однако, если выбрано это решение, должна существовать база данных, содержащая все подробные данные (которая не является промежуточной областью), и какой-то модуль должен отвечать за агрегирование и обработку данных. Hadoop можно использовать в качестве хранилища всех подробных данных и позволить MapReduce выполнять все агрегаты. В этой архитектуре Hadoop содержит большие объемы данных - именно сюда данные загружаются в первую очередь. Затем запускаются задания MapReduce для эффективного агрегирования и обработки этих данных и их копирования в базу данных хранилища данных. Итак, Hadoop выполняет все агрегаты. Другими словами, он выполняет всю обработку данных до того, как они становятся доступными для отчетов и анализа - Hadoop используется в качестве механизма ETL. Как известно, ETL — комплекс методов, реализующих процесс переноса исходных данных из различных источников в аналитическое приложение или поддерживающее его хранилище данных. Задача процесса ETL - извлечь и очистить данные из отдельной исходной системы, интегрировать их и загрузить в систему хранилища данных [5].

Эта архитектура также может быть полезна, когда так называемые мульти структурированные данные хранятся и используются для анализа и отчетности. В случае много структурированных данных им не назначается никакая схема. Это как если бы данные хранятся в том необработанном формате, в котором они были получены. В этом случае схема должна быть получена при чтении данных. Это называется схемой при чтении. Когда хранимые данные имеют схему, например, большинство данных, хранящихся в базах данных SQL, это называется схемой при записи. Назначение схемы данным при их чтении требует большой обработки. Опять же, задания MapReduce могут делать это параллельно и, следовательно, быстро. Это делает эту архитектуру очень актуальной, когда много структурированные данные должны обрабатываться средой хранилища данных. Много структурированные данные хранятся в

хранилище данных Hadoop, и MapReduce назначает схему, когда данные должны быть скопированы в хранилище данных.

Для каждого из этих подходов требуется функциональность ETL в сочетании с Hadoop. В первом подходе ETL используется для копирования данных из хранилища данных на основе SQL в Hadoop и далее в витрины данных, во втором подходе ETL используется для копирования данных из производственных баз данных в Hadoop и из Hadoop в хранилище данных на основе SQL , в третьем подходе ETL копирует данные из производственных систем в Hadoop и из Hadoop в витрины данных, а в четвертом подходе ETL извлекает данные из промежуточной области на основе SQL, загружает их в Hadoop и копирует в хранилище данных на основе SQL.

В связи с этим, сформулируем требования к инструментам ETL для эффективной поддержки Hadoop:

- Продвинутость: Поскольку Hadoop MapReduce может распределять обработку логики по огромному количеству процессоров, он потенциально обладает высокой масштабируемостью. MapReduce не имеет проблем с параллельным выполнением фильтров данных, манипуляциями со строками, математическими функциями, строками, агрегаты и так далее. Чтобы использовать эту мощь, инструменты интеграции должны уметь продвинуть: столько же логики ETL в задания MapReduce. Другими словами, они должны иметь возможность создавать задания MapReduce для извлечения данных из HDFS и позволять Hadoop выполнять большую часть логики преобразования.
- Масштабируемость данных: Инструменты ETL должны быть оптимизированы внутри для обработки огромных объемов данных.
- Концепции NoSQL: Инструменты ETL должны понимать, как обрабатывать много структурированные, неструктурированные (например, текстовые) и структурированные данные, хранящиеся в HDFS. Они должны уметь преобразовывать такие концепции в плоские

реляционные концепции и наоборот.

- Двунаправленный: Инструменты ETL должны иметь возможность читать и писать в Hadoop. В частности, запись данных в Hadoop еще не является общей функцией инструментов ETL.
- Схема при чтении: Инструменты ETL должны поддерживать схему при чтении. Другими словами, они должны иметь возможность назначать схему данным, хранящимся в HDFS, в момент извлечения данных.
- Прозрачность хранилища данных: Собственный API MapReduce очень подробный и сложный. Разработчики должны подробно понимать, как работает технология и каковы эффективные стратегии обработки. Такой низкоуровневый интерфейс плохо сказывается на производительности и обслуживании. Кроме того, в большинстве отделы бизнес-аналитики не владеют необходимыми навыками для эффективной работы с MapReduce. Инструменты ETL, поддерживающие Hadoop, должны скрывать технические и низкоуровневые API Hadoop. Для разработчика все должно быть прозрачным, независимо от того, предназначено ли задание ETL для извлечения данных из базы данных SQL или Hadoop. Только в этом случае гарантируется высокая производительность.

Когда в 2005 году была основана компания Talend, они были первым поставщиком программного обеспечения ETL с открытым исходным кодом. В ноябре 2006 года они выпустили свой первый продукт - инструмент ETL под названием Talend Open Studio.

Разработчики, использующие Talend, видят одну общую среду выполнения. Независимо от того, нужно ли извлекать данные из базы данных SQL, плоского файла, службы или Hadoop, интерфейс остается одинаковым. Когда код ETL разработан, он развертывается в этой общей среде выполнения.

Внутренне эта общая среда выполнения состоит из нескольких сред выполнения. В настоящее время Talend поддерживает четыре среды

выполнения: Java, SQL, Camel и Hadoop MapReduce. На основе кода ETL Talend генерирует собственный оптимизированный код для этих сред. Так, например, разработчик, пишущий код для извлечения данных из исходной системы, не должен иметь дело с техническими аспектами этой системы. Для каждой среды генерируется оптимизированный код. Итак, если код ETL должен выполняться в Hadoop MapReduce, оптимизированный код MapReduce генерируется средой выполнения, разработанной специально для Hadoop. Затем этот код помещается в MapReduce для параллельного выполнения.

Таким образом, разработчики Talend не видят MapReduce. Они видят свои знакомые, графические, высокоуровневый интерфейс для проектирования ETL-заданий. Эти спецификации используются Talend для генерации кода MapReduce, который запускается в массовом параллельном режиме. Как будто Talend Open Studio - это среда проектирования и разработки, а Hadoop - среда выполнения.

Приведем преимущества «сокрытия» Hadoop: повышение производительности и простота обслуживания. Дизайнеры могут разрабатывать задания ETL, не учитывая специфики Hadoop и используемых технологий хранения данных; обучение разработчиков ETL работе с Hadoop не требуется; в случае добавления новых функций в Hadoop или изобретения новых подходов к программированию эффективного кода MapReduce, разработчики Talend могут развернуть их, не изменяя свои спецификации ETL. Это будет просто вопрос регенерации кода.

Поскольку особенности Hadoop скрыты, относительно легко встроить Hadoop в существующую архитектуру хранилища данных. Представьте, что существующая архитектура использует сервер базы данных SQL в качестве промежуточной области. Если можно разработать более оптимизированную архитектуру путем замены сервера базы данных SQL

на Hadoop, только существующие данные должны быть скопированы в Hadoop, а существующий код ETL должен быть восстановлен для Hadoop.

Выводы. Таким образом, были описаны четыре подхода, в которых Hadoop можно использовать для оптимизации системы поддержки принятия решений. Оптимизация в этой ситуации подразумевает снижение затрат, повышение производительности и улучшение масштабируемости данных. Очевидно, есть и другие способы использования Hadoop для оптимизации системы хранилищ данных, но эти подходы наиболее вероятны и используются сегодня. Обратите внимание, что в одной и той же СППР можно использовать несколько подходов.

Подводя итог, можно сказать, что сокрытие технологии Hadoop для разработчиков ETL упрощает внедрение Hadoop в текущую систему хранилища данных. Другими словами, становится проще встраивать Hadoop и получать бизнес-преимущества, предлагаемые Hadoop, такие как, высокая масштабируемость данных и низкие расходы. Устраняя технические проблемы, связанные с Hadoop, Talend ETL делает Hadoop простым инструментом для оптимизации хранилищ данных.

В заключение можно сказать, что способность Hadoop обрабатывать большие хранилища данных с низкими затратами делает его самым оптимальным инструментом для снижения затрат на хранение и управления хранилищем данных, а также для повышения производительности обработки данных, анализа и отчетности.

ЛИТЕРАТУРА

1. https://www.tadviser.ru/index.php/Статья:Большие_данные_(Big_Data)_мировой_рынок
2. https://ru.wikipedia.org/wiki/Hadoop
3. https://m.habr.com/ru/post/240405/
4. https://www.ibm.com/developerworks/ru/library/bd-hadoopyarn/index.html
5. Kimball R. The Data Warehouse Lifecycle Toolkit, 2nd Edition: Practical Techniques for Building Data Warehouse and Business Intelligence Systems/ Kimball R., Ross M., etc. – John Wiley & Sons, 2008.