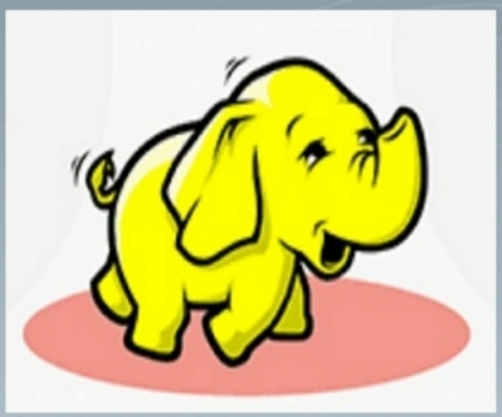
Технология хранения и обработки больших данных

Hadoop



Тимур Машнин

12+

Тимур Машнин

Технология хранения и обработки больших данных Hadoop

«ЛитРес: Самиздат»

2021

Машнин Т.

Технология хранения и обработки больших данных Hadoop / Т. Машнин — «ЛитРес: Самиздат», 2021

ISBN 978-5-532-96881-3

Apache Hadoop - это платформа для распределенной обработки больших наборов данных на кластерах компьютеров с использованием простых моделей программирования. В этой книге вы познакомитесь с общей архитектурой платформы, компонентами стека, такими как HDFS и MapReduce, приложениями Hadoop.

ISBN 978-5-532-96881-3

Содержание

Тимур Машнин
Технология хранения и обработки больших данных Hadoop

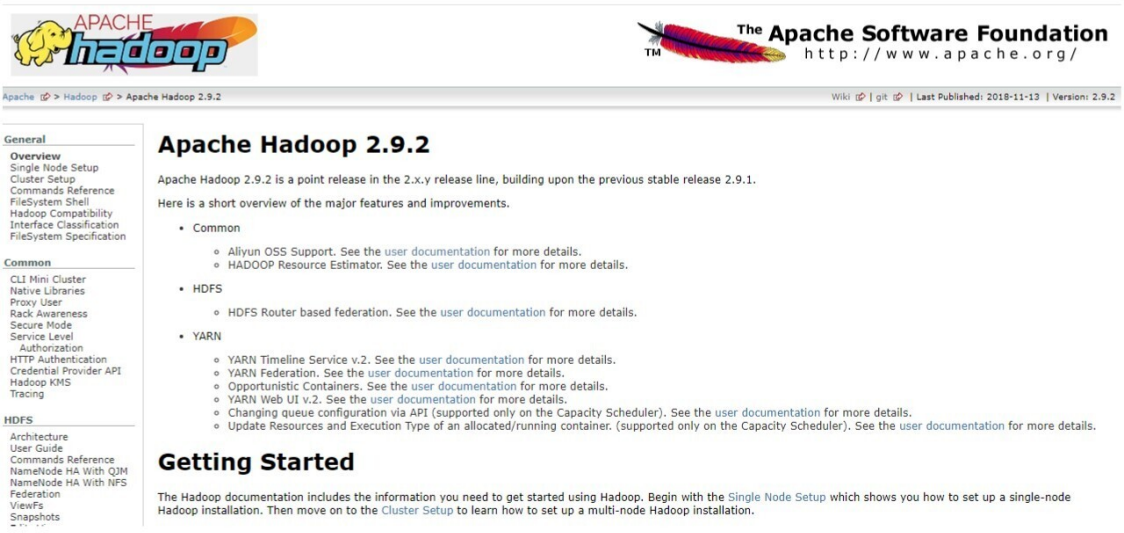
Введение

Tech Solutions

Hadoop

Введение

Hadoop – это программная платформа с открытым исходным кодом Apache для хранения и крупномасштабной обработки больших наборов данных в распределенной среде кластеров компьютеров с использованием простых моделей программирования.



Hadoop предназначен для масштабирования от отдельных серверов до тысяч машин, каждая из которых обеспечивает локальные вычисления и хранилище.

Фреймворк Hadoop был создан Дагом Каттингом и Майком Кафареллой в 2005 году.

Первоначально этот фреймворк был разработан для поддержки распространения проекта Nutch Search Engine построения поисковых систем.

Даг, который в то время работал в Yahoo, а сейчас является главным архитектором в Cloudera, назвал этот проект в честь слона своего сына.

Его сын назвал своего игрушечного слона Hadoop, и Даг использовал это имя, чтобы так назвать свой проект.

Давайте посмотрим, что делает фреймворк Hadoop таким интересным, масштабируемым и удобным в использовании.

Hadoop начинался как простая среда пакетной обработки.

Идея, лежащая в основе Hadoop, заключается в том, что вместо перемещения данных в вычисления мы переносим вычисления в данные.

И в основе системы Hadoop лежит масштабируемость.

Все модули в Hadoop разработаны с фундаментальным предположением о том, что аппаратное обеспечение рано или поздно выходит из строя.

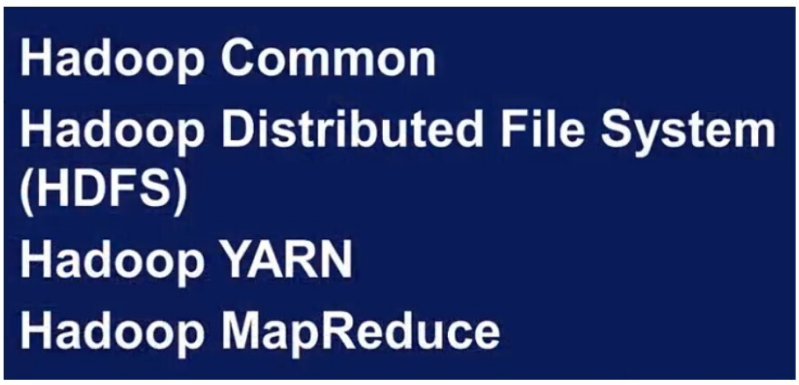
То есть предположением, что отдельная машина или стойка машин, или большой кластер или суперкомпьютер, все они в какой-то момент выйдут из строя, или некоторые их компоненты выйдут из строя.

И компоненты Apache Hadoop – MapReduce и HDFS изначально были созданы на основе Google MapReduce и файловой системы Google.

Еще одна очень интересная вещь, которую приносит Hadoop, – это новый подход к данным.

Новый подход заключается в том, что мы можем сохранить все данные, которые у нас есть, и мы можем взять эти данные и читать данные, создавая схему, во время чтения.

Вместо того, чтобы тратить время на создание схемы, пытаясь подогнать данные к схеме, которую мы создали заранее, мы сохраняем все данные в приблизительном формате, а затем проецируем их в схему на лету, пока мы эти данные читаем.



Фреймворк Apache Hadoop содержит четыре основных компонента.

Это Hadoop Common, распределенная файловая система Hadoop или HDFS, Hadoop MapReduce и Hadoop YARN.

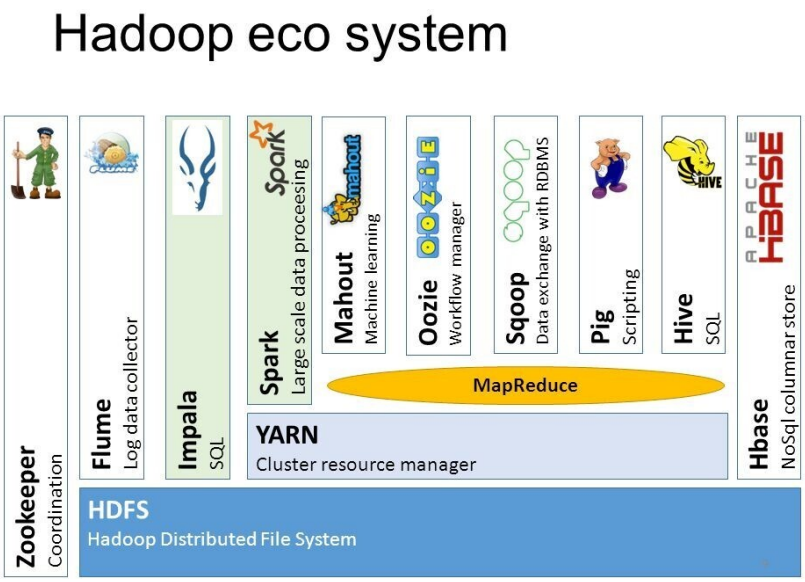
Hadoop Common содержит библиотеки и утилиты, необходимые для других модулей Hadoop.

Распределенная файловая система Hadoop хранит данные на обычном компьютере, обеспечивая очень высокую совокупную пропускную способность по всему кластеру компьютеров.

Hadoop YARN – это платформа управления ресурсами, которая отвечает за управление вычислительными ресурсами в кластере и их использование в при планировании пользователей и приложений.

И Hadoop MapReduce – это модель программирования, которая масштабирует данные по множеству процессов.

И все модули фреймворка Hadoop разработаны с фундаментальным предположением, что аппаратное обеспечение выходит из строя.



Если вы посмотрите на HDFS, YARN, MapReduce и всю платформу в целом, она состоит из многочисленных приложений, и каждое из этих приложений создано с учетом этого предположения.

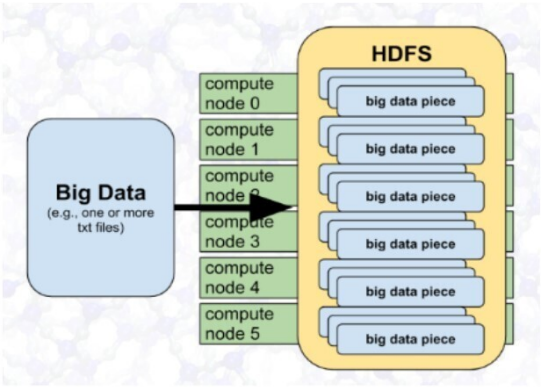
У нас есть различные приложения, такие как Apache PIG, Apache Hive, HBase и другие.

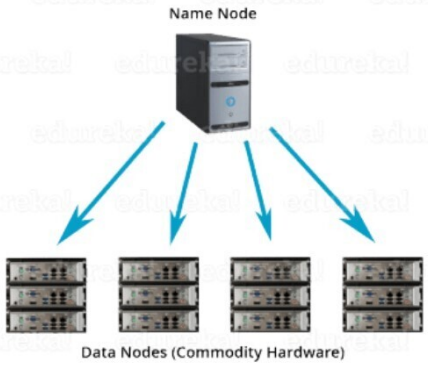
И для конечного пользователя, через Java-код MapReduce, он может получить доступ к любому из этих приложений.

И мы можем строить различного вида системы из этих приложений.

Проекты Apache PIG и Apache Hive предоставляют интерфейсы высокого уровня, обеспечивая доступ к данным через пользовательский интерфейс.

Сам фреймворк Hadoop в основном написан на языке программирования Java и проект также содержит несколько приложений на нативном языке C и утилиты командной строки.





Теперь, давайте немного поговорим о распределенной файловой системе Hadoop.

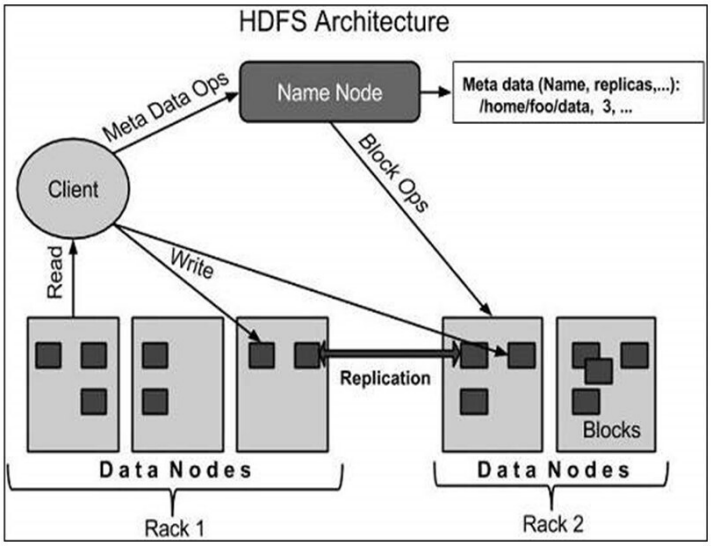
Что такое HDFS по своей сути?

Это распределенная, масштабируемая и переносимая файловая система, написанная на Java для поддержки фреймворка Hadoop.

Каждый Hadoop кластер обычно состоит из одного узла Namenode и кластера узлов Datanode, которые и формируют этот кластер.

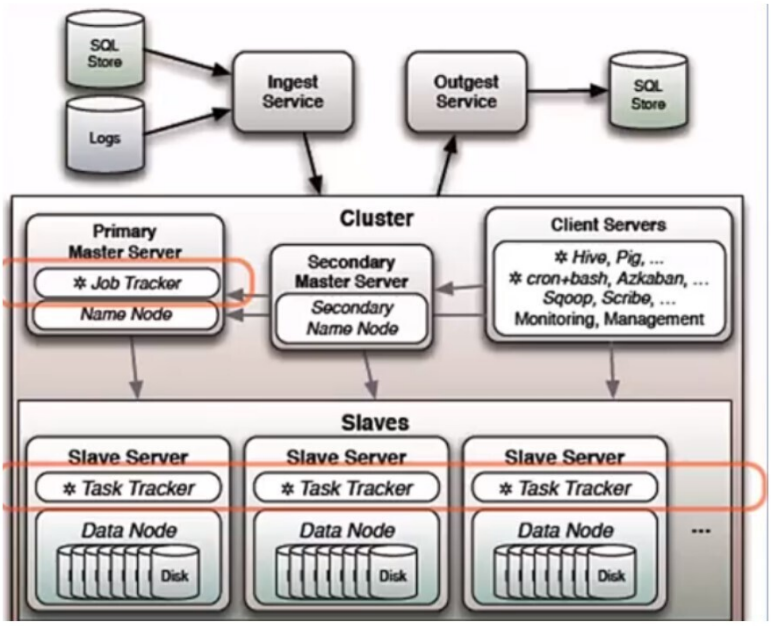
И каждая система HDFS хранит большие файлы, как правило, в диапазоне от гигабайтов до терабайтов.

И надежность системы HDFS достигается путем репликации многочисленных хостов.



Также файловая система HTFS поддерживает так называемый вторичный узел NameNote, который регулярно подключается к первичному узлу NameNote и создает снимки его состояния, запоминая, что система сохраняет в локальных и удаленных каталогах.

В каждой системе, основанной на Hadoop, содержится какая-то версия движка MapReduce.



Типичный движок MapReduce содержит средство отслеживания работы, в которое клиентские приложения могут отправлять задания MapReduce.

И этот трекер работы передает задачи всем доступным трекерам задач, которые есть в кластере.

Таким образом, классический Hadoop MapReduce представляет собой один процесс JobTracker и произвольное количество процессов TaskTracker, или по-другому один мастер узел и множество узлов slave.

MapReduce выполняет работу над огромным набором данных, обрабатывая данные и сохраняя их в HDFS таким образом, что извлечение данных производится проще, чем в традиционном хранилище.

Модель MapReduce следует принципам функционального программирования, вследствие чего пользовательские вычисления выполняются как функции map и reduce, обрабатывающие данные в виде пар ключ-значение.

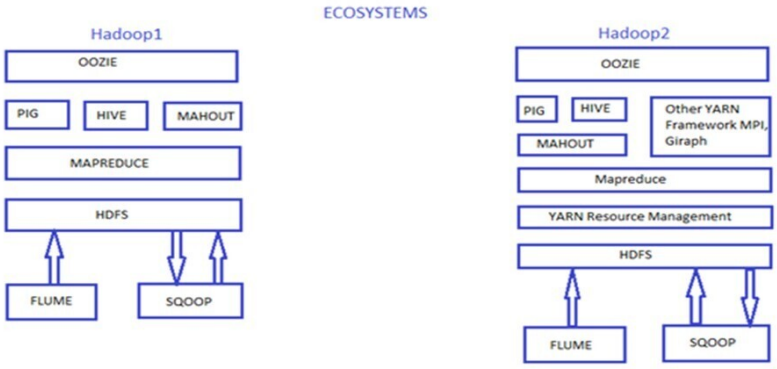
Hadoop предоставляет высокоуровневый программный интерфейс для реализации пользовательских функций map и reduce на различных языках.

Также Hadoop предоставляет инфраструктуру для выполнения заданий MapReduce в виде серий задач map и reduce.

Задачи map вызывают функции map для обработки наборов входных данных.

Затем задачи reduce вызывают функции reduce для обработки промежуточных данных, сгенерированных функциями map, формируя окончательные выходные данные.

Задачи map и reduce выполняются изолированно друг от друга, что обеспечивает параллельность и отказоустойчивость вычислений.



Hadoop версии 1 содержал компоненты HDFS и Map Reduce.

И Hadoop версии 1 разрабатывался только для выполнения заданий MapReduce.

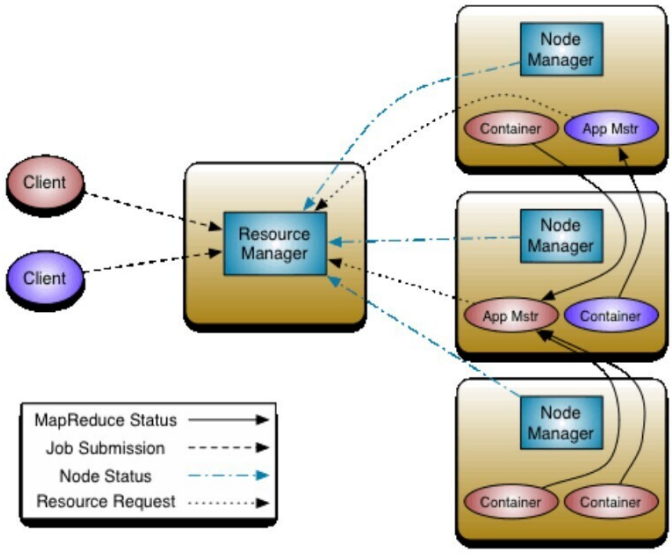
А Hadoop версии 2 уже содержит компоненты HDFS и YARN/Map Reduce версии 2.

В классическом Map Reduce, когда мастер узел перестает работать, тогда все его узлы slave автоматически перестают работать.

И мы должны перезапустить весь кластер и заново начать выполнять работу.

Это единственный сценарий, когда выполнение работы может прерваться, и это создает единственную точку отказа.

Компонент YARN или Yet Another Resource Negotiator решает эту проблему благодаря своей архитектуре.



YARN основывается на концепции нескольких мастер узлов и нескольких подчиненных slave узлов, и если один мастер узел выйдет из строя, тогда другой мастер узел возобновит процесс и продолжит выполнение.

Классический Map Reduce отвечает как за управление ресурсами, так и за обработку данных.

В Hadoop версии 2, YARN разделяет функций управления ресурсами и планирования/мониторинга заданий на отдельные демоны.

YARN – это универсальная платформа для запуска любого распределенного приложения, и здесь Map Reduce – это распределенное приложение, которое работает поверх YARN.

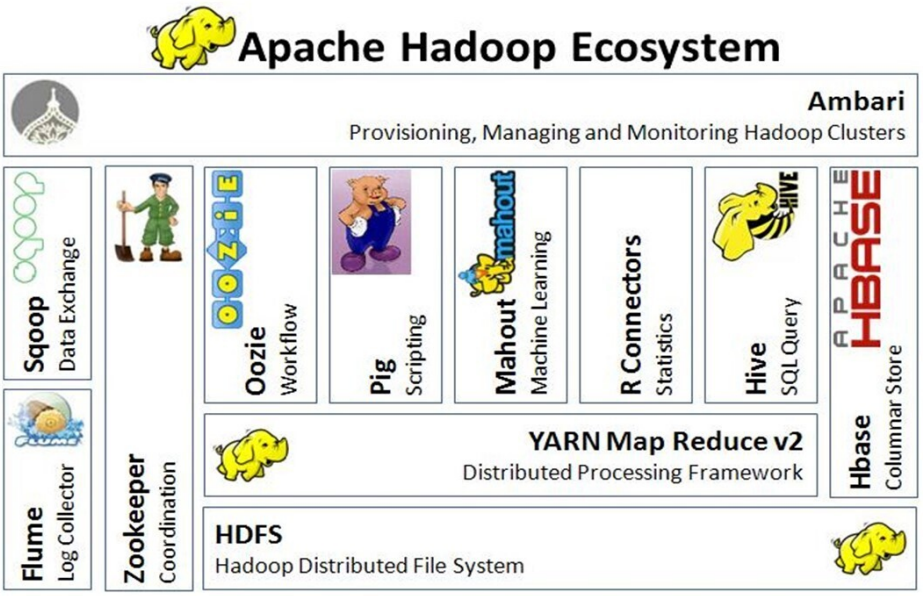
Таким образом, YARN отвечает за управление ресурсами, то есть решает, какая работа будет выполняться и какой системой.

Тогда как Map Reduce является фреймворком программирования, который отвечает за то, как выполнить конкретную работу, используя два компонента mapper и reducer.

YARN отделяет компоненты управления ресурсами от компонентов обработки, и YARN не сводится только к MapReduce.

Диспетчер ресурсов resource manager YARN оптимизирует использование кластера и поддерживает другие рабочие процессы, кроме Map Reduce.

Поэтому здесь мы можем добавлять дополнительные программные модели, такие как обработка графов или итеративное моделирование, которые могут обрабатывать данные, используя те же кластеры и общие ресурсы.



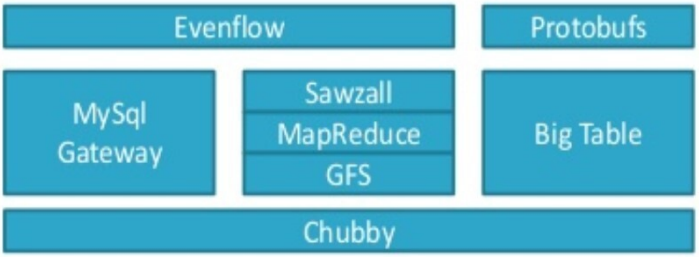
Поверх HDFS и Yarn могут работать множество компонентов, и эта архитектура также развивалась с течением времени.

Давайте посмотрим на историю и посмотрим, как вся эта экосистема Hadoop развивалась и росла со временем.

Как вы можете заметить, у многих из этих приложений смешные имена.

Как мы можем понять весь этот зоопарк, и как мы можем понять, что делает каждое из этих приложений?

Проект Hadoop возник из концепции Google MapReduce и идеи о том, как можно обрабатывать очень большие объемы данных.



Здесь показан стек Google Big Data.

И он начинается с файловой системы Google GFS.

В Google подумали, что будет хорошей идеей использовать большое количество распределенного дешевого хранилища, и попытаться разместить там много данных.

И придумать какой-то фреймворк, который позволил бы обрабатывать все эти данные.

Таким образом, у Google появился свой оригинальный MapReduce, и они хранили и обрабатывали большие объемы данных.

Затем в Google сказали, что это действительно здорово, но нам бы очень хотелось иметь доступ к этим данным и обращаться к ним на языке, похожем на SQL.

Поэтому они создали шлюз MySQL Gateway, чтобы адаптировать данные в кластере MapReduce и иметь возможность запрашивать эти данных.

Затем они поняли, что им нужен специальный язык высокого уровня для доступа к MapReduce в кластере и отправки работы.

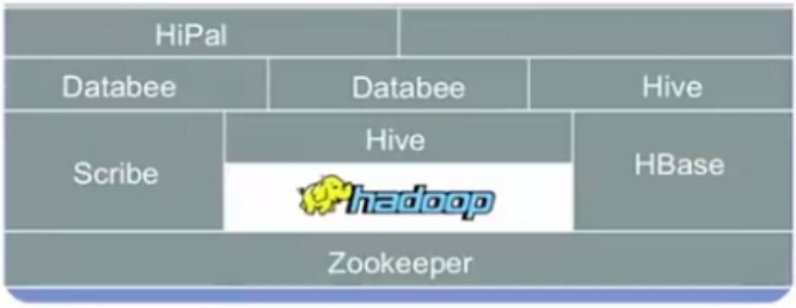
Так появился Sawzall.

Затем появился Evenflow и позволил связывать воедино сложные рабочие нагрузки и координировать сервисы и события.

Затем появился Дремель. Dremel – это хранилище и менеджер метаданных, который позволяет управлять данными и обрабатывать очень большой объем неструктурированных данных.

И затем, конечно, вам нужно что-то, чтобы координировать все это между собой.

Так появился Chubby в качестве системы координации, которая управляет всеми продуктами в этой экосистеме, обрабатывающей большие объемы данных.



Здесь показан стек Facebook Big Data.

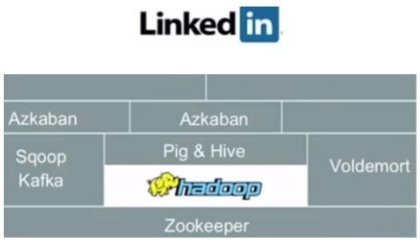
И мы видим, что стек Facebook выглядит очень похожим.

Здесь есть Zookeeper, аналог Chubby, цель которого хранение и управление конфигурациями систем.

Здесь есть HBase, и таблицы в HBase служат входом и выходом для работы MapReduce.

И здесь Hive и Databee, которые обеспечивает SQL запросы.

И есть Scribe, который используется для агрегации лог данных, передаваемых в режиме реального времени с большого количества серверов.

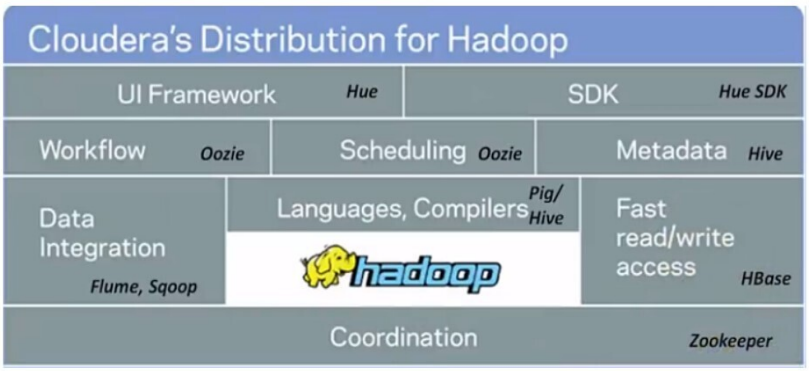


Затем, если мы посмотрим на стек Yahoo, вы увидите, что они используют те же компоненты, некоторые из них с другими именами, но для тех же целей.

LinkedIn также имеет свою версию этого стека.

И опять же, вы можете видеть, что здесь те же компоненты, некоторые из которых имеют свои реализации.

Таким образом, вы можете видеть, что из всех этих стеков возникает шаблон, который используют разные организации.



И здесь показан Hadoop стек CDH – Cloudera's distribution for Hadoop компании Cloudera.

Cloudera – это американская компания, разработчик дистрибутивов Apache Hadoop и ряда программных продуктов экосистемы Hadoop.

В этом стеке у нас есть Sqoop, инструмент, предназначенный для эффективной передачи больших данных между Hadoop и структурированными хранилищами данных, такими как реляционные базы данных.

И есть Flume – распределенный сервис для агрегирования больших объемов лог данных.

Здесь используется HBase для случайной записи и чтения данных, хранящихся в HDFS.

Oozie используется в качестве движка координации и рабочего процесса.

И Pig и Hive обеспечивают языки высокого уровня запросов данных.

И наконец здесь используется Zookeeper в качестве службы координации в основе этого стека.

И мы можем скачать и запустить виртуальную машину Cloudera, которая позволяет запускать все эти различные сервисы и узнавать, как они работают, без необходимости установки сервера.

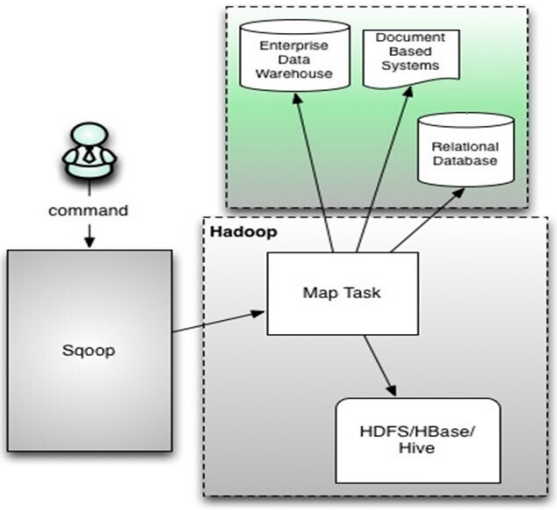
Но сначала давайте поговорим о различных инструментах, которые мы будем использовать поверх платформы Hadoop.

С развитием вычислительной техники стало возможным управлять огромными объемами данных, которые раньше мы могли обрабатывать только на суперкомпьютерах.

Настоящий прорыв произошел, когда такие компании, как Yahoo, Google и Facebook пришли к пониманию, что им нужно что-то сделать, чтобы обрабатывать и монетизировать эти огромные объемы данных, которые они собирают.

В результате были созданы различные инструменты и собраны стеки Big Data.

И давайте начнем обсуждение этих инструментов с Apache Sqoop.

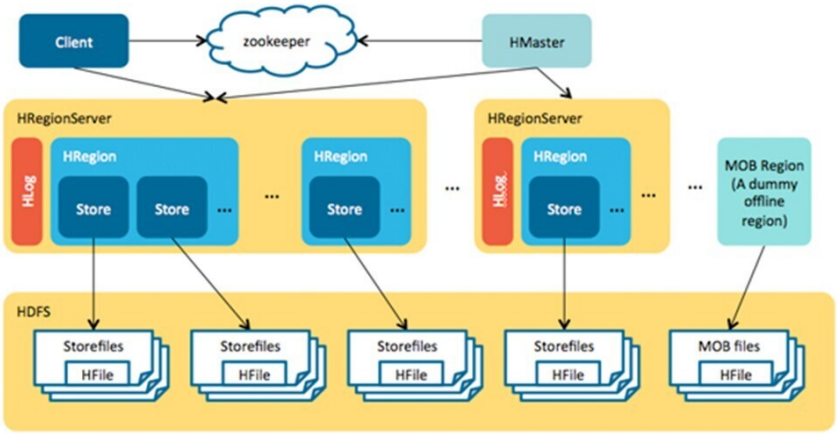


Sqoop означает SQL для Hadoop.

Это простой инструмент командной строки, который позволяет импортировать отдельные таблицы или целые базы данных в систему HDFS.

И этот инструмент генерирует классы Java, чтобы можно было взаимодействовать с данными, которые мы импортировали.

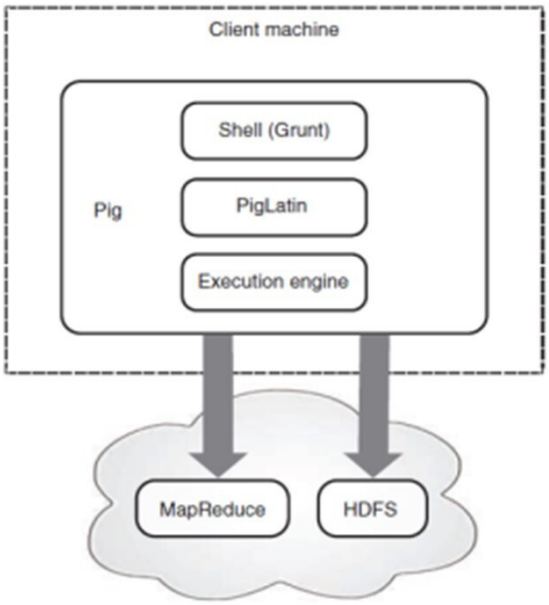
С этим инструментом Вы можете работать с данными базы данных SQL в среде Hadoop и использовать Map Reduce для запуска заданий с этими данными.



Следующий инструмент – это Hbase.

Hbase является ключевым компонентом стека Hadoop, так как он предназначен для приложений, которым требуется быстрый произвольный доступ к большому набору данных.

И Hbase основывается на Google Big Table и может обрабатывать большие таблицы данных, объединяющие миллиарды строк и миллионы столбцов.



Pig – это язык скриптов, это платформа высокого уровня для создания программ MapReduce с использованием Hadoop.

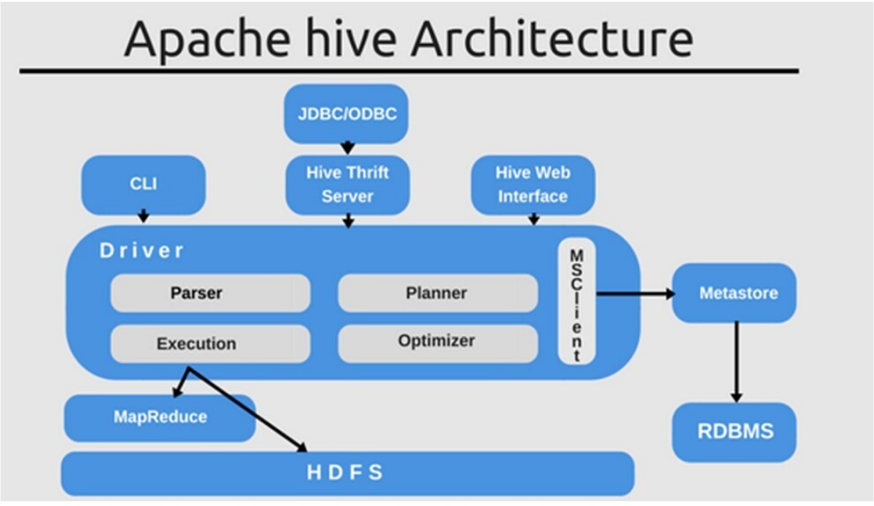
Этот язык называется Pig Latin, и он предназначен для задач анализа данных как потоков данных.

Pig самодостаточен, и вы можете выполнят все необходимые манипуляции в Hadoop, просто используя pig.

Кроме того, в pig, вы можете использовать код на разных языках, таких как JRuby, JPython и Java.

И наоборот, вы можете выполнять скрипты PIG на других языках.

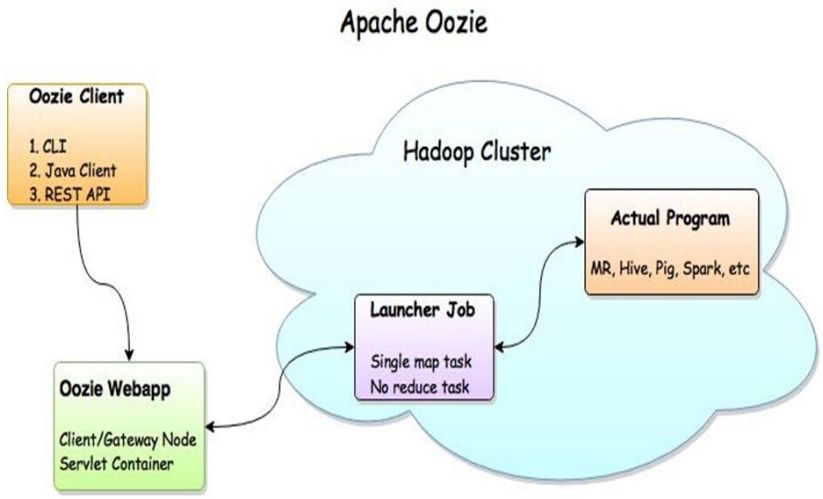
Таким образом, в результате вы можете использовать PIG в качестве компонента для создания гораздо более крупных и более сложных приложений.



Программное обеспечение Apache Hive облегчает запросы и управление большими наборами данных, которые находятся в распределенном хранилище файлов.

Hive предоставляет механизм для проектирования структуры поверх этих данных и позволяет использовать SQL-подобные запросы для доступа к данным, которые хранятся в этом хранилище данных.

И этот язык запросов называется Hive QL.

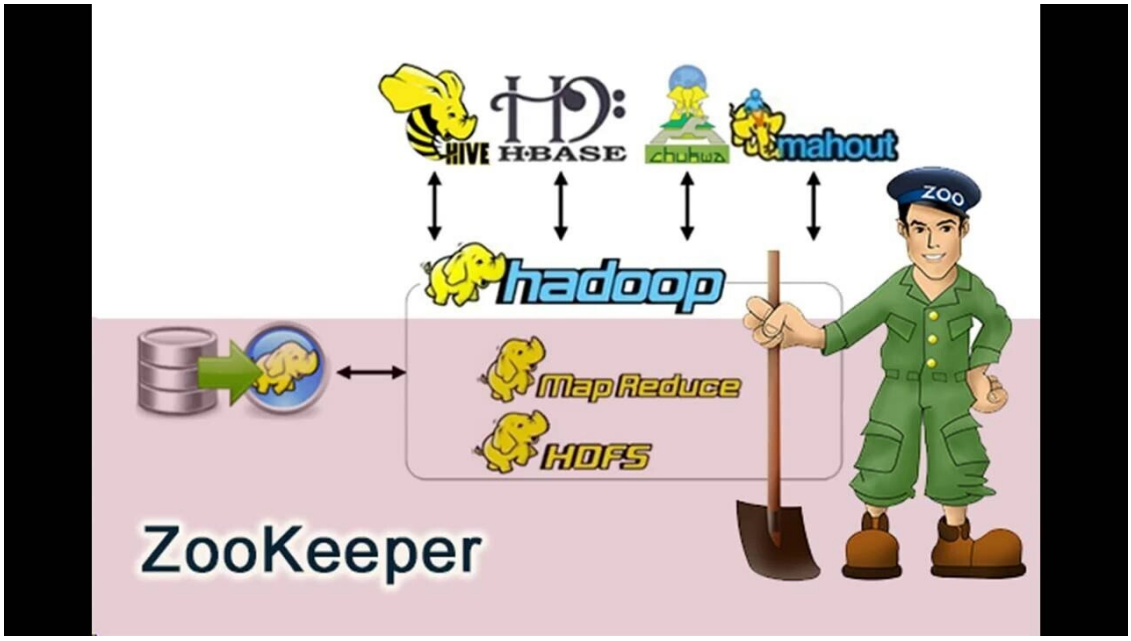


Oozie – это система планирования рабочих процессов, которая управляет всеми нашими заданиями Hadoop.

Задания рабочего процесса Oozie – это то, что мы называем DAG или Directed Graphs.

Задания координатора Oozie – это периодические задания рабочего процесса Oozie, которые запускаются по частоте или доступности данных.

Oozie интегрирован с остальной частью стека Hadoop и может поддерживать сразу несколько различных заданий Hadoop.



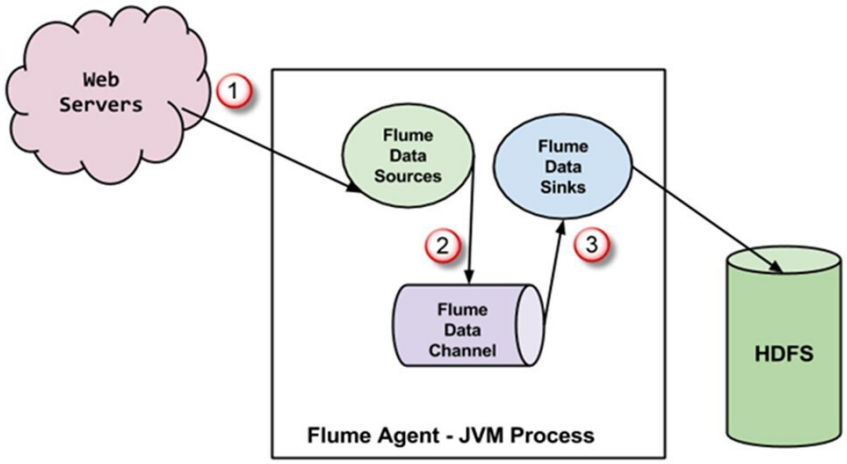
Следующий инструмент – это Zookeeper.

У нас есть большой зоопарк сумасшедших диких животных, и мы должны держать их вместе и как-то их организовывать.

Это как раз то, что делает Zookeeper.

Он предоставляет операционные сервисы для кластера Hadoop.

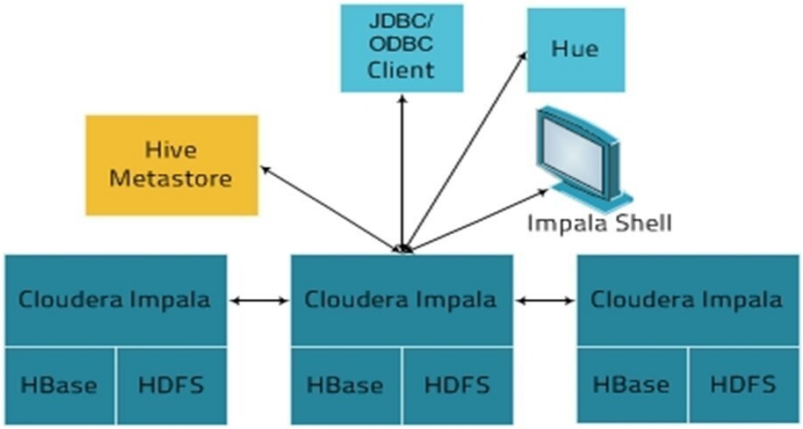
Он предоставляет службу распределенной конфигурации и службу синхронизации, поэтому он может синхронизировать все эти задания и реестр имен для всей распределенной системы.



Инструмент Flume – это распределенный сервис для эффективного сбора и перемещения больших объемов данных.

Он имеет простую и очень гибкую архитектуру, основанную на потоковых данных.

И Flume использует простую расширяемую модель данных, которая позволяет применять различные виды аналитических онлайн приложений.



Еще один инструмент – это Impala, который был разработан специально для Cloudera, и это механизм запросов, работающий поверх Hadoop.

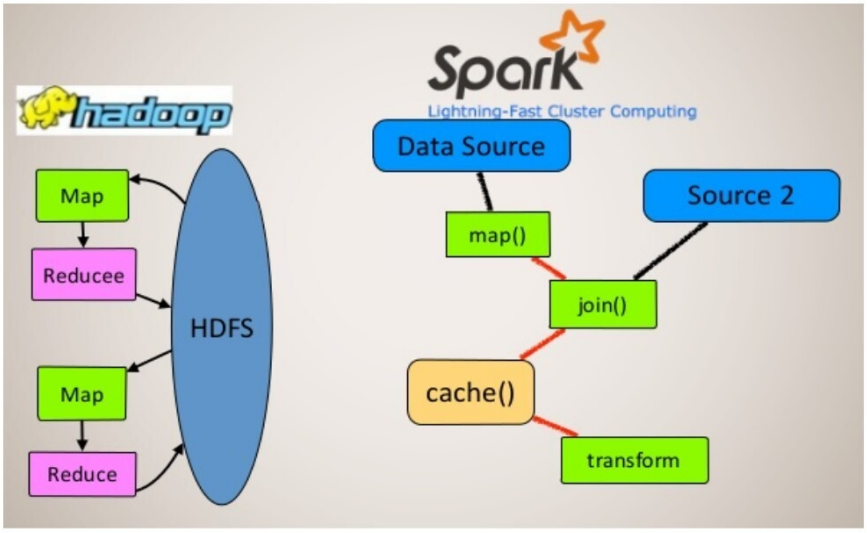
Impala привносит в Hadoop технологию масштабируемой параллельной базы данных.

И позволяет пользователям отправлять запросы с малыми задержками к данным, хранящимся в HTFS или Hbase, не сопровождая это масштабными перемещениями и манипулированием данными.

Impala интегрирована с Hadoop и работает в той же экосистеме.

Это обеспечивает масштабируемую технологию параллельных баз данных на вершине Hadoop.

И это позволяет отправлять SQL-подобные запросы с гораздо более высокими скоростями и с гораздо меньшей задержкой.



Еще один дополнительный компонент, это Spark.

Хотя Hadoop широко используется для анализа распределенных данных, в настоящее время существует ряд альтернатив, которые предоставляют некоторые интересные преимущества по сравнению с традиционной платформой Hadoop.

И Spark – это одна из таких альтернатив.

Apache Spark – это фреймворк экосистемы Hadoop с открытым исходным кодом для реализации распределённой обработки данных.

В отличие от классического обработчика Hadoop, реализующего двухуровневую концепцию MapReduce с дисковым хранилищем, Spark использует специализированные примитивы для рекуррентной обработки в оперативной памяти, благодаря чему позволяет получать значительный выигрыш в скорости работы для некоторых классов задач, в частности, возможность многократного доступа к загруженным в память пользовательским данным делает библиотеку привлекательной для алгоритмов машинного обучения.

И Spark поддерживает язык Scala, и предоставляет уникальную среду для обработки данных.

Для управления кластерами Spark поддерживает автономные нативные кластеры Spark, или вы можете запустить Spark поверх Hadoop Yarn.

Что касается распределенного хранилища, Spark может взаимодействовать с любой системой хранения, включая HDFS, Amazon S3 или с каким-либо другим пользовательским решением.

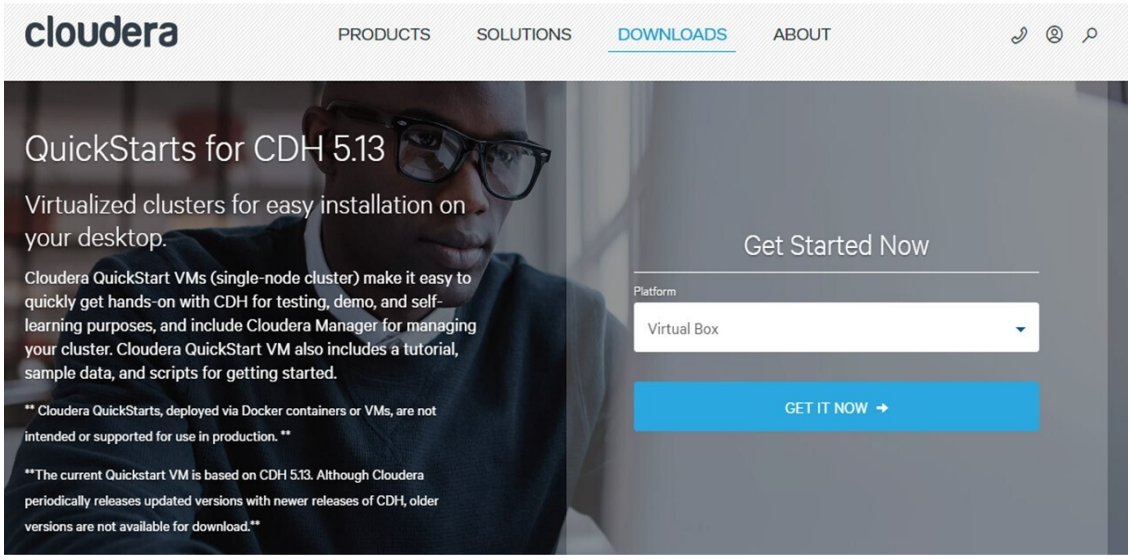
Cloudera QuickStart VM

Tech Solutions

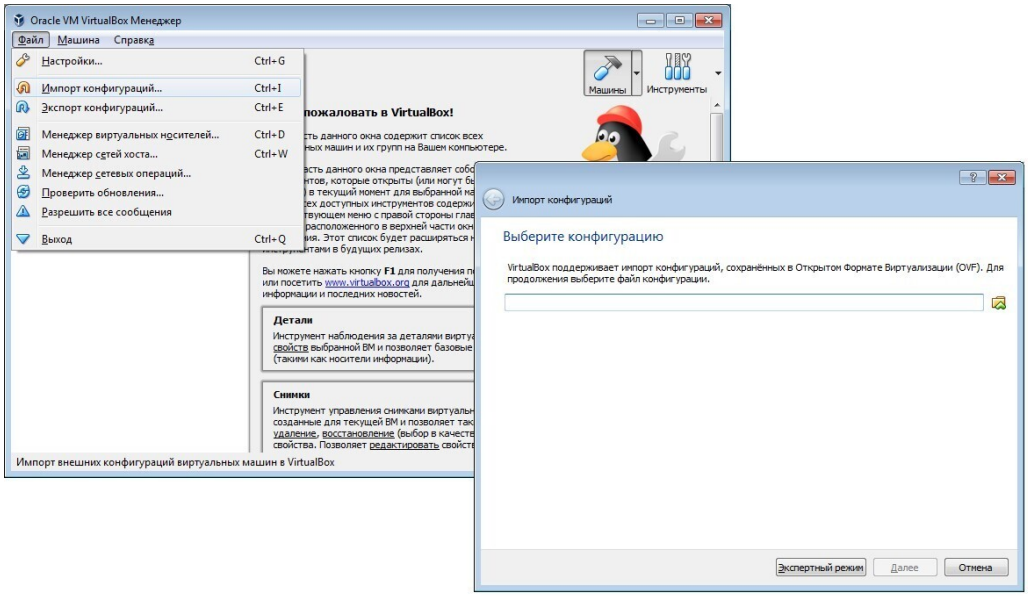
Hadoop

Cloudera QuickStart VM

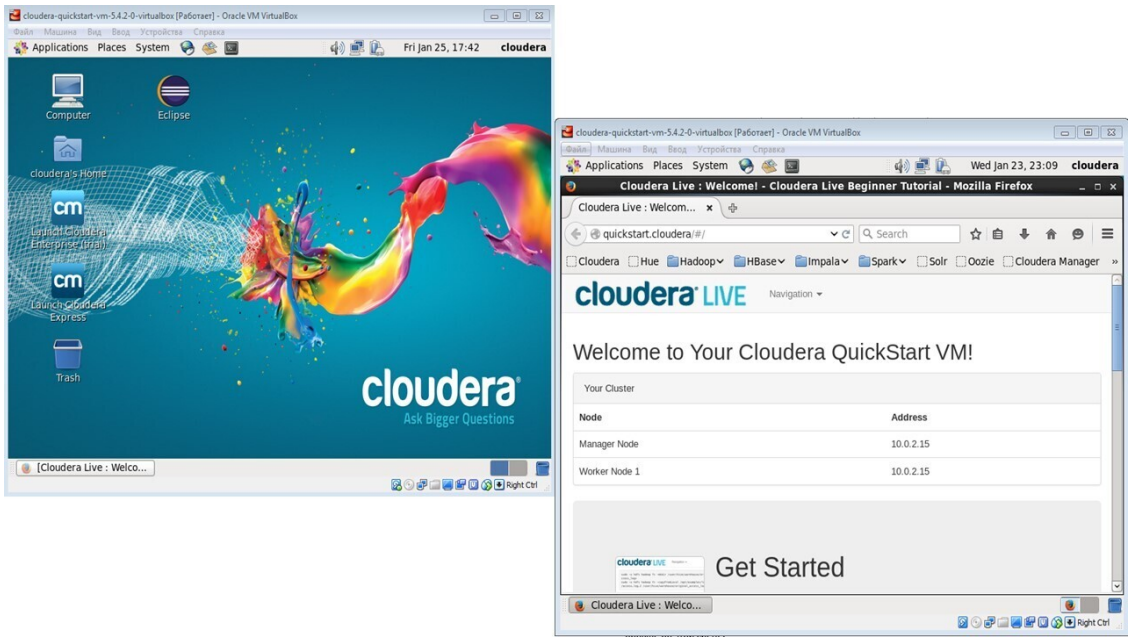
Для начала работы нам нужно скачать виртуальную машину Cloudera, позволяющую ознакомиться со стеком Cloudera Hadoop.



После скачивания и распаковки архива, запустим виртуальную машину.



Для этого в VirtualBox импортируем скачанную конфигурацию ovf.



После запуска виртуальной машины Cloudera QuickStart вы увидите рабочий стол и открытый браузер.

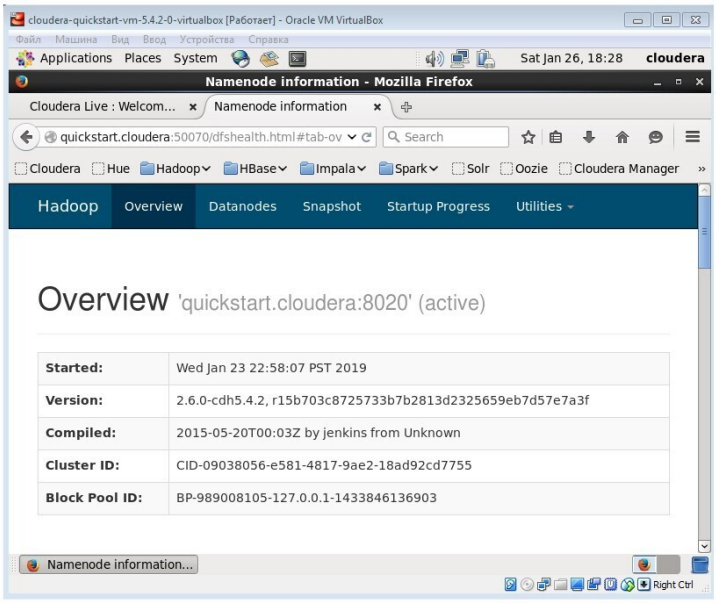
И если вы посмотрите на этот браузер, вы увидите, что здесь представлено несколько разных сервисов Cloudera.

Здесь есть Hue, Hadoop, HBase, Impala, Spark, и т. д.

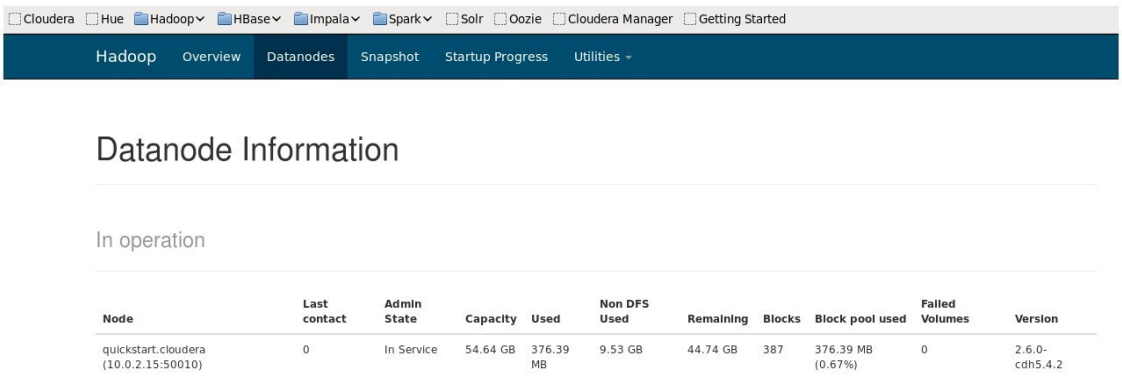
Это все приложения стека Cloudera Hadoop.

Здесь браузер выступает как клиент, для доступа к этим сервисам, запущенным на виртуальной машине, для доступа с помощью URL адреса.

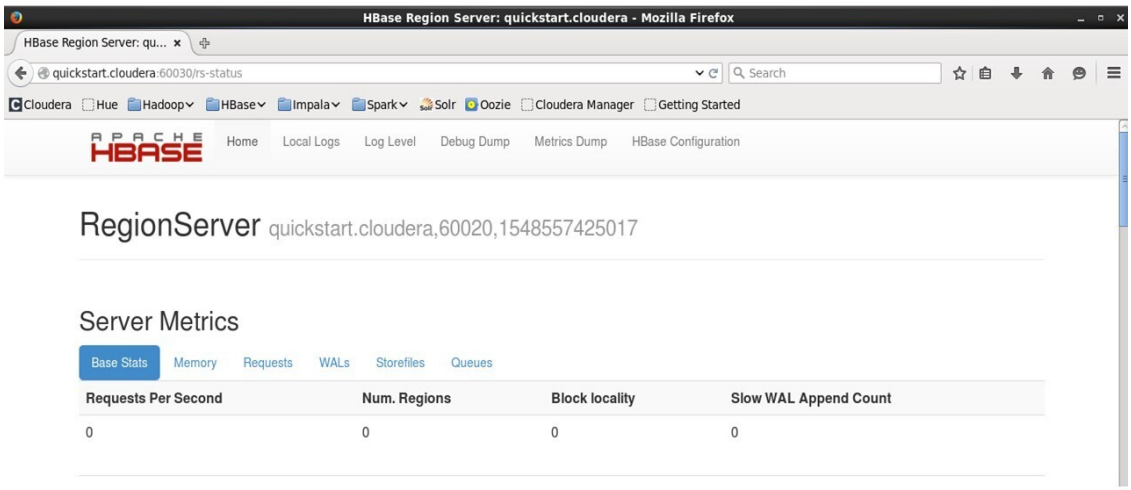
И давайте пройдемся по ним и узнаем, что они нам могут предоставить.



Откроем вкладку Overview NameNode Hadoop.
Здесь мы видим обзор нашего стека Hadoop.
Мы можем видеть, когда произошла инициализация этого стека.
И этот обзор дает нам полную сводку по всем конфигурациям, количеству файлов и т. д.



Давайте откроем вкладку Datanodes.
Этот сервис позволяет посмотреть на все имеющиеся у нас Datanodes.
Напомним, что кластер HDFS состоит из одного NameNode, главного сервера, который управляет пространством имен файловой системы и регулирует доступ клиентов к файлам.
И существуют узлы данных Datanodes, обычно по одному на узел кластера, которые управляют хранилищем, подключенным к узлам.

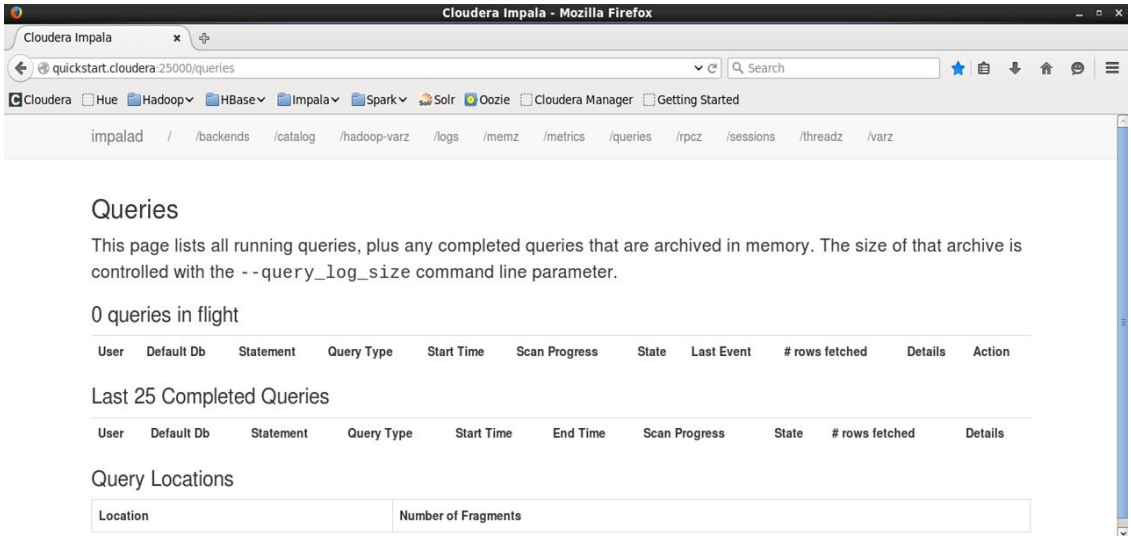


Откроем вкладку RegionServer HBase/

HBase – это столбцовое хранилище данных, которое хранит неструктурированные данные в файловой системе Hadoop.

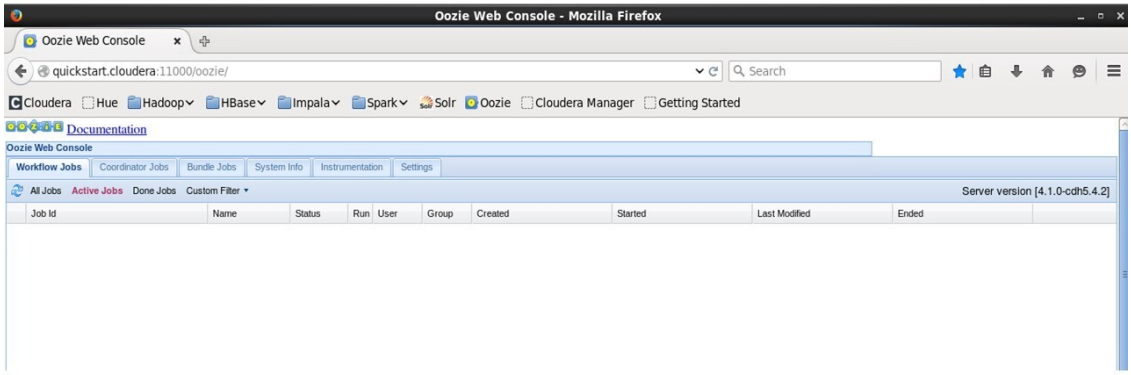
Здесь показывается количество запросов, которые делаются для чтения и записи в базу данных HBase.

И мы можем видеть все вызовы и задачи, которые были переданы в базу данных.



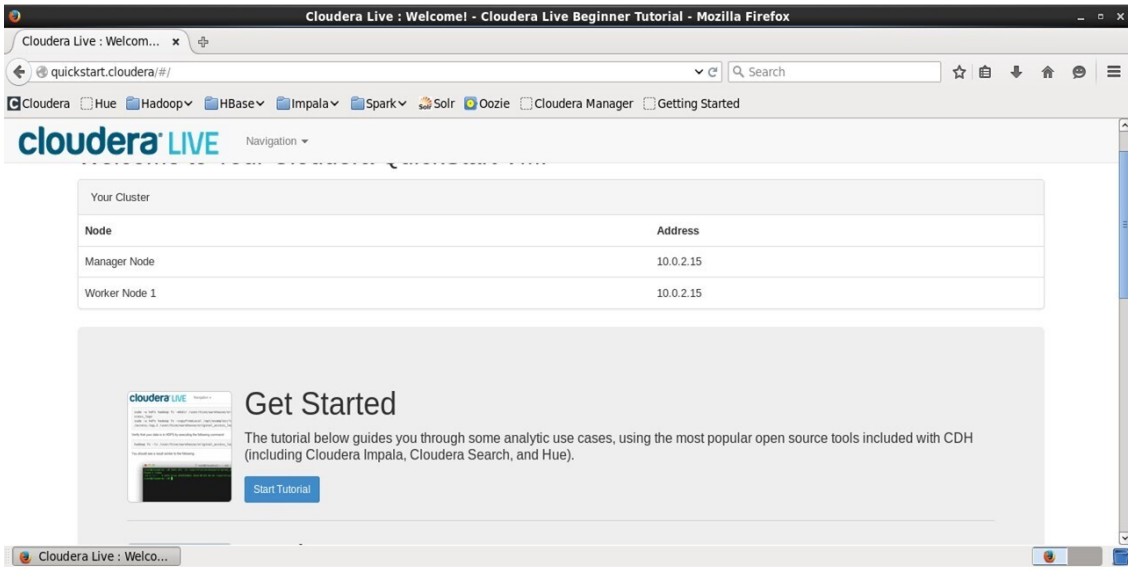
Impala позволяет нам отправлять высокопроизводительные SQL-подобные запросы к данным, хранящимся в HDFS.

И здесь мы можем посмотреть последние 25 выполненных запросов, мы можем посмотреть на запросы, которые происходят прямо сейчас, мы можем посмотреть на местоположения и фрагменты, к которым были отправлены эти запросы.



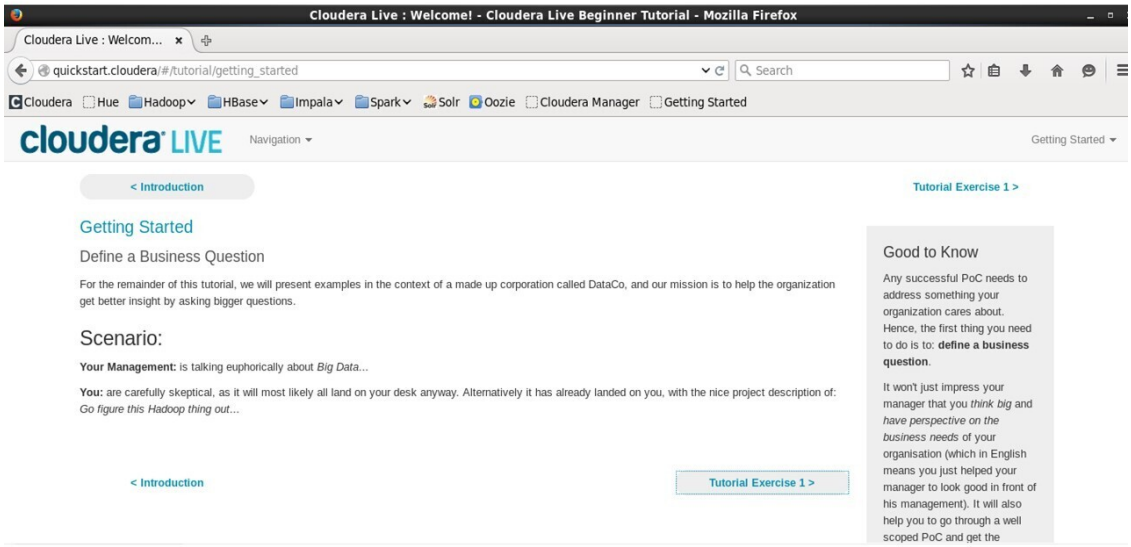
Далее, давайте откроем вкладку Oozie.

Здесь мы можем увидеть количество отправленных заданий, когда они были запущены, и т. д.

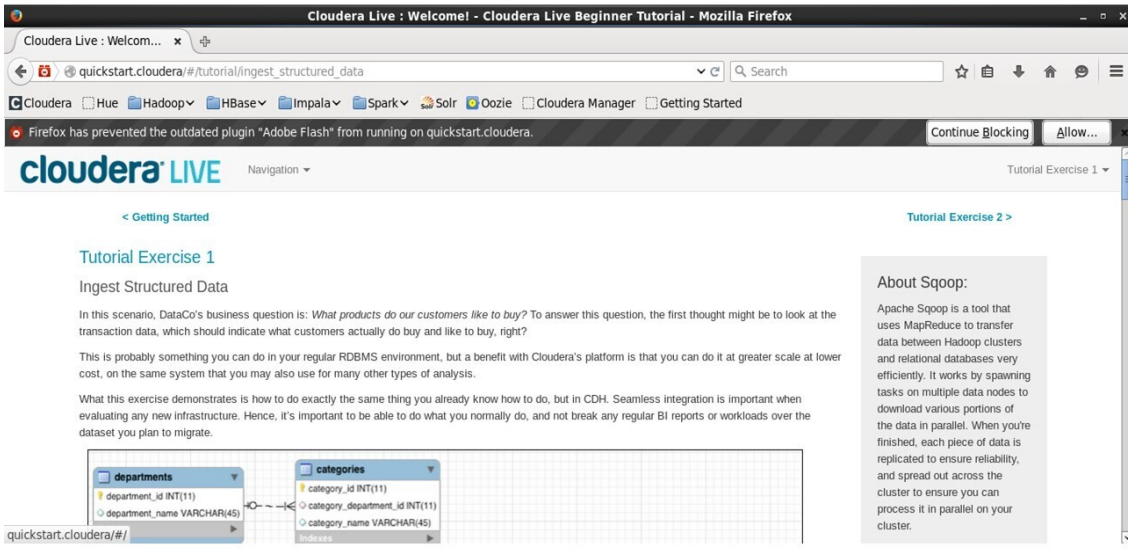


Теперь, давайте вернемся к исходной веб-странице, странице приветствия, и нажмем Start Tutorial.

И этот урок предложит нам введение в стек Cloudera.



На этой странице говорится, что в этом уроке представлены примеры в контексте созданной корпорации под названием DataCo.



И вопрос первого упражнения – какие продукты любят покупать клиенты корпорации?

Чтобы ответить на этот вопрос, вы можете посмотреть на данные транзакций, которые должны указать, что клиенты покупают.

Вероятно, вы можете это сделать в обычной реляционной базе данных.

Но преимущество платформы Cloudera заключается в том, что вы можете делать это в большем масштабе при меньших затратах.

Здесь сбоку есть информация о Scoop.

Это инструмент, который использует Map Reduce для эффективной передачи данных между кластером Hadoop и реляционной базой данных.

Он работает путем порождения нескольких узлов данных, чтобы загружать различные части данных параллельно.

И по окончании, каждый фрагмент данных будет реплицирован для обеспечения доступности и распределения по кластеру, чтобы вы могли параллельно обрабатывать данные в кластере.

И в платформу Cloudera включены две версии Sqoop.

Sqoop1 – это толстый клиент.

И Scoop2 состоит из центрального сервера и тонкого клиента, который вы можете использовать для подключения к серверу.

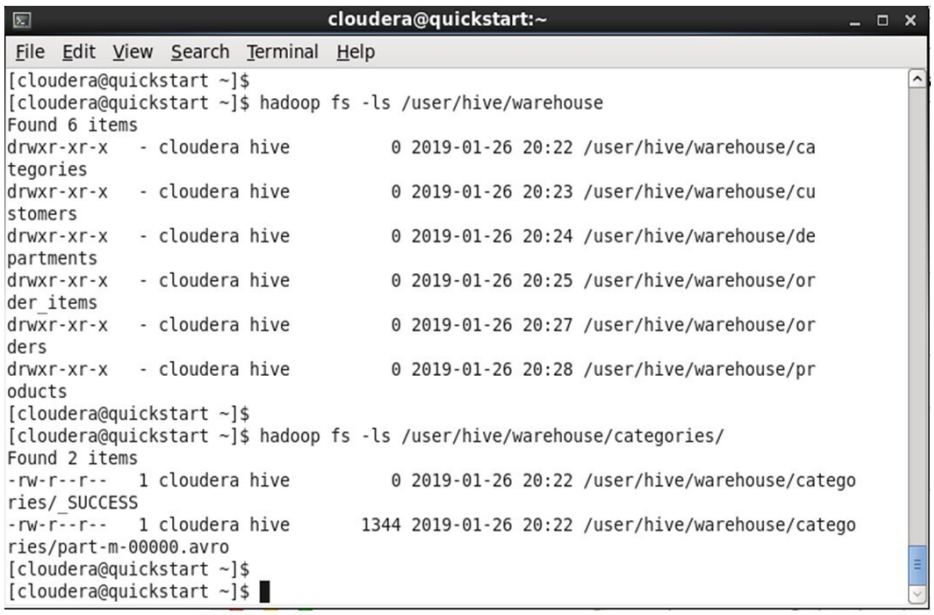
Ниже, вы можете посмотреть структуру таблицы данных.

Чтобы проанализировать данные транзакций на платформе Cloudera, нам нужно ввести их в распределенную файловую систему Hadoop (HDFS).

И нам нужен инструмент, который легко переносит структурированные данные из реляционной базы данных в HDFS, сохраняя при этом структуру.

И Apache Sqoop является этим инструментом.

С помощью Sqoop мы можем автоматически загружать данные из MySQL в HDFS, сохраняя при этом структуру.



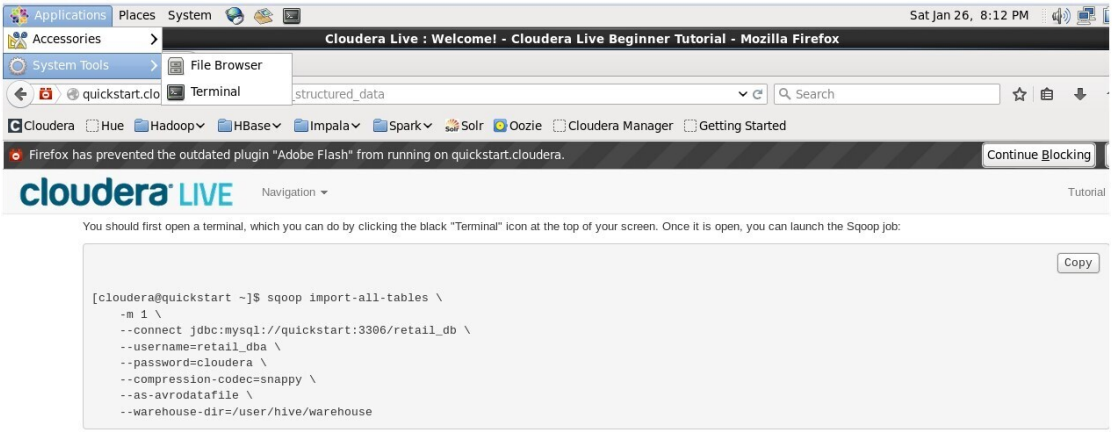
Вверху в меню откроем терминал, и запустим это задание Sqoop.

Эта команда запускает задания MapReduce для экспорта данных из базы данных MySQL и размещения этих файлов экспорта в формате Avro в HDFS.

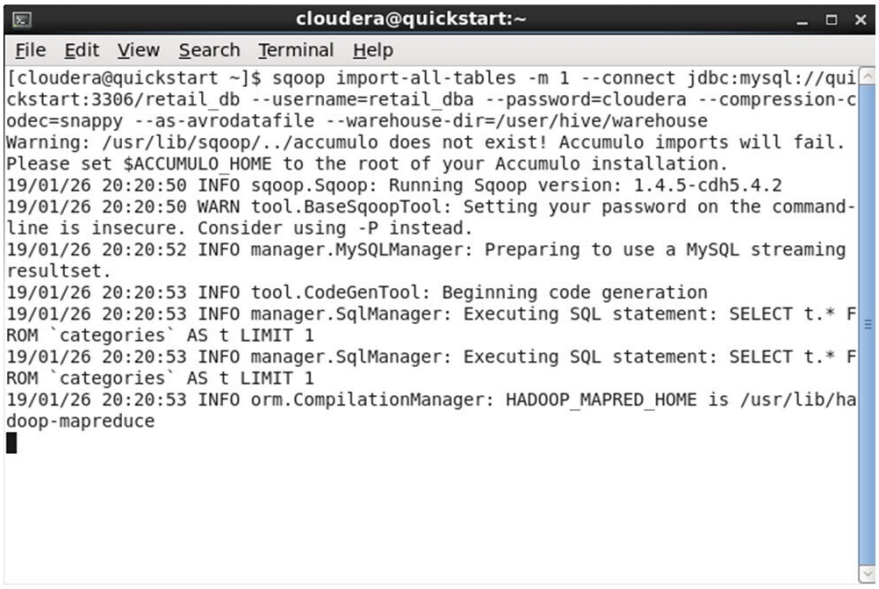
Эта команда также создает схему Avro, чтобы мы могли легко загрузить таблицы Hive для последующего использования в Impala.

Impala – это механизм аналитических запросов.

И Avro – это формат файлов, оптимизированный для Hadoop.

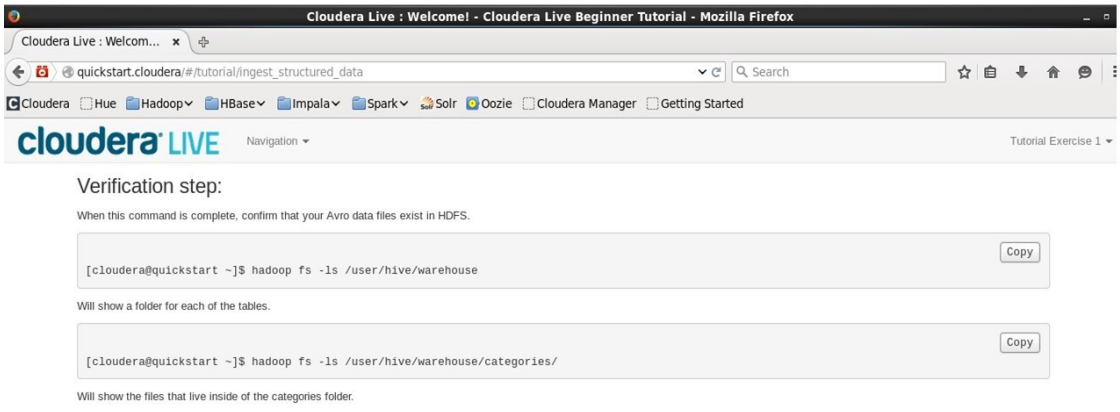


Таким образом, мы скопируем код и запустим команду в терминале.

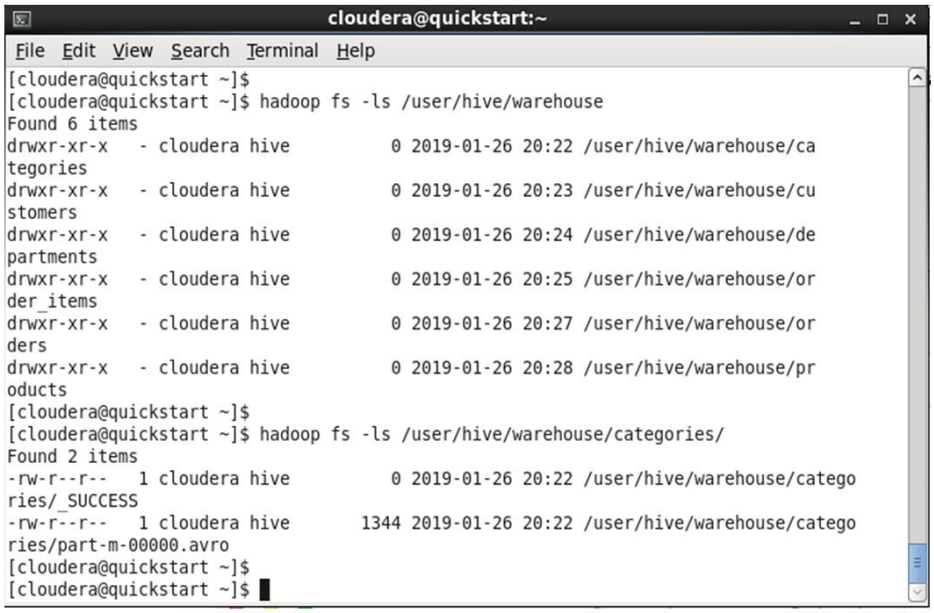


После выполнения задания, чтобы подтвердить, что данные существуют в HDFS, мы скопируем следующие команды в терминал.

Которые покажут папку для каждой из таблиц и покажут файлы в папке категорий.



Инструмент Sqoop также должен был создать файлы схемы для этих данных.



И эта команда должна показать avsc схемы для шести таблиц базы данных.

Таким образом, схемы и данные хранятся в отдельных файлах.

И схема применяется к данным, только когда данные запрашиваются.

И это то, что мы называем схемой на чтение.

Это дает гибкость при запросе данных с помощью SQL.

И это отличие от традиционных баз данных, которые требуют, чтобы у вас была четкая схема, прежде чем вводить в базу какие-либо данные. Здесь мы вводим данные, а уже потом применяем к ним схему.

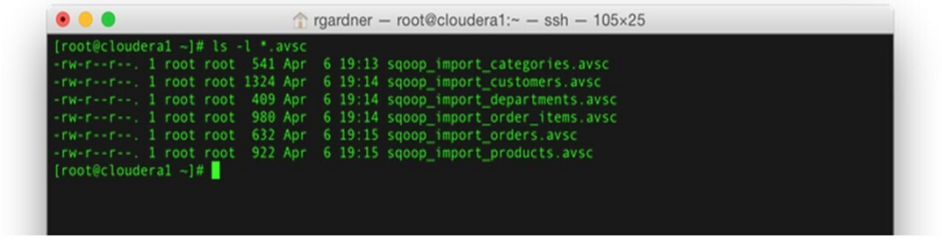
Sqoop should also have created schema files for this data in your home directory.

Avro schema files:

```
[cloudera@quickstart ~]$ ls -l *.avsc
```

Copy

Should show .avsc files for the six tables that were in our retail_db.



Теперь, так как мы хотим использовать Apache Hive, нам понадобятся файлы схем.

Поэтому с помощью этой команду скопируем их в HDFS, где Hive может легко получить к ним доступ.

Вы могли заметить, что мы импортировали данные в каталоги Hive.

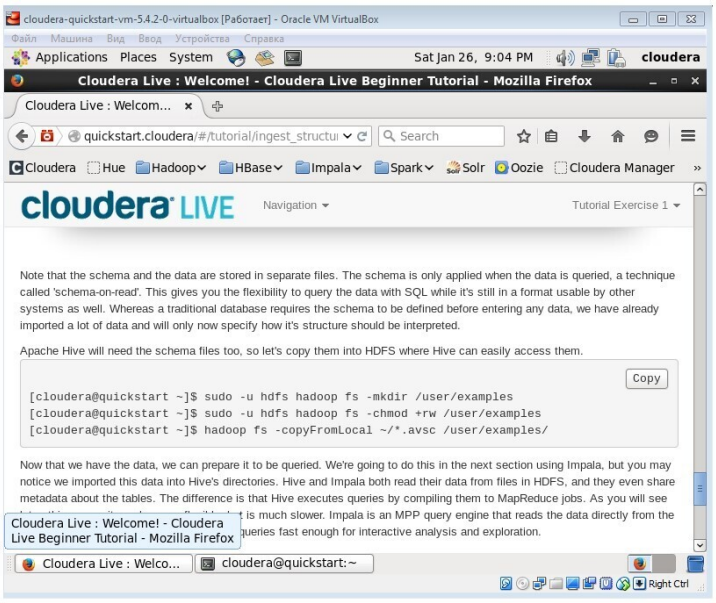
И Hive и Impala читают данные из файла в HDFS, и они даже обмениваются метаданными о таблицах.

Отличие состоит в том, что Hive выполняет запросы, компилируя их в задания MapReduce.

В то время как Impala является механизмом системы параллельных запросов, которые считывают данные непосредственно из самой файловой системы, в более быстром и интерактивном режиме.

Таким образом, мы загрузили данные с помощью Sqoop в HTFS, преобразовав их в формат Avro, и импортировали файлы схем, для их использования при запросе этих данных.

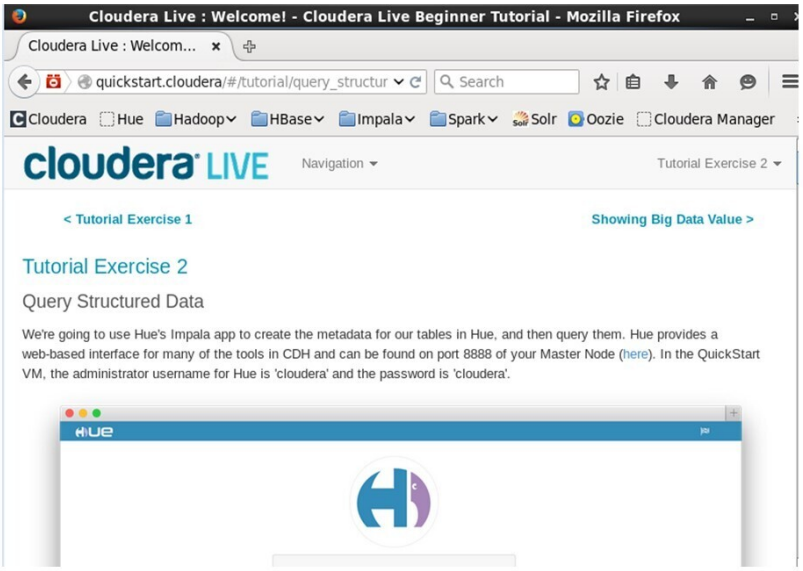
И теперь, давайте перейдем к следующему упражнению.



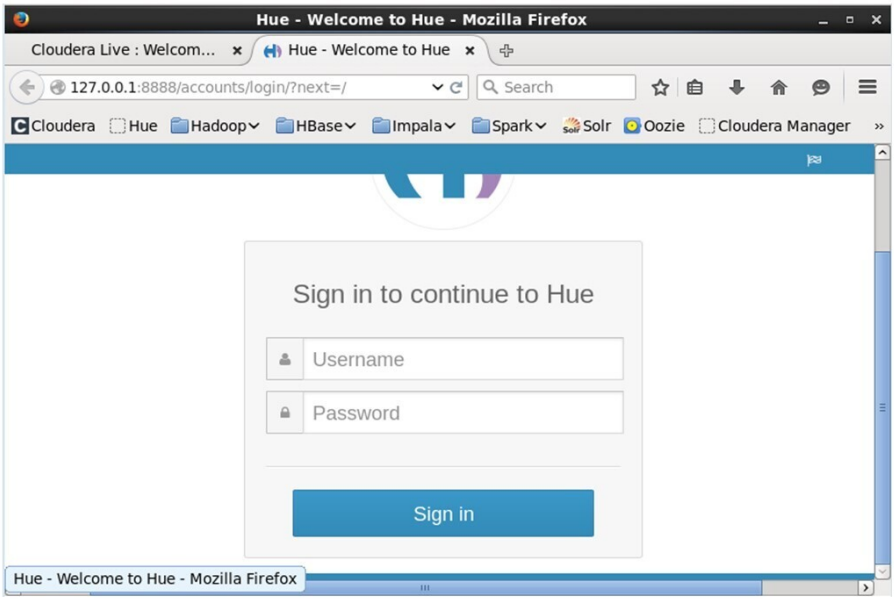
Здесь мы будем использовать Hue, приложение Impala, для создания метаданных для наших таблиц.

Мы создадим эти метаданные, а затем сделаем запрос к нашей таблице используя Hue.

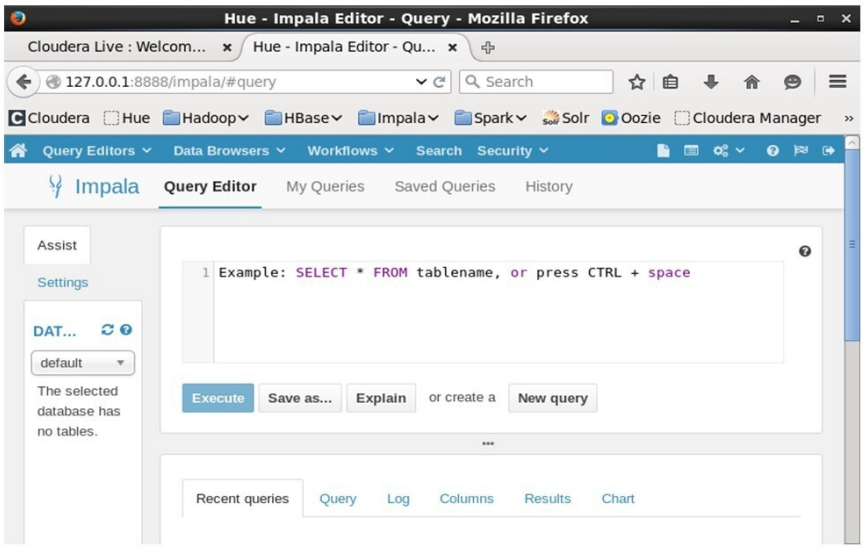
Hue предоставляет веб-интерфейс, который доступен на порту 8888.



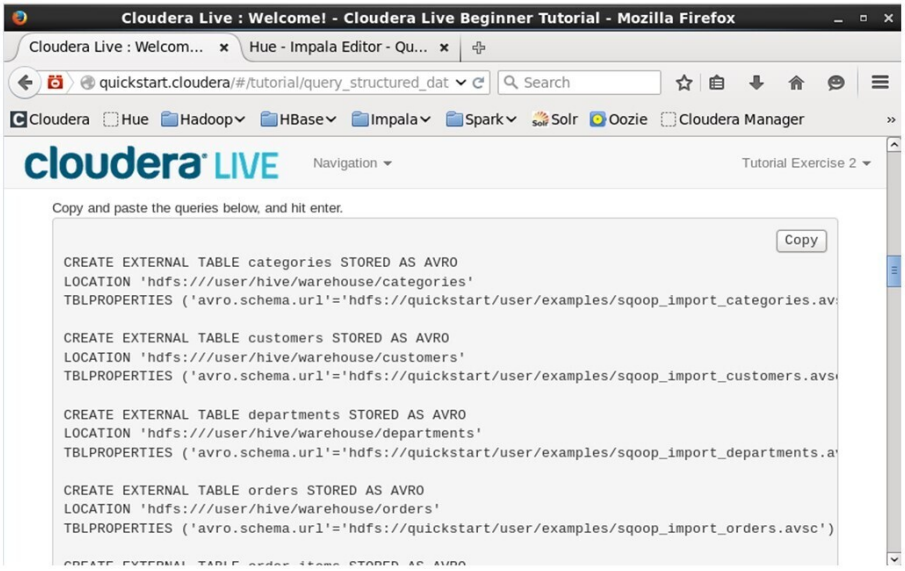
Чтобы войти в Hue, введем cloudera в качестве имени пользователя и пароля.



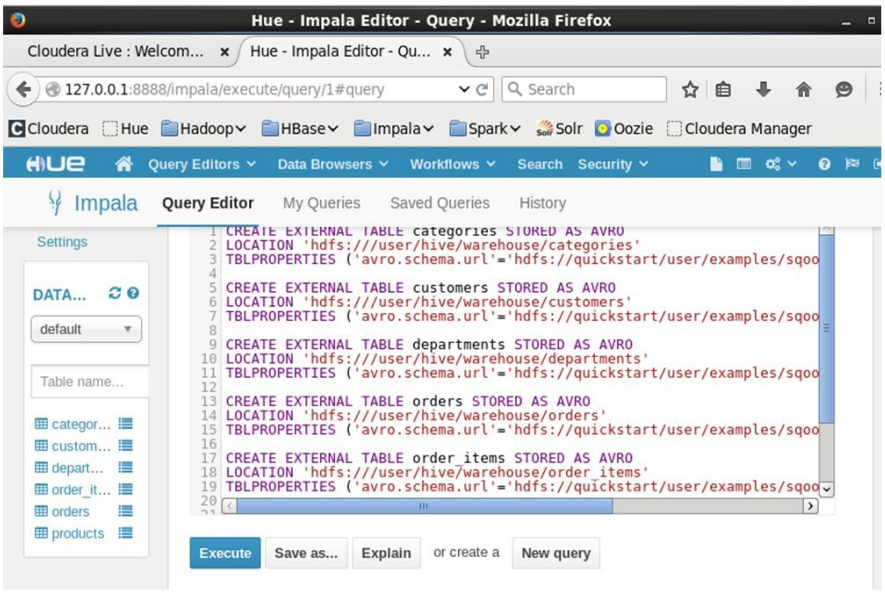
Далее в меню Query Editors откроем Impala.



Скопируем и вставим код, который создаст таблицы.



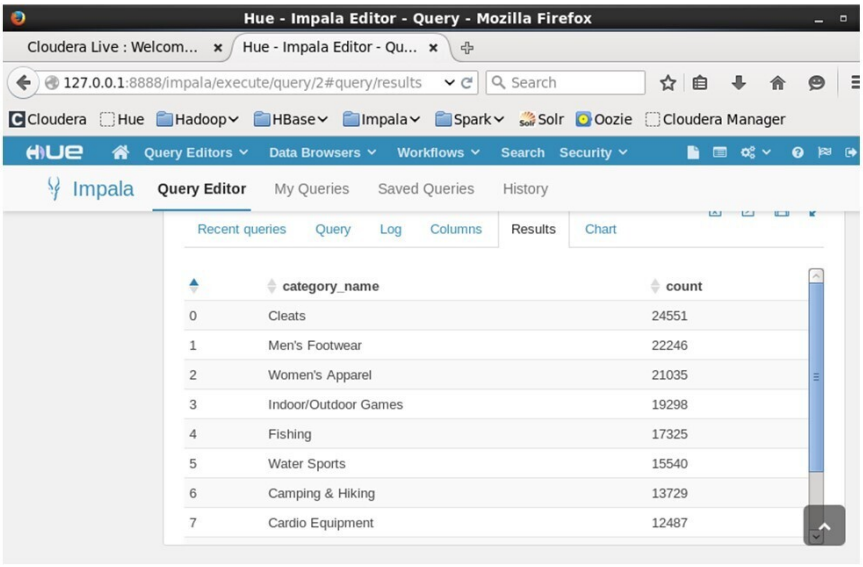
И обновим данные в левой колонке, чтобы увидеть созданные таблицы.



Теперь, когда данные доступны для запросов, мы можем ответить на вопрос, какие продукты покупают клиенты.

Для этого скопируем и вставим SQL запросы для расчета общей выручки по продукту и отображения 10 лучших продуктов, приносящих доход.

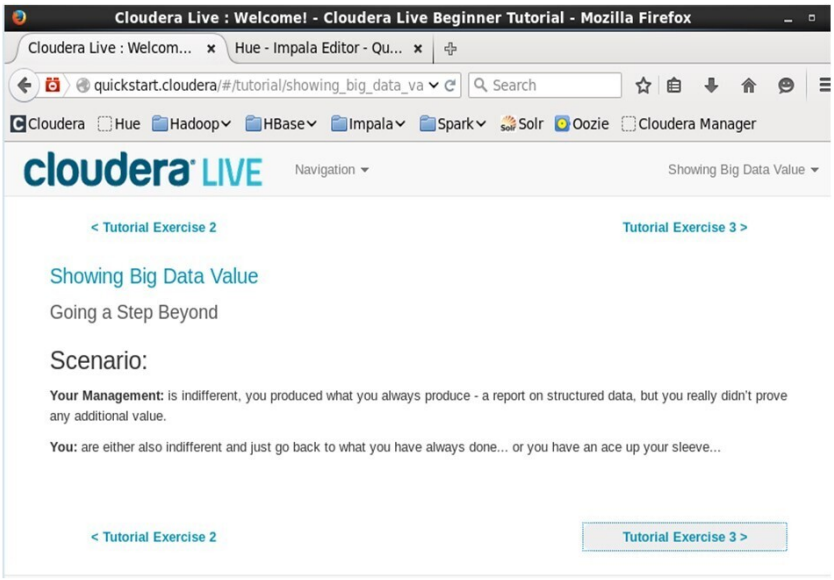
После выполнения, в Hue, мы увидим результаты запроса.



Таким образом мы узнали, как создавать и запрашивать таблицы с помощью Impala.

Теперь, давайте перейдем к следующему уроку.

И далее мы должны посмотреть, какие преимущества дает стек Cloudera по сравнению с традиционными системами.



Здесь мы попытаемся соотнести структурированные данные с неструктурированными данными и сможем ответить на вопрос – являются ли наиболее просматриваемые товары наиболее продаваемыми.

Конец ознакомительного фрагмента.