**ИНСТРУКЦИЯ ПОЛЬЗОВАТЕЛЯ**

**RT DATALAKE**

2021

ОГЛАВЛЕНИЕ

[1 ОПИСАНИЕ RT DATALAKE 4](#_Toc96429536)

[2 РАБОТА С HDFS 5](#_Toc96429537)

[2.1 Общая информация 5](#_Toc96429538)

[2.2 Shell-команды 6](#_Toc96429539)

[2.2.1 [classpath] Вывод пути к классу 7](#_Toc96429540)

[2.2.2 [fs] Взаимодействие с распределённой файловой системой 7](#_Toc96429541)

[2.2.3 [envvars] Отображение вычисленных переменных сред 18](#_Toc96429542)

[2.2.4 [fetchdt] Получение токена делегирования 18](#_Toc96429543)

[2.2.5 [fsck] Проверка файловой системы 19](#_Toc96429544)

[2.2.6 [getconf] Получение информации о настройках 20](#_Toc96429545)

[2.2.7 [groups] Получение информации о группах 20](#_Toc96429546)

[2.2.8 [httpfs] Запуск HTTP 20](#_Toc96429547)

[2.2.9 [lsSnapshottableDir] Получение списка каталогов снапшотов 21](#_Toc96429548)

[2.2.10 [jmxget] Получение дампа информации JMX из службы 21](#_Toc96429549)

[2.2.11 [oev] Редактор файлов 21](#_Toc96429550)

[2.2.12 [oiv] Редактор изображений 22](#_Toc96429551)

[2.2.13 [snapshotDiff] Определение разницы между снапшотами 23](#_Toc96429552)

[2.2.14 [version] Просмотр версии HDFS 23](#_Toc96429553)

[2.3 Secondary NameNode 23](#_Toc96429554)

[2.4 Checkpoint Node 24](#_Toc96429555)

[2.5 Backup Node 27](#_Toc96429556)

[2.6 Import Checkpoint 28](#_Toc96429557)

[2.7 Balancer 28](#_Toc96429558)

[2.8 Rack Awareness 30](#_Toc96429559)

[2.9 Безопасный режим 30](#_Toc96429560)

[2.10 Команда fsck 30](#_Toc96429561)

[2.11 Команда fetchdt 31](#_Toc96429562)

[2.12 Recovery Mode 32](#_Toc96429563)

[2.13 Upgrade и Rollback 32](#_Toc96429564)

[2.14 DataNode Hot Swap Drive 34](#_Toc96429565)

[2.15 Права доступа к файлам и безопасность 34](#_Toc96429566)

[2.16 Масштабируемость 34](#_Toc96429567)

[3 РАБОТА С YARN 36](#_Toc96429568)

[4 РАБОТА С HUE 38](#_Toc96429569)

[4.1 Общая информация 38](#_Toc96429570)

[4.1.1 Интерфейс 38](#_Toc96429571)

[4.1.2 Средство импорта 40](#_Toc96429572)

[4.1.3 Документы 41](#_Toc96429573)

[4.1.4 Смена языка 41](#_Toc96429574)

[4.2 Редактор (Editor) 41](#_Toc96429575)

[4.2.1 Концепции 42](#_Toc96429576)

[4.2.2 SQL 46](#_Toc96429577)

[4.2.3 Задания 47](#_Toc96429578)

[4.3 Дашборды (Dashboards) 49](#_Toc96429579)

[4.3.1 Концепции 49](#_Toc96429580)

[4.3.2 Базы данных 50](#_Toc96429581)

[4.4 Браузеры (Browsers) 50](#_Toc96429582)

[4.4.1 Таблицы 50](#_Toc96429583)

[4.4.2 Файловые системы 50](#_Toc96429584)

[4.4.3 Файлы и каталоги 54](#_Toc96429585)

[4.4.4 Загрузка zip-архивов 55](#_Toc96429586)

[4.4.5 Корзина для мусора 55](#_Toc96429587)

[4.4.6 Смена владельца, группы или разрешений 55](#_Toc96429588)

[4.4.7 Просмотр и редактирование файлов 56](#_Toc96429589)

[4.4.8 Разрешения Sentry 57](#_Toc96429590)

[4.4.9 Задания 57](#_Toc96429591)

[4.5 Планировщик задач (Scheduler) 59](#_Toc96429592)

[4.5.1 Workflows 59](#_Toc96429593)

[4.5.2 Расписание 60](#_Toc96429594)

[4.5.3 Bundle 62](#_Toc96429595)

ОПИСАНИЕ RT DATALAKE

**RT DataLake** представляет собой масштабируемую платформу с открытым исходным кодом, предназначенную для хранения, обработки и анализа больших объёмов данных.

Платформа спроектирована так, чтобы обеспечить быструю, лёгкую и незатратную загрузку данных различного формата из большого количества источников.

Платформа представляет собой комплекс сервисов, которые обеспечивают не только хранение и обработку данных, но и управление, безопасность и операции.

Платформа включает в себя Hadoop, который состоит из **MapReduce**, Hadoop Distributed File System (**HDFS**) и Yet Another Resource Negotiator (**YARN**), а также **Hive**, **Knox**.

Все компоненты интегрированы друг с другом и протестированы на совместимость.

РАБОТА С HDFS

Общая информация

**HDFS** — это основное распределённое хранилище, используемое приложениями **Hadoop**. Кластер **HDFS** в основном состоит из NameNode, который управляет метаданными файловой системы, и DataNode, в которых хранятся фактические данные. Основные взаимодействия между NameNode, DataNodes и клиентами следующие: клиенты связываются с NameNode для изменения метаданных файла и выполняют фактический ввод-вывод файла непосредственно с помощью DataNodes.

Далее приведены некоторые из основных функций, которые могут представлять интерес для пользователей:

**Hadoop**, включая **HDFS**, хорошо подходит для распределённого хранения и распределенной обработки: он отказоустойчив, масштабируем и чрезвычайно прост в расширении. **MapReduce**, хорошо известный своей простотой и применимостью для распределённых приложений, является неотъемлемой частью **Hadoop**.

**HDFS** легко настраивается с конфигурацией по умолчанию, которая подходит для многих установок. Но при этом в большинстве случаев в зависимости от специфики решаемых задач конфигурацию необходимо настраивать.

**Hadoop** написан на Java и поддерживается на всех основных платформах.

**Hadoop** поддерживает shell-подобные команды для непосредственного взаимодействия с **HDFS**.

NameNode и Datanodes имеют встроенные веб-интерфейсы, которые позволяют легко проверять текущее состояние кластера.

Новые функции и улучшения регулярно внедряются в **HDFS**. Ниже приведен перечень полезных функций **HDFS**:

Права доступа к файлам и аутентификация;

Rack awareness — учёт физического местоположения узла при планировании задач и выделении хранилища;

Safemode — административный режим для сопровождения;

fsck — утилита для диагностики работоспособности файловой системы, поиска отсутствующих файлов или блоков;

fetchdt — утилита для извлечения DelegationToken и сохранения его в файле в локальной системе;

Balancer — инструмент для балансировки кластера, когда данные неравномерно распределены между DataNodes;

Upgrade и rollback — после обновления программного обеспечения можно выполнить откат до состояния **HDFS** перед обновлением в случае непредвиденных проблем;

Secondary NameNode — выполняет периодические checkpoints пространства имен и помогает поддерживать размер файла, содержащего журнал изменений **HDFS**, в определённых пределах в NameNode;

Checkpoint node — выполняет периодические checkpoints пространства имен и помогает минимизировать размер журнала с изменениями **HDFS**, хранящийся в NameNode. NameNode позволяет использовать несколько узлов Checkpoint одновременно, если в системе не зарегистрированы резервные узлы;

Backup node — расширение узла Checkpoint. В дополнение к checkpointing он также получает поток изменений от NameNode и поддерживает свою собственную копию пространства имён в памяти, которая всегда синхронизируется с активным состоянием пространства имен NameNode. Только один резервный узел может быть зарегистрирован с помощью NameNode одновременно.

NameNode и DataNode запускают внутренний веб-сервер для отображения базовой информации о текущем состоянии кластера. В конфигурации по умолчанию главная страница NameNode находится по адресу <http://namenode-name:9870/>. На ней перечислены узлы данных в кластере и основные статистические данные кластера. Веб-интерфейс также можно использовать для просмотра файловой системы, используя ссылку Browse the file system на главной странице NameNode.

**HDFS** имеет один NameNode для каждого кластера. В настоящее время общий объём памяти, доступный на NameNode, является основным ограничением масштабируемости. В очень больших кластерах увеличение среднего размера файлов, хранящихся в **HDFS**, помогает увеличить размер кластера без увеличения требований к памяти для NameNode.

Shell-команды

Для работы с **HDFS** через командную строку необходимо использовать нативный shell-клиент для **HDFS**.

Для использования shell-команд, с помощью которых можно взаимодействовать с **HDFS**, необходимо в терминале запустить скрипт bin/hdfs.

При запуске скрипта bin/hdfs без аргументов будет выведен перечень всех возможных команд.

Чтобы запустить конкретную команду, необходимо воспользоваться синтаксисом и перечнем команд, приведёнными ниже.

Общий синтаксис команд:

|  |
| --- |
| hadoop [SHELL\_OPTIONS] COMMAND [GENERIC\_OPTIONS] [COMMAND\_OPTIONS] |

Описание параметров и классов команд представлено ниже.

Таблица — Параметры и классы

| COMMAND\_OPTIONS | Описание |
| --- | --- |
| SHELL\_OPTIONS | Общий набор shell-параметров. |
| GENERIC\_OPTIONS | Общий набор параметров, поддерживаемый несколькими командами. |
| COMMAND COMMAND\_OPTIONS | В следующих пунктах раздела описаны различные пользовательские команды с их параметрами. |

[classpath] Вывод пути к классу

Команда classpath выводит путь к классу, необходимый для получения jar-файла Hadoop и необходимых библиотек.

Если вызывается без аргументов, выводит путь к классам, установленных командными скриптами, который будет содержать символы в записях пути к классам. Дополнительные параметры распечатывают путь к классам после раскрытия подстановочных знаков или записывают путь к классам в манифест файла jar. Последнее полезно в средах, где нельзя использовать подстановочные знаки, а расширенный путь к классам превышает максимальную поддерживаемую длину командной строки.

Синтаксис команды:

|  |
| --- |
| hadoop classpath [--glob |--jar <path> |-h |--help] |

В команде могут быть использованы следующие параметры команды COMMAND\_OPTION:

--glob — развернуть подстановочные знаки;

--jar path — записать путь к классам как манифест в jar с именем path;

-h, --help — распечатать справку.

[fs] Взаимодействие с распределённой файловой системой

Команда fs предназначена для работы с распределённой файловой системой **HDFS** через командную строку.

Синтаксис команды:

|  |
| --- |
| hadoop fs [COMMAND [COMMAND\_OPTIONS]] |

Команды и их параметры COMMAND\_OPTIONS представлены в разделах ниже.

[appendToFile] Добавление источников

Команда appendToFile позволяет добавить один или несколько источников из локальной файловой системы в целевую файловую систему.

Синтаксис команды:

|  |
| --- |
| hadoop fs -appendToFile <localsrc> ... <dst> |

Пример команды:

|  |
| --- |
| hadoop fs -appendToFile localfile1 localfile2 /dir |

Команда добавит файлы localfile1 и localfile2 в каталог HDFS dir.

[cat] Чтение файлов небольших объёмов

Команда cat выводит источник в стандартный поток вывода — stdout.

Синтаксис команды:

|  |
| --- |
| hadoop fs –cat <URI> |

Пример команды:

|  |
| --- |
| hadoop fs -cat /dir/file.txt |

Команда выведет файл в stdout.

Если файл, выводимый в stdout, содержит большой объём данных, то процедура будет выполняться долго.

Пример команды для вывода 100 строк из файла:

|  |
| --- |
| hadoop fs -cat /dir/file.txt | head -n 100 |

Команда выведет только 100 первых строк файла в stdout.

[chgrp] Изменение группы-владельца файла

Команда chgrp позволяет изменять групповую ассоциацию файлов. Пользователь должен обладать привилегиями — быть владельцем файлов или суперпользователем.

Синтаксис команды:

|  |
| --- |
| hadoop fs -chgrp [-r] GROUP URI [URI ...] |

Пример команды:

|  |
| --- |
| hadoop fs -chgrp mygroup /dir/file.txt |

Команда сменит для файла file.txt группу-владельца на mygroup.

Чтобы рекурсивно сменить группу-владельца для всего каталога (с подкаталогами), необходимо использовать параметр -r.

Пример команды для рекурсивной смены владельца:

|  |
| --- |
| hadoop fs -chgrp –r mygroup /dir |

Команда сменит для каталога dir и всех его подкаталогов группу-владельца на mygroup.

[chmod] Изменение прав доступа к файлу

Команда chmod позволяет изменять права доступа к файлам. Пользователь должен обладать привилегиями — быть владельцем файлов или суперпользователем.

Синтаксис команды:

|  |
| --- |
| hadoop fs -chmod [-r] <MODE[,MODE]... | OCTALMODE> URI [URI ...] |

Пример команды:

|  |
| --- |
| hadoop fs -chmod 777 /dir |

Команда изменит для каталога dir права rwx (чтение, выполнение и запись) для всех, кто имеет доступ к серверу (владельца, группы, остальных).

Чтобы рекурсивно изменить права доступа для всего каталога (с подкаталогами), необходимо использовать параметр -r.

Пример команды для рекурсивного изменения прав доступа:

|  |
| --- |
| hadoop fs -chmod –r 766 /dir |

Команда изменит для каталога dir и всех его подкаталогов права доступа rwx для владельца, rw (чтение, запись) для групп и остальных.

[chown] Изменение владельца файла

Команда chown позволяет сменить владельца файла. Пользователь должен обладать привилегиями — быть суперпользователем.

Синтаксис команды:

|  |
| --- |
| hadoop fs -chown [-R] [OWNER][:[GROUP]] URI [URI ] |

Пример команды:

|  |
| --- |
| hadoop fs -chown hduser /dir |

Команда сменит для каталога dir владельца на hduser.

Чтобы рекурсивно cменить владельца для всего каталога (с подкаталогами), необходимо использовать параметр -r.

Пример команды для рекурсивной смены владельца:

|  |
| --- |
| hadoop fs -chown –r hduser /dir |

Команда сменит для каталога dir и всех его подкаталогов владельца на hduser.

[copyFromLocal] Копирование локального файла в HDFS

Команда copyFromLocal позволяет копировать локальный файл в HDFS.

Синтаксис команды:

|  |
| --- |
| hadoop fs -copyFromLocal <локальный src> URI |

Пример команды:

|  |
| --- |
| hadoop fs -copyFromLocal localfile /dir/file |

Команда скопирует localfile из локального каталога в каталог HDFS dir с переименованием файла в file.

[copyToLocal] Копирование файла из HDFS в локальную файловую систему

Команда copyToLocal позволяет копировать файл из HDFS в локальную файловую систему.

Синтаксис команды:

|  |
| --- |
| hadoop fs -copyToLocal [-ignorecrc] [-crc] URI <локальное местоназначение> |

Пример команды:

|  |
| --- |
| hadoop fs -copyToLocal /dir/file localfile |

Команда скопирует file из каталога HDFS dir в локальную файловую систему с переименованием файла в localfile.

[count] Подсчёт количества файлов

Команда count позволяет подсчитать количество каталогов, файлов и байтов по путям, которые соответствуют указанному шаблону файла

Синтаксис команды:

|  |
| --- |
| hadoop fs -count [-q] <paths> |

Пример команды:

|  |
| --- |
| hadoop fs -count /dir |

Команда подсчитает количество файлов в каталоге dir.

Чтобы вывести столбцы QUOTA, REMAINING\_QUATA, SPACE\_QUOTA, REMAINING\_SPACE\_QUOTA, DIR\_COUNT, FILE\_COUNT, CONTENT\_SIZE, FILE\_NAME, необходимо использовать параметр -q.

Пример команды:

|  |
| --- |
| hadoop fs -count -q /dir |

Команда выведет информацию о каталоге dir по вышеописанным столбцам.

[cp] Копирование файлов небольших объёмов

Команда cp позволяет скопировать файлы из одного места в другое.

Если копируемый файл содержит большой объём данных, то процедура будет выполняться долго.

Синтаксис команды:

|  |
| --- |
| hadoop fs -cp [-f] URI [URI ...] <местоназначение> |

Пример команды:

|  |
| --- |
| hadoop fs -cp /dir/file1 /otherDir/file2 |

Команда скопирует file1 из каталога dir в каталог otherDir с переименованием файла в file2.

[du] Вывод размеров файлов

Команда du позволяет вывести информацию о размерах файлов и каталогов, содержащихся в данном каталоге, или длину файла, если это всего лишь файл.

Синтаксис команды:

|  |
| --- |
| hadoop fs -du [-s] [-h] URI [URI ...] |

Пример команды:

|  |
| --- |
| hadoop fs -du /user /dir |

Команда выведет размеры каталогов user и dir.

Чтобы вывести совокупную сводку длин файлов, необходимо использовать параметр -s.

Пример команды для вывода общего размера каталога или файла:

|  |
| --- |
| hadoop fs -du -s /dir |

Команда выведет информацию об общем размере всех файлов каталога dir.

Чтобы вывести размеры файлов в удобном для чтения виде (в Мб), необходимо использовать параметр -h.

Пример команды для вывода размеров файлов в удобочитаемом виде:

|  |
| --- |
| hadoop fs -du -h /user |

Команда выведет информацию о размерах каждого файла каталога user в удобном виде.

[dus] Вывод совокупного размера файлов

Команда dus позволяет вывести информацию об общем размере каталога или файла. Является альтернативой команды hdfs dfs -du -s URI.

Синтаксис команды:

|  |
| --- |
| hadoop fs -dus URI [URI ...] |

Пример команды:

|  |
| --- |
| hadoop fs -dus /dir |

Команда выведет информацию об общем размере всех файлов каталога dir.

[expunge] Очистка корзины

Команда expunge позволяет очистить корзину.

Когда файл удаляется в корзину (команда rm), он не сразу удаляется из **HDFS**, а переименовывается в файл в каталоге /trash. Пока файл остается там, его можно восстановить.

Синтаксис команды:

|  |
| --- |
| hadoop fs -expunge |

Пример команды:

|  |
| --- |
| hadoop fs -expunge |

Команда очистит содержимое корзины.

[help] Вывод всех команд или информации по конкретной команде

Команда help позволяет получить список всех команд в скрипте **HDFS**.

Синтаксис команды:

|  |
| --- |
| hadoop fs -help |

Команда выведет список возможных команд.

Если в параметрах команды help указать название конкретной команды, то команда выведет информацию по введённой команде.

Синтаксис команды:

|  |
| --- |
| hadoop s fs –help <название команды> |

Пример команды:

|  |
| --- |
| hadoop fs –help ls |

Команда выведет информацию по команде ls.

[get] Копирование файла из HDFS в локальную файловую систему

Команда get позволяет копировать файл из HDFS в локальную файловую систему.

Синтаксис команды:

|  |
| --- |
| hadoop fs -get [-ignorecrc] [-crc] <src> <localdst> |

Пример команды:

|  |
| --- |
| hadoop fs -get /dir/file localfile |

Команда скопирует file из каталога HDFS dir в локальную файловую систему с переименованием файла в localfile.

Чтобы скопировать файлы, которые не прошли проверку кода циклической избыточности (CRC, Cyclic Redundancy Code), необходимо использовать параметр -ignorecrc.

Пример команды:

|  |
| --- |
| hadoop fs -get –ignorecrc /dir/file localfile |

CRC является распространенным методом обнаружения ошибок передачи данных. Файлы контрольной суммы CRC имеют расширение. crc и используются для проверки целостности данных другого файла. Чтобы скопировать эти файлы, необходимо указать параметр -crc.

Пример команды:

|  |
| --- |
| hadoop fs -get –crc /dir/file localfile |

[getfacl] Вывод списков контроля доступа файлов

Команда getfacl позволяет вывести списки контроля доступа (ACL, Access Control List) файлов и каталогов.

Синтаксис команды:

|  |
| --- |
| hadoop fs -getfacl [-R] <path> |

Пример команды:

|  |
| --- |
| hadoop fs -getfacl /dir |

Команда выведет списки контрля доступа к каталогу dir.

Чтобы рекурсивно вывести списки контроля доступа для всего каталога (с подкаталогами), необходимо использовать параметр -r.

Пример команды для рекурсивного вывода:

|  |
| --- |
| hadoop fs -getfacl -r /dir |

Команда dsdtltn ACL для каталога dir и всех его подкаталогов.

[getmerge] Объединение файлов и запись в локальную файловую систему

Команда getmerge позволяет объединять файлы в src и записывать результат в указанный локальный файл назначения.

Чтобы добавить символ новой строки в конце каждого файла, необходимо указать параметр addnl.

Синтаксис команды:

|  |
| --- |
| hadoop fs -getmerge <src> <localdst> [addnl] |

Пример команды:

|  |
| --- |
| hadoop fs -getmerge /dir/ ./file.txt |

[ls] Вывод списка всех файлов

Команда ls используется для вывода списка (листинга) всех каталогов и статистики файлов.

Синтаксис команды:

|  |
| --- |
| hadoop fs -ls <args> |

Пример использования команды:

|  |
| --- |
| hadoop fs -ls / |

Команда выведет следующую информацию по корневому каталогу **HDFS**:

количество элементов в каталоге;

файлы, которые содержатся в каталоге;

статистику с указанием прав доступа и типа файла (подкаталога), фактора репликации (у каталога — нет), принадлежности файла (пользователю / группе), времени создания.

[lsr] Вывод рекурсивного списка всех файлов

Команда lsr используется для вывода рекурсивного списка (листинга) всех каталогов (и подкаталогов) и статистики файлов. Является рекурсивной версией команды ls и используется в случае необходимости вывода иерархии каталогов.

Синтаксис команды:

|  |
| --- |
| hadoop fs -lsr <args> |

Пример использования команды:

|  |
| --- |
| hadoop fs -lsr / |

[mkdir] Создание каталога

Команда mkdir создает новый каталог.

Синтаксис команды:

|  |
| --- |
| hadoop fs -mkdir [-p] <paths> |

Пример использования команды:

|  |
| --- |
| hadoop fs -mkdir /data/new\_path |

Команда создаст каталог new\_path по пути /data.

[moveFromLocal] Перемещение локального файла в HDFS

Команда moveFromLocal позволяет переместить локальный файл в HDFS.

Синтаксис команды:

|  |
| --- |
| hadoop fs -moveFromLocal <localsrc> <dst> |

Пример команды:

|  |
| --- |
| hadoop fs -moveFromLocal localfile /dir/file |

Команда скопирует localfile из локального каталога в каталог HDFS dir с переименованием файла в file, а затем удалит исходный файл localfile.

[moveToLocal] Перемещение файла из HDFS в локальную файловую систему

Команда moveToLocal позволяет переместить файл из HDFS в локальную файловую систему.

Синтаксис команды:

|  |
| --- |
| hadoop fs -moveToLocal [-crc] <src> <dst> |

Пример команды:

|  |
| --- |
| hadoop fs -moveToLocal /dir/file localfile |

Команда скопирует file из каталога HDFS dir в локальную файловую систему с переименованием файла в localfile, а затем удалит исходный файл.

[mv] Перемещение файла

Команда mv предназначена для перемещения файла из одного места в другое.

Физически перемещение данных не происходит. Происходит указание Namenode, что файл теперь будет храниться в другом каталоге.

Синтаксис команды:

|  |
| --- |
| hadoop fs -mv URI [URI ...] <dest> |

Пример использования команды:

|  |
| --- |
| hadoop fs –mv /dir/file1 /otherDir |

Команда переместит file1 из каталога dir в каталог otherDir.

[put] Копирование локального файла в HDFS

Команда put позволяет копировать локальный файл в HDFS.

Синтаксис команды:

|  |
| --- |
| hadoop fs -put <локальный src> ... <местоназначение> |

Пример команды:

|  |
| --- |
| hadoop fs -put localfile /dir/file |

Команда скопирует localfile из локального каталога в каталог HDFS dir с переименованием файла в file.

[rm] Удаление файла в корзину

Команда rm удаляет файл в корзину.

При случайном удалении большого файла его очень трудно восстановить. Поэтому файл сначала помещается в корзину. Корзина — специальный для каждого пользователя каталог. Файлы из него можно удалить или скопировать на прежнее место. Автоматическая очистка корзины задана в параметрах HDFS. Если нужно освободить место в HDFS, то можно удалить файл из корзины.

Синтаксис команды:

|  |
| --- |
| hadoop fs -rm [-skipTrash] URI [URI ...] |

Пример команды:

|  |
| --- |
| hadoop fs -rm /dir/file |

Команда удалит file из каталога dir в корзину.

Чтобы удалить безвозвратно файл, необходимо использовать параметр ‑skiptrash.

Пример команды для безвозвратного удаления файла:

|  |
| --- |
| hadoop fs -rm -skipTrash /dir/file |

Команда удалит file из каталога dir безвозвратно.

Чтобы рекурсивно удалить весь каталог (с подкаталогами), необходимо использовать параметр -r.

Пример команды для рекурсивного удаления каталога:

|  |
| --- |
| hadoop fs -rm -r /dir |

Команда удалит каталог dir со всеми подкаталогами.

[rmr] Рекурсивное удаление файла в корзину

Команда rmr рекурсивно удаляет каталог (со всеми подкаталогами) в корзину.

Синтаксис команды:

|  |
| --- |
| hadoop fs -rmr [-skipTrash] URI [URI ...] |

Пример команды:

|  |
| --- |
| hadoop fs -rmr /dir |

Команда удалит каталог dir со всеми подкаталогами.

[setfacl] Управление списками контроля доступа файлов

Команда setfacl позволяет взаимодействовать со списками контроля доступа (ACL, Access Control List) — устанавливать, модифицировать и удалять ACL для файлов и каталогов.

Синтаксис команды:

|  |
| --- |
| hadoop fs -setfacl [-r] [-b|-k -m|-x <acl\_spec> <path>]|[--set <acl\_spec> <path>] |

В команде могут быть использованы следующие параметры:

-b — удалить все записи ACL кроме базовых. Записи для пользователя, группы и остальных сохраняются для совместимости с битами разрешений;

-k — удалить ACL по умолчанию;

-r — рекурсивно применять операции ко всем файлам и каталогам;

-m — изменить ACL. Новые записи добавляются в ACL, а существующие записи сохраняются;

-x — удалить указанные записи ACL. Остальные записи ACL сохраняются;

--set — полностью заменить ACL, удалив все существующие записи. acl\_spec должен включать записи для пользователя, группы и остальных для совместимости с битами разрешений;

acl\_spec — список записей ACL, разделённых запятыми;

path — файл или каталог для изменения.

Примеры команды:

|  |
| --- |
| hadoop fs -setfacl -m user:hadoop:rw- /file  hadoop fs -setfacl -x user:hadoop /file  hadoop fs -setfacl -b /file  hadoop fs -setfacl -k /dir  hadoop fs -setfacl --set user::rw-,user:hadoop:rw-,group::r--,other::r-- /file  hadoop fs -setfacl -R -m user:hadoop:r-x /dir  hadoop fs -setfacl -m default:user:hadoop:r-x /dir |

[setrep] Изменение коэффициента репликации файла

Команда setrep предназначена для изменения коэффициента репликации для указанного файла или каталога.

Синтаксис команды:

|  |
| --- |
| hadoop fs -setrep [-r] [-w] <numReplicas> <path> |

В команде могут быть использованы следующие параметры:

-w — требует, чтобы команда дождалась завершения репликации.

-r — для каталогов команда рекурсивно изменяет коэффициент репликации всех файлов в дереве каталогов с корнем по пути.

Пример команды:

|  |
| --- |
| hadoop fs -setrep -w 3 /user/hadoop/dir1 |

[stat] Вывод статической информации о пути

Команда stat предназначена для вывода статистической информации о пути.

Синтаксис команды:

|  |
| --- |
| hadoop fs -stat URI [URI ...] |

Пример команды:

|  |
| --- |
| hadoop fs -stat path |

Команда выведет статистику о пути path.

[tail] Чтение последних строк файла

Команда tail позволяет без чтения всего файла вывести только последние строки файла (последний килобайт). Например, можно использовать для чтения логов.

Синтаксис команды:

|  |
| --- |
| hadoop fs -tail [-f] URI |

Пример использования команды:

|  |
| --- |
| hadoop fs -tail /dir/file.txt |

Команда выведет последние строки файла.

[test] Вывод атрибутов файла

Команда test выводит атрибуты указанного файла или каталога.

Синтаксис команды:

|  |
| --- |
| hadoop fs -test -[ezd] URI |

В команде могут быть использованы следующие параметры:

-e — проверяет, существует ли файл, и возвращает 0 в случае истины;

-z — проверяет, имеет ли файл нулевую длину, и возвращает 0 в случае истины;

-d — проверяет, является ли путь каталогом, и возвращает 0, если истина.

Пример использования команды:

|  |
| --- |
| hadoop fs -test -e filename |

[text] Чтение файлов

Команда text является аналогом команды cat, но при этом может служить для разархивирования файлов.

Синтаксис команды:

|  |
| --- |
| hadoop fs -text <scr> |

Пример использования команды:

|  |
| --- |
| hadoop fs -text /dir/file.gz |

Команда извлечёт архив и выведет его содержимое.

[touchz] Создание пустого файла

Команда touchz позволяет создать файл нулевой длины.

Синтаксис команды:

|  |
| --- |
| hadoop fs -touchz URI [URI ...] |

Пример использования команды:

|  |
| --- |
| hadoop fs -touchz /dir |

Команда создаст нулевой файл в каталоге dir.

[envvars] Отображение вычисленных переменных сред

Команда envvars позволяет отобразить вычисленные переменные среды Hadoop.

Синтаксис команды:

|  |
| --- |
| hadoop envvars |

[fetchdt] Получение токена делегирования

Команда fetchdt позволяет получить токен делегирования от NameNode.

Синтаксис команды:

|  |
| --- |
| hadoop fetchdt <opts> <token\_file\_path> |

В команде могут быть использованы следующие параметры команды COMMAND\_OPTION:

--webservice NN\_Url — UR-адрес для связи с NN (начинается с http или https);

--renewer name — имя обновителя токена делегирования;

--cancel — отменить токен делегирования;

--renew — обновить токен делегирования. Токен делегирования должен быть получен с использованием параметра –renewer name;

--print — распечатать токен делегирования;

token\_file\_path — путь к файлу, в котором будет храниться токен.

[fsck] Проверка файловой системы

Команда fsck позволяет осуществлять проверку файловой системы.

Синтаксис команды:

|  |
| --- |
| hadoop fsck <path>  [-list-corruptfileblocks |  [-move | -delete | -openforwrite]  [-files [-blocks [-locations | -racks | -replicaDetails | -upgradedomains]]]  [-includeSnapshots] [-showprogress]  [-storagepolicies] [-maintenance]  [-blockId <blk\_Id>] |

В команде могут быть использованы следующие параметры команды COMMAND\_OPTION:

path — начать проверку с этого пути;

-delete — удалить поврежденные файлы;

-files — распечатать проверяемые файлы;

-files -blocks — распечатать отчёт о блоке;

-files -blocks -locations — распечатать расположение каждого блока;

-files -blocks -racks — распечатать топологию сети для расположения узлов данных;

-files -blocks -replicaDetails — распечатать детали каждой реплики;

-files -blocks -upgradedomains — распечатать домены обновления для каждого блока;

-includeSnapshots — включить данные моментального снимка, если указанный путь указывает на каталог таблицы моментальных снимков или под ним есть каталоги таблиц моментальных снимков;

-list-correctedfileblocks — распечатать список отсутствующих блоков и файлов, которым они принадлежат;

-move — переместить поврежденные файлы в /lost + found;

-openforwrite — распечатать файлы, открытые для записи;

-showprogress — распечатать точки для прогресса вывода. По умолчанию выключено (нет прогресса);

-storagepolicies — распечатать сводку политики хранения для блоков;

-main maintenance — распечатать сведения об узле состояния обслуживания;

-blockId — распечатать информацию о блоке.

[getconf] Получение информации о настройках

Команда getconf позволяет получить информацию о настройках из каталогов конфигураций.

Синтаксис команды:

|  |
| --- |
| hadoop getconf -namenodes  hadoop getconf -secondaryNameNodes  hadoop getconf -backupNodes  hadoop getconf -journalNodes  hadoop getconf -includeFile  hadoop getconf -excludeFile  hadoop getconf -nnRpcAddresses  hadoop getconf -confKey [key] |

В команде могут быть использованы следующие параметры команды COMMAND\_OPTION:

-namenodes — получить список узлов NameNode в кластере;

-secondaryNameNodes — получить список узлов SecondaryNameNode в кластере;

-backupNodes — получить список узлов резервного копирования в кластере;

-journalNodes — получить список узлов журнала в кластере;

-includeFile — получить путь к файлу, который определяет узлы данных, которые имеют доступ к кластеру;

-excludeFile — получить путь к файлу исключения, который определяет узлы данных, которые доступа к кластеру не имеют;

-nnRpcAddresses — получить адреса rpc NameNode;

-confKey [key] — получить конкретный ключ из конфигурации.

[groups] Получение информации о группах

Команда groups позволяет получить информацию о группах пользователей с учётом одного или нескольких имён пользователей.

Синтаксис команды:

|  |
| --- |
| hadoop groups [username ...] |

[httpfs] Запуск HTTP

Команда httpfs позволяет запустить сервер HttpFS, HTTP-шлюз **HDFS**.

Синтаксис команды:

|  |
| --- |
| hadoop httpfs |

[lsSnapshottableDir] Получение списка каталогов снапшотов

Команда lsSnapshottableDir позволяет получить список каталогов снапшотов. Когда он запускается от имени суперпользователя, он возвращает все каталоги снапшотов. В противном случае он возвращает те каталоги, которые принадлежат текущему пользователю.

Синтаксис команды:

|  |
| --- |
| hadoop lsSnapshottableDir [-help] |

В команде могут быть использованы следующие параметры команды COMMAND\_OPTION:

-help — получение справки по получению.

[jmxget] Получение дампа информации JMX из службы

Команда jmxget позволяет получить дамп информации JMX из службы.

Синтаксис команды:

|  |
| --- |
| hadoop jmxget [-localVM ConnectorURL | -port port | -server mbeanserver | -service service] |

В команде могут быть использованы следующие параметры команды COMMAND\_OPTION:

-help — справка по печати;

-localVM ConnectorURL — подключиться к виртуальной машине на том же компьютере;

-port mbean server port — указать порт сервера mbean, если он отсутствует, он попытается подключиться к серверу MBean на той же виртуальной машине;

-server — указать сервер mbean (по умолчанию localhost);

-service NameNode|DataNode — указать службу jmx (NameNode по умолчанию).

[oev] Редактор файлов

Команда oev позволяет вывести просмотрщик и редактор файлов оффлайн.

Синтаксис команды:

|  |
| --- |
| hadoop oev [OPTIONS] -i INPUT\_FILE -o OUTPUT\_FILE |

В команде могут быть использованы следующие параметры команды COMMAND\_OPTION:

обязательные:

-i, -inputFile arg — редактирует файл для обработки, расширение xml (без учёта регистра) означает формат XML, любое другое имя файла означает двоичный формат;

-o, -outputFile arg — имя выходного файла. Если указанный файл существует, он будет перезаписан, формат файла определяется параметром –p;

необязательные:

-f, -fix-txids — перенумеровать идентификаторы транзакций во входных данных, чтобы не было пропусков или недопустимых идентификаторов транзакций;

-h, -help — показать информацию об использовании и выйти;

-r, -restore — при чтении журналов редактирования двоичных файлов использовать режим восстановления. Это даст вам возможность пропустить повреждённые части журнала редактирования;

-p, -processor arg — выберите тип процессора, который будет применяться к файлу изображения;

-v, -verbose — более подробный вывод, распечатывает имена файлов ввода и вывода, для процессоров, которые пишут в файл, также выводятся на экран. Для больших файлов изображений это значительно увеличит время обработки (по умолчанию false).

[oiv] Редактор изображений

Команда oiv позволяет вывести просмотрщик и редактор изображений оффлайн.

Синтаксис команды:

|  |
| --- |
| hadoop oiv [OPTIONS] -i INPUT\_FILE |

В команде могут быть использованы следующие параметры команды COMMAND\_OPTION:

обязательные:

-i | --inputFile input file — укажите входной файл fsimage (или файл XML, если используется процессор ReverseXML) для обработки;

необязательные:

-o, -outputFile output file — укажите имя выходного файла, если указанный процессор вывода создает его. Если указанный файл уже существует, он автоматически перезаписывается (по умолчанию вывод на стандартный вывод). Если входной файл является файлом XML, он также создаёт <outputFile>.md5;

-p, -processor processor — укажите процессор изображения, который будет применяться к файлу изображения. В настоящее время допустимые варианты: Web (по умолчанию), XML, Delimited, FileDistribution и ReverseXML;

-addr address — укажите адрес (хост:порт) для прослушивания. (localhost:5978 по умолчанию). Эта опция используется с веб-процессором;

-maxSize size — укажите диапазон [0, maxSize] размеров файлов для анализа в байтах (по умолчанию 128 ГБ). Эта опция используется с процессором FileDistribution;

-step size — укажите степень детализации распределения в байтах (по умолчанию 2 МБ). Эта опция используется с процессором FileDistribution;

-format — форматировать результат в удобочитаемом виде, а не в байтах. (по умолчанию false). Эта опция используется с процессором FileDistribution;

-delimiter arg — строка-разделитель для использования с процессором с разделителями;

-t, -temp temporary dir — использовать временный каталог для кеширования промежуточного результата для генерации выходных данных с разделителями. Если не установлен, процессор с разделителями создает пространство имен в памяти перед выводом текста;

-h, -help — показать использование инструмента и справочную информацию и выйти.

[snapshotDiff] Определение разницы между снапшотами

Команда snapshotDiff позволяет определить разницу между снапшотами.

Синтаксис команды:

|  |
| --- |
| hadoop snapshotDiff <path> <fromSnapshot> <toSnapshot> |

[version] Просмотр версии HDFS

Команда version позволяет посмотреть текущую версию **HDFS**.

Синтаксис команды:

|  |
| --- |
| hadoop version |

Secondary NameNode

NameNode хранит изменения в файловой системе в виде журнала изменений, добавляемого к родному файлу. Когда NameNode запускается, сервис считывает состояние **HDFS** из файла образа, fsimage, а затем применяет изменения из файла журнала. После чего он записывает новое состояние **HDFS** в fsimage и начинает обычную работу с пустым файлом правок. Поскольку NameNode объединяет fsimage и редактирует файлы только во время запуска, файл журнала изменений может со временем стать очень большим. Другим побочным эффектом большего файла журнала изменений является то, что следующий перезапуск NameNode занимает больше времени.

В свою очередь Secondary NameNode время от времени объединяет файлы журнала изменений с fsimage и сохраняет размер журнала в пределах лимита. Поскольку требования к памяти у вторичного и основного NameNode одинаковы, Secondary NameNode обычно запускается на другом узле, чем NameNode.

Запуск процесса контрольной точки на Secondary NameNode управляется двумя параметрами конфигурации:

dfs.namenode.checkpoint.period — по умолчанию установлено значение на 1 час; определяет максимальную задержку между двумя последовательными контрольными точками;

dfs.namenode.checkpoint.txns — по умолчанию установлено значение на 1 миллион; определяет количество незарегистрированных транзакций на NameNode, которые будут вызывать следующую контрольную точку, даже если период её не был достигнут.

Secondary NameNode хранит последнюю контрольную точку в каталоге, который структурирован таким же образом, как и каталог основного NameNode. Получается, образ с контрольной точки всегда готов к считыванию активным NameNode при необходимости.

Пример использования:

|  |
| --- |
| hdfs secondarynamenode [-checkpoint [force]] | [-format] | [-geteditsize] |

Где:

-checkpoint [force] — контрольные точки SecondaryNameNode, если EditLog size> = fs.checkpoint.size. Если используется принудительно, контрольная точка не зависит от размера EditLog;

-format — форматирование локального хранилища во время запуска;

-geteditsize — вывод количества незарегистрированных транзакций на NameNode.

Checkpoint Node

NameNode сохраняет свое пространство имён с помощью двух файлов: fsimage, который является последней контрольной точкой пространства имен, и журнал изменений (log) в пространстве имён с момента контрольной точки. Когда NameNode запускается, он объединяет fsimage и журнал изменений, чтобы обеспечить актуальное представление метаданных файловой системы. Затем NameNode перезаписывает fsimage с новым состоянием **HDFS** и начинает новый журнал изменений.

Checkpoint Node периодически создаёт контрольные точки пространства имён. Он загружает fsimage и изменения из активного NameNode, объединяет их локально и загружает новое изображение обратно в активный NameNode. Checkpoint Node обычно работает на компьютере, отличном от NameNode, поскольку их требования к памяти одинаковы. Запуск Checkpoint Node осуществляется командой bin/hdfs namenode -checkpoint на указанном в конфигурационном файле узле.

Расположение Checkpoint Node (или Backup) и сопровождающего его веб-интерфейса настраивается с помощью переменных конфигурации dfs.namenode.backup.address и dfs.namenode.backup.http-address.

Запуск процесса контрольной точки на Checkpoint Node контролируется двумя параметрами конфигурации:

dfs.namenode.checkpoint.period — по умолчанию установлено значение на 1 час; определяет максимальную задержку между двумя последовательными контрольными точками;

dfs.namenode.checkpoint.txns — по умолчанию установлено значение на 1 миллион; определяет количество незарегистрированных транзакций на NameNode, которые будут вызывать срочную контрольную точку, даже если период её не был достигнут.

Checkpoint Node хранит последнюю контрольную точку в каталоге, который структурирован таким же образом, как и каталог NameNode. Таким образом контрольное изображение всегда доступно для чтения из NameNode при необходимости.

|  |
| --- |
| **Внимание.**  В файле конфигурации кластера можно указать несколько Checkpoint Node. |

Пример использования:

|  |
| --- |
| hdfs namenode [-backup] |  [-checkpoint] |  [-format [-clusterid cid ] [-force] [-nonInteractive] ] |  [-upgrade [-clusterid cid] [-renameReserved<k-v pairs>] ] |  [-upgradeOnly [-clusterid cid] [-renameReserved<k-v pairs>] ] |  [-rollback] |  [-rollingUpgrade <rollback |started> ] |  [-importCheckpoint] |  [-initializeSharedEdits] |  [-bootstrapStandby [-force] [-nonInteractive] [-skipSharedEditsCheck] ] |  [-recover [-force] ] |  [-metadataVersion ] |

Где:

-backup — запуск Backup Node;

-checkpoint — запуск Checkpoint Node;

-format [-clusterid cid] — форматирование указанного NameNode. Запускает NameNode, форматирует его и затем выключает. Будет генерировать NameNodeFormatException, если имя dir уже существует и если переформатирование для кластера отключено;

-upgrade [-clusterid cid] [-renameReserved <k-v pairs>] — Namenode должен быть запущен с опцией обновления до новой версии Hadoop;

-upgradeOnly [-clusterid cid] [-renameReserved <k-v pairs>] — обновляет и закрывает указанный NameNode;

-rollback — откат NameNode до предыдущей версии. Следует использовать после остановки кластера и разворачивания старой версии Hadoop;

-rollingUpgrade <rollback|started> — query: запрос текущего состояния обновления; prepare: подготовка нового обновления; finalize: завершение текущего обновления;

-importCheckpoint — загрузка изображения из каталога контрольных точек и сохранение его в текущем. Контрольная точка dir читается из свойства dfs.namenode.checkpoint.dir;

-initializeSharedEdits — форматирование нового общего каталога изменений и копирование его в достаточное количество сегментов журнала для запуска резервного NameNode;

-bootstrapStandby [-force] [-nonInteractive] [-skipSharedEditsCheck] — позволяет загружать резервные каталоги хранилища NameNode путём копирования последнего снимка пространства имён из активного NameNode. Используется при первой настройке кластера высокой доступности. Опция -force или -nonInteractive имеет то же значение, что и -format. Опция -skipSharedEditsCheck пропускает проверку правок, которая гарантирует достаточное количество правок, уже находящихся в общем каталоге, для запуска с последней контрольной точки;

-recover [-force] — восстановление потерянных метаданных в поврежденной файловой системе;

-metadataVersion — печать версии метаданных программного обеспечения и образа (при условии наличия настроенных каталогов).

Backup Node

Backup Node обеспечивает ту же функциональность контрольных точек, что и Checkpoint Node, а также поддерживает в памяти обновленную копию пространства имён файловой системы, которая всегда синхронизируется с активным состоянием NameNode. Наряду с принятием потока изменений файловой системы журнала из NameNode и сохранением его на диске, Backup Node также применяет эти изменения к своей собственной копии пространства имён в памяти, создавая таким образом резервную копию пространства имён.

Backup Node не нужно загружать fsimage и редактировать файлы с активного NameNode для создания контрольной точки, как это требуется для Checkpoint Node или Secondary NameNode, поскольку Backup Node уже имеет актуальный статус состояния пространства имён в памяти. Процесс создания контрольной точки Backup узла является более эффективным, поскольку ему нужно только сохранить пространство имён в локальном файле fsimage и сбросить изменения.

Поскольку Backup Node поддерживает копию пространства имён в памяти, его требования к RAM такие же, как и у узла NameNode.

|  |
| --- |
| **Внимание.**  NameNode поддерживает один Backup Node. Узлы Checkpoint не могут быть зарегистрированы, если используется Backup. |

Backup Node настраивается так же, как узел Checkpoint, начиная с bin/hdfs namenode -backup.

Расположение Backup Node (или Checkpoint) и его веб-интерфейса настраивается с помощью переменных конфигурации dfs.namenode.backup.address и dfs.namenode.backup.http-address.

Использование Backup Node обеспечивает возможность запуска NameNode без постоянного хранилища, делегируя всю ответственность за сохранение состояния пространства имён Backup Node. Для этого необходимо запустить NameNode с параметром -importCheckpoint, а также не указывать постоянные каталоги хранения типа edits dfs.namenode.edits.dir в конфигурации NameNode.

Пример использования приведён в главе Checkpoint Node.

Import Checkpoint

В случае потери всех копий образов и файлов транзакций в NameNode может быть импортирована последняя контрольная точка. Для этого необходимо:

1. Создать пустой каталог, указанный в переменной конфигурации dfs.namenode.name.dir.
2. Указать местоположение каталога контрольных точек в переменной конфигурации dfs.namenode.checkpoint.dir.
3. Запустить NameNode с параметром -importCheckpoint.

При этом NameNode загружает контрольную точку из каталога dfs.namenode.checkpoint.dir, а затем сохраняет её в каталог NameNode, заданный в dfs.namenode.name.dir. Если в dfs.namenode.name.dir содержится допустимый образ, NameNode выдаёт ошибку. NameNode проверяет консистентность образа в dfs.namenode.checkpoint.dir, но не изменяет его каким-либо образом.

Пример использования приведён в главе Checkpoint Node.

Balancer

Данные **HDFS** не всегда могут быть размещены равномерно по всем DataNode. Одной из распространённых причин является добавление новых DataNodes в существующий кластер. При размещении новых блоков (данные для файла хранятся в виде серии блоков) NameNode учитывает различные параметры, прежде чем выбирать узлы DataNodes для получения этих блоков. Некоторые из них:

1. Политика для хранения одной из реплик блока на том же узле, что и узел, который записывает блок.
2. Необходимо распределить различные реплики блока по стойкам, чтобы кластер мог пережить потерю всей стойки.
3. Одна из реплик обычно размещается в той же стойке, что и узел, выполняющий запись в файл, что снижает количество операций ввода-вывода между стойками.
4. Данные **HDFS** распределяются равномерно по узлам DataNodes в кластере.

Из-за множества конкурирующих соображений данные могут быть неравномерно размещены между узлами DataNodes. **HDFS** предоставляет инструмент для администраторов, который анализирует размещение блоков и осуществляет перебалансировку данных через DataNode.

Пример использования:

|  |
| --- |
| hdfs balancer  [-policy <policy>]  [-threshold <threshold>]  [-exclude [-f <hosts-file> | <comma-separated list of hosts>]]  [-include [-f <hosts-file> | <comma-separated list of hosts>]]  [-source [-f <hosts-file> | <comma-separated list of hosts>]]  [-blockpools <comma-separated list of blockpool ids>]  [-idleiterations <idleiterations>]  [-runDuringUpgrade] |

Где:

-policy <policy> — datanode (по умолчанию): кластер сбалансирован, если каждый datanode сбалансирован; blockpool: кластер сбалансирован, если каждый пул блоков в каждой datanode сбалансирован;

-threshold <threshold> — процент ёмкости диска; перезапись порога по умолчанию;

-exclude -f <hosts-file> | <comma-separated list of hosts> — исключение указанных datanode из балансировки;

-include -f <hosts-file> | <comma-separated list of hosts> — включение только указанных datanode для балансировки;

-source -f <hosts-file> | <comma-separated list of hosts> — выбор только указанных datanode в качестве исходных узлов;

-blockpools <comma-separated list of blockpool ids> — работа балансировщика только на blockpools, включенных в этот список;

-idleiterations <iterations> — максимальное количество холостых итераций перед выходом; перезапись количества итераций по умолчанию (5);

-runDuringUpgrade — следует ли запускать балансировщик во время текущего обновления **HDFS**. Действие не влияет на используемое пространство на перегруженных машинах, поэтому нежелательно;

-h|--help — показывает используемый инструмент и справочную информацию.

Для остановки процесса перебалансировки администратор может просто нажать комбинацию клавиш Ctrl-C.

|  |
| --- |
| **Внимание.**  Политика blockpool является более строгой, чем политика datanode. |

Помимо указанных параметров введена функция закрепления (pinning feature) для предотвращения перемещения определённых реплик балансировщиком или mover. По умолчанию функция отключена и может быть включена с помощью свойства конфигурации dfs.datanode.block-pinning.enabled. В включенном состоянии она влияет только на блоки, которые записываются в указанные в вызове create() узлы. Функция полезна при необходимости локального сохранения данных для таких приложений, как HBase regionserver.

Rack Awareness

Кластер **HDFS** может распознавать топологию стоек, в которых размещены все узлы. Важно настроить эту топологию, чтобы оптимизировать емкость и использование данных.

Безопасный режим

Во время запуска NameNode загружает состояние файловой системы из fsimage и файла журнала изменений. Затем, чтобы не начать преждевременную репликацию блоков, он ожидает, когда DataNodes сообщат о своих блоках, несмотря на то, что в кластере уже существует достаточное количество реплик. В течение этого времени NameNode остается в безопасном режиме (Safemode). Безопасный режим для NameNode по существу является режимом только для чтения для кластера **HDFS**, где он не допускает каких-либо изменений в файловой системе или блоках. Обычно NameNode выходит из Safemode автоматически после того, как узлы DataNodes сообщают, что большинство блоков файловой системы доступно.

При необходимости система **HDFS** может быть размещена в безопасном режиме явно с помощью команды bin/hdfs dfsadmin -safemode. Включён ли режим Safemode отображается на первой странице NameNode.

Команда fsck

**HDFS** поддерживает команду fsck для проверки различных несоответствий. Команда предназначена для сообщения о проблемах с различными файлами, например, об отсутствующих блоках для файла или недостаточно реплицированных блоках. Но в отличие от традиционной утилиты fsck для собственных файловых систем, эта команда не исправляет обнаруженные ошибки.

Обычно NameNode автоматически исправляет большинство сбоев. По умолчанию fsck игнорирует открытые файлы, но предоставляет возможность выбора всех файлов во время создания отчётов. Команда fsck системы **HDFS** не является командой оболочки **Hadoop**, она может быть запущена путем bin/hdfs fsck.

Команда fsck может быть запущена по всей файловой системе или только на подмножестве файлов.

Пример использования:

|  |
| --- |
| hdfs fsck <path>  [-list-corruptfileblocks |  [-move | -delete | -openforwrite]  [-files [-blocks [-locations | -racks | -replicaDetails | -upgradedomains]]]  [-includeSnapshots] [-showprogress]  [-storagepolicies] [-maintenance]  [-blockId <blk\_Id>] |

Где:

path — начало проверки с указанного пути;

-delete — удаление повреждённых файлов;

-files — отобразить проверяемые файлы;

-files -blocks — отобразить отчёт о блокировке;

-files -blocks -locations — отобразить расположение каждого блока;

-files -blocks -racks — отобразить топологию сети для расположения узлов данных;

-files -blocks -replicaDetails — отобразить каждую копию реплики;

-files -blocks -upgradedomains — отобразить обновления доменов для каждого блока;

-includeSnapshots — включить данные снапшотов, если указанный путь указывает на каталог снапшотов или под ним есть каталоги снапшотов;

-list-corruptfileblocks — отобразить список отсутствующих блоков и файлов, к которым они принадлежат;

-move — переместить поврежденные файлы в /lost+found;

-openforwrite — отобразить файлы, открытые для записи;

-showprogress — отобразить точки для прогресса в выводе; по умолчанию выключено (без прогресса);

-storagepolicies — отобразить сводную политику хранения для блоков;

-maintenance — отобразить детали узла;

-blockId — отобразить информацию о блоке.

Команда fetchdt

Для получения Delegation Token и сохранения его в файле в локальной системе **HDFS** поддерживает команду fetchdt. Этот токен впоследствии можно использовать для доступа к защищённому серверу (например, NameNode) с незащищённого клиента. Утилита использует RPC или HTTPS (через Kerberos) для получения токена и, следовательно, требует наличия тикетов Kerberos перед запуском (необходимо запустить kinit для получения тикетов). В **HDFS** команда fetchdt не является командой оболочки **Hadoop** и может быть запущена как bin/hdfs fetchdt DTfile. После получения токена можно запустить команду **HDFS** без тикетов Kerberos, указав переменную среды HADOOP\_TOKEN\_FILE\_LOCATION в файл токена делегирования.

Пример использования:

|  |
| --- |
| hdfs fetchdt <opts> <token\_file\_path> |

Где:

--webservice NN\_Url — URL для связи с NN (начинается с http или https);

--renewer name — обновленное название токена делегирования;

--cancel — отменить токен делегирования;

--renew — обновить токен делегирования. Токен делегирования должен быть получен с использованием параметра -renewer name;

--print — отобразить токен делегирования;

token\_file\_path — путь к файлу для хранения токена.

Recovery Mode

Как правило, обычно настраивается несколько мест хранения метаданных. И в случае, если хранилище повреждается, метаданные можно прочитать из других хранилищ. Однако, что делать, если единственное доступное хранилище повреждено? Для таких ситуаций существует специальный режим запуска NameNode, называемый режимом восстановления — Recovery Mode, который позволяет восстановить большую часть данных.

Запуск NameNode в режиме восстановления осуществляется следующим образом: namenode -recover.

В режиме восстановления NameNode в командной строке интерактивно запрашивает о возможных действиях, которые необходимо предпринять для восстановления данных. Для того, чтобы не получать эти запросы, можно указать опцию -force, которая заставляет режим Recovery Mode всегда выбирать первый вариант действий, обычно являющийся самым разумным.

|  |
| --- |
| **Внимание.**  Поскольку режим Recovery Mode может привести к потере данных, всегда необходимо перед его использованием делать резервную копию журнала изменений и fsimage. |

Upgrade и Rollback

При обновлении **Hadoop** на существующем кластере, как и при любом обновлении программного обеспечения, возможны новые ошибки или несовместимые изменения, которые влияют на существующие приложения и не были обнаружены ранее. Система позволяет администраторам вернуться к более ранней версии **Hadoop** и откатить кластер до состояния, в котором он находился до обновления. Но **HDFS** может иметь одну такую резервную копию. Перед обновлением администраторам необходимо удалить существующую резервную копию с помощью команды bin/hadoop dfsadmin -finalizeUpgrade. Далее кратко описывается типичная процедура обновления:

1. Перед обновлением программного обеспечения **Hadoop** завершить работу, если имеется существующая резервная копия.
2. Остановить кластер и распространить новую версию **Hadoop**.
3. Запустить новую версию с параметром -upgrade (sbin/start-dfs.sh -upgrade).
4. Завершить обновление, как только новая **HDFS** считается работающей (возможно, после нескольких дней эксплуатации). Важно обратить внимание, что пока кластер не будет завершён, удаление файлов, существовавших до обновления, не освобождает реальное дисковое пространство на узлах данных DataNodes.
5. При необходимости вернуться к старой версии:

Остановить кластер и распространить более раннюю версию **Hadoop**;

Выполнить команду отката в namenode (bin/hdfs namenode -rollback);

Запустить кластер с возможностью отката (sbin/start-dfs.sh -rollback).

При обновлении **HDFS** до новой версии необходимо переименовать или удалить все пути, зарезервированные в новой версии **HDFS**, в ином случае, когда NameNode встречает зарезервированный путь во время обновления, выдает ошибку:

|  |
| --- |
| /.reserved is a reserved path and .snapshot is a reserved path component in this version of HDFS. Please rollback and delete or rename this path, or upgrade with the -renameReserved [key-value pairs] option to automatically rename these paths during upgrade. |

Указание -upgrade -renameReserved [optional key-value pairs] заставляет NameNode автоматически переименовывать любые зарезервированные пути, найденные во время запуска. Например, чтобы переименовать все пути с именами .snapshot на .my-snapshot и .reserved на .my-reserved, необходимо указать -upgrade -renameReserved .snapshot=.my-snapshot,.reserved=.my-reserved.

Если с ключом -renameReserved не указаны пары ключ-значение, NameNode добавляет суффиксы зарезервированным путям в виде .<LAYOUT-VERSION>.UPGRADE\_RENAMED, например, .snapshot.-51.UPGRADE\_RENAMED.

Есть некоторые оговорки к этому процессу переименования и по возможности рекомендуется перед обновлением сначала выполнять hdfs dfsadmin -saveNamespace. Это связано с тем, что может возникнуть несогласованность данных, если операция журнала изменений ссылается на место назначения автоматически переименованного файла.

DataNode Hot Swap Drive

Datanode поддерживает возможность горячей замены дисков, поэтому пользователь может добавлять или заменять узлы DataNode в **HDFS**, не выключая их. Далее кратко описана типичная процедура замены Hot Swap Drive:

1. При наличии новых каталогов хранения пользователь должен отформатировать их и подключить соответствующим образом.
2. Пользователь обновляет конфигурацию DataNode dfs.datanode.data.dir, чтобы отразить каталоги томов данных, которые будут использоваться.
3. Пользователь запускает dfsadmin -reconfig datanode HOST:PORT start для инициализации процесса реконфигурации. Пользователь может использовать dfsadmin -reconfig datanode HOST:PORT status для запроса текущего состояния задачи реконфигурации.
4. После завершения задачи реконфигурации пользователь может безопасно размонтировать удалённые каталоги томов данных и физически удалить диски.

Права доступа к файлам и безопасность

Права доступа к файлам схожи с правами доступа к файлам на других известных платформах, таких как Linux. В настоящее время безопасность ограничена простыми правами доступа к файлам. Пользователь, запускающий NameNode, для **HDFS** рассматривается как суперпользователь. Будущие версии **HDFS** будут поддерживать сетевые протоколы аутентификации, такие как Kerberos, для аутентификации пользователей и шифрования передачи данных.

Масштабируемость

В настоящее время **Hadoop** может работать на кластерах с тысячами узлов, где **HDFS** имеет по одному NameNode для каждого кластера. Таким образом общий объём памяти, доступный на NameNode, является основным ограничением масштабируемости. В очень больших кластерах увеличение среднего размера файлов, хранящихся в **HDFS**, способствует увеличению размера кластера без повышения требований к памяти для NameNode.

|  |
| --- |
| **Внимание.**  Конфигурация по умолчанию может не подходить для очень больших кластеров. |

РАБОТА С YARN

Основная идея **YARN** состоит в том, чтобы разделить функции управления ресурсами и планирования/мониторинга заданий на отдельные демоны. Суть заключается в том, чтобы иметь общий ResourceManager и ApplicationMaster для каждого приложения. Приложение — это отдельная работа или группа работ.

ResourceManager и NodeManager образуют среду для вычисления данных. ResourceManager — это высший орган, который распределяет ресурсы между всеми приложениями в системе. NodeManager — это агент инфраструктуры для каждой машины, который отвечает за контейнеры, отслеживает использование их ресурсов (процессор, память, диск, сеть) и сообщает об этом в ResourceManager/Scheduler.

ApplicationMaster для каждого приложения, по сути, является библиотекой, специфичной для платформы, и на него возложена задача согласования ресурсов из ResourceManager и работа с NodeManager(-ами) для выполнения и мониторинга задач.

ResourceManager имеет два основных компонента: Scheduler и ApplicationsManager.

Scheduler отвечает за распределение ресурсов между различными запущенными приложениями с учетом известных ограничений ёмкости, очередей и т.д. Scheduler является чистым планировщиком в том смысле, что он не выполняет никакого мониторинга или отслеживания состояния приложения. Кроме того, он не даёт никаких гарантий относительно перезапуска сбойных задач из-за ошибок приложения или оборудования. Планировщик выполняет свою функцию планирования на основе требований к ресурсам приложений на базе абстрактного понятия resource Container, включающего в себя такие элементы, как память, процессор, диск, сеть и т.д.

Планировщик имеет подключаемую политику, которая отвечает за распределение ресурсов кластера между различными очередями, приложениями и т.д.

ApplicationsManager отвечает за приём заявок, согласование первого контейнера для выполнения приложения определенным ApplicationMaster и предоставляет сервис для перезапуска контейнера ApplicationMaster при сбое. ApplicationMaster для каждого приложения отвечает за согласование подходящих resource Container с планировщиком, отслеживание их состояния и мониторинг прогресса.

**MapReduce** в hadoop-3.x поддерживает API-совместимость с предыдущей стабильной версией (hadoop-2.x). Это означает, что все задания **MapReduce** по-прежнему выполняются поверх **YARN** только с учётом перекомпиляции.

**YARN** поддерживает понятие резервирования ресурсов через ReservationSystem — компонент, который позволяет пользователям определять профиль ресурсов over-time и задавать временные ограничения (сроки), а так же резервировать ресурсы для обеспечения предсказуемого выполнения важных заданий. ReservationSystem отслеживает ресурсы over-time, выполняет управление допуском для резервирования и динамично инструктирует базовый планировщик, чтобы гарантировать, что резервирование полностью выполнено.

С целью масштабирования **YARN** за пределы нескольких тысяч узлов поддерживается понятие Federation через функцию YARN Federation. Федерация позволяет прозрачно соединять несколько кластеров (подкластеров) yarn и делать их единым массивным кластером. Это применимо для достижения большего масштаба и/или для того, чтобы несколько независимых кластеров могли использоваться вместе для очень объёмных заданий.

РАБОТА С HUE

Общая информация

**Hue** представляет собой web-интерфейс, который позволяет пользователям интерактивно взаимодействовать с кластером **Hadoop** и его компонентами.

**Hue** содержит следующие приложения:

1. **Редактор (Editor)** — служит для отправки запросов к данным как на SQL, так и с использованием других языков запросов. **Editor** также имеет функции интеллектуального автозаполнения, поиска и тегирования данных.

**Браузеры (Browsers)** — служат для поиска, просмотра и выполнения действия с данными или заданиями (jobs) кластера.

**Дашборд (Dashboard)** — служит для интерактивного изучения данных без использования языков программирования, а только с использованием функций drag&drop и выбора данных кликом.

**Планировщик (Scheduler)** — служит для создания заданий (workflows) и планирования их автоматического выполнения по триггеру. Интерфейс мониторинга показывает прогресс, логи и разрешает такие действия, как приостановка или остановка заданий.

Интерфейс

Интерфейс **Hue** является одностраничным приложением и содержит следующие области:

1. Верхняя панель с быстрым действием, глобальным поиском и областью уведомлений справа.

Меню со ссылками на различные приложения и быстрый способ импорта данных.

Расширенный быстрый просмотр на левой вспомогательной панели.

Основная область приложения.

Правая вспомогательна панель для текущего приложения. Она включена для редакторов (editors), а в случае, например, с **Hive**, она предлагает оперативную помощь, быстрый просмотр используемых таблиц в запросе и многое другое. Если ваш экземпляр **Hue** подключён к службе SQL-оптимизатора, панель может отображать предложения по вашим запросам.

Различные приложения сгруппированы в 4 основные концептуальные области.

Поиск

Панель поиска всегда доступна в верхней части экрана, а также предлагает поиск документов и поиск по метаданным, если **Hue** настроен для доступа к серверу метаданных.

Поиск предлагает следующие возможности:

1. **Встроенный поиск и теги**.

**Hue** позволяет искать любую таблицу, представление или столбец во всех базах данных в кластере.

Кроме того, имеется возможность помечать объекты тегами для их классификации и группировки по проектам. Эти теги доступны для поиска.

Автозаполнение панели поиска предлагает список аспектов и предварительное заполнение верхних значений. Нажатие клавиши ввода приводит список доступных объектов, которые можно открыть и исследовать далее во всплывающем окне, в помощнике или непосредственно в приложении для просмотра таблиц.

**Детальный поиск**.

По умолчанию возвращаются только таблицы и представления. Для поиска столбцов, разделов, баз данных используйте фильтр **Тип**:

Пример поиска:

table:customer — поиск таблицы клиента;

table:tax\* tags:finance — перечисление всех таблиц, начинающихся с tax и тегированных как finance;

owner:admin type:field usage — список всех полей, созданных пользователем с правами admin, которые соответствуют строке использования;

parentPath:"/default/web\_logs" type:FIELD originalName:b\* — перечисление всех столбцов, начинающихся с b таблицы web\_logs в базе данных default.

Левая вспомогательная панель

С помощь левой вспомогательной панели можно найти документы **Hue**, файлы **HDFS** и **S3** и другое. При щелчке правой кнопкой мыши по элементам отобразится список возможных действий с элементом.

Правая вспомогательная панель

Содержимое правой вспомогательной панели зависит от контекста выбранного приложения и отображает текущие таблицы или доступные UDF.

Всплывающее окно

Всплывающее окно предлагает быстрый способ просмотреть образцы данных и другую статистику по базам данных, таблицам и столбцам. Вы можете открыть всплывающее окно из SQL Assist или щёлкнуть правой кнопкой мыши на любой объект SQL (таблица, столбец, функция и прочие).

Визуализация

Визуализация удобна для построения хронологических данных или, когда подмножества строк имеют один и тот же атрибут, они будут складываться вместе.

Средство импорта

Цель Средства импорта — разрешить ad-hoc запросы к данным, которых ещё нет в кластере, тем самым ускорить self-service аналитику.

Если вы хотите импортировать свои собственные данные вместо образцов таблиц, откройте Средство импорта из левого меню или из **+** на левой вспомогательной панели.

Чтобы создать новые SQL-таблицы из файлов:

1. Выберите тип источника.

Выберите тип объекта назначения.

SQL-таблицы

Хотя можно создавать таблицы, выполняя соответствующие команды DDL- и HQL-запросов Hive, проще создать таблицу с помощью мастера создания таблиц:

1. **Из файла**.

Чтобы создать таблицу из файла, необходимо:

В диспетчере импорта выбирать источник из **File**.

Выбрать тип таблицы.

Файлы можно перетаскивать, выбирать из **HDFS** или S3 (если настроено), и их форматы определяются автоматически. Мастер также помогает при выполнении расширенных функций, таких как разделение таблиц, таблицы Kudu и вложенные типы.

**Вручную**.

Чтобы создать таблицу вручную, необходимо:

В диспетчере импорта выбрать **Manually**.

Следовать инструкциям мастера, чтобы создать таблицу. Основные шаги:

* назвать таблицу;
* выбрать формат записи;
* настроить сериализацию записей, указав разделители для столбцов, коллекций и ключей карты;
* выбрать формат файла;
* указать место для данных вашей таблицы;
* указать столбцы, указав имя и выбрав тип для каждого столбца;
* указать столбцы раздела, указав имя и выбрав тип для каждого столбца.

Индексирование

На вкладке **Search** на панели навигации выберите **Index**.

Введите имя для новой коллекции и выберите наш файл данных отзывов из HDFS. Затем нажмите **Next**.

Традиционные базы данных

**Hue** позволяет принимать данные из традиционных баз данных.

Документы

**Hue** позволяет сохранять документы, а также обмениваться ими с другими пользователями.

Обмен

Совместное использование происходит на главной странице или через верхнее правое меню приложения. Можно выбрать пользователей и группы с разрешениями на чтение или запись.

Импорт/экспорт

Через главную страницу сохраненные документы можно экспортировать для резервного копирования или переноса в другой **Hue**.

Смена языка

Язык определяется автоматически из браузера или ОС.

Язык может быть установлен вручную пользователем на странице **My profile**. Перейдите в **My Profile** —> **Step2 Profile and Groups** —> **Language Preference** и выберите нужный язык.

Редактор (Editor)

**Hue** ориентирован на SQL, но поддерживает и иные языки. Он позволяет работать с интеллектуальным автозаполнением, поиском и тегированием данных, а также поддержкой запросов.

Любой редактор может быть помечен звёздочкой рядом с его именем, чтобы он стал редактором по умолчанию и целевой страницей при входе в систему.

Во-первых, в файле hue.ini вам нужно обавить соответствующую информацию о подключении к базе данных в разделе librdbms:

|  |
| --- |
| [librdbms]  [[databases]]  [[[postgresql]]]  nice\_name=PostgreSQL  name=music  engine=postgresql\_psycopg2  port=5432  user=hue  password=hue  options={} |

Во-вторых, необходимо добавить новый интерпретатор в приложение для notebook. Это позволит зарегистрировать новый тип базы данных как тип сниппета в приложении Notebook. Для редакторов запросов, использующих базу данных, совместимую с Django, имя в скобках должно совпадать с именем конфигурации базы данных в разделе librdbms (например, ‑postgresql). Интерфейс будет установлен на rdbms. Это сообщает **Hue** использовать драйвер librdbms и соответствующую информацию для подключения к базе данных. Например, с указанной выше конфигурацией подключения postgresql в разделе librdbms можно добавить интерпретатор PostgreSQL со следующей конфигурацией notebook:

|  |
| --- |
| [notebook]  [[interpreters]]  [[[postgresql]]]  name=PostgreSQL  interface=rdbms |

Концепции

Выполнение запросов

|  |
| --- |
| **Примечание.**  Чтобы выполнить запрос, вы должны авторизоваться в **Hue** как пользователь, у которого также есть учётная запись пользователя Unix на удалённом сервере. |

Чтобы выполнить запрос, необходимо:

1. Чтобы выполнить часть запроса, выделите один или несколько операторов запроса.

Щёлкните **Execute**. Откроется окно **Query Results** с результатами вашего запроса.

Чтобы просмотреть журнал выполнения запроса, переключите курсор **Log** слева от индикатора выполнения. Вы можете использовать информацию на этой вкладке для отладки вашего запроса;

Чтобы просмотреть столбцы запроса, разверните значок **Columns**. При нажатии на метку столбца выполняется прокрутка к столбцу. Имена и типы можно фильтровать;

Чтобы развернуть строку, дважды щёлкните по ней или щёлкните номер строки;

Чтобы заблокировать строку, щёлкните значок замка в столбце номера строки;

выполните поиск, щёлкнув значок лупы на вкладке результатов или нажав Ctrl/Cmd + F.

Если в запросе есть несколько операторов (разделённых точкой с запятой), щёлкните **Next** на панели запроса с несколькими операторами, чтобы выполнить оставшиеся операторы.

Если у вас есть несколько операторов, достаточно поместить курсор в оператор, который вы хотите выполнить, активный оператор обозначается синей меткой.

|  |
| --- |
| **Примечание.**  Чтобы выполнить запрос, используйте Ctrl/Cmd + Enter. |

|  |
| --- |
| **Примечание.**  На панели журналов вы можете просмотреть любые задания **MapReduce**, созданные в результате запроса. |

Скачивание и сохранение результатов запроса

|  |
| --- |
| **Внимание.**  Вы можете сохранить результаты в файл только в том случае, если результаты были созданы заданием **MapReduce**.  Данный способ сохранения является предпочтительным, когда результат большой (например, > 1 млн строк). |

Чтобы скачать и сохранить результат запроса, выполните одно из следующих действий:

1. Щёлкните **Download as CSV**, чтобы загрузить результаты в файл значений, разделенных запятыми, который можно использовать в других приложениях.

Нажмите **Download as XLS**, чтобы загрузить результаты в файл таблицы Microsoft Office Excel.

Нажмите **Save**, чтобы сохранить результаты в таблице или файле **HDFS**:

Чтобы сохранить результаты в новой таблице, выберите **In a new table**, введите имя таблицы и нажмите **Save**.

Чтобы сохранить результаты в файле HDFS, выберите **In an HDFS directory**, введите путь и нажмите **Save**.

Расширенные настройки запроса

На панели вверху Pедактора можно указать параметры, представленные в таблице ниже.

Таблица — Параметры Редактора

| Параметр | Описание |
| --- | --- |
| Database | База данных, содержащая определения таблиц. |
| Settings | Переопределение настроек **Hive** и **Hadoop** по умолчанию. Чтобы настроить новый параметр:   1. Щёлкните **Add**. 2. В поле **Key** введите имя переменной конфигурации **Hive** или **Hadoop**. 3. В поле **Value** введите значение, которое вы хотите использовать для переменной.   Например, чтобы переопределить каталог, в котором создаются структурированные журналы запросов **Hive**, вы должны ввести hive.querylog.location для **Key** и путь для **Value**. |
| File resources | Параметр для локальной доступности файлов во время выполнения запроса в кластере **Hadoop**. **Hive** использует Hadoop Distributed Cache для распространения добавленных файлов на все машины в кластере во время выполнения запроса.   1. Нажмите **Add**, чтобы настроить новый параметр. 2. В раскрывающемся меню **Type** выберите один из следующих вариантов:   **jar** — добавляет указанные ресурсы в путь к классам Java;  **archive** — разархивирует указанные ресурсы при их распространении;  **file** — добавляет указанные ресурсы в распределённый кэш. Обычно это может быть скрипт преобразования (или аналогичный), который нужно выполнить.   1. В поле **Path** введите путь к файлу или щёлкните **Browse**, чтобы просмотреть и выбрать файл.   **Примечание.** Нет необходимости указывать файлы, используемые в скрипте преобразования, если файлы доступны по одному и тому же пути на всех машинах в кластере **Hadoop**. |
| User-defined functions | Специальные функции, определяемые пользователем.   1. Нажмите **Add**, чтобы настроить новый параметр. 2. Укажите имя функции в поле **Name** и укажите имя класса для **Classname**. 3. Вы должны указать jar-файл для пользовательских функций в **File resources**. Чтобы включить в запрос пользовательскую функцию, добавьте в запрос **$** (знак доллара) перед именем функции. Например, если MyTable — это имя определяемой пользователем функции в запросе, вы должны ввести: SELECT $MyTable |
| Parameterization | Указывает, что диалоговое окно должно отображаться для ввода значений параметров при выполнении запроса, содержащего строку $parametername. Включено по умолчанию. |

Автозаполнение

Инструмент автозаполнения знает все тонкости диалекта **Hive** SQL и предлагает ключевые слова, функции, столбцы, таблицы, базы данных в зависимости от структуры оператора и положения курсора.

Инструмент предлагает завершение не только для операторов SELECT, но и других операторов, таких как: DDL и DML, INSERT, CREATE, ALTER, DROP и прочих.

Инструмент осуществляет следующие типы автозаполнения:

1. **Автозаполнение столбцов**.

Если в предложении FROM появляется несколько таблиц, включая производные и объединённые таблицы, инструмент объединит столбцы из всех таблиц и при необходимости добавит соответствующие префиксы. Он также содержит информацию о ваших алиасах, последних просмотрах и сложных типах и будет включать их. Он будет автоматически возвращать любые зарезервированные слова или экзотические имена столбцов, где это необходимо, чтобы предотвратить любые ошибки.

**Автозаполнение ключевых слов**.

Инструмент автозаполнения предлагает ключевые слова в зависимости от положения курсора в операторе. Там, где это возможно, он даже будет предлагать более одного слова за раз. В тех частях, где порядок имеет значение, но ключевые слова являются необязательными, например, после FROM tbl, он будет перечислять предложения ключевых слов в ожидаемом порядке, с первым ожидаемым вверху. Итак, после FROM tbl ключевое слово WHERE указано выше GROUP BY.

**UDF**.

Инструмент автозаполнения предлагает функции. Для каждого предложения функции добавляется дополнительная панель в раскрывающемся списке автозаполнения, показывающая документацию и подпись функции. Автозаполнение хранит информацию об ожидаемых типах аргументов и будет предлагать только те столбцы или функции, которые соответствуют аргументу в позиции курсора в списке аргументов.

**Подзапросы, коррелированные или нет**.

При редактировании подзапросов инструмент будет делать предложения только в рамках подзапроса. Для коррелированных подзапросов также принимаются во внимание внешние таблицы.

В настройках Редактора вы можете отключить автозаполнение в реальном времени или вообще отключить автозаполнение. Чтобы получить доступ к этим настройкам, откройте Редактор и, наведя курсор на область с кодом, нажмите **CTRL + ,** (или на **Mac CMD + ,**) и настройки отобразятся.

Автозаполнение обращается к бэкэнду, чтобы получить данные для таблиц, баз данных и прочих. По умолчанию он истекает через 5 секунд, но как только он будет получен, он кэшируется для следующего раза. Тайм-аут можно настроить в конфигурации сервера **Hue**.

Если автозаполнение не может интерпретировать оператор, то раскрывающийся список не появится.

Переменные

Переменные могут иметь значения по умолчанию, например, ${n = 10} вместо простого ${n}.

Проверка синтаксиса

Подчёркивание красного цвета будет отображать неправильный синтаксис, чтобы запрос можно было исправить перед отправкой. Правый щелчок отобразит предложения.

Режим презентации

Превращает список запросов, разделённых точкой с запятой, в интерактивную презентацию. Режим отлично подходит для демонстрации или базовой отчётности.

SQL

Используйте Редактор запросов с любой базой данных, совместимой с JDBC или Django.

Hive

Обеспечена поддержка отправки нескольких запросов при использовании **Tez**. Вы можете включить его с помощью этой настройки:

|  |
| --- |
| [beeswax]  max\_number\_of\_sessions=10 |

Другие

Помимо Hive Редактор запросов совместим со следующими БД:

1. MySQL.

Oracle.

KSQL / Kafka SQL.

Solr SQL.

Presto.

PostgreSQL.

Redshift.

BigQuery.

AWS Athena.

Spark SQL.

Phoenix.

Kylin.

И другие.

Задания

Приложение Редактор позволяет создавать и отправлять задания в кластер. Вы можете включать переменные в свои задания, чтобы вы и другие пользователи могли вводить значения для переменных при выполнении вашего задания.

Все настройки дизайна задания, кроме **Name** и **Description**, поддерживают использование переменных вида $variable\_name. Когда вы запустите задание, появится диалоговое окно, в котором вы сможете указать значения переменных.

Параметры передаются скрипту или команде. Параметры выражаются с использованием [JSP 2.0 Specification (JSP.2.3) Expression Language], что позволяет использовать переменные, функции и сложные выражения в качестве параметров.

Задания содержат классы, представленные в таблице ниже.

Таблица — Классы заданий

| Класс | Описание |
| --- | --- |
| Name | Определяет задание и набор его свойств и параметров. |
| Description | Описание задания.  Описание отображается в диалоговом окне, которое появляется, если вы указываете переменные для задания. |
| Advanced | Расширенные настройки:   1. **Is shared** — укажите, следует ли предоставлять доступ к действию всем пользователям. 2. **Oozie parameters** — параметры для передачи в **Oozie**. |
| Prepare | Указывает пути для создания или удаления перед запуском задания рабочего процесса. |
| Params | Параметры задания. |
| Job Properties | Свойства задания.  Чтобы установить значение свойства, нажмите **Add Property**:   1. **Property name** — имя свойства конфигурации. Заполнение этого поля обеспечивается автозаполнением, поэтому вы можете ввести несколько первых символов имени свойства, а затем выбрать необходимое из раскрывающегося списка. 2. Установите значение для свойства. |
| Files | Файлы для передачи заданию.  Эквивалентно параметру Hadoop -files. |
| Archives | Файлы в виде архивов для передачи заданию.  Эквивалентно параметру Hadoop –archives. |

MapReduce

Дизайн задания **MapReduce** состоит из функций **MapReduce**, написанных на Java. Вы можете создать проект задания **MapReduce** из существующих классов mapper и reducer без необходимости писать основной класс Java. Вы должны указать классы mapper и reducer, а также другие свойства **MapReduce** в настройке **Job Properties**.

Таблица — Классы заданий MapReduce

| Класс | Описание |
| --- | --- |
| Jar path | Полный путь к файлу JAR, содержащему классы, реализующие функции mapper и reducer. |

Java

Дизайн задания Java состоит из основного класса, написанного на Java.

Таблица — Классы заданий Java

| Класс | Описание |
| --- | --- |
| Jar path | Полный путь к файлу JAR, содержащему основной класс. |
| Main class | Основной класс для вызова программы. |
| Args | Аргументы, передаваемые в основной класс. |
| Java opts | Параметры, передаваемые JVM. |

Shell

Дизайн задания Shell состоит из shell-команд.

Таблица — Классы заданий Shell

| Класс | Описание |
| --- | --- |
| Command | Shell-команда. |
| Capture output | Укажите, нужно ли записывать вывод команды. |

Дашборды (Dashboards)

Дашборды — это интерактивный способ быстро и легко изучить ваши данные. Никакого программирования не требуется, и анализ выполняется перетаскиванием и щелчками.

Концепции

Просто перетащите и отпустите виджеты, которые связаны между собой. Это отлично подходит для изучения новых наборов данных или мониторинга без ввода текста.

Импорт

Любой файл CSV можно перетащить и вставить в индекс за несколько кликов с помощью Data Import Wizard. Индексированные данные можно сразу же запросить, а их показатели/измерения можно будет очень быстро изучить.

Просмотр

Левая панель метаданных перечисляет ифномрацию о столбцах и просмотреть их содержимое во всплывающем окне образца.

Запрос

Поле поиска поддерживает фильтрацию данных поля в реальном времени по префиксу и поставляется с автозаполнением синтаксиса Solr, чтобы сделать запросы интуитивно понятными и быстрыми. Любое поле можно проверить на предмет его максимальных значений статистики. Этот анализ происходит очень быстро, поскольку данные индексируются.

Базы данных

Solr

С помощью Solr верхняя панель поиска предлагает полное автозаполнение для всех значений индекса.

Функция **More like This** позволяет вам выбирать поля, которые вы хотели бы использовать для поиска похожих записей. Это отличный способ найти похожие проблемы, клиентов, людей и прочее по списку атрибутов.

SQL

SQL также поддерживается дашбордами.

Браузеры (Browsers)

Браузеры **Hue** поддерживают ваш Data Catalog. Они позволяют легко искать, просматривать и выполнять действия с данными или заданиями в локальных кластерах.

Таблицы

Браузер таблиц позволяет вам управлять базами данных, таблицами и разделами хранилища метаданных, используемых **Hive**. Вы можете использовать Metastore Manager для выполнения следующих операций:

1. **Базы данных**:

Выбор базы данных;

Создание базы данных;

Удаление базы данных.

**Таблицы**.

Создание таблиц;

Просмотр таблиц;

Сброс таблиц;

Просмотр данных таблицы и метаданных (столбцы, разделы и прочее);

Импорт данных в таблицу;

Фильтрация, сортировка и просмотр разделов.

Файловые системы

Браузер файлов позволяет просматривать и управлять файлами и каталогами в HDFS, S3 или ADLS.

С помощью файлового браузера вы можете:

1. Создавать файлы и каталоги, загружать и скачивать файлы, загружать zip-архивы, а также переименовывать, перемещать и удалять файлы и каталоги. Также можно изменить владельца, группу и права доступа к файлу или каталогу.

Осуществлять поиск файлов, каталогов, владельцев и групп.

Просматривать и редактировать файлы как текстовые или двоичные.

HDFS

**Hue** поддерживает один кластер **HDFS**. Этот кластер должен быть определён в подразделе [[[default]]].

1. fs\_defaultfs:: — это эквивалент fs.defaultFS (также известного как fs.default.name) в конфигурации Hadoop.

webhdfs\_url:: — вы также можете установить это как URL-адрес HttpFS. Значение по умолчанию — порт HTTP на NameNode.

hadoop\_conf\_dir:: — это каталог конфигурации HDFS, обычно /etc/hadoop/conf.

S3

**Hue** можно настроить для чтения и записи в настроенную учётную запись S3, а пользователи получают возможность автозаполнения и могут напрямую запрашивать и сохранять данные в S3 без какого-либо промежуточного перемещения/копирования в HDFS.

Создание таблиц Hive непосредственно из S3

Hue Metastore Import Data Wizard может создавать внешние таблицы **Hive** непосредственно из каталогов данных в S3. Это позволяет запрашивать данные S3 через SQL из Hive без перемещения или копирования данных в HDFS или Hive Warehouse.

Чтобы создать внешнюю таблицу **Hive** из S3, перейдите в приложение Metastore, выберите нужную базу данных и затем щёлкните значок **Create a new table from a file** в правом верхнем углу.

Введите имя таблицы и необязательное описание, а в инструменте выбора файлов **Input File or Directory** выберите файловую систему S3A, перейдите к родительскому каталогу, содержащему желаемые файлы данных, и нажмите кнопку **Select this folder**. В раскрывающемся списке **Load Data** должен автоматически быть выбран параметр **Create External Table**, который указывает, что эта таблица будет напрямую ссылаться на каталог внешних данных.

Выберите параметры разделителя и определения столбца для входных файлов и, наконец, нажмите **Create Table**, когда будете готовы создать таблицу **Hive**. После создания вы должны увидеть детали вновь созданной таблицы в Metastore.

Сохранение результаты запроса в S3

Мы можем перейти в Редактор **Hive** и запросить данные непосредственно из S3. Эти запросы могут объединять таблицы и объекты, которые поддерживаются S3, HDFS или обоими. Затем результаты запроса можно легко сохранить обратно в S3.

Настройка S3

Чтобы добавить учётную запись S3 в **Hue**, вам необходимо настроить **Hue** с действительными учётными данными S3, включая идентификатор ключа доступа и секретный ключ доступа AWSCredentials.

Эти ключи могут безопасно храниться в скрипте, который выводит фактический ключ доступа и секретный ключ на стандартный вывод для чтения **Hue**. Чтобы использовать файлы скриптов, добавьте следующий раздел в файл конфигурации hue.ini:

|  |
| --- |
| [aws]  [[aws\_accounts]]  [[[default]]]  access\_key\_id\_script=/path/to/access\_key\_script  secret\_access\_key\_script= /path/to/secret\_key\_script  allow\_environment\_credentials=false  region=us-east-1 |

В качестве альтернативы (но не рекомендуется для производственной или безопасной среды) вы можете установить значения access\_key\_id и secret\_access\_key равными текстовым значениям ваших ключей:

|  |
| --- |
| [aws]  [[aws\_accounts]]  [[[default]]]  access\_key\_id=s3accesskeyid  secret\_access\_key=s3secretaccesskey  allow\_environment\_credentials=false  region=us-east-1 |

В качестве региона необходимо указать регион AWS, соответствующий учётной записи S3. По умолчанию этот регион будет установлен на us-east-1.

ADLS

Изучение ADLS в файловом браузере Hue

После успешной настройки **Hue** для подключения к ADLS мы можем просмотреть все доступные папки в учётной записи, щёлкнув корень ADLS. Отсюда мы можем просматривать существующие ключи (как каталоги, так и файлы) и создавать, переименовывать, перемещать, копировать или удалять существующие каталоги и файлы. Кроме того, мы можем напрямую загружать файлы в ADLS.

Создание таблиц Hive непосредственно из ADLS

Мастер импорта браузера таблиц **Hue** может создавать внешние таблицы Hive непосредственно из файлов в ADLS. Это позволяет запрашивать данные ADLS через SQL из Hive без перемещения или копирования данных в HDFS или Hive Warehouse.

Чтобы создать внешнюю таблицу Hive из ADLS, перейдите в Бразуер таблиц, выберите нужную базу данных и щёлкните значок плюса в правом верхнем углу. Выберите файл с помощью средства выбора файлов и перейдите к файлу в ADLS.

Сохранение результатов запроса в ADLS

Мы можем перейти в Редактор **Hive** и запросить данные непосредственно из ADLS. Эти запросы могут объединять таблицы и объекты, которые поддерживаются ADLS, HDFS или обоими. Затем результаты запроса можно сохранить обратно в ADLS.

Настройка ADLS

Чтобы добавить учётную запись ADLS в **Hue**, вам необходимо настроить **Hue** с действительными учётными данными ADLS, включая идентификатор клиента, пароль клиента и идентификатор клиента. Эти ключи можно безопасно хранить в скрипте, который выводит фактический ключ доступа и секретный ключ в стандартный вывод для чтения **Hue**. Чтобы использовать файлы скриптовв, добавьте следующий раздел в файл конфигурации hue.ini:

|  |
| --- |
| [adls]  [[azure\_accounts]]  [[[default]]]  client\_id\_script=/path/to/client\_id\_script.sh  client\_secret\_script=/path/to/client\_secret\_script.sh  tenant\_id\_script=/path/to/tenant\_id\_script.sh  [[adls\_clusters]]  [[[default]]]  fs\_defaultfs=adl://.azuredatalakestore.net  webhdfs\_url=https://.azuredatalakestore.net |

В качестве альтернативы (но не рекомендуется для производственных или безопасных сред) вы можете установить значение client\_secret в виде обычного текста:

|  |
| --- |
| [adls]  [[azure\_account]]  [[[default]]]  client\_id=adlsclientid  client\_secret=adlsclientsecret  tenant\_id=adlstenantid  [[adls\_clusters]]  [[[default]]]  fs\_defaultfs=adl://.azuredatalakestore.net  webhdfs\_url=https://.azuredatalakestore.net |

Файлы и каталоги

Вы можете использовать Браузер файлов для просмотра файлов ввода и вывода ваших заданий **MapReduce**. Как правило, вы можете сохранять файлы вывода в /tmp или в своём домашнем каталоге, если ваш системный администратор настроил его для вас. У вас должны быть соответствующие разрешения для управления файлами других пользователей.

Создание каталогов

1. В окне обозревателя файлов выберите **New —> Directory**.

В диалоговом окне **Create Directory** введите имя каталога и нажмите **Submit**.

Смена каталогов

1. Щёлкните имя каталога или точки родительского каталога в окне Браузера файлов.

Щёлкните значок карандаша, введите имя каталога и нажмите **Enter**.

Чтобы перейти в домашний каталог, щёлкните **Home** в поле пути вверху окна Браузера файлов.

|  |
| --- |
| **Примечание.**  Кнопка **Home** неактивна, если у вас нет домашнего каталога. Попросите администратора **Hue** создать для вас домашний каталог. |

Создание файлов

1. В окне Браузера файлов выберите **New —> File**.

В диалоговом окне **Create File** введите имя файла и нажмите **Submit**.

Загрузка файлов

Вы можете загружать текстовые и двоичные файлы в **HDFS**.

1. В окне Браузера файлов перейдите к каталогу, в который вы хотите загрузить файл.

Выберите **Upload —> Files**.

В открывшемся окне нажмите **Upload a File**, чтобы найти и выбрать файлы, которые вы хотите загрузить, а затем нажмите **Open.**

Скачивание файлов

Вы можете скачивать текстовые и двоичные файлы в **HDFS**.

1. В окне Браузера файлов установите флажок рядом с файлом, который вы хотите скачать.

Нажмите кнопку **Download**.

Загрузка zip-архивов

Вы можете распаковывать zip-архивы в **HDFS**. Архив извлекается в каталог с именем archivename.

1. В окне Браузера файлов перейдите к каталогу, в который вы хотите загрузить архив.

Выберите **Upload —> Zip file**.

В открывшемся окне нажмите **Upload a zip file**, чтобы найти и выбрать архив, который вы хотите загрузить, а затем нажмите **Open**.

Корзина для мусора

Браузер файлов поддерживает корзину для мусора **HDFS** (домашний каталог /.Trash) для хранения файлов и каталогов до их окончательного удаления.

Файлы в папке имеют полный путь к удалённым файлам (для возможности их восстановления при необходимости) и контрольные точки. Время, в течение которого файл или каталог остается в корзине, зависит от свойств **HDFS**.

Чтобы просмотреть корзину, в окне Браузера файлов щёлкните **Trash**.

Смена владельца, группы или разрешений

|  |
| --- |
| **Примечание.**  Только суперпользователь **Hadoop** может изменять владельца, группу или права доступа к файлу или каталогу. Пользователь, запускающий **Hadoop**, является суперпользователем **Hadoop**.  Учётная запись суперпользователя **Hadoop** не обязательно совпадает с учётной записью суперпользователя **Hue**.  Если вы создаёте пользователя **Hue** с тем же именем пользователя и паролем, что и суперпользователь **Hadoop**, то этот пользователь **Hue** может изменить владельца, группу или права доступа к файлу или каталогу. |

Владелец или группа

1. В окне Браузера файлов установите флажок рядом с файлом или каталогом, владельца или группу которого вы хотите изменить.

Выберите **Change Owner/Group** в меню **Options**.

В диалоговом окне **Change Owner/Group**:

Выберите нового пользователя из раскрывающегося меню **User**;

Выберите новую группу из раскрывающегося меню **Group**;

Установите флажок **Recursive**, чтобы применить изменение.

Нажмите **Submit**, чтобы внести изменения.

Разрешения

1. В окне Браузера файлов установите флажок рядом с файлом или каталогом, разрешения которого вы хотите изменить.

Нажмите кнопку **Change Permissions**.

В диалоговом окне **Change Permissions** выберите разрешения, которые вы хотите назначить, и нажмите **Submit**.

Просмотр и редактирование файлов

Вы можете просматривать и редактировать файлы как текстовые или двоичные.

Просмотр

1. В окне Браузера файлов щёлкните файл, который хотите просмотреть. Браузер файлов отображает первые 4096 байт файла в окне инструмента просмотра файлов.

Если размер файла превышает 4096 байт, используйте кнопки навигации по блоку (**First Block**, **Previous Block**, **Next Block**, **Last Block**) для прокрутки файла блок за блоком. В полях **Viewing Bytes** отображается диапазон байтов, который вы просматриваете в данный момент.

Чтобы переключить представление с текстового на двоичное, щёлкните **View as Binary**, чтобы просмотреть шестнадцатеричный дамп.

Чтобы переключить представление с двоичного на текстовое, щёлкните **View as Text**.

Редактирование

1. Если вы просматриваете текстовый файл, щёлкните **Edit File**. Браузер файлов отображает содержимое файла в окне Редактора файлов.

Отредактируйте файл и затем нажмите **Save** или **Save as**, чтобы сохранить файл.

Разрешения Sentry

Роли и права устройства безопасности можно редактировать непосредственно в интерфейсе безопасности.

SQL

Права доступа необходимо изменять через интерфейс **Hive**.

Solr

Права доступа Solr можно редактировать прямо через интерфейс.

Для перечисления наборов, запросов и создания наборов:

|  |
| --- |
| Admin=\*->action=\*  Collection=\*->action=\*  Schema=\*->action=\*  Config=\*->action=\* |

Задания

Браузер заданий (Job Browser) позволяет проверять несколько типов заданий, выполняемых в кластере.

Браузер заданий представляет задания и задачи уровнями. Верхний уровень — это список заданий, и вы можете просмотреть список задач этого задания. Затем вы можете просмотреть запуски задачи и свойства каждого запуска, такие как состояние, время начала и окончания, а также размер выходных данных. Для устранения проблем с неудачными заданиями вы также можете просмотреть логи каждого запуска.

Если есть выполняемые задания, Браузер заданий отобразит их перечень.

Дашборд

1. Чтобы отфильтровать задания по их статусу (например, **Running** или **Completed**), выберите необходимый статус в раскрывающемся меню **Job status**.

Для фильтрации по пользователю, который запустил задание, введите имя пользователя в поле **User Name**.

Чтобы выполнить фильтрацию по имени задания, введите имя в поле **Text**.

Чтобы очистить фильтры, выберите **All States** в раскрывающемся меню **Job status** и удалите любой текст в полях **User Name** и **Text**.

Просмотр информации о заданиях

|  |
| --- |
| **Примечание.**  На любом уровне вы можете просмотреть журнал объекта, щёлкнув значок изображения в столбце журналов. |

Чтобы просмотреть информацию об отдельном задании:

1. В окне Браузера заданий щёлкните **View** справа от задания, которое необходимо просмотреть. Здесь отображается страница **Job** для задания, а недавние задачи, связанные с выбарнным заданием, отображаются на вкладке **Tasks**.

Щёлкните вкладку **Metadata**, чтобы просмотреть метаданные для этого задания.

Щёлкните вкладку **Counters**, чтобы просмотреть метрики счётчика для задания.

Чтобы просмотреть подробную информацию о задачах, связанных с заданием:

1. В окне **Job** щёлкните ссылку **View All Tasks** справа над списком **Recent Tasks**. Здесь перечислены все задачи, связанные с выбранным заданием.

Щёлкните **Attempts** справа от задачи, чтобы просмотреть попытки запуска этой задачи.

Чтобы просмотреть информацию об отдельной задаче:

1. В окне **Job** щёлкните ссылку **View** справа от задания. Будут отображены попытки запуска, связанные с выбранной задачей.

Щёлкните вкладку **Metadata**, чтобы просмотреть метаданные для выбранной задачи. Будут отображены метаданные, связанные с задачей.

Чтобы просмотреть счётчики Hadoop для задачи, щёлкните вкладку **Counters**. Будут отображены метрики счётчика, связанные с задачей.

Чтобы вернуться в окно **Job** для этого задания, щёлкните номер задания на панели состояния в левой части окна.

Чтобы просмотреть сведения о попытке выполнения задачи:

1. В окне **Job Task** щёлкните ссылку **View** справа от попытки запуска задания. Метаданные, связанные с попыткой запуска, отображаются на вкладке **Metadata**.

Чтобы просмотреть счётчики Hadoop для попытки выполнения задачи, щёлкните вкладку **Counters**. Будут отображены метрики счётчика, связанные с попыткой запуска.

Чтобы просмотреть журналы, связанные с попыткой выполнения задачи, щёлкните вкладку **Logs**. Будут отображены журналы, связанные с попыткой выполнения задачи.

Чтобы вернуться к списку задач для текущего задания, щёлкните номер задачи на панели состояния в левой части окна.

Типы

YARN (MapReduce)

Все задания, выполняемые в Resource Manager, будут автоматически отображены. Информация будет извлечена соответственно, если задание было перемещено на один из серверов истории.

Планировщик задач (Scheduler)

Приложение **Scheduler** позволяет создавать процессы (workflows), а затем планировать их регулярное автоматическое выполнение.

Интерфейс мониторинга показывает прогресс, логи и разрешает такие действия, как приостановка или остановка workflow.

Приложение **Workflow** — это набор действий (actions), организованных в ориентированный ациклический граф (DAG).

Приложение включает в себя два типа узлов:

1. Поток управления (control flow) — начало, конец, разветвление, соединение, решение и уничтожение.

Действие (action) — задания (jobs).

Приложение **Coordinator** позволяет определять и выполнять повторяющиеся и взаимозависимые workflow jobs. Приложение **Coordinator** определяет условия, при которых может происходить выполнение workflow.

Приложение **Bundle** позволяет вам группировать набор (bundle) приложений **Coordinator**.

Workflows

В **Workflow Editor** вы можете легко выполнять операции с узлами действий (action nodes) и узлами управления (control nodes).

Узлы действий (Action nodes)

**Workflow Editor** поддерживает перетаскивание узлов действий (action nodes). Когда вы перемещаете действие над другими действиями, активные области подсвечиваются. Если в процессе есть действия, сами действия подсвечиваются, а также области над и под действиями. Если вы перетаскиваете действие на существующее действие, к рабочему процессу добавляются развлетвление и соединение.

1. Добавьте действия в процесс, нажав image, и перетащите действие в процесс. Отобразится экран редактирования узла:

Задайте свойства действия и нажмите **Done**. Каждое действие в процессе должно иметь уникальное имя.

Скопируйте действие, нажав кнопку image.

Действие открывается на экране редактирования узла.

Отредактируйте свойства действия и нажмите **Done**. Действие добавляется в конец процесса.

Удалите действие, нажав image.

Отредактируйте действие, нажав image.

Измените положение действия, щёлкнув левой кнопкой мыши и перетащив действие в новое место.

Узлы управления (Control nodes)

1. Создайте развлетвление и соединение, поместив действие поверх другого действия.

Удалите развлетвление и соединение, перетащив разветвление и поместив её вне области развлетвления.

Преобразуйте разветвление в решение, нажав image.

Чтобы отредактировать решение:

Нажмите image.

Заполните предикаты, которые определяют, какое действие следует выполнить, и выберите действие по умолчанию из раскрывающегося списка.

Щёлкните **Done**.

|  |
| --- |
| **Примечание.**  worfklows.xml и их job.properties также могут быть выбраны и выполнены напрямую через Браузер файлов. |

Расписание

В Менеджере-координаторе вы можете создавать приложения-координаторы и отправлять их на исполнение.

Редактирование координатора

В Редакторе координатора вы указываете свойства координатора и наборы данных, с которыми будет работать процесс, запланированный координатором, путём пошагового перехода между экранами в мастере. Вы также можете перейти к конкретным шагам и вернуться к ним, щёлкнув вкладки шагов над экранами.

1. Введите имя, выберите процесс, установите флажок **Is shared checkbox**, чтобы поделиться заданием, и нажмите **Next**. Если Редактор координатора был открыт после планирования процесса, процесс будет сформирован.

Выберите, сколько раз координатор будет запускаться для каждой указанной единицы, время начала и окончания координатора, часовой пояс времени начала и окончания и нажмите **Next**. Время должно быть выражено в формате UTC. Например, для запуска в 22:00 по PST укажите время начала 6:00 по UTC следующего дня (+8 часов) и установите в поле **Timezone** значение America/Los\_Angeles.

Щёлкните **Add**, чтобы выбрать входной набор данных, и щёлкните **Next**.

Нажмите **Add**, чтобы выбрать выходной набор данных. Щёлкните **Save coordinator** или **Next**, чтобы указать дополнительные настройки.

Чтобы предоставить доступ к координатору всем пользователям, установите флажок **Isshared**.

Заполните параметры для передачи, свойства, которые определяют, как долго координатор будет ждать до истечения тайм-аута, сколько координаторов может запускаться и ждать одновременно, а также политику выполнения координатора.

Щёлкните **Save coordinator**.

Создание набора данных

1. В Редакторе координатора выполните одно из следующих действий:

Щёлкните **here** на панели Inputs или Outputs в верхней части редактора.

На панели слева щёлкните ссылку **Create new**. Приступите к редактированию набора данных.

Редактирование набора данных

1. Введите имя для набора данных.

В полях **Start** и **Frequency** укажите, когда и как часто набор данных будет доступен.

В поле **URI** укажите шаблон URI для расположения набора данных. Чтобы создать URI и пути URI, содержащие даты и отметки времени, вы можете указать переменные ${YEAR}, ${MONTH}, ${DAY}, ${HOUR}, ${MINUTE}. Например: hdfs://foo:9000/usr/app/stats/${YEAR}/${MONTH}/data.

В поле **Instance** нажмите кнопку, чтобы выбрать экземпляр данных по умолчанию, один или диапазон экземпляров данных. Например, если frequency==DAY, окно последних скользящих 5 дней (не включая сегодняшний) будет выражено как начало: -5 и конец: -1. Установите флажок **advanced**, чтобы отобразить поле, в котором можно указать функцию EL-координатора.

Укажите часовой пояс даты начала.

В поле **Done flag** укажите флаг, который определяет, когда входные наборы данных больше не готовы.

Bundle

Пакет (bundle) состоит из набора расписаний.

Создание bundle

1. Нажмите **Create** в правом верхнем углу.

В поле **Name** введите имя.

В поле **Kick off time** выберите время начала.

Установите флажок **Is shared**, чтобы разрешить всем пользователям доступ к workflow.

Щёлкните **Save**. Откроется **Bundle Editor**. Приступите к редактированию bundle.

Редактирование bundle

В **Bundle Editor** вы указываете свойства, переходя по экранам мастера. Вы также можете перейти к конкретным шагам и вернуться к ним, щёлкнув вкладки шагов над экранами.

1. Щёлкните **Add**, чтобы выбрать coordinator, с которого будет запущен пакет.

Выберите время начала. Время должно быть выражено в формате UTC. Например, для запуска в 22:00 по PST укажите время начала 6:00 по UTC следующего дня (+8 часов).

Чтобы поделиться bundle со всеми пользователями, установите флажок **Is shared**.

Щёлкните **Next**, чтобы указать дополнительные параметры, или щёлкните **Save bundle**.

Введите параметры для передачи в **Oozie**.

Щёлкните **Save bundle**.