cedrusdata

Инструкция по эксплуатации CedrusData

ООО «Кверифай Лабс»
ОГРН 1217800163790
ИНН 7811766769
КПП 781101001

Введение 3
1. Основные понятия 3
2. Конфигурация 4
3. Запуск узла 7
4. Подключение к кластеру 7

Введение

CedrusData это высокопроизводительная распределенная платформа для сквозного анализа всех данных предприятия в облаке и on-premise через единую точку доступа с SQL интерфейсом.

CedrusData основана на распределенном SQL-движке Trino (https://trino.io/) и включает дополнительный функционал управления и мониторинга (в том числе в облачных инфраструктурах), улучшения производительности, профессиональную документацию и поддержку.

CedrusData может быть установлена из архива или из Docker-образа. Краткая инструкция по установке доступна в документации:

- Установка из архива: https://docs.cedrusdata.ru/latest/installation/archive.html.
- Установка из Docker-образа: https://docs.cedrusdata.ru/latest/installation/docker.html.

Данный документ представляет собой руководство по эксплуатации кластера CedrusData, установленного из архива.

1. Основные понятия

1.1. Узел

Кластер CedrusData состоит из одного или нескольких независимых процессов, называемых **узлами**, запущенных на одном или нескольких ЭВМ.

Узел может выполнять две ключевые роли - coordinator и worker.

Coordinator принимает SQL-запросы от пользователя, координирует выполнение SQL-запросов в кластере, отдает результаты SQL-запросов пользователю.

Worker выполняет отдельные части SQL-запроса на основе команд от coordinator и пересылает результаты другим узлам.

Для выполнения SQL-запросов, кластер CedrusData должен иметь как минимум один coordinator и worker. Coordinator может опционально выполнять роль worker, что позволяет выполнять SQL-запросы в кластере, состоящим из одного узла.

Подробное описание ролей узлов доступно в документации https://docs.cedrusdata.ru/latest/design.html.

1.2. Каталог

Плагин это дополнительный функционал, который может быть динамически подключен к узлу CedrusData. Плагин представляет собой набор скомпилированных Java-классов и иных ресурсов, необходимых для корректной работы целевого функционала. Плагины являются частью архива CedrusData, и находятся в директории `plugin/`.

Коннектор это часть функционала плагина, которая позволяет работать с источником данных определенного типа. Например, существует отдельный коннектор для работы с СУБД Postgres, а так же отдельный коннектор для работы с озерами данных под управлением Hive Metastore. Полный список коннекторов доступен в документации https://docs.cedrusdata.ru/latest/connector.html.

Каталог это подключение к конкретному источнику данных с использованием одного из коннекторов CedrusData. Каталог содержит набор схем, таблиц, представлений, процедур, и иных сущностей, которые можно использовать в SQL-запросах. Подробное описание каталогов доступно в документации https://docs.cedrusdata.ru/latest/design.html.

2. Конфигурация

По умолчанию узел сконфигурирован как coordinator с ролью worker и следующими каталогами:

- `tpch` - каталог TPCH коннектора со сгенерированными TPC-H данными (см. https://docs.cedrusdata.ru/latest/connector/tpch.html).
- `tpcds` - каталог TPCDS коннектора со сгенерированными TPC-DS данными (см. https://docs.cedrusdata.ru/latest/connector/tpcds.html).
- `memory` - каталог Memory коннектора для хранения данных в памяти (см. https://docs.cedrusdata.ru/latest/connector/memory.html).
- `jmx` - каталог JMX коннектора для доступа к MBeans (см. https://docs.cedrusdata.ru/latest/connector/jmx.html).

Если вы хотите изменить роли узла или иные его параметры или изменить состав каталогов, необходимо провести конфигурацию.

При необходимости изменения конфигурации уже запущенного узла, необходимо остановить узел, изменить конфигурацию, запустить узел.

2.1. Конфигурация узла

Базовая конфигурация узла находится в файле `etc/config.properties`. Подробное описание всех доступных параметров конфигурации узла доступно в документации https://docs.cedrusdata.ru/latest/admin.html.

2.1.1. Coordinator

Для конфигурации узла как coordinator, необходимо задать следующие свойства:

```
coordinator=true
http-server.http.port=<указать локальный порт>
discovery.uri=<указать URL текущего узла со схемой http или https>
```

- `coordinator=true` - задает узлу роль coordinator.
- `http-server.http.port` - указывает порт, который будет использовать узел для коммуникации с другими узлами.
- `discovery.uri` - указывает URL, по которому происходит регистрация узла в кластере. Для узла типа coordinator URL должен указывать на сетевой интерфейс текущего узла, а портом должно являться значение, указанное в параметре `http-server.http.port`.

Чтобы coordinator так же имел роль worker и мог выполнять SQL-запросы, необходимо добавить параметр `node-scheduler.include-coordinator` со значением `true`.

Пример 1: конфигурации coordinator узла, который находится на компьютере с IP адресом `192.168.1.25`, использует порт `8080`, и может выполнять SQL-запросы:

```
coordinator=true
http-server.http.port=8080
discovery.uri=http://192.168.1.25:8080
node-scheduler.include-coordinator=true
```

2.1.2. Worker

Для конфигурации узла как worker, необходимо задать следующие свойства:

```
coordinator=false
http-server.http.port=<указать локальный порт>
discovery.uri=<указать значение discovery.uri coordinator>
```

- `coordinator=false` - задает узлу роль worker.
- `http-server.http.port` - указывает порт, который будет использовать узел для коммуникации с другими узлами.
- `discovery.uri` указывает URL, по которому происходит регистрация узла в кластере. Для узла типа worker значение данного параметра должно совпадать со значением этого же параметра на узле coordinator.

Чтобы coordinator так же имел роль worker и мог выполнять SQL-запросы, необходимо добавить следующее свойство со значением `true`:

```
node-scheduler.include-coordinator=true
```

Пример 2: конфигурация worker узла, который находится на компьютере с IP адресом `192.168.1.26`, использует порт `8080`, и взаимодействует с coordinator узлом, расположенным на компьютере с IP адресом `192.168.1.25`:

```
coordinator=false
http-server.http.port=8080
discovery.uri=http://192.168.1.25:8080
```

2.1.3. Несколько узлов на одном компьютере

CedrusData допускает запуск нескольких узлов на одном компьютере. Такая конфигурация чаще всего используется для тестирования и изучения возможностей продукта.

Для запуска нескольких узлов на одном компьютере необходимо убедиться, что узлы работают на разных портах.

Пример конфигурации coordinator узла, который находится на компьютере с IP адресом `192.168.1.25`, использует порт `8080`:

```
coordinator=true
http-server.http.port=8080
discovery.uri=http://192.168.1.25:8080
```

Пример 3: конфигурация worker узла, запущенного на том же компьютере, который использует порт `8081`:

```
coordinator=false
http-server.http.port=8081
discovery.uri=http://192.168.1.25:8080
```

2.2. Конфигурация каталогов

В момент запуска узел CedrusData формирует список каталогов, к данным которых можно запускать SQL-запросы. Для этого узел составляет список файлов с расширением `*.properties` в директории `etc/catalog`. Для каждого найденного файла CedrusData создает каталог, имя которого совпадает с именем файла без расширения. Например, файлу `my_postgres.properties` будет соответствовать каталог `my_postgres`.

В каждом найденном файле должно присутствовать свойство `connector.name`, которое определяет коннектор, который будет использован для данного каталога. Каждый из доступных коннекторов имеет уникальное имя. Полный список коннекторов доступен в документации https://docs.cedrusdata.ru/latest/connector.html.

В дополнение к `connector.name`, файл конфигурации каталога может содержать специфичные для коннектора параметры. Например, коннектор к озеру данных на основе Hive Metastore требует наличие адреса процесса Hive Metastore, а коннектор к Postgres требует наличие строки подключения к экземпляру Postgres. Некоторые коннекторы не имеют обязательных параметров конфигурации. В таком случае в файле `*.properties` достаточно указать имя коннектора в параметре `connector.name`. Список доступных параметров доступен в документации коннектора.

Для создания нового каталога необходимо создать файл `etc/catalog/<имя_каталога>.properties`, указать имя коннектора в параметре `connector.name`, указать иные необходимые для работы коннектора параметры, после чего запустить (или перезапустить) узел.

Для изменения параметров каталога, необходимо внести требуемые изменения в файл `etc/catalog/<имя_каталога>.properties` и перезапустить узел.

Для удаления каталога необходимо удалить файл `etc/catalog/<имя_каталога>.properties` и перезапустить узел.

Пример 4: конфигурации каталога с названием *hive*, который использует Hive коннектор (https://docs.cedrusdata.ru/latest/connector/hive.html) для подключения к озеру данных под управлением Hive Metastore. Параметр `hive.metastore.uri` специфичен для Hive Connector, и задает IP адрес компьютера и порт, на котором запущен Hive Metastore. Параметр `hive.allow-drop-table` разрешает выполнение операции `DROP TABLE` над таблицами Hive Metastore, что запрещено по умолчанию.

`etc/catalog/hive.properties`

```
connector.name=hive
hive.metastore.uri=thrift://192.168.1.25:9083
hive.allow-drop-table=true
```

Пример 5: конфигурация каталога с названием *jmx*, который использует JMX коннектор (https://docs.cedrusdata.ru/latest/connector/hive.html). Данный коннектор не имеет обязательных параметров конфигурации, поэтому файл содержит только имя коннектора:

`etc/catalog/jmx.properties`

```
connector.name=jmx
```

3. Запуск узла

Для создания кластера CedrusData необходимо запустить один или несколько узлов, среди которых должен быть минимум один coordinator и worker.

Управление жизненным циклом узла происходит с помощью утилиты `bin/launcher`.

- `bin/launcher run` - запускает узел в текущем процессе консоли.
- `bin/launcher start` - запускает узел в фоновом режиме и печатает в консоли PID процесса.
- `bin/launcher restart` - перезапускает запущенный узел.
- `bin/launcher stop` - останавливает запускает узел.
- `bin/launcher kill` - принудительно останавливает запускает узел.
- `bin/launcher --help` - отображает справку об использовании утилиты.

По умолчанию, используется конфигурация узла в директории `etc/`. Для использования конфигурации из другой директории необходимо передать параметр `--etc-dir=<путь>`.

По умолчанию, операционные данные узла записываются в директорию `data/cedrus/`. Для использования другой директории необходимо передать параметр `--data-dir=<путь>`.

Для управления узлом, запущенного с явно заданными значениями параметров `--etc-dir` или `--data-dir`, необходимо передать в команду `restart`, `stop` или `kill` эти же параметры.

Пример 6: запуск узла с конфигурацией из директории по умолчанию `etc/`, который будет записывать операционные данные в директорию по умолчанию `deta/cedrus`:

```
bin/launcher start
```

Пример 7: запуск узла с конфигурацией из директории `etc/coordinator`, который будет записывать операционные данные в директорию `data/coordinator`:

```
bin/launcher start --etc-dir=etc/coordinator --data-dir=data/coordinator
```

Пример 8: остановка узла с конфигурацией из директории `etc/coordinator`, который записывает операционные данные в директорию `data/coordinator`:

```
bin/launcher stop --etc-dir=etc/coordinator --data-dir=data/coordinator
```

4. Подключение к кластеру

Для подключения к кластеру CedrusData и выполнения SQL-запросов можно использовать утилиту командной строки *Trino CLI*, либо JDBC драйвер.

4.1. Trino CLI

Trino CLI это утилита командной строки, которая позволяет подключиться к кластеру CedrusData и выполнить SQL-запрос. Полная документация утилиты расположена по адресу https://docs.cedrusdata.ru/latest/client/cli.html.

- *bin/trino* - подключается к кластеру в интерактивном режиме.
- *bin/trino --execute* "<*sql*>" - подключается к кластеру, выполняет SQL-запрос, печатает результат в консоль.
- *bin/trino --help* - печатает справку по работе с командой.

Пример 9: подключиться к кластеру, запущенному на текущем узел, и выполнить запрос к таблице `orders` в схеме `sf1` в каталоге `tpch`:

```
bin/trino --execute "select count(*) from tpch.sf1.orders"
```

4.2. JDBC драйвер

Для подключения к кластеру с помощью JDBC драйвера необходимо задать URL в формате `jdbc:trino://<host>:<port>/[<catalog>]/[<schema>]` и непустое имя пользователя.

- `<host>` - имя компьютера, на котором запущен coordinator узел.
- `<port>` - порт, который использует coordinator узел.
- `<catalog>` - каталог по умолчанию (опционально).
- `<schema>` - схема по умолчанию (опционально).

и порт coordinator узла, указанный в `etc/config.properties:http-server.http.port`.

Пример 10: строка подключения к coordinator узлу, запущенному на компьютере с IP адресом `192.168.1.25` на порту `8080`.

```
jdbc:trino://192.168.1.25:8080
```

Пример 11: строка подключения к тому же coordinator узлу, с использованием каталога по умолчанию `tpch`, и схемы по умолчанию `sf1`.

```
jdbc:trino://192.168.1.25:8080/tpch/sf1
```