Сколковский институт науки и технологий

Руководство системного администратора

Платформа DATASKAI
версия: 0.0.1

Москва
2020

Оглавление

1 Введение **1**

2 Требования к аппаратному обеспечению **2**

3 Требования к программному обеспечению **3**

4 Установка **4**
4.1 Подсистема сбора данных . . . 4
4.1.1 Установка внешних зависимостей . . . 4
4.1.1.1 Шина данных . . . 4
4.1.1.2 Шлюз импорта данных . . . 4
4.1.2 Проверка внешних зависимостей . . . 4
4.1.3 Инструкции по установке . . . 5
4.1.3.1 Source-коннекторы . . . 5
4.1.3.2 Обработчики данных . . . 5
4.1.3.3 Sink-коннекторы . . . 5
4.2 Подсистема хранения данных . . . 5
4.2.1 Установка внешних зависимостей . . . 5
4.2.1.1 PostgreSQL . . . 5
4.2.1.2 HDFS . . . 5
4.2.2 Проверка внешних зависимостей . . . 5
4.2.3 Инструкции по установке . . . 6
4.2.3.1 Data Service . . . 6
4.3 Подсистема обработки данных . . . 7
4.4 Подсистема анализа данных . . . 7
4.4.1 External dependencies . . . 7
4.4.2 Checking external dependencies . . . 7

5 Администрирование **8**
5.1 Подсистема сбора данных . . . 8
5.2 Подсистема хранения данных . . . 8
5.2.1 Data Service . . . 8
5.2.1.1 Описание файла конфигурации . . . 8
5.2.1.2 Запуск сервиса . . . 9
5.2.1.3 Остановка сервиса . . . 10
5.2.1.4 Удаление сервиса . . . 10
5.3 Подсистема обработки данных . . . 10
5.4 Подсистема анализа данных . . . 10

6 Устранение неполадок **11**

7 Дополнительная информация **12**

Глава 1

Введение

Глава 2

Требования к аппаратному обеспечению

Глава 3

Требования к программному обеспечению

Глава 4

Установка

4.1 Подсистема сбора данных

Установка подсистемы включает в себя установку следующих внешних зависимостей и компонентов.

Внешние зависимости:

- Hortonworks Data Platform
- Apache Kafka
- Confluent REST Proxy

Компоненты:

- Source-коннекторы
- Обработчики данных
- Sink-коннекторы

4.1.1 Установка внешних зависимостей

4.1.1.1 Шина данных

Установите Apache Kafka версии 2.0.0 или выше. Рекомендованная процедура установки:

- Установите Hortonworks Data Platform
- Установите и сконфигурируйте Apache Kafka следуя данной инструкции

4.1.1.2 Шлюз импорта данных

Установите Confluent REST Proxy используя инструкцию по установке.

4.1.2 Проверка внешних зависимостей

Примечание: добавьте сюда описание

4.1.3 Инструкции по установке

4.1.3.1 Source-коннекторы

Установка source-коннекторов производится отдельно в соответствии с документацией к каждому коннектору.

4.1.3.2 Обработчики данных

Установка обработчиков данных производится отдельно в соответствии с документацией к каждому обработчику.

4.1.3.3 Sink-коннекторы

Установка sink-коннекторов производится отдельно в соответствии с документацией к каждому коннектору.

4.2 Подсистема хранения данных

Установка подсистемы включает в себя установку следующих внешних зависимостей и компонентов.

Внешние зависимости:

- PostgreSQL
- Hortonworks Data Platform

Компоненты:

- Документация Data Service

4.2.1 Установка внешних зависимостей

Набор используемых хранилищ данных может различаться в зависимости от прикладной области. Минимальная конфигурация требует установки PostgreSQL.

4.2.1.1 PostgreSQL

Поддерживается PostgreSQL версии 11 и выше. Обратитесь к официальной документации PostgreSQL за инструкциями по установке.

4.2.1.2 HDFS

Поддерживается HDFS 3.0 и выше. Рекомендуемая процедура установки заключается в развертывании HDFS с использованием Hortonworks Data Platform.

4.2.2 Проверка внешних зависимостей

Примечание: добавьте сюда описание

4.2.3 Инструкции по установке

4.2.3.1 Data Service

Подготовка к установке

Приложение запускается в контейнере, поэтому первый шаг - это настройка docker.

При построении образа используются следующие образы docker:

- ubuntu:18.04 и выше
- openjdk:11 и выше
- python:3.6 и выше

Сборка/установка

Этапы сборки/установки:

1) This service should be built with Spring Framework. It uses automatic configuration using config file, so first step you must edit application-compose_api.yml configuration file (see configuration section below).

2) Перейдите в директорию platform-proto-java. Постройте файлы .jar с помощью gradle:

```
./gradlew :data-service-impl:clean && ./gradlew :data-service-impl:bootJar
```

3) Создайте директорию для логирования (в данном примере **workdir/logs**), постройте образ docker и запустите контейнер.

```
mkdir -p workdir/logs

cd data-service-impl

docker build -t idcp/data-service-api .

docker run -d \
--name data-service-api \
--mount type=bind,source=${PWD}/../workdir/logs,target=/logs \
-p 8170:8170 \
-e SPRING_PROFILES_ACTIVE='compose_api' \
-v ${PWD}/application-compose_api.yml:/application-compose_api.yml \
idcp/data-service-api
```

..также вы можете использовать docker-compose:

```
docker-compose -f docker-compose.<profile>.yml build data-service-api

docker-compose -f docker-compose.<profile>.yml up -d data-service-api
```

4) Проверяем статус контейнера:

```
docker ps --filter "name=data-service-api"
```

STATUS **Up** означает, что контейнер успешно запущен.

4.3 Подсистема обработки данных

4.4 Подсистема анализа данных

The installation of the subsystem includes the installation of the following external dependencies and components.

Dependencies:

- 1
- 2

Components:

- 1
- 2

4.4.1 External dependencies

4.4.2 Checking external dependencies

Глава 5

Администрирование

5.1 Подсистема сбора данных

5.2 Подсистема хранения данных

Note: include components install pages, e.g.:

5.2.1 Data Service

5.2.1.1 Описание файла конфигурации

Данный сервис использует файл конфигурации Spring Boot - application.yml .

```
spring:
  profiles:
    active: development
  datasource:
    driver-class-name: org.postgresql.Driver
    url: jdbc:postgresql://localhost:5432/${dataservice.namespace}
    username: <username>
    password: <password>
    hikari:
      minimum-idle: 1
      maximum-pool-size: 4
dataservice:
  data-catalog: DataCatalogPSQLImpl
  data-query: PSQLDataQueryImpl
  namespace: <namespace>
  schema: public
  proxy: psql
server:
  port : 8170
logging:
  level:
    platform: DEBUG
    org.springframework: INFO
    org.hibernate: ERROR
```

Файл конфигурации включает в себя:

КОНФИГУРАЦИЯ SPRING

PROFILES - позволяет определить, какие профили используют данную конфигурацию.
spring.profiles.active - указывает какой профиль активный (string, например, «development»).

Соединение с базой данных также можно настроить автоматически с помощью *DATASOURCE*
spring.datasource.driver-class-name - полное имя JDBC-драйвера (string, например, «org.postgresql.Driver»).
spring.datasource.url - JDBC url базы данных ‘jdbc:postgresql://<hostname>:<port>/${dataservice.namespace}‘ (url, например, ‘jdbc:postgresql://localhost:5432/${dataservice.namespace}‘).
spring.datasource.username - логин для подключения к базе данных (string, например, «username»).
spring.datasource.username - пароль для подключения к базе данных (string, например, «password»).
spring.datasource.hikari.* - настройка hikari .

КОНФИГУРАЦИЯ СЕРВИСА

dataservice.data-catalog - Реализация бина DataCatalogPSQLImpl доступа к каталогу данных.
dataservice.data-query - Реализация бина PSQLDataQueryImpl доступа к данным.
dataservice.namespace - название базы данных (string, например «test-data-api»).
dataservice.schema - название схемы (string, например «public»).
dataservice.proxy - название прокси (string, например «psql»).

КОНФИГУРАЦИЯ ВСТРОЕННОГО СЕРВЕРА

server.port - HTTP-порт сервера (int, например 8190).

НАСТРОЙКА ЛОГИРОВАНИЯ

logging.level.* - уровень логирования. Например, ‘logging.level.org.springframework=INFO‘.

5.2.1.2 Запуск сервиса

Для запуска выполните команду docker:

```
docker start data-service-api
```

..или используйте docker-compose:

```
docker-compose up -d data-service-api
```

После запуска можно отправить запрос к сервису, например:

```
curl 'http://127.0.0.1:8170/datasets'
```

Пример ответа:

```
[
    {"namespace":"aerophm","name":"aids_reports_01","versions":["1.0"]},
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    {"namespace":"aerophm","name":"aids_reports_02","versions":["1.0"]},
    {"namespace":"aerophm","name":"aircrafts","versions":["1.0"]}
]
```

5.2.1.3 Остановка сервиса

Для остановки сервиса используйте docker CLI:

```
docker stop data-service-api
```

..или используйте docker-compose:

```
docker-compose down data-service-api
```

5.2.1.4 Удаление сервиса

Чтобы удалить сервис используйте команду docker:

```
docker rm data-service-api
```

..или используйте docker-compose:

```
docker-compose rm data-service-api
```

5.3 Подсистема обработки данных

5.4 Подсистема анализа данных

Глава 6

Устранение неполадок

Глава 7

Дополнительная информация

Построено Sphinx, git commit: 01f31f4bd8134d549f4e6afab4830fdf3d9ed15b.