Руководство системного администратора программного обеспечения

Data Quality Framework

Аннотация

Настоящий документ представляет собой руководство системного администратора программного обеспечения (далее – ПО) Data Quality Framework (далее – ПО DQF), в котором определяется порядок установки, настройки и администрирования ПО.

Исключительные права на ПО DQF принадлежат ООО «Клин Дейта».

Перед использованием необходимо ознакомиться с документом «Руководство по развертыванию и установке программного обеспечения Data Quality Framework».

Оглавление

1. Область применения

Настоящий документ предназначен для пользователей, в должностные обязанности которых входят функции по администрированию ПО DQF.

Настоящее Руководство системного администратора предназначено для:

- подготовки к установке и настройке дистрибутива ПО DQF;
- мониторинга событий в сбои работоспособности ПО DQF;
- управления настройками ПО DQF.

1.1. Требования к уровню подготовки персонала

К уровню подготовки администратора ПО DQF предъявляются следующие требования:

- знание технической и рабочей документации на ПО DQF;
- наличие опыта работы по установке, настройке и поддержке ПО DQF;
- базовые знания Linux, Java, работы с консолью в оболочке Bash (написание командных срок);
- базовые навыки администрирования ПО RabbitMQ, Kafka, Docker.

1.2. Управление доступом

Установка и администрирование ПО DQF осуществляются системным администратором, имеющим доступ к программно-аппаратной инфраструктуре, на которой будет произведена установка ПО DQF.

2. Общие сведения

Data Quality Framework (DQF) – программное обеспечение, предназначенное для выявления ошибок и предотвращения внесения недостоверных (противоречивых) данных. Функциональные возможности ПО DQF позволяют с помощью специальных правил и алгоритмов произвести верификацию данных проверяемого объекта и выявить ошибки, произвести настройку проверок данных без доработки программного обеспечения в соответствии с предметной областью и заданной конфигурацией (бизнес-логики) исполняемой проверки.

ПО DQF позволяет:

– выполнять преднастроенные алгоритмы и формировать результаты их выполнения в виде сведений об ошибках;

– осуществлять контроль выполнения алгоритмов;

– предоставлять результаты проверок;

– осуществлять запрос дополнительных данных, в том числе из внешних информационных систем, необходимых для проведения проверок.

3. Установка и настройка ПО DQF

3.1. Подготовка ИТ–инфраструктуры

Для установки ПО DQF ИТ-инфраструктура в части аппаратной платформы и сетевой инфраструктуры должна быть подготовлена в соответствии с требованиями, приведенными в данном разделе.

3.1.1. Требования к аппаратной платформе

Для установки ПО DQF необходимо обеспечить настройку сервера приложения в следующей конфигурации:

- Процессор:
  - Intel Xeon Ice Lake или новее (Intel Xeon Silver 4310 и более старшие версии) или AMD EPYC второго поколения или новее (AMD EPYC 7502P и более старшие версии);
  - от 6 ядер.
- ОЗУ:
  - от 16 Гб.

3.1.2. Требования к сетевой инфраструктуре

Пропускная способность канала между сервером приложений и внешними информационными системами, направляющими запросы в ПО DQF, должна составлять не менее 1 Гбит/с.

Пропускная способность канала между сервером приложений и брокером сообщений должна составлять не менее 1 Гбит/с.

3.2. Установка ПО DQF

Правила и порядок установки ПО DQF приведены в документе «Руководство по развертыванию и установке программного обеспечения Data Quality Framework». Для перезапуска ПО DQF используется порядок действий, приведенный в разделе 4.3 документа «Руководство по развертыванию и установке программного обеспечения Data Quality Framework».

3.3. Обеспечение отказоустойчивости ПО DQF

Отказоустойчивая схема развертывания ПО DQF подразумевает установку двух экземпляров программного обеспечения, которые работают одновременно. Поступающие запросы от внешних информационных систем в брокер очередей балансируются между

экземплярами ПО DQF, а при отказе одного экземпляров программного обеспечения все новые запросы направляются на второй. При восстановлении работоспособности первого экземпляра ПО DQF балансировщик распределяет нагрузку между обоими экземплярами.

Для реализации отказоустойчивой схемы требуется:

- приобретение дополнительной лицензии на установку экземпляра ПО DQF;
- два сервера приложений;
- установка и настройка балансировщик нагрузки - HAproxy версии 2.0.

3.4. Логирование в ПО DQF

Для получения логов при использовании ПО DQF в консоли сервера необходимо ввести команду: docker logs dqf-engine-standalone, где dqf-engine-standalone, является названием контейнера приложения.

Логирование ведется в стандартный поток ввода-вывода, просмотр которого доступен из запущенного контейнера приложения.

Пример журнала логов ПО DQF:

```
2022-09-09 19:18:25.326 DEBUG 1 --- [ntContainer#0-1]
r.c.e.e.algorithm.spi.AlgorithmFactory   : Опрашиваю, поддерживает ли фабрика
алгоритм: unknown_algorithm_id
2022-09-09 19:18:25.327 ERROR 1 --- [ntContainer#0-1]
c.d.e.s.c.ComplexAlgorithmFileRepository : Ошибка поиска алгоритма
unknown_algorithm_id при чтении директории /opt/dqf/engine/complex

java.nio.file.NoSuchFileException:
/opt/dqf/engine/complex/unknown_algorithm_id.ca.yaml
```

3.5. Управление файлами конфигураций комплексных алгоритмов

Администратору ПО DQF доступно управление файлами конфигураций комплексных алгоритмов ПО DQF, а именно:

- загрузка файлов;
- редактирование файлов;
- удаление файлов.

Управление файлами конфигураций комплексных алгоритмов осуществляется в директории, все вносимые изменения (загрузка/редактирование/удаление комплексных алгоритмов) отображаются в директории и в ПО DQF.

Для загрузки новых файлов конфигураций комплексных алгоритмов администратору ПО DQF необходимо скопировать их в директорию, указанную при установке ПО DQF. Правила формирования настроек при установке ПО DQF приведены в документе «Руководство по развертыванию и установке программного обеспечения Data Quality Framework». Например, если смонтированная директория при установке ПО DQF

была /etc/configurations и в данной директории доступны два файла: check_balance.ca.yaml, check_limits.ca.yaml, - соответственно в ПО DQF будут отображаться загруженные конфигурации комплексных алгоритмов check_balance и check_limits, определенные в соответствующих файлах. Так, при размещении нового файла конфигурации в директории, новый комплексный алгоритм будет доступен в ПО DQF.

Параметры файлов конфигураций комплексных алгоритмов следующие:

– формат файла – Yaml 1.2.2;

– кодировка - UTF-8;

– расширение - .ca.yaml.

3.6. Формирование сообщений-запросов и получение сообщений-ответов с использованием брокера очередей

Ниже приведено описание формата сообщения-запроса (для ПО DQF – входящего сообщения). Формат заголовка сообщения приведен в таблице (Таблица 1), формат тела сообщения приведен в таблице (Таблица 2). Тело является JSON-объектов в кодировке UTF-8.

Таблица 1 - Формат заголовка сообщения-запроса (для ПО DQF - входящего сообщения)

Название	Обязателен	Тип	Пример	Описание
message_id	нет	Строка	1	Используется для журналирования.
correlation_id	нет	Строка	1	Если передан в сообщении-запросе - будет возвращен в сообщении-ответе.

Таблица 2 - Формат тела сообщения-запроса (для ПО DQF - входящего сообщения)

Название	Обязателен	Тип	Пример(ы)	Описание
algorithmId	да	Строка	my_complex_algo	Идентификатор комплексного алгоритма для выполнения проверки.
objects	да	Любой валидный JSON, не являющийся null	1. {"attr": "value"} 2. ["arrayValue"] 3. 42 4. 3.14 5. true	Объекты для проверки. Будут переданы в контекст исполнения.

Название	Обязателен	Тип	Пример(ы)	Описание
			6. false	

Пример сообщения-запроса (входящего сообщения для ПО DQF):

```
{
  "algorithmId": "check_balance",
  "objects": {
    "balance": 42
  }
}
```

Ниже приведено описание формата сообщения-ответа (для ПО DQF – исходящего сообщения). Формат заголовка сообщения приведен в таблице (Таблица 3**Таблица *1***), формат тела сообщения приведен в таблице (Таблица 4**Таблица *2***). Тело является JSON-объектов в кодировке UTF-8.

Таблица 3 - Формат заголовка сообщения-ответа (для ПО DQF - исходящего сообщения)

Название	Обязателен	Тип	Описание
message_id	да	Строка	Идентификатор сообщения-ответа. Будет случайно сгенерирован перед отправкой.
correlation_id	нет	Строка	Будет равен соответствующему заголовку во сообщении-запросе.
content_encoding	да	Строка	Кодировка сообщения-ответа. Всегда UTF-8
content_type	да	Строка	Тип тела сообщения-ответа. Всегда application/json

Таблица 4 - Формат тела сообщения-ответа (для ПО DQF - исходящего сообщения)

Название	Обязателен	Тип	Описание
correlationId	да, может быть null	Строка	Идентификатор корреляции. Равен значению заголовка correlation_id, сообщения-запроса, если был передан, иначе null.
status	да	Логический литерал	Статус выполнения комплексного алгоритма. Значение и его интерпретация зависит от конкретного комплексного алгоритма.

Название	Обязателен	Тип	Описание
variables	да	json-объект	Переменные, которые были заполнены в рамках выполнения комплексного алгоритма. Значение и интерпретация зависят от конкретного комплексного алгоритма.

Пример сообщения-ответа (исходящего сообщения для ПО DQF):

```
{
  "correlationId": null,
  "status": true,
  "variables": {
    "limit": 50000,
    "limitCurrency": "RUB"
  }
}
```

При возникновении проблем при обработке запроса (например, был передан неизвестный идентификатор комплексного алгоритма или в файле-конфигурации комплексного алгоритма была допущена ошибка) в очередь для ответов будет отправлено сообщение с информацией об ошибке. Формат заголовка сообщения об ошибке приведен в таблице (Таблица 5**Таблица *1***), формат тела сообщения об ошибке приведен в таблице (Таблица 6**Таблица *2***). Тело является JSON-объектов в кодировке UTF-8.

Таблица 5 - Формат заголовка сообщения об ошибке

Название	Обязателен	Тип	Описание
message_id	да	Строка	Идентификатор сообщения-ответа. Будет случайно сгенерирован перед отправкой.
correlation_id	нет	Строка	Будет равен соответствующему заголовку в сообщении-запросе.
content_encoding	да	Строка	Кодировка сообщения-ответа. Всегда UTF-8
content_type	да	Строка	Тип тела сообщения-запроса. Всегда application/json

Таблица 6 - Формат тела сообщения об ошибке

Название	Обязателен	Тип	Описание
correlationId	да, может быть null	Строка	Идентификатор корреляции. Равен значению заголовка correlation_id, сообщения-запроса, если был передан, иначе null.

Название	Обязателен	Тип	Описание
exception	да	json-объект	Информация об ошибке с сообщением и трейсингом для поиска места возникновения.

Пример сообщения об ошибке (приведена только часть сообщения для сокращения объема документа (приведен первый и последний элемент массива stackTrace)):


```
{
  "correlationId": null,
  "exception": {
    "message": "Ошибка в позиции (3:17@37 - 3:41@61): Пропущена закрывающая
кавычка: `'`\n| '$.objects.balance > 0\n> ^----------------------^\n",
    "localizedMessage": "Ошибка в позиции (3:17@37 - 3:41@61): Пропущена
закрывающая кавычка: `'`\n| '$.objects.balance > 0\n> ^----------------------
^\n"
    "suppressed": [],
    "cause": {
      "cause": null,
      "stackTrace": [
         {
           "classLoaderName": null,
           "moduleName":null,
           "moduleVersion":null,
           "methodName":"parse",
           "fileName":"StringTextExpressionParser.java",
           "lineNumber":92,

"className":"ru.cleandata.execution.engine.impl.expression.text.parser",
           "nativeMethod":false
         },
         {
           "classLoaderName": null,
           "moduleName":java.base,
           "moduleVersion":"11.0.15",
           "methodName":"run",
           "fileName":"Thread.java",
           "lineNumber":829,
           "className":"java.lang.Thread",
           "nativeMethod":false
         }
       ],
       "configLocation":{
         "start":{
           "lineNumber":3,
           "columnNumber":17,
           "streamOffset":37
         },
         "end":{
           "lineNumber":3,
           "columnNumber":41,
           "streamOffset":61
         }
       }
    }
  }
}
```