ИНФРАСТРУКТУРА ЭЛЕКТРОННОГО ПРАВИТЕЛЬСТВА

ВЫПОЛНЕНИЕ РАБОТ ПО РАЗВИТИЮ ТИПОВОГО ТИРАЖИРУЕМОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ВИТРИН ДАННЫХ

Витрина данных НСУД

Руководство администратора ПО «Витрина данных Лайт»

Версия 1.13.0

Листов 77

Москва, 2024

СОДЕРЖАНИЕ

АННОТАЦИЯ

В данном программном документе приведено «Руководство администратора» программного обеспечения «Витрина данных Лайт» (далее – программа), предназначенного для загрузки публикуемых данных в отдельную базу данных на стороне поставщика данных, а также для формирования отдельной базы данных в соответствии с результатами выполнения запросов на предоставление или репликации данных со стороны получателя данных. «Витрина данных Лайт», в отличии от ПО «Витрина данных НСУД», имеет ограниченный функционал, невысокие требования к аппаратному обеспечению и предназначена для небольших организаций.

В разделе «Общие сведения о программе» указаны назначение и возможности программы и сведения о технических и программных средствах, обеспечивающих выполнение данной программы.

В разделе «Настройка программы» приведены описания действий по настройке программы на состав технических и программных средств.

В разделе «Резервное копирование» приведены рекомендации по выполнению планового резервного копирования, контролю результатов резервного копирования, восстановлению информации из резервных копий.

В разделе «Дополнительные возможности» описание дополнительных разделов функциональных возможностей программы и способов их выбора.

В разделе «Сообщения администратору» приведены описания сообщений, выдаваемых в ходе выполнения настройки, проверки программы, а также в ходе выполнения программы, описание их содержания и действий, которые необходимо предпринять по этим сообщениям.

В разделе «REST API» приведены описания взаимодействия конечных точек со сторонними программами для обработки информации Программы.

В приложении к руководству администратора приведены настройки межсетевого экрана ОС Linux, а также инструкция по эксплуатации модуля CSV-Uploader.

1 ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММЕ

1.1 Обозначение и наименование программы

Полное наименование: Типовое тиражируемое программное обеспечение витрин данных конфигурации «Лайт».

Условное обозначение: ПО «Витрина данных Лайт».

1.2 Назначение программы

Национальная система управления данными (далее – НСУД) представляет собой систему, состоящую из взаимосвязанных элементов информационно-технологического, организационного, методологического, кадрового и нормативно-правового характера и обеспечивающую достижение целей и выполнение задач, обозначенных в Концепции Национальной системы управления данными, утвержденной распоряжением Правительства Российской Федерации от 3 июня 2019 года № 1189-р.

НСУД предназначена для управления информацией, содержащейся в информационных системах органов и организаций государственного сектора, а также в информационных ресурсах, созданных в целях реализации полномочий органов и организаций государственного сектора (далее – государственные данные) и для осуществления информационного обмена между Поставщиками и Получателями данных, присоединившимися к НСУД (далее – Участники НСУД).

Управление процессами информационного обмена между Участниками НСУД осуществляется средствами федеральной государственной информационной системы «Единая информационная платформа Национальной системы управления данными» (далее – ФГИС «ЕИП НСУД»).

Для передачи данных между Участниками НСУД используется среда взаимодействия НСУД, состоящая из Системы межведомственного электронного взаимодействия 3.0 (далее – СМЭВ) и (или) подсистемы обеспечения доступа к данным СМЭВ (далее – ПОДД СМЭВ) (СМЭВ 4.0), обеспечивающих транспорт и процессинг данных, а также агентов ПОДД СМЭВ, устанавливаемых на стороне Участников НСУД.

Программа является частью НСУД и предназначена для загрузки публикуемых данных в отдельную БД на стороне Поставщика данных. Программа представляет собой типовое программное обеспечение, устанавливаемое на стороне поставщиков данных.

1.3 Возможности программы

Программа обеспечивает выполнение следующих функций:

- автоматическая настройка взаимосвязей между компонентами программы;
- автоматический запуск всех необходимых компонентов программы после установки;
- автоматическая настройка витрины и структуры ее таблиц на основании содержимого XML-файла, загружаемого через пользовательский web-интерфейс;
- выгрузка шаблона через графический интерфейс (для упрощения процесса

подготовки загружаемых данных);

- загрузка данных в витрину:
  - через графический интерфейс;
  - REST API;
  - файловый обмен.
- настройка параметров работы витрины через графический интерфейс;
- выполнение запросов на предоставление данных в соответствии с протоколом ПОДД через механизмы СМЭВ ПОДД.

1.4 Обеспечивающие технические и программные средства

Сведения о рекомендуемых технических и программных средствах описаны в документе «Техническое описание системы».

2 НАСТРОЙКА ПРОГРАММЫ

2.1 Настройка на состав технических средств

Сервер, на которым будет установлена программа, должен соответствовать техническим характеристикам, указанным в документе «Техническим описание системы».

2.2 Настройка на состав программных средств

Все предварительные действия необходимые перед установкой программы, процесс установки и проверка корректной установки программы описан в документе «Руководство по установке».

2.2.1 Настройка ProStore

Настройка ProStore заключается в настройке составляющих его компонентов и осуществляется путём внесения изменений в описание файла *application.yml* – основной конфигурационный файл, в котором задана логика и порядок работы *Сервиса исполнения запросов* (`query-execution`). Программная конфигурация, в частности, включает сетевые адреса, сетевые порты и идентификаторы компонентов для взаимосвязи между ними, пути на дисковых пространствах для обработки пользовательских и служебных данных, а также метаданных.

Внимание:
Не рекомендуется менять настройки при первоначальной установке программы с помощью Ansible. Все необходимые настройки для корректной работы будут сконфигурированы автоматически.

2.2.1.1 Настройка Сервиса исполнения запросов (query-execution)

Файл *application.yml* – основной конфигурационный файл, в котором задана логика и порядок работы Сервиса исполнения запросов (query-execution). Для первоначальной установки используйте значения «по умолчанию».

Пример полного yml-файла со всеми конфигурируемыми атрибутами, приведен в Приложении 1 (см. Приложение 1).

1. Настройка ProStore:

   – `DTM_CORE_PLUGINS_ANALYTICAL` - настройка профилей приоритетности СУБД для запросов аналитики;
   – `DTM_CORE_PLUGINS_DICTIONARY` - настройка профилей приоритетности СУБД для запросов ключ-значение;
   – `DTM_CORE_PLUGINS_UNDEFINED` - настройка профилей приоритетности СУБД для не указанной категории запросов;
   – `DTM_CORE_HTTP_PORT` - номер порта, на который Сервис исполнения запросов ожидает входящие запросы от JDBC-драйвера;
   – `DTM_NAME` - имя среды для формирования полного наименования датамартов;
   – `CORE_TIME_ZONE` - настройки временной зоны;

- DTM_CORE_METRICS_ENABLED - настройки генерации метрики Сервиса исполнения запросов;
- DTM_CORE_TASK_POOL_SIZE - максимальный объем пула задач в Сервисе исполнения запросов;
- DTM_CORE_TASK_TIMEOUT - интервал времени завершения задачи, выполняемой в Сервисе исполнения запросов.

2. Оптимизация работы сокета TCP_NODELAY:
   - DTM_CORE_HTTP_TCP_NODELAY - настройка режима оптимизации работы сокета TCP_NODELAY;
   - DTM_CORE_HTTP_TCP_FAST_OPEN - настройка режима TCP FAST_OPEN;
   - DTM_CORE_HTTP_TCP_QUICK_ACK - настройка режима оптимизации работы сокета TCP_QUICKACK.

3. Настройки для EDML операторов:
   - EDML_DATASOURCE - тип СУБД-источника;
   - EDML_DEFAULT_CHUNK_SIZE – размер chunk по умолчанию;
   - EDML_STATUS_CHECK_PERIOD_MS - период проверки статуса плагина в миллисекундах;
   - EDML_FIRST_OFFSET_TIMEOUT_MS - интервал времени ожидания до таймаута в миллисекундах при работе с первым смещением;
   - EDML_CHANGE_OFFSET_TIMEOUT_MS - интервал времени ожидания до таймаута в миллисекундах при работе с первым смещением в топике Kafka.

4. Настройка Zookeeper-серверов:
   - ZOOKEEPER_DS_ADDRESS - сетевой адрес хоста Zookeeper для служебной БД;
   - ZOOKEEPER_DS_CONNECTION_TIMEOUT_MS - интервал времени ожидания (в миллисекундах) соединения с хостом Zookeeper для служебной БД до достижения таймаута;
   - ZOOKEEPER_DS_SESSION_TIMEOUT_MS - интервал времени бездействия (в миллисекундах) соединения с хостом Zookeeper для служебной БД до достижения таймаута;
   - ZOOKEEPER_DS_CHROOT - корневой путь к хосту Zookeeper для служебной БД;
   - ZOOKEEPER_KAFKA_ADDRESS - сетевой адрес хоста Zookeeper для брокера сообщений Kafka;
   - ZOOKEEPER_KAFKA_CONNECTION_TIMEOUT_MS - интервал времени ожидания (в миллисекундах) соединения с хостом Zookeeper для брокера сообщений Kafka до

достижения таймаута;

- ZOOKEEPER_KAFKA_SESSION_TIMEOUT_MS - интервал времени бездействия (в миллисекундах) соединения с хостом Zookeeper для брокера сообщений Kafka до достижения таймаута;
- ZOOKEEPER_KAFKA_CHROOT - корневой путь к хосту Zookeeper для брокера сообщений Kafka.

5. Настройка Kafka-серверов:

- KAFKA_INPUT_STREAM_TIMEOUT_MS – интервал времени ожидания (в миллисекундах) входного потока данных для брокера сообщений Kafka до достижения таймаута;
- KAFKA_STATUS_EVENT_ENABLED - разрешение на публикацию событий;
- KAFKA_STATUS_EVENT_TOPIC - наименование топика Kafka, в который публикуются события;
- STATUS_MONITOR_URL - сетевой адрес, порт и путь к Сервису мониторинга статусов Kafka.

6. Настройки кэширования запросов:

- CACHE_INITIAL_CAPACITY - начальная емкость кэша;
- CACHE_MAXIMUM_SIZE - максимальный размер кэша;
- CACHE_EXPIRE_AFTER_ACCESS_MINUTES - время (в минутах) устаревания кэша после последнего момента доступа к нему.

2.2.2 Настройка ПОДД-адаптера - Модуль исполнения запросов

2.2.2.1 Конфигурация ПОДД-адаптера - Модуль исполнения запросов (application.yml)

Файл application.yml – основной конфигурационный файл **ПОДД-адаптера - Модуль исполнения запросов**, в котором задана логика и порядок работы адаптера: получение входящих запросов, их обработка, подключение к Сервису формирования документов (секция: printable-forms-service), настройки логирования (секция: logging), а также другие настройки необходимые для корректной работы адаптера. Хинт пагинации FORCE_LLR определен в переменных среды.

2.2.2.2 Пример файла application.yml

Приведем типовую структуру файла и возможные настройки **ПОДД-адаптера - Модуль исполнения запросов**. Следует учитывать, что в конфигурационном файле следует задавать только те настройки, которые необходимы для решения текущих бизнес-задач.

```
http-server:
  port: ${HTTP_PORT:8090}

environment:
  name: ${ENVIRONMENT_NAME:test}
```

```yaml
executor:
  reader-pool-size: ${EXECUTOR_READER_POOL_SIZE:20}
  max-execute-time: ${EXECUTOR_MAX_EXECUTE_TIME:600}
  log-pool-size: ${EXECUTOR_LOG_POOL_SIZE:20}

send:
  channel-size: ${SEND_CHANNEL_SIZE:1}
  compress: ${SEND_COMPRESS:none}
  max-message-size: ${SEND_MAX_MESSAGE_SIZE:800000}

query:
  data-source-type:
    for-listagg: ${DATA_SOURCE_TYPE_LISTAGG:ADP}
    statistics-request: ${DATA_SOURCE_TYPE_STATISTIC:ADP}
  force-llr-for-order: ${FORCE_LLR_FOR_ORDER:true}
  force-llr-for-all: ${FORCE_LLR_FOR_ALL:false}
  llr-rows-limit: ${LLR_ROWS_LIMIT:200}
  fetch-size: ${FETCH_SIZE:1000}

zookeeper:
  connection-string: ${ZOOKEEPER_DS_ADDRESS:localhost}
  connection-timeout-ms: ${ZOOKEEPER_DS_CONNECTION_TIMEOUT_MS:30000}
  session-timeout-ms: ${ZOOKEEPER_DS_SESSION_TIMEOUT_MS:86400000}
  chroot: ${ZOOKEEPER_DS_CHROOT:/adapter}

prostore-rest-client:
  # Признак использования rest-api для взаимодействия с простором.
  enabled: ${PS_REST_CLIENT_ENABLED:true}
  host: ${PS_HOST:localhost}
  port: ${PS_PORT:9195}
  http:
    max-pool-size: ${PS_MAX_POOL_SIZE:8}

printable-forms-service:
  host: ${PFS_HOST:localhost}
  port: ${PFS_PORT:8080}
  pool-size: ${PFS_POOL_SIZE:10}
  timeout: ${PFS_TIMEOUT:30}

kafka:
  agent.topic.prefix: ${AGENT_TOPIC_PREFIX:}
  max-concurrent-handle: ${KAFKA_MAX_CONCURRENT_HANDLE:1000}
  commit-interval: ${KAFKA_COMMIT_INTERVAL:5s}
  external:
    bootstrap.servers: ${KAFKA_BOOTSTRAP_SERVERS:localhost:9092}
    topic.prefix: ${EXTERNAL_TOPIC_PREFIX:${agent.topic.prefix}}
  internal:
    bootstrap.servers: ${PS_KAFKA:localhost:9092}
    topic.prefix: ${INTERNAL_TOPIC_PREFIX:${agent.topic.prefix}}
  consumer:
    query-request:
      topic: ${kafka.external.topic.prefix}query.rq
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.external.bootstrap.servers}
        group.id: ${kafka.external.topic.prefix}query.consumer
        auto.offset.reset: earliest
        enable.auto.commit: false
    query-cancel-request:
      topic: ${kafka.external.topic.prefix}cancel.rq
      commit-interval: ${kafka.commit-interval}
```

```
      property:
        bootstrap.servers: ${kafka.external.bootstrap.servers}
        group.id: ${kafka.external.topic.prefix}cancel.query.consumer
        auto.offset.reset: earliest
        enable.auto.commit: false
    metadata-request:
      topic: ${kafka.external.topic.prefix}metadata.rq
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.external.bootstrap.servers}
        group.id: ${kafka.external.topic.prefix}metadata.consumer
        auto.offset.reset: earliest
        enable.auto.commit: false
    metadata-new-data-request:
      topic: ${kafka.external.topic.prefix}metadata.newdata.rq
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.external.bootstrap.servers}
        group.id: ${kafka.external.topic.prefix}metadata.newdata.consumer
        auto.offset.reset: earliest
        enable.auto.commit: false
    statistics-request:
      topic: ${kafka.external.topic.prefix}statistics.rq
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.external.bootstrap.servers}
        group.id: ${kafka.external.topic.prefix}statistics.rq.consumer
        auto.offset.reset: earliest
        enable.auto.commit: false
    report-request:
      topic: ${kafka.external.topic.prefix}procedure.query.rq
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.external.bootstrap.servers}
        group.id: ${kafka.external.topic.prefix}report.rq.consumer
        auto.offset.reset: earliest
        enable.auto.commit: false
  producer:
    query-result: ${kafka.external.topic.prefix}query.rs
    query-error: ${kafka.external.topic.prefix}query.err
    query-estimation-result: ${kafka.external.topic.prefix}query.estimation.rs
    query-cancel-result: ${kafka.external.topic.prefix}cancel.rs
    query-cancel-error: ${kafka.external.topic.prefix}cancel.err
    metadata-result: ${kafka.external.topic.prefix}metadata.rs
    metadata-error: ${kafka.external.topic.prefix}metadata.err
    metadata-newdata-result: ${kafka.external.topic.prefix}metadata.newdata.rs
    metadata-newdata-error: ${kafka.external.topic.prefix}metadata.newdata.err
    statistics-result: ${kafka.external.topic.prefix}statistics.rs
    statistics-error: ${kafka.external.topic.prefix}statistics.err
    report-result: ${kafka.external.topic.prefix}query.rs
    report-error: ${kafka.external.topic.prefix}query.err
    property:
      bootstrap.servers: ${kafka.external.bootstrap.servers}
    internal:
      mppr-query-request: ${kafka.internal.topic.prefix}mppr.delegate.rq
      tp-delete-tmp: ${kafka.internal.topic.prefix}tp.delete.tmp
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}

statistics:
  enabled: ${STATISTICS_ENABLED:false}
```

```
  timeout-min: ${STATISTICS_TIMEOUT_MIN:60}
  datamarts:
    - name: demo_dev
      tables:
        - name: all_types
          columns:
            - varchar_c
            - char_c
            - bigint_c

logging:
  request-response:
    query-request: ${QUERY_REQUEST_LOG_ENABLED:false}
    query-response: ${QUERY_RESPONSE_LOG_ENABLED:false}
    pf-request: ${PF_REQUEST_LOG_ENABLED:false}
    pf-response: ${PF_RESPONSE_LOG_ENABLED:false}

metrics:
  port: ${METRICS_PORT:9837}
```

2.2.2.3 Параметры конфигурации

Настройка конфигурации **ПОДД-адаптера - Модуль исполнения запросов** осуществляется путем редактирования параметров настроек в файле `application.yml`.

Пример конфигурации файла `application.yml` для **ПОДД-адаптера - Модуль исполнения запросов** см. в разделе Пример файла application.yml.

В файле конфигурации **ПОДД-адаптера - Модуль исполнения запросов** могут быть настроены следующие секции:

- `http-server` - указывается порт для подключения;
- `environment` - указывается название окружения (`test`, `prod` и т.д.);
- `executor` - настраивается размер пула для запросов;
- `send` - настраиваются ограничения на размер загружаемого файла;
- `query` - настройка выполнения запросов;
- `zookeeper` - подключения в Zookeeper;
- `prostore-rest-client` - блок параметров конфигурирования взаимодействия с ProStore. Если false - будет использоваться JDBC-драйвер.
- `prostore`- указываются настройки подключения к ProStore;
- `printable-forms-service` - настройки подключения к Сервису формирования документов;
- `kafka` - настройки параметров подключения к шине данных Apache Kafka;
- `statistics` - управление статистикой;
- `logging` - настройка сохранения лог-файла;
- `metrics` - настройка получения метрик.

2.2.2.3.1 Секция http-server

В секции `http-server` указывается порт веб-сервера.

Например:

```
http:
  port: ${HTTP_PORT:8090}
```

Параметры настроек

– `port` - порт веб-сервера, например: `HTTP_PORT:8090`.

2.2.2.3.2 Секция environment

Секция `environment` предназначена для настройки параметров окружения.

Например:

```
environment:
  name: ${ENVIRONMENT_NAME:test}
```

Параметры настроек

– `name` - название окружения (test, prod и т.д.), например: `ENVIRONMENT_NAME:test`.

2.2.2.3.3 Секция executor

Секция `executor` предназначена для указания размера пула для чтения Kafka и времени выполнения задач.

Например:

```
executor:
  reader-pool-size: ${EXECUTOR_READER_POOL_SIZE:20}
  max-execute-time: ${EXECUTOR_MAX_EXECUTE_TIME:600}
  log-pool-size: ${EXECUTOR_LOG_POOL_SIZE:20}
```

Параметры настроек

– `reader-pool-size` - размер пула для чтения Kafka, например `EXECUTOR_READER_POOL_SIZE:20`;

– `max-execute-time` - максимальное время выполнения задачи (сек), например `EXECUTOR_MAX_EXECUTE_TIME:600`;

– `log-pool-size` - Размер используемого пула для журналирования запросов и ответов, например `EXECUTOR_LOG_POOL_SIZE:20`.

2.2.2.3.4 Секция send

В секции `send` настраиваются ограничения на размер загружаемого файла.

Например:

```
send:
  channel-size: ${SEND_CHANNEL_SIZE:1}
  compress: ${SEND_COMPRESS:none}
  max-message-size: ${SEND_MAX_MESSAGE_SIZE:800000}
```

Параметры настроек

– `channel-size` - размер канала на отправку сообщения, например `SEND_CHANNEL_SIZE:10`;

– `compress` - сжатие выгружаемых сообщений (none или zstd), например `SEND_COMPRESS:none`;

- `max-message-size` - максимальный размер отправляемого сообщения, например `SEND_MAX_MESSAGE_SIZE:800000`.

2.2.2.3.5 Секция query

В секции `query` выполняется настройка выполнения запросов.
Например:

```
query:
  data-source-type:
    for-listagg: ${DATA_SOURCE_TYPE_LISTAGG:ADP}
    statistics-request: ${DATA_SOURCE_TYPE_STATISTIC:ADP}
  force-llr-for-order: ${FORCE_LLR_FOR_ORDER:true}
  force-llr-for-all: ${FORCE_LLR_FOR_ALL:false}
  llr-rows-limit: ${LLR_ROWS_LIMIT:200}
  fetch-size: ${FETCH_SIZE:1000}
```

Параметры настроек

- `data-source-type` - выполнение запроса с LISTAGG на (ADB/ADP), например `DATA_SOURCE_TYPE:ADB`;
- `force-llr-for-order` - выполнение ORDER BY запроса с использованием пагинации, например `FORCE_LLR_FOR_ORDER:true`;
- `force-llr-for-all` - выполнение всех запросов через LLR, например `FORCE_LLR_FOR_ALL:false`;
- `llr-rows-limit` - ограничение выгрузки через ЛЛР, при использовании в запросе лимита со значением меньшим, чем указанное значение, например `LLR_ROWS_LIMIT:200`, будет использован режим LLR;
- `fetch-size` - размер выгрузки через JDBC, например `FETCH_SIZE:1000`.

> Внимание:
> Для опции `force-llr-for-order` параметр `false` можно устанавливать только при развертывании витрины на единственной БД ADP

2.2.2.3.6 Секция zookeeper

Секция `zookeeper` определяет настройки подключения в zookeeper DS.
Например:

```
zookeeper:
  connection-string: ${ZOOKEEPER_DS_ADDRESS:t5-adsp-01.ru-central1.internal}
  connection-timeout-ms: ${ZOOKEEPER_DS_CONNECTION_TIMEOUT_MS:30000}
  session-timeout-ms: ${ZOOKEEPER_DS_SESSION_TIMEOUT_MS:86400000}
  chroot: ${ZOOKEEPER_DS_CHROOT:/adapter}
```

Параметры настроек

- `connection-string` - Подключение в Zookeeper DS, например `ZOOKEEPER_DS_ADDRESS:t5-adsp-01.ru-central1.internal`;
- `connection-timeout-ms` - Zookeeper DS таймаут подключения, например `ZOOKEEPER_DS_CONNECTION_TIMEOUT_MS:30000`;

– `session-timeout-ms` - Zookeeper DS таймаут сессии, например `ZOOKEEPER_DS_SESSION_TIMEOUT_MS:86400000`;

– `chroot` - Zookeeper DS chroot path, например `ZOOKEEPER_DS_CHROOT:/adapter`.

2.2.2.3.7 Секция prostore-rest-client

В секции `prostore-rest-client` реализован блок параметров конфигурирования взаимодействия с ProStore.

Например:

```
prostore-rest-client:
  enabled: true
  host: ${PS_HOST:localhost}
  port: ${PS_PORT:9195}
  http:
    max-pool-size: ${PS_MAX_POOL_SIZE:8}
```

Параметры настроек

– `host` - адрес Prostore, например `PS_HOST:localhost`;

– `port` - порт Prostore, например `PS_PORT:9195`;

– `max-pool-size` - максимальное число подключений к Prostore, например `PS_MAX_POOL_SIZE:8`.

2.2.2.3.8 Секция printable-forms-service

Секция `printable-forms-service` определяет настройки подключения к Сервису формирования документов.

Например:

```
printable-forms-service:
  host: ${PFS_HOST:localhost}
  port: ${PFS_PORT:8080}
  pool-size: ${PFS_POOL_SIZE:10}
  timeout: ${PFS_TIMEOUT:30}
```

Параметры настроек

– `host` - адрес сервера формирования документов, например `PFS_HOST:localhost`;

– `port` - порт сервера формирования документов, например `PFS_PORT:8080`;

– `pool-size` - размер пула соединений для ПФ, например `PFS_POOL_SIZE:10`;

– `timeout` - таймаут переподключения к сервису формирования документов (секунды), например `PFS_TIMEOUT:30`.

2.2.2.3.9 Секция kafka

В секции `kafka` собраны настройки параметров подключения к шине данных Apache Kafka.

Например:

```
kafka:
  agent.topic.prefix: ${AGENT_TOPIC_PREFIX:}
  max-concurrent-handle: ${KAFKA_MAX_CONCURRENT_HANDLE:1000}
  commit-interval: ${KAFKA_COMMIT_INTERVAL:5s}
```

```
external:
  bootstrap.servers: ${KAFKA_BOOTSTRAP_SERVERS:localhost:9092}
  topic.prefix: ${EXTERNAL_TOPIC_PREFIX:${agent.topic.prefix}}
internal:
  bootstrap.servers: ${PS_KAFKA:localhost:9092}
  topic.prefix: ${INTERNAL_TOPIC_PREFIX:${agent.topic.prefix}}
consumer:
  query-request:
    topic: ${kafka.external.topic.prefix}query.rq
    max-concurrent-handle: ${kafka.max-concurrent-handle}
    commit-interval: ${kafka.commit-interval}
    property:
      bootstrap.servers: ${kafka.external.bootstrap.servers}
      group.id: ${kafka.external.topic.prefix}query.consumer
      auto.offset.reset: earliest
      enable.auto.commit: false
  query-cancel-request:
    topic: ${kafka.external.topic.prefix}cancel.rq
    commit-interval: ${kafka.commit-interval}
    property:
      bootstrap.servers: ${kafka.external.bootstrap.servers}
      group.id: ${kafka.external.topic.prefix}cancel.query.consumer
      auto.offset.reset: earliest
      enable.auto.commit: false
  metadata-request:
    topic: ${kafka.external.topic.prefix}metadata.rq
    commit-interval: ${kafka.commit-interval}
    property:
      bootstrap.servers: ${kafka.external.bootstrap.servers}
      group.id: ${kafka.external.topic.prefix}metadata.consumer
      auto.offset.reset: earliest
      enable.auto.commit: false
  metadata-new-data-request:
    topic: ${kafka.external.topic.prefix}metadata.newdata.rq
    commit-interval: ${kafka.commit-interval}
    property:
      bootstrap.servers: ${kafka.external.bootstrap.servers}
      group.id: ${kafka.external.topic.prefix}metadata.newdata.consumer
      auto.offset.reset: earliest
      enable.auto.commit: false
  statistics-request:
    topic: ${kafka.external.topic.prefix}statistics.rq
    commit-interval: ${kafka.commit-interval}
    property:
      bootstrap.servers: ${kafka.external.bootstrap.servers}
      group.id: ${kafka.external.topic.prefix}statistics.rq.consumer
      auto.offset.reset: earliest
      enable.auto.commit: false
  report-request:
    topic: ${kafka.external.topic.prefix}procedure.query.rq
    max-concurrent-handle: ${kafka.max-concurrent-handle}
    commit-interval: ${kafka.commit-interval}
    property:
      bootstrap.servers: ${kafka.external.bootstrap.servers}
      group.id: ${kafka.external.topic.prefix}report.rq.consumer
      auto.offset.reset: earliest
      enable.auto.commit: false
producer:
  query-result: ${kafka.external.topic.prefix}query.rs
  query-error: ${kafka.external.topic.prefix}query.err
  query-estimation-result: ${kafka.external.topic.prefix}}query.estimation.rs
  query-cancel-result: ${kafka.external.topic.prefix}cancel.rs
```

```
    query-cancel-error: ${kafka.external.topic.prefix}cancel.err
    metadata-result: ${kafka.external.topic.prefix}metadata.rs
    metadata-error: ${kafka.external.topic.prefix}metadata.err
    metadata-newdata-result: ${kafka.external.topic.prefix}metadata.newdata.rs
    metadata-newdata-error: ${kafka.external.topic.prefix}metadata.newdata.err
    statistics-result: ${kafka.external.topic.prefix}statistics.rs
    statistics-error: ${kafka.external.topic.prefix}statistics.err
    report-result: ${kafka.external.topic.prefix}query.rs
    report-error: ${kafka.external.topic.prefix}query.err
    property:
      bootstrap.servers: ${kafka.external.bootstrap.servers}
    internal:
      mppr-query-request: ${kafka.internal.topic.prefix}mppr.delegate.rq
      tp-delete-tmp: ${kafka.internal.topic.prefix}tp.delete.tmp
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
```

Параметры конфигурации

– `topic` - префикс для топиков агента ПОДД, например `AGENT_TOPIC_PREFIX`.

2.2.2.3.10 Секция statistics

Секция `statistics` предназначена для управления статистикой.

Например:

```
statistics:
  enabled: ${STATISTICS_ENABLED:false}
  timeout-min: ${STATISTICS_TIMEOUT_MIN:60}
  datamarts:
    - name: demo_dev
      tables:
        - name: all_types
          columns:
            - varchar_c
            - char_c
            - bigint_c
```

Параметры конфигурации

– `enabled` - включение (true)/ выключение (false) расчета статистики, например `STATISTICS_ENABLED:false`;

– `timeout-min` - время обновления статистики (минуты), например `STATISTICS_TIMEOUT_MIN:60`.

2.2.2.3.11 Секция logging

Секция `logging` предназначена для настройки параметров логирования.

Например:

```
logging:
  request-response:
    query-request: ${QUERY_REQUEST_LOG_ENABLED:false}
    query-response: ${QUERY_RESPONSE_LOG_ENABLED:false}
    pf-request: ${PF_REQUEST_LOG_ENABLED:false}
    pf-response: ${PF_RESPONSE_LOG_ENABLED:false}
```

Параметры конфигурации

– `query-request` - журналирование query запросов, например

QUERY_REQUEST_LOG_ENABLED:false;

- query-response - журналирование query ответов, например QUERY_RESPONSE_LOG_ENABLED:false;
- pf-request - журналирование запросов на сервис формирования документов, например PF_REQUEST_LOG_ENABLED:false;
- pf-response - журналирование ответов от сервиса формирования документов, например PF_RESPONSE_LOG_ENABLED:false.

2.2.2.3.12 Секция metrics

Секция metrics предназначена для настройки параметров метрик.
Например:

```
metrics:
  port: ${METRICS_PORT:9837}
```

Параметры конфигурации

- port - Порт для метрик, например METRICS_PORT:9837.

Описание формата взаимодействия между Агентом ПОДД и *ПОДД-адаптером - Модуль исполнения запросов* (название топиков, формат сообщений, схема взаимодействия) описан в документе Спецификация модуля ПОДД-адаптера - Модуль исполнения запросов.

2.2.3 Настройка ПОДД-адаптер – Модуль MPPR

2.2.3.1 Конфигурация ПОДД-адаптера - Модуль MPPR (application.yml)

Файл application.yml – основной конфигурационный файл модуля, в котором задана его логика и порядок работы модуля: получение входящих запросов, их обработка, а также настройка подключения к ядру витрины (секция: prostore), настройка метрик (секция: metrics), а также другие настройки необходимые для корректной работы адаптера.

2.2.3.2 Пример файла application.yml

Приведем типовую структуру файла и возможные настройки **ПОДД-адаптера - Модуль MPPR**. Следует учитывать, что в конфигурационном файле следует задавать только те настройки, которые необходимы для решения текущих бизнес-задач.

```
http-server:
  port: ${HTTP_PORT:8085}

environment:
  name: ${ENVIRONMENT_NAME:test}

executor:
  reader-pool-size: ${EXECUTOR_READER_POOL_SIZE:20}
  max-execute-time: ${EXECUTOR_MAX_EXECUTE_TIME:600}
  log-pool-size: ${EXECUTOR_LOG_POOL_SIZE:20}

send:
  channel-size: ${SEND_CHANNEL_SIZE:1}
  timeout: ${SEND_TIMEOUT:30}
  delete-topic: ${SEND_DELETE_TOPIC:true}
  compress: ${SEND_COMPRESS:none}
```

```
prostore-rest-client:
  host: ${PS_HOST:localhost}
  port: ${PS_PORT:9195}
  http:
    max-pool-size: ${PS_MAX_POOL_SIZE:8}

prostore:
  kafka:
    message-limit: ${PS_MESSAGE_LIMIT:1000}
    zk-url: ${PS_ZK_KAFKA_URL:localhost:2181}
    statusEventTopic:
      topic: ${PS_STATUS_EVENT_TOPIC:status.event}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
        group.id: ${kafka.internal.topic.prefix}podd-adapter-mppr-status-event
        auto.offset.reset: earliest
        enable.auto.commit: false

kafka:
  agent.topic.prefix: ${AGENT_TOPIC_PREFIX:}
  max-concurrent-handle: ${KAFKA_MAX_CONCURRENT_HANDLE:10}
  commit-interval: ${KAFKA_COMMIT_INTERVAL:5s}
  external:
    bootstrap.servers: ${KAFKA_BOOTSTRAP_SERVERS:localhost:9092}
    topic.prefix: ${EXTERNAL_TOPIC_PREFIX:${kafka.agent.topic.prefix}}
  internal:
    bootstrap.servers: ${PS_KAFKA:localhost:9092}
    topic.prefix: ${INTERNAL_TOPIC_PREFIX:${kafka.agent.topic.prefix}}
  consumer:
    query-request:
      topic: ${kafka.internal.topic.prefix}mppr.delegate.rq
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
        group.id: ${kafka.internal.topic.prefix}mppr.query.consumer
        auto.offset.reset: earliest
        enable.auto.commit: false
        max.poll.records: 1
        max.poll.interval.ms: 600000
    delta-request:
      topic: ${kafka.internal.topic.prefix}mppr.delta.rq
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
        group.id: ${kafka.internal.topic.prefix}mppr.delta.consumer
        auto.offset.reset: earliest
        enable.auto.commit: false
        max.poll.records: 1
        max.poll.interval.ms: 600000
    download-data:
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
        group.id: ${kafka.internal.topic.prefix}mppr.x.query.consumer
        auto.offset.reset: earliest
        enable.auto.commit: false
        max.poll.records: 1
  producer:
    query-result: ${kafka.external.topic.prefix}query.rs
    query-error: ${kafka.external.topic.prefix}query.err
```

```
    delta-result: ${kafka.external.topic.prefix}delta.rs
    delta-error: ${kafka.external.topic.prefix}delta.err
    property:
      bootstrap.servers: ${kafka.external.bootstrap.servers}
    internal:
      tp-delete-tmp: ${kafka.internal.topic.prefix}tp.delete.tmp
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}

metrics:
  port: ${METRICS_PORT:9843}

logging:
  scl.delta:
    enabled: ${SCL_DELTA_ENABLED:false}
  request-response:
    delta-request: ${DELTA_REQUEST_LOG_ENABLED:false}
    delta-response: ${DELTA_RESPONSE_LOG_ENABLED:false}
    query-request: ${QUERY_REQUEST_LOG_ENABLED:false}
    query-response: ${QUERY_RESPONSE_LOG_ENABLED:false}
```

2.2.3.3 Параметры конфигурации

Настройка конфигурации **ПОДД-адаптера - Модуль MPPR** осуществляется путем редактирования параметров настроек в файле `application.yml`.

Пример файла application.yml для **ПОДД-адаптера - Модуль MPPR**.

В файле конфигурации **ПОДД-адаптера - Модуль MPPR** могут быть настроены следующие секции:

- `http-server` - указывается порт веб-сервера;
- `environment` - название окружения (`test`, `prod` и т.д.);
- `executor` - предназначена для указания размера пула для запросов;
- `send` - настраиваются ограничения на размер загружаемого файла;
- `prostore-rest-client` - блок параметров конфигурирования взаимодействия с ProStore. Если false - будет использоваться JDBC-драйвер;
- `prostore` - настройка подключения к серверу и базе данных ProStore;
- `kafka` - настройки параметров подключения к шине данных Apache Kafka;
- `metrics` - настройка получения метрик;
- `logging` - настройки журналирования запросов и ответов;

2.2.3.3.1 Секция http-server

В секции `http-server` указывается порт веб-сервера.

Например:

```
http-server:
  port: ${HTTP_PORT:8085}
```

Параметры настроек

- `port` - порт веб-сервера, например: `HTTP_PORT:8085`.

2.2.3.3.2 Секция environment

В секции `environment` указывается среда разработки (dev, test, stable, prod)
Например:

```
environment:
  name: ${ENVIRONMENT_NAME:test}
```

Параметры настроек

- `name` - Название окружения, например `ENVIRONMENT_NAME:test`.

2.2.3.3.3 Секция executor

Секция `executor` предназначена для указания размера пула для чтения Kafka и времени выполнения задач.

Например:

```
executor:
  reader-pool-size: ${EXECUTOR_READER_POOL_SIZE:20}
  max-execute-time: ${EXECUTOR_MAX_EXECUTE_TIME:600}
  log-pool-size: ${EXECUTOR_LOG_POOL_SIZE:20}
```

Параметры настроек

- `reader-pool-size` - размер пула для чтения Kafka, например `EXECUTOR_READER_POOL_SIZE:20`;
- `max-execute-time` - максимальное время выполнения задачи (сек), например `EXECUTOR_MAX_EXECUTE_TIME:600`;
- `log-pool-size` - размер пула используемого для журналирования запросов и ответов, например `EXECUTOR_LOG_POOL_SIZE:20`.

2.2.3.3.4 Секция send

В секции `send` настраиваются ограничения на размер загружаемого файла.

Например:

```
send:
  channel-size: ${SEND_CHANNEL_SIZE:1}
  timeout: ${SEND_TIMEOUT:30}
  delete-topic: ${SEND_DELETE_TOPIC:true}
  compress: ${SEND_COMPRESS:none}
```

Параметры настроек

- `channel-size` - размер канала на отправку сообщения, например `SEND_CHANNEL_SIZE:10`;
- `timeout` - таймаут вычитывания данных из топика (сек), например `SEND_TIMEOUT:30`
- `delete-topic` - удаление внешнего топика после выгрузки, например `SEND_DELETE_TOPIC:true`;
- `compress` - сжатие выгружаемых сообщений (none или zstd), например `SEND_COMPRESS:none`.

2.2.3.3.5 Секция prostore-rest-client

В секции `prostore-rest-client` реализован блок параметров конфигурирования взаимодействия с ProStore.

Например:

```
prostore-rest-client:
  # Признак использования rest-api для взаимодействия с простором.
  enabled: ${PS_REST_CLIENT_ENABLED:true}
  host: ${PS_HOST:localhost}
  port: ${PS_PORT:9195}
  http:
    max-pool-size: ${PS_MAX_POOL_SIZE:8}
```

Параметры настроек

- `host` - адрес Prostore, например `PS_HOST:localhost`;
- `port` - порт Prostore, например `PS_PORT:9195`;
- `max-pool-size` - максимальное число подключений к Prostore, например `PS_MAX_POOL_SIZE:8`.

2.2.3.3.6 Секция prostore

В секции `prostore` осуществляется настройка подключения к серверу и базе данных ProStore.

Например:

```
prostore:
  kafka:
    message-limit: ${PS_MESSAGE_LIMIT:1000}
    zk-url: ${PS_ZK_KAFKA_URL:localhost:2181}
    statusEventTopic:
      topic: ${PS_STATUS_EVENT_TOPIC:status.event}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
        group.id: ${kafka.internal.topic.prefix}podd-adapter-mppr-status-event
        auto.offset.reset: earliest
        enable.auto.commit: false
```

Параметры настроек

- `message-limit` - лимит сообщений, например `PS_MESSAGE_LIMIT:1000`;
- `zk-url` - адрес сервера zookeeper для загрузки данных в Prostore, например `PS_ZK_KAFKA_URL:localhost:2181`.

2.2.3.3.7 Секция kafka

Секция `kafka` определяет настройки взаимодействия через ПОДД-адаптер между Поставщиком данных (`producer`) и Получателем данных (`consumer`).

В секции `kafka` собраны настройки параметров подключения к шине данных Apache Kafka.

Например:

```
kafka:
  agent.topic.prefix: ${AGENT_TOPIC_PREFIX:}
  max-concurrent-handle: ${KAFKA_MAX_CONCURRENT_HANDLE:10}
```

```
  commit-interval: ${KAFKA_COMMIT_INTERVAL:5s}
  external:
    bootstrap.servers: ${KAFKA_BOOTSTRAP_SERVERS:localhost:9092}
    topic.prefix: ${EXTERNAL_TOPIC_PREFIX:${agent.topic.prefix}}
  internal:
    bootstrap.servers: ${PS_KAFKA:localhost:9092}
    topic.prefix: ${INTERNAL_TOPIC_PREFIX:${agent.topic.prefix}}
  consumer:
    query-request:
      topic: ${kafka.internal.topic.prefix}mppr.delegate.rq
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
        group.id: ${kafka.internal.topic.prefix}mppr.query.consumer
        auto.offset.reset: earliest
        enable.auto.commit: false
        max.poll.records: 1
        max.poll.interval.ms: 600000
    delta-request:
      topic: ${kafka.internal.topic.prefix}mppr.delta.rq
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
        group.id: ${kafka.internal.topic.prefix}mppr.delta.consumer
        auto.offset.reset: earliest
        enable.auto.commit: false
        max.poll.records: 1
        max.poll.interval.ms: 600000
    download-data:
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
        group.id: ${kafka.internal.topic.prefix}mppr.x.query.consumer
        auto.offset.reset: earliest
        enable.auto.commit: false
        max.poll.records: 1
  producer:
    query-result: ${kafka.external.topic.prefix}query.rs
    query-error: ${kafka.external.topic.prefix}query.err
    delta-result: ${kafka.external.topic.prefix}delta.rs
    delta-error: ${kafka.external.topic.prefix}delta.err
    property:
      bootstrap.servers: ${kafka.external.bootstrap.servers}
    internal:
      tp-delete-tmp: ${kafka.internal.topic.prefix}tp.delete.tmp
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
```

Параметры конфигурации

- `topic` - префикс для топиков агента ПОДД, например `AGENT_TOPIC_PREFIX`.

2.2.3.3.8 Секция metrics

Секция `metrics` предназначена для настройки параметров метрик:
Например:

```
metrics:
  port: ${METRICS_PORT:9843}
```

Параметры конфигурации

– `port` - Порт для метрик, например `METRICS_PORT:9843`.

2.2.3.3.9 Секция logging

Секция `logging` предназначена для настройки журналирования запросов и ответов Например:

```
logging:
  scl.delta:
    enabled: ${SCL_DELTA_ENABLED:false}
  request-response:
    delta-request: ${DELTA_REQUEST_LOG_ENABLED:false}
    delta-response: ${DELTA_RESPONSE_LOG_ENABLED:false}
    query-request: ${QUERY_REQUEST_LOG_ENABLED:false}
    query-response: ${QUERY_RESPONSE_LOG_ENABLED:false}
```

LOG_FORMAT - Логирование в формате (JSON/TEXT) - указывается в `logback.xml`

2.2.4 Настройка Модуля подписок

2.2.5.1 Конфигурация модуля ПОДД-адаптер - Модуль подписок (application.yml)

Файл `application.yml` – основной конфигурационный файл модуля, в котором задана его логика и порядок работы модуля: настройка подключения к **Prostore** (секция: `prostore`), подключение к **Брокеру сообщений Kafka**, **Zookeeper**, а также , порядок обработки запросов между Получателем и Поставщиком данных (секция: `kafka`), настройка метрик (секция: `metrics`) и другие настройки необходимые для корректной работы адаптера.

2.2.4.2 Пример файла application.yml

Приведем типовую структуру файла и возможные настройки **ПОДД-адаптера - Модуль подписок**. Следует учитывать, что в конфигурационном файле следует задавать только те настройки, которые необходимы для решения текущих бизнес-задач.

```
environment:
  name: ${ENVIRONMENT_NAME:test}

http-server:
  port: ${HTTP_PORT:8085}

executor:
  reader-pool-size: ${EXECUTOR_READER_POOL_SIZE:20}

zookeeper:
  connection-string: ${ZOOKEEPER_DS_ADDRESS:localhost}
  connection-timeout-ms: ${ZOOKEEPER_DS_CONNECTION_TIMEOUT_MS:30000}
  session-timeout-ms: ${ZOOKEEPER_DS_SESSION_TIMEOUT_MS:86400000}
  chroot: ${ZOOKEEPER_DS_CHROOT:/adapter}

migration:
  enabled: ${MIGRATION_ENABLE:false}
  old-connection-string: ${OLD_ZOOKEEPER_DS_ADDRESS:localhost}

table-metadata:
  cache:
    enabled: false

prostore-rest-client:
  # Признак использования rest-api для взаимодействия с простором.
  enabled: ${PS_REST_CLIENT_ENABLED:true}
  host: ${PS_HOST:localhost}
```

```yaml
  port: ${PS_PORT:9195}
  http:
    max-pool-size: ${PS_MAX_POOL_SIZE:8}

prostore:
  status-event-topic:
    topic: ${PS_STATUS_EVENT_TOPIC:status.event}
    property:
      bootstrap.servers: ${kafka.internal.bootstrap.servers}
      group.id: ${kafka.external.topic.prefix}replicator-status-event
      auto.offset.reset: earliest
      enable.auto.commit: false

subscription:
  consumer:
    # режим отмены подписки на потребителе. Возможные значения rename, drop, none
    cancel-mode: none

# Массив описания standalone таблиц, участвующих в репликации
#standalone-tables: []
# Пример описания
#standalone-tables:
#  - table: "misdm05.readable_book"
#    anchor: "update_at"
#    soft-delete: "delete_at"


kafka:
  agent.topic.prefix: ${AGENT_TOPIC_PREFIX:}
  max-concurrent-handle: ${KAFKA_MAX_CONCURRENT_HANDLE:10}
  commit-interval: ${KAFKA_COMMIT_INTERVAL:5s}
  external:
    bootstrap.servers: ${KAFKA_BOOTSTRAP_SERVERS:localhost:9092}
    topic.prefix: ${EXTERNAL_TOPIC_PREFIX:${kafka.agent.topic.prefix}}
  internal:
    bootstrap.servers: ${PS_KAFKA:localhost:9092}
    topic.prefix: ${INTERNAL_TOPIC_PREFIX:${kafka.agent.topic.prefix}}
  consumer:
    subscription-request:
      topic: ${kafka.external.topic.prefix}replication.rq
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.external.bootstrap.servers}
        group.id: ${kafka.external.topic.prefix}replicator-subscription-request
        auto.offset.reset: earliest
        enable.auto.commit: false
    subscription-cancel-request:
      topic: ${kafka.external.topic.prefix}replication.cancel.rq
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.external.bootstrap.servers}
        group.id: ${kafka.external.topic.prefix}replicator-subscription-cancel-request
        auto.offset.reset: earliest
        enable.auto.commit: false
    subscription-consumer-cancel-request:
      topic: ${kafka.external.topic.prefix}replication.cancel.in.rq
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.external.bootstrap.servers}
```

```
      group.id: ${kafka.external.topic.prefix}}replicator-subscription-consumer-
cancel-request
      auto.offset.reset: earliest
      enable.auto.commit: false
    subscription-storage-request:
      topic: ${kafka.external.topic.prefix}replication.in.rq
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.external.bootstrap.servers}
        group.id: ${kafka.external.topic.prefix}replicator-subscription-storage-request
        auto.offset.reset: earliest
        enable.auto.commit: false
    delta-request:
      topic: ${kafka.external.topic.prefix}delta.rq
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.external.bootstrap.servers}
        group.id: ${kafka.external.topic.prefix}replicator-delta-request
        auto.offset.reset: earliest
        enable.auto.commit: false
    delta-apply-notification:
      topic: ${kafka.internal.topic.prefix}subscription.in
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
        group.id: ${kafka.internal.topic.prefix}replicator-delta-apply-notification
        auto.offset.reset: earliest
        enable.auto.commit: false
    mppw-delta-apply-result:
      topic: ${kafka.internal.topic.prefix}mppw.delta.in.rs
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
        group.id: ${kafka.internal.topic.prefix}replicator-delta-apply-result
        auto.offset.reset: earliest
        enable.auto.commit: false

  producer:
    subscription-result: ${kafka.external.topic.prefix}replication.rs
    subscription-error: ${kafka.external.topic.prefix}replication.err
    subscription-cancel-error: ${kafka.external.topic.prefix}replication.cancel.rs
    subscription-cancel-result: ${kafka.external.topic.prefix}replication.cancel.rs
    subscription-consumer-cancel-result:
${kafka.external.topic.prefix}replication.cancel.in.rs
    subscription-storage-result: ${kafka.external.topic.prefix}replication.in.rs
    subscription-storage-error: ${kafka.external.topic.prefix}replication.in.err
    delta-error: ${kafka.external.topic.prefix}delta.err
    delta-apply-error: ${kafka.external.topic.prefix}delta.in.err
    delta-apply-result: ${kafka.external.topic.prefix}delta.in.rs
    delta-notification: ${kafka.external.topic.prefix}delta.notification
    property:
      bootstrap.servers: ${kafka.external.bootstrap.servers}
    internal:
      mppr-delta-request: ${kafka.internal.topic.prefix}mppr.delta.rq
      mppw-delta-apply-request: ${kafka.internal.topic.prefix}mppw.delta.in.rq
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
```

```
metrics:
  port: ${METRICS_PORT:9837}

log:
  replRequest: ${REPL_REQUEST_LOG_ENABLED:false}
  replResponse: ${REPL_RESPONSE_LOG_ENABLED:false}

backup:
  zk-path: ${REPLICATOR_BACKUP_ZK_PATH:/${environment.name}/podd-adapter-replicator}
  commandTopic: ${BACKUP_COMMAND_TOPIC:adapter.command}
  adapterCommandBroadcast:
${REPLICATOR_COMMAND_BROADCAST_TOPIC:adapter.command.broadcast}
  backupTopic: ${BACKUP_TOPIC:adapter.backup}
  statusTopic: ${STATUS_TOPIC:adapter.status}
  timeout: ${BACKUP_TIMEOUT:PT180s}
  idleDelay: ${BACKUP_DELAY:500}
  kafka:
    consumer:
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
        group.id: ${REPLICATOR_BACKUP_GROUP_ID:podd_adapter_replicator_adapter_command}
        auto.offset.reset: latest
    producer:
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
```

2.2.4.3 Параметры конфигурации

Настройка конфигурации **ПОДД-адаптера - Модуль подписок** осуществляется путем редактирования параметров настроек в файле `application.yml`.

В файле конфигурации **ПОДД-адаптера - Модуль подписок** могут быть настроены следующие секции:

- `environment` - указывается название окружения (`test`, `prod` и т.д.);
- `http-server` - настройки порта подключения;
- `executor` - масштабирования нагрузки на модуль;
- `zookeeper` – параметры подключения к Zookeeper;
- `migration` - настройки миграции;
- `prostore-api-client` - блок параметров конфигурирования взаимодействия с ProStore. Если false - будет использоваться JDBC-драйвер;
- `subscription` - настройки подписки;
- `kafka` - настройки параметров подключения к шине данных Apache Kafka;
- `log` - настройка сохранения лог-файла;
- `metrics` - настройка получения метрик;
- `backup` - настройки бекапирования.

2.2.4.3.1 Секция environment

В секции `environment` указывается среда разработки (dev, test, stable, prod)
Например:

```
environment:
  name: ${ENVIRONMENT_NAME:test}
```

Параметры настроек

– `name` - Название окружения, например `ENVIRONMENT_NAME:test`.

2.2.4.3.2 Секция http-server

Секция `http-server` предназначена для настройки порта и протокола передачи данных (одно из значений `http` или `https`).

Например:

```
http-server:
  port: ${HTTP_PORT:8085}
```

Параметры настроек

– `port` - порт веб-сервера, например: `HTTP_PORT:8085`.

2.2.4.3.3 Секция executor

Секция `executor` предназначена для масштабирования нагрузки на модуль. Увеличить или уменьшить нагрузку можно с помощью указания размера пула (`reader-pool-size`) чтения из Kafka.

Например:

```
executor:
  reader-pool-size: ${EXECUTOR_READER_POOL_SIZE:20}
```

Параметры настроек

– `reader-pool-size` - размер пула для чтения Kafka, например `EXECUTOR_READER_POOL_SIZE:20`.

2.2.4.3.4 Секция zookeeper

Секция `zookeeper` предназначена для настройки параметров подключения к Zookeeper.

Например:

```
zookeeper:
  connection-string: ${ZOOKEEPER_DS_ADDRESS:t5-adsp-01.ru-central1.internal}
  connection-timeout-ms: ${ZOOKEEPER_DS_CONNECTION_TIMEOUT_MS:30000}
  session-timeout-ms: ${ZOOKEEPER_DS_SESSION_TIMEOUT_MS:86400000}
  chroot: ${ZOOKEEPER_DS_CHROOT:/adapter}
```

Параметры настроек

– `ZOOKEEPER_DS_ADDRESS` - адрес сервера Zookeeper DS;

– `ZOOKEEPER_DS_SESSION_TIMEOUT_MS` - таймаут подключения к Zookeeper DS,максимальное время ожидания для выявления сбоев потребителей, указывается в миллисекундах (MS.);

– `ZOOKEEPER_DS_SESSION_TIMEOUT_MS` - таймаут сессии, максимальное время ожидания подключения к Zookeeper. Если ответ не получен до истечения установленного значения, клиент повторно отправляет запрос при необходимости. Указывается в

миллисекундах (MS.);

- ZOOKEEPER_DS_CHROOT - Zookeeper DS chroot path.

2.2.4.3.5 Секция migration

Секция migration реализована настройка миграции зукипера для задачи бекапирования.
Например:

```
migration:
  enabled: ${MIGRATION_ENABLE:false}
  old-connection-string: ${OLD_ZOOKEEPER_DS_ADDRESS:localhost}
```

Параметры настроек

- enabled - подключение миграции, например {MIGRATION_ENABLE:false};
- old-connection-string - адрес ZOOKEEPER, например {OLD_ZOOKEEPER_DS_ADDRESS:localhost}.

2.2.4.3.6 Секция prostore-rest-client

В секции prostore-rest-client реализован блок параметров конфигурирования взаимодействия с ProStore.
Например:

```
prostore-rest-client:
  enabled: true
  host: ${PS_HOST:t5-prostore-01.ru-central1.internal}
  port: ${PS_PORT:9195}
  http:
    max-pool-size: ${PS_MAX_POOL_SIZE:8}
```

Параметры настроек

- host - адрес Prostore, например PS_HOST:t5-prostore-01.ru-central1.internal;
- port - порт Prostore, например PS_PORT:9195;
- max-pool-size - максимальное число подключений к Prostore, например PS_MAX_POOL_SIZE:8.

2.2.4.3.7 Секция subscription

Секция subscription предназначена для настройки подписки на потребителе.
Например

```
subscription:
  consumer:
    # режим отмены подписки на потребителе. Возможные значения rename, drop, none
    cancel-mode: none
```

Параметры конфигурации

- cancel-mode - режим отмены подписки на потребителе. Возможные значения rename, drop, none.

2.2.4.3.8 Секция kafka

Секция kafka предназначена для настройки параметров подключения к шине данных Apache Kafka (используется для взаимодействия с ПОДД-адаптером) и настройки

взаимодействия через топики модуля *ПОДД-адаптер - Модуль исполнения запросов*.

Модуль взаимодействует через следующие топики:

- Запрос создания подписки (Поставщик данных): replication.rq/rs/err;
- Запрос отмены подписки (Поставщик данных): replication.cancel.rq/rs/err;
- Запрос дельты (Поставщик данных): delta.rq/mppr.delta.rq;
- Запрос создания структуры по подписке (Получатель данных): replication.in.rq/rs/err;
- Запрос применения дельты (Получатель данных): subscription.in/delta.in.rs/delta.in.err;
- Статусы с Prostore (Поставщик данных): status.event/delta.notification.

Например:

```
kafka:
  agent.topic.prefix: ${AGENT_TOPIC_PREFIX:}
  max-concurrent-handle: ${KAFKA_MAX_CONCURRENT_HANDLE:10}
  commit-interval: ${KAFKA_COMMIT_INTERVAL:5s}
  external:
    bootstrap.servers: ${KAFKA_BOOTSTRAP_SERVERS:localhost:9092}
    topic.prefix: ${EXTERNAL_TOPIC_PREFIX:${kafka.agent.topic.prefix}}
  internal:
    bootstrap.servers: ${PS_KAFKA:localhost:9092}
    topic.prefix: ${INTERNAL_TOPIC_PREFIX:${kafka.agent.topic.prefix}}
  consumer:
    subscription-request:
      topic: ${kafka.external.topic.prefix}replication.rq
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.external.bootstrap.servers}
        group.id: ${kafka.external.topic.prefix}replicator-subscription-request
        auto.offset.reset: earliest
        enable.auto.commit: false
    subscription-cancel-request:
      topic: ${kafka.external.topic.prefix}replication.cancel.rq
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.external.bootstrap.servers}
        group.id: ${kafka.external.topic.prefix}replicator-subscription-cancel-request
        auto.offset.reset: earliest
        enable.auto.commit: false
    subscription-consumer-cancel-request:
      topic: ${kafka.external.topic.prefix}replication.cancel.in.rq
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.external.bootstrap.servers}
        group.id: ${kafka.external.topic.prefix}}replicator-subscription-consumer-
cancel-request
        auto.offset.reset: earliest
        enable.auto.commit: false
    subscription-storage-request:
      topic: ${kafka.external.topic.prefix}replication.in.rq
      max-concurrent-handle: ${kafka.max-concurrent-handle}
```

```
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.external.bootstrap.servers}
        group.id: ${kafka.external.topic.prefix}replicator-subscription-storage-request
        auto.offset.reset: earliest
        enable.auto.commit: false
    delta-request:
      topic: ${kafka.external.topic.prefix}delta.rq
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.external.bootstrap.servers}
        group.id: ${kafka.external.topic.prefix}replicator-delta-request
        auto.offset.reset: earliest
        enable.auto.commit: false
    delta-apply-notification:
      topic: ${kafka.internal.topic.prefix}subscription.in
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
        group.id: ${kafka.internal.topic.prefix}replicator-delta-apply-notification
        auto.offset.reset: earliest
        enable.auto.commit: false
    mppw-delta-apply-result:
      topic: ${kafka.internal.topic.prefix}mppw.delta.in.rs
      max-concurrent-handle: ${kafka.max-concurrent-handle}
      commit-interval: ${kafka.commit-interval}
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
        group.id: ${kafka.internal.topic.prefix}replicator-delta-apply-result
        auto.offset.reset: earliest
        enable.auto.commit: false

  producer:
    subscription-result: ${kafka.external.topic.prefix}replication.rs
    subscription-error: ${kafka.external.topic.prefix}replication.err
    subscription-cancel-error: ${kafka.external.topic.prefix}replication.cancel.rs
    subscription-cancel-result: ${kafka.external.topic.prefix}replication.cancel.rs
    subscription-consumer-cancel-result:
${kafka.external.topic.prefix}replication.cancel.in.rs
    subscription-storage-result: ${kafka.external.topic.prefix}replication.in.rs
    subscription-storage-error: ${kafka.external.topic.prefix}replication.in.err
    delta-error: ${kafka.external.topic.prefix}delta.err
    delta-apply-error: ${kafka.external.topic.prefix}delta.in.err
    delta-apply-result: ${kafka.external.topic.prefix}delta.in.rs
    delta-notification: ${kafka.external.topic.prefix}delta.notification
    property:
      bootstrap.servers: ${kafka.external.bootstrap.servers}
    internal:
      mppr-delta-request: ${kafka.internal.topic.prefix}mppr.delta.rq
      mppw-delta-apply-request: ${kafka.internal.topic.prefix}mppw.delta.in.rq
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
```

Параметры конфигурации

– `AGENT_TOPIC_PREFIX` - значение префикса для топиков. Топики взаимодействия с *ПОДД-адаптером - Модуль исполнения запросов* (см. раздел «Спецификация модуля *ПОДД-адаптер-Модуль исполнения запросов*»).

2.2.4.3.9 Секция log

Секция `log` предназначена для настройки параметров логирования.
Например:

```
log:
  replRequest: ${REPL_REQUEST_LOG_ENABLED:false}
  replResponse: ${REPL_RESPONSE_LOG_ENABLED:false}
```

Параметры конфигурации

- `repl-request` - журналировать запросы к модулю подписок, например `REPL_REQUEST_LOG_ENABLED:false`;
- `repl-response` - журналировать ответы модуля подписок, например `REPL_RESPONSE_LOG_ENABLED:false`.

2.2.4.3.10 Секция metrics

Секция `metrics` предназначена для настройки параметров метрик.
Например:

```
metrics:
  port: ${METRICS_PORT:9837
```

Параметры конфигурации

- `port` - порт для получения метрик, например `9837`.

2.2.4.3.11 Секция backup

Секция `backup` предназначена для настроек бекапирования модуля.
Например:

```
backup:
  zk-path: ${REPLICATOR_BACKUP_ZK_PATH:/${environment.name}/podd-adapter-replicator}
  commandTopic: ${BACKUP_COMMAND_TOPIC:adapter.command}
  adapterCommandBroadcast:
${REPLICATOR_COMMAND_BROADCAST_TOPIC:adapter.command.broadcast}
  backupTopic: ${BACKUP_TOPIC:adapter.backup}
  statusTopic: ${STATUS_TOPIC:adapter.status}
  timeout: ${BACKUP_TIMEOUT:PT180s}
  idleDelay: ${BACKUP_DELAY:500}
  kafka:
    consumer:
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
        group.id: ${REPLICATOR_BACKUP_GROUP_ID:podd_adapter_replicator_adapter_command}
        auto.offset.reset: latest
    producer:
      property:
        bootstrap.servers: ${kafka.internal.bootstrap.servers}
```

Параметры настроек

- `zk-path` - путь к корневой ноде zookeeper для бэкапирования, например `{COUNTER_BACKUP_ZK_PATH:/${environment.name}/counter-provider/counters}`;
- `commandTopic` - топик команд бэкапирования, например: `{BACKUP_COMMAND_TOPIC:adapter.command}`;

– `backupTopic` - топик для отправки забэкапированных данных, например: `{BACKUP_TOPIC:adapter.backup}`;

– `statusTopic` - топик для отправки статусов бэкапирования, например: `{STATUS_TOPIC:adapter.status}`.

2.2.5 Настройка Модуля группировки чанков репликации

2.2.5.1 Конфигурация Модуля группировки чанков репликации (application.yml)

Файл `application.yml` – основной конфигурационный файл модуля, в котором описаны подключение к сервису Kafka, порт веб-сервера, и настройки журналирования запросов и ответов.

2.2.5.2 Пример файла application.yml

Приведем типовую структуру файла и возможные настройки **Модуля группировки чанков репликации**. Следует учитывать, что в конфигурационном файле следует задавать только те настройки, которые необходимы для решения текущих бизнес-задач.

```
http-server:
  port: ${HTTP_PORT:8084}

kafka:
  agent.topic.prefix: ${AGENT_TOPIC_PREFIX:}
  # максимальное количество обработчиков входящих запросов
  max-concurrent-handle: ${KAFKA_MAX_CONCURRENT_HANDLE:100}
  # периодичность фиксации оффсета обработанных сообщений
  commit-interval: ${KAFKA_COMMIT_INTERVAL:5s}
  external:
    bootstrap.servers: ${KAFKA_BOOTSTRAP_SERVERS:localhost:9092}
    topic.prefix: ${EXTERNAL_TOPIC_PREFIX:${kafka.agent.topic.prefix}}
  internal:
    bootstrap.servers: ${PS_KAFKA:localhost:9092}
    topic.prefix: ${INTERNAL_TOPIC_PREFIX:${kafka.agent.topic.prefix}}
  consumer:
    delta-apply-request: ${kafka.external.topic.prefix}delta.in.rq
    property:
      bootstrap.servers: ${kafka.external.bootstrap.servers}
      group.id: ${kafka.external.topic.prefix}podd-adapter-group-repl
      auto.offset.reset: earliest
      enable.auto.commit: false
  producer:
    refresh-interval: 60s
    delta-apply-notification: ${kafka.internal.topic.prefix}subscription.in
    property:
      bootstrap.servers: ${kafka.internal.bootstrap.servers}

logging:
  scl.delta:
      enabled: ${SCL_DELTA_ENABLED:false}

metrics:
  port: ${METRICS_PORT:9837}
```

2.2.5.3 Параметры конфигурации

Настройка конфигурации **Модуля группировки чанков репликации** осуществляется путем редактирования параметров настроек в файле `application.yml`.

Пример файла application.yml для **Модуля группировки чанков репликации**.

В файле конфигурации **Модуля группировки чанков репликации** могут быть настроены следующие секции:

- `http-server` - указывается порт веб-сервера;
- `kafka` - настройки параметров подключения к шине данных Apache Kafka;
- `logging` - настройки журналирования запросов и ответов;
- `metrics` - настройка порта для получения метрик.

2.2.5.3.1 Секция http-server

В секции `http-server` указывается порт веб-сервера.

Например:

```
http-server:
  port: ${HTTP_PORT:8084}
```

Параметры настроек

- `port` - порт веб-сервера, например: `HTTP_PORT:8084`.

2.2.5.3.2 Секция kafka

Секция `kafka` определяет настройки взаимодействия через ПОДД-адаптер между Поставщиком данных (`producer`) и Получателем данных (`consumer`).

В секции `kafka` собраны настройки параметров подключения к шине данных Apache Kafka.

Например:

```
kafka:
  agent.topic.prefix: ${AGENT_TOPIC_PREFIX:}
  max-concurrent-handle: ${KAFKA_MAX_CONCURRENT_HANDLE:100}
  commit-interval: ${KAFKA_COMMIT_INTERVAL:5s}
  external:
      bootstrap.servers: ${KAFKA_BOOTSTRAP_SERVERS:localhost:9092}
      topic.prefix: ${EXTERNAL_TOPIC_PREFIX:${kafka.agent.topic.prefix}}
  internal:
      bootstrap.servers: ${PS_KAFKA:localhost:9092}
      topic.prefix: ${INTERNAL_TOPIC_PREFIX:${kafka.agent.topic.prefix}}
  consumer:
    delta-apply-request: ${kafka.external.topic.prefix}delta.in.rq
    property:
      bootstrap.servers: ${kafka.external.bootstrap.servers}
      group.id: ${kafka.external.topic.prefix}podd-adapter-group-repl
      auto.offset.reset: earliest
      enable.auto.commit: false
  producer:
    refresh-interval: 60s
    delta-apply-notification: ${kafka.internal.topic.prefix}subscription.in
    property:
      bootstrap.servers: ${kafka.internal.bootstrap.servers}
```

Параметры конфигурации

- `topic` - префикс для топиков агента ПОДД, например `AGENT_TOPIC_PREFIX`;
- `max-concurrent-handle` - максимальное количество обработчиков входящих запросов, например `KAFKA_MAX_CONCURRENT_HANDLE:100`;

– `commit-interval` - периодичность фиксации оффсета обработанных сообщений, например `KAFKA_COMMIT_INTERVAL:5s`.

2.2.5.3.3 Секция logging

Секция `logging` предназначена для настройки журналирования запросов и ответов
Например:

```
logging:
scl.delta:
    enabled: ${SCL_DELTA_ENABLED:false}
```

Параметры конфигурации

– `enabled` - Журналировать события SCL delta, например: `SCL_DELTA_ENABLED:false`.

LOG_FORMAT - Логирование в формате (JSON/TEXT) - указывается в `logback.xml`.

2.2.5.3.4 Секция metrics

Секция `metrics` предназначена для настроек порта получения метрик.
Например:

```
metrics:
  port: ${METRICS_PORT:9837}
```

Параметры настроек

– `port` - Порт для получения метрик, например `{METRICS_PORT:9837}`.

2.2.6 Настройка CSV-Uploader

2.2.6.1 Конфигурация CSV-uploader (application.yml)

Файл `application.yml` – основной конфигурационный файл CSV-uploader, в котором задана логика и порядок работы загрузчика, а также другие настройки необходимые для корректной работы адаптера.

2.2.6.1.1 Пример файла application.yml

```
# Kafka Prostore
.kafkaUrl: &kafkaUrl ${KAFKA_BOOTSTRAP_SERVERS:localhost:9092}

http-server:
        # Порт для старта веб сервера
        port: ${HTTP_PORT:8080}
        # Включить веб-сервер
        enabled: ${HTTP_ENABLED:true}

send:
        # Размер отправляемой порции данных
        chunk-row-count: ${CHUNK_ROW_COUNT:1000}
        # Размер буфера на чтение файла
        file-buffer-size: ${FILE_BUFFER_SIZE:1048576}
        # Количество Job на чтение
        read-job-count: ${READ_JOB_COUNT:4}
        # Размер Channel для сериализации
        serialize-channel-size: ${SERIALIZE_CHANNEL_SIZE:20}
        # Количество Job на сериализацию
        serialize-job-count: ${SERIALIZE_JOB_COUNT:4}
        # Размер Job на отправку
        send-channel-size: ${SEND_CHANNEL_SIZE:20}
```

```
        # Количество Job на отправку
        send-job-count: ${SEND_JOB_COUNT:4}

file-size:
        # Ограничение на размер отправляемого файла (мегабайты)
        restriction: ${SEND_FILE_SIZE_RESTRICTION:1024}

logging.level:
        root: info
        ru.itone: debug

environment:
        # Название окружения
        name: ${ENVIRONMENT_NAME:test}
        # Папка для ошибочных файлов
        error-folder: ${ENVIRONMENT_ERROR_FOLDER:error}

zookeeper:
        # Адрес сервера zookeeper
        connection-string: ${ZK_CONNECTION:localhost}
        # Таймаут сессии
        session-timeout-ms: ${ZK_SESSION_TIMEOUT_MS:30000}
        # Таймаут подключения
        connection-timeout-ms: ${ZK_CONNECTION_TIMEOUT_MS:86400000}
        chroot: ${ZOOKEEPER_DS_CHROOT:/adapter}

migration:
        enabled: ${MIGRATION_ENABLE:false}

prostore-rest-client:
        # Признак использования rest-api для взаимодействия с простором.
        enabled: ${PS_REST_CLIENT_ENABLED:true}
        host: ${PS_HOST:localhost}
        port: ${PS_PORT:9195}
        http:
                max-pool-size: ${PS_MAX_POOL_SIZE:8}

prostore:
        zookeeper:
                # Адрес сервера zookeeper для загрузки данных в простор
                connection-string: ${ZK_PROSTORE_CONNECTION:locahost:2181}

validation:
        enable: ${VALIDATION_ENABLE:true}
        rest-uploader-url: ${REST_UPLOADER_URL:http://localhost:8081}
        # обязательность использования ФЛК
        mandator: ${VALIDATION_MANDATOR:false}

upload:
        # требуется токен для аутентификации на rest-uploader
        jwt-auth: ${JWT_AUTH:false}

kafka:
        create-topic:
                # Количество партиций на загрузку через EDML
                num-partitions: ${EDML_UPLOAD_NUM_PARTITIONS:1}
                # Фактор репликации при создании топика
                replication-factor: ${EDML_UPLOAD_REPLICATION_FACTOR:1}
        topic:
                # Топик для журналирования
                journal-log: journal.log
                flk-log: flk.log
```

```
        consumer:
                # Количество партиций на выгрузку через EDML
                num-partitions: ${EDML_DOWNLOAD_NUM_PARTITIONS:1}
                property:
                        bootstrap.servers: *kafkaUrl
                        group.id: csv-uploader
                        auto.offset.reset: earliest
                        enable.auto.commit: true
        producer:
                property:
                        bootstrap.servers: *kafkaUrl

csv-parser:
        separator: ${CSV_PARSER_SEPARATOR:;}
        quote-char: ${CSV_PARSER_QUOTE_CHAR:"}
        escape-char: ${CSV_PARSER_ESCAPE_CHAR:'}
        field-as-null: ${CSV_PARSER_FIELD_AS_NULL:EMPTY_SEPARATORS}

metrics:
        port: ${METRICS_PORT:9837}

backup:
        zk-path: ${CSV_UPLOADER_BACKUP_ZK_PATH:/${environment.name}/csv-uploader/config}
        commandTopic: ${BACKUP_COMMAND_TOPIC:adapter.command}
        backupTopic: ${BACKUP_TOPIC:adapter.backup}
        statusTopic: ${STATUS_TOPIC:adapter.status}
        kafka:
                consumer:
                        property:
                                bootstrap.servers: ${KAFKA_BOOTSTRAP_SERVERS:localhost}
                                group.id:
${CSV_UPLOADER_BACKUP_GROUP_ID:csv_uploader_adapter_command}
                                auto.offset.reset: latest
                producer:
                        property:
                                bootstrap.servers: ${KAFKA_BOOTSTRAP_SERVERS:localhost}

jet-connector:
        use: ${JET_CONNECTOR_USE:false}
```

2.2.6.2 Параметры конфигурации

Настройка конфигурации **CSV-uploader** осуществляется путем редактирования параметров настроек в файле `application.yml`. Некоторые настройки доступны для редактирования через пользовательский интерфейс модуля, например, Настройка отображения количества записей в *Журнале операций* и Запуск по расписанию.

Пример конфигурации файла `application.yml` для **CSV-uploader** см. в разделе Пример файла application.yml.

В файле конфигурации **CSV-uploader** могут быть настроены следующие секции:

- `kafkaUrl` - URL для доступа к Kafka;
- `http-server` - настройки порта подключения;
- `send` - настройка отправки файлов;
- `file-size` - ограничение на размер отправляемого файла (мегабайты);
- `logging.level` - настройка сохранения лог-файла;
- `environment` - определяет значение среды разработки;

- `zookeeper` - настройка подключения Zookeeper;
- `migration` - настройка миграции зукипера для задачи бекапирования;
- `prostore-rest-client` - блок параметров конфигурирования взаимодействия с ProStore;
- `prostore` - адрес сервера zookeeper для загрузки данных в ProStore;
- `validation` - включение/выключение механизма валидации загрузки с помощью rest-uploader сервиса;
- `upload` - требование токена для аутентификации на rest-uploader;
- `kafka` - настройка подключения к шине данных Apache Kafka;
- `csv-parser` - настройка парсинга CSV;
- `metrics` - настройка получения метрик;
- `backup` - настройка бэкапирования модуля;
- `jet-connector` - подготовлен для оптимизации задержек записи.

2.2.6.2.1 Секция kafkaUrl

В секция `kafkaUrl` указывается URL-адрес для доступа к Apache Kafka (ProStore).
Например:

```
.kafkaUrl: &kafkaUrl ${KAFKA_BOOTSTRAP_SERVERS:dev-dtm-one05.ru-central1.internal:9092}
```

Параметры конфигурации

`KAFKA_BOOTSTRAP_SERVERS` - URL-адрес для доступа к Apache Kafka (ProStore).

2.2.6.2.2 Секция http

Секция `http` предназначена для настройки порта и протокола передачи данных (одно из значений `http` или `https`).

Например:

```
http:
       port: ${HTTP_PORT:8080}
       enabled: ${HTTP_ENABLED:true}
```

Параметры конфигурации

- `port` - порт для старта веб-сервера;
- `enabled` - статус включения/отключения веб-сервера.

2.2.6.2.3 Секция query-executor

Секция `query-executor` определяет настройки настройка получения входящих запросов.
Например:

```
query-executor:
       reader-pool-size: ${READER_POOL_SIZE:20}
       rest-pool-size: ${REST_POOL_SIZE:5}
       writer-pool-size: ${WRITER_POOL_SIZE:1}
       kafka-pool-size: ${KAFKA_POOL_SIZE:20}
```

Параметры конфигурации

- `reader-pool-size` - размер пула на чтение, например: `READER_POOL_SIZE:20`;

- `rest-pool-size` - размер пула на REST, например: `REST_POOL_SIZE:5`;
- `writer-pool-size` - размер пула на запись, например: `WRITER_POOL_SIZE:1`;
- `kafka-pool-size` - размер kafka пула на отправку, например `KAFKA_POOL_SIZE:20`.

2.2.6.2.4 Секция send

Секция `send` определяет настройки отправки файлов.
Например:

```
send:
        chunk-row-count: ${CHUNK_ROW_COUNT:1000}
        file-buffer-size: ${FILE_BUFFER_SIZE:1048576}
        read-job-count: ${READ_JOB_COUNT:4}
        serialize-channel-size: ${SERIALIZE_CHANNEL_SIZE:20}
        serialize-job-count: ${SERIALIZE_JOB_COUNT:4}
        send-channel-size: ${SEND_CHANNEL_SIZE:20}
        send-job-count: ${SEND_JOB_COUNT:4}
```

Параметры конфигурации:

- `chunk-row-count` - размер отправляемой порции данных, например `CHUNK_ROW_COUNT:100`;
- `file-buffer-size` - размер буфера на чтение файла, например `FILE_BUFFER_SIZE:1048576`;
- `read-job-count` - количество Job на чтение, например `READ_JOB_COUNT:4`;
- `serialize-channel-size` - размер `Channel` для сериализации, например `SERIALIZE_CHANNEL_SIZE:20`;
- `serialize-job-count` - количество задач на сериализацию, например `SERIALIZE_JOB_COUNT:4`;
- `send-channel-size` - размер задач на отправку, например `SEND_CHANNEL_SIZE:20`;
- `send-job-count` - количество задач на отправку, например `SEND_JOB_COUNT:4`;

2.2.6.2.5 Секция file-size

Секция `file-size` отвечает за ограничение на размер отправляемого файла (мегабайты)

```
file-size:
        #
        restriction: ${SEND_FILE_SIZE_RESTRICTION:1024}
```

Параметры конфигурации

- `restriction` - ограничение на размер оправляемого файла (мегабайты), например `SEND_FILE_SIZE_RESTRICTION:1024`.

2.2.6.2.6 Секция logging.level

Секция `logging.level` определяет настройки записи логирования.
Например:

```
logging.level:
        root: info
        ru.itone: debug
```

2.2.6.2.7 Секция environment

Секция `environment` определяет значение среды разработки (например, значение `test`, `prod` и т.д).

Например:

```
environment:
        name: ${ENVIRONMENT_NAME:test}
        error-folder: ${ENVIRONMENT_ERROR_FOLDER:error}
```

Параметры конфигурации

- `name` - название окружения, например `ENVIRONMENT_NAME:test`;
- `error-folder` - папка для ошибочных файлов, например `ENVIRONMENT_ERROR_FOLDER:error`.

2.2.6.2.8 Секция zookeeper

Секция `zookeeper` предназначена для настройки параметров подключения к серверу Zookeeper.

Например:

```
zookeeper:
        connection-string: ${ZK_CONNECTION:t5-ads-02.ru-central1.internal}
        session-timeout-ms: ${ZK_SESSION_TIMEOUT_MS:30000}
        connection-timeout-ms: ${ZK_CONNECTION_TIMEOUT_MS:86400000}
```

Параметры конфигурации

- `connection-string` - адрес сервера Zookeeper, например `ZK_CONNECTION:t5-ads-02.ru-central1.internal`;
- `session-timeout-ms` - таймаут сессии, например `ZK_SESSION_TIMEOUT_MS:30000`;
- `connection-timeout-ms` - таймаут подключения, например `ZK_CONNECTION_TIMEOUT_MS:86400000`.

2.2.6.2.9 Секция migration

В секции `migration` реализована настройка миграции зукипера для задачи бекапирования

Например:

```
migration:
        enabled: ${MIGRATION_ENABLE:false}
```

Параметры настроек

- `enabled` - включение миграции (по умолчанию выключена), например `{MIGRATION_ENABLE:false}`.

2.2.7.2.10 Секция prostore-rest-client

В секции `prostore-rest-client` реализован блок параметров конфигурирования

взаимодействия с ProStore.

Например:

```
prostore-rest-client:
       enabled: ${PS_REST_CLIENT_ENABLED:true}
       host: ${PS_HOST:localhost}
       port: ${PS_PORT:9195}
       http:
              max-pool-size: ${PS_MAX_POOL_SIZE:8}
```

Параметры настроек

- `host` - адрес Prostore, например `PS_HOST:localhost`;
- `port` - порт Prostore, например `PS_PORT:9195`;
- `max-pool-size` - максимальное число подключений к Prostore, например `PS_MAX_POOL_SIZE:8`.

2.2.6.2.11 Секция prostore

В секции `prostore` указывается адрес сервера zookeeper для загрузки данных в Prostore.

Например:

```
prostore:
       zookeeper:
              connection-string: ${ZK_PROSTORE_CONNECTION:locahost:2181}
```

Параметры настроек

- `connection-string` - адрес сервера zookeeper для загрузки данных в Prostore, например `ZK_PROSTORE_CONNECTION:locahost:2181`.

2.2.6.2.12 Секция validation

В секции validation реализован механизм настройки валидации ФЛК.

Например:

```
enable: ${VALIDATION_ENABLE:true}
rest-uploader-url: ${REST_UPLOADER_URL:http://localhost:8081}
mandator: ${VALIDATION_MANDATOR:false}
```

Параметры конфигурации

- `enable` - валидация включена (по умолчанию), например `{VALIDATION_ENABLE:true}`;
- `rest-uploader-url` - URL к сервису rest-uploader для выполнения валидации, например `{REST_UPLOADER_URL:http://localhost:8081}`;
- `mandator` - обязательность использования ФЛК, например `{VALIDATION_MANDATOR:false}`.

2.2.6.2.13 Секция upload

В секции upload реализована настройка требования токена для аутентификации на REST-Uploader (если `true`, то при переключении на вкладку **Загрузка** появляется модальное окно для задания токена в текстовом виде и кнопка **Сохранить**)

Например:

```
upload:
        jwt-auth: ${JWT_AUTH:false}
```

Параметры конфигурации

– `jwt-auth` - требование токена для аутентификации на REST-Uploader, например `{JWT_AUTH:false}`.

2.2.6.2.14 Секция kafka

Секция `kafka` предназначена для настройки параметров подключения к шине данных Apache Kafka.

Например:

```
kafka:
        create-topic:
                num-partitions: ${EDML_UPLOAD_NUM_PARTITIONS:1}
                replication-factor: ${EDML_UPLOAD_REPLICATION_FACTOR:1}
        topic:
                journal-log: journal.log
        consumer:
                num-partitions: ${EDML_DOWNLOAD_NUM_PARTITIONS:1}
                property:
                        bootstrap.servers: *kafkaUrl
                        group.id: csv-uploader
                        auto.offset.reset: earliest
                        enable.auto.commit: true
        producer:
                property:
                        bootstrap.servers: *kafkaUrl
```

Параметры конфигурации

– `num-partitions` - количество партиций на загрузку через EDML, например `EDML_UPLOAD_NUM_PARTITIONS:1`;

– `replication-factor` - фактор репликации при создании топика, например `EDML_UPLOAD_REPLICATION_FACTOR:1`.

2.2.6.2.15 Секция csv-parser

> Внимание:
>
> При загрузке файлов с форматно-логическим контролем, важно, чтобы настройки секции csv-parser были одинаковы в модулях CSV-Uploader(если используется его UI),REST-Uploader и DATA-Uploader.

Секция `csv-parser` - настройка парсинга CSV.

Например:

```
csv-parser:
        separator: ${CSV_PARSER_SEPARATOR:;}
        quote-char: ${CSV_PARSER_QUOTE_CHAR:"}
        escape-char: ${CSV_PARSER_ESCAPE_CHAR:'}
        field-as-null: ${CSV_PARSER_FIELD_AS_NULL:EMPTY_SEPARATORS}
```

Параметры конфигурации

– `separator` - символ разделителя полей, например `CSV_PARSER_SEPARATOR:;`;

- `quote-char` - символ кавычки, например `CSV_PARSER_QUOTE_CHAR:"`;
- `escape-char` - символ экранирования, например `CSV_PARSER_ESCAPE_CHAR:'`;
- `field-as-null` - способ определения null поля, например `CSV_PARSER_FIELD_AS_NULL:EMPTY_SEPARATORS`.

Дополнительное описание параметров

1. Параметр `CSV_PARSER_ESCAPE_CHAR` работает следующим образом: если символ экранирования и символ кавычки равны `"`, то будет использован RFC4180Parser, который считывает все символы между двумя двойными кавычками, при этом двойная кавычка в тексте поля должна быть экранирована двойной кавычкой (Например `"поле, ""содержащее двойную кавычку"""` будет считано как `поле, "содержащее двойную кавычку"`). В противном случае будет использован CSVParser, использующий символ экранирования для обозначения «непечатаемых символов».
2. Параметр `CSV_PARSER_FIELD_AS_NULL` может принимать следующие значения:
    - `EMPTY_SEPARATORS` - два разделителя полей (см. csv-parser/separator) подряд считаются null. Например: строка [aaa,,ccc] содержит значения [«aaa», null, «bbb»], а строка [aaa,»»,ccc] содержит значения [«aaa», «», «bbb»].
    - `EMPTY_QUOTES` - два «ограничителя строки» (см. csv-parser/escape-char) подряд считаются null. Например: строка [aaa,»»,ccc] содержит значения [«aaa», null, «bbb»], а строка [aaa,,ccc] содержит значения [«aaa», «», «bbb»].
    - `BOTH` - оба варианта (см. `EMPTY_SEPARATORS` и `EMPTY_QUOTES`) считаются null. Например: обе строки [aaa,»»,ccc] и [aaa,,bbb] содержат одинаковое значение [«aaa», null, «bbb»].
    - `NEITHER` - ни один из вариантов (см. `EMPTY_SEPARATORS` и `EMPTY_QUOTES`) не считается null. Например: обе строки [aaa,»»,ccc] и [aaa,,bbb] содержат одинаковое значение [«aaa», «», «bbb»].

2.2.6.2.16 Секция metrics

Секция `metrics` предназначена для настройки параметров метрик.
Например:

```
metrics:
  port: ${METRICS_PORT:9837}
```

Параметры конфигурации

- `port` - Порт для метрик, например `METRICS_PORT:9837`.

2.2.6.2.17 Секция backup

Секция `backup` предназначена для настроек бекапирования модуля.
Например:

```
backup:
  zk-path: ${COUNTER_BACKUP_ZK_PATH:/${environment.name}/counter-provider/counters}
  commandTopic: ${BACKUP_COMMAND_TOPIC:adapter.command}
  backupTopic: ${BACKUP_TOPIC:adapter.backup}
  statusTopic: ${STATUS_TOPIC:adapter.status}
  kafka:
    consumer:
      property:
        bootstrap.servers: ${KAFKA_BOOTSTRAP_SERVERS:localhost:9092}
        group.id: ${COUNTER_BACKUP_GROUP_ID:counter_provider_adapter_command}
        auto.offset.reset: latest
    producer:
      property:
        bootstrap.servers: ${KAFKA_BOOTSTRAP_SERVERS:localhost:9092}
```

Параметры настроек

- `zk-path` - путь к корневой ноде zookeeper для бэкапирования, например `{COUNTER_BACKUP_ZK_PATH:/${environment.name}/counter-provider/counters}`;
- `commandTopic` - топик команд бэкапирования, например: `{BACKUP_COMMAND_TOPIC:adapter.command}`;
- `backupTopic` - топик для отправки забэкапированных данных, например: `{BACKUP_TOPIC:adapter.backup}`;
- `statusTopic` - топик для отправки статусов бэкапирования, например: `{STATUS_TOPIC:adapter.status}`.

2.2.6.2.18 Секция jet-connector

Секция `jet-connector` предназначена для оптимизации задержек записи данных. Например:

```
jet-connector:
use: ${JET_CONNECTOR_USE:false}
```

Параметры настроек

- `use` - флаг активации jet connector'а, например `{JET_CONNECTOR_USE:false}`;

Таблица 2.1 Область применения

Режим/СУБД	ADB	ADP	ADQM	ADG
Чтение	Нет	Нет	Нет	Нет
Запись	В перспективе	Да	Нет	Нет

Чтобы воспользоваться Jet коннектором требуется вместо upload external table создавать readable external table, указывающую на топик, как отражено в документации Prostore

В модуль MPPW не передается информация о том, в какую БД физически будут загружаться данные, синтаксис Простора един для всех поддерживаемых СУБД. Конкретная СУБД указывается лишь в настройках Простора.

Даже если загрузка данных выполняется в более чем одну базу, на работе адаптеров это не сказывается.

При формировании запросов в табличными параметрами, в том числе при регистрации

РЗ, необходимо явно перечислять поля таблиц, по которым будет выполняться фильтрация записей или объединение таблиц.

Переключение с **jet-connector** на **kafka postgres writer** и наоборот допускается лишь при завершенных операциях загрузки данных.

> Предупреждение:
>
> Jet коннектор в настоящее время применим лишь для ADP, что делает его применимым, только для исталляций одной единственной СУБД ADP. Это ограничение остается на уровне документации при использовании JET коннектора с другими базами должна появиться ошибка

2.2.7 Настройка Агента ПОДД

Порядок установки и описание настроек Агента ПОДД см. в документе: **«17514186.СМЭВ-2021-3.О1.РА Руководство администратора ПОДД СМЭВ Часть 1. Агент ПОДД СМЭВ»**.

Описание формата взаимодействия между Агентом ПОДД и ПОДД-адаптером (название топиков, формат сообщений, схема взаимодействия) описан в документе «Спецификация модуля ПОДД-адаптера - Модуль исполнения запросов».

2.2.9 Настройка взаимодействия программы с Агентом ПОДД

После установки программы и *Агента ПОДД* надо настроить их взаимодействие между собой. Для этого:

1. Настройте *Агента ПОДД* и *ПОДД-адаптер* на работу с одним и тем же брокером сообщения Kafka:
   - Если вместе с *Агентом ПОДД* устанавливается брокер сообщений Kafka, а *Агент ПОДД* преднастроен на работу именно с этим экземпляром брокера сообщений, то укажите адрес этого брокера сообщений в конфигурационном файле *ПОДД-адаптера* (*application.yml*), параметр `kafkaUrl`.
   - Если вместе с *Агентом ПОДД* не устанавливается брокер сообщений Kafka, то в *Агенте ПОДД* согласно его документации настройте работу с брокером сообщений Kafka, установленным с программой. Для этого используйте адрес сервера Kafka из конфигурационного файла *ПОДД-адаптера* (*application.yml*), параметр `kafkaUrl`.
1. Настройте названия топиков (см. Таблица 2.2) для обмена сообщениями в конфигурационном файле *ПОДД-адаптера* (*application.yml*).

Таблица 2.2 Название топиков для обмена сообщениями между ПОДД-адаптером и Агентом ПОДД

№	Назначение	Настройка	Значение по умолчанию
1	Получение запросов	`client.kafka.query.consumer.rqTopicName`	`query.rq`
2	Ответы на запросы	`client.kafka.query.producer.rsTopicName`	`query.rs`

№	Назначение	Настройка	Значение по умолчанию
3	Ошибки запросов	`client.kafka.query.producer.errTopicName`	`query.err`
4	Результат запроса оценки	`client.kafka.query.estimateTopicName`	`query.query.estimation.rs`

Формат обмена электронными сообщениями с ПОДД-адаптер описан в разделе Спецификация модуля ПОДД-адаптера - Модуль исполнения запросов.

3 ЗАПУСК И ОСТАНОВКА ПРОГРАММЫ

Программа не имеет графического интерфейса. При необходимости любой из сервисов/модулей можно остановить и запустить заново.

Остановка модуля выполняется при помощи Docker команды:

```
docker start dtm-adapter-reader-das.local
```

В случае, если модуль поставляется как jar-файл, то следует выполнить команду:

```
java
    [-Dconfig.location=<путь до application.yml> ]
    [-Dloader.path=file:dtm-jdbc-***.jar ]
    [-Dlogging.config=logback.xml]
    -jar <путь до rest-uploader.jar>
```

где, команды заключенные в [] выполняются опционально. *** - номер версии JDBC-драйвера.

Для ручной остановки и запуска необходимо подключиться по SSH на сервер и с правами sudo, найти процесс, который содержит jar-файл и остановить.

Например, для CSV-Uploader команда будет следующей:

```
ps aux | grep csv-uploader
```

Описание первоначального запуска системы описано в документе **«Руководство по установке»**.

4 РЕЗЕРВНОЕ КОПИРОВАНИЕ

В программе необходимо настроить резервное копирование для базы данных ProStore (PostrgeSQL).

Для этого следует использовать рекомендованную разработчиками PostrgeSQL программу для создания резервных копий базы данных PostgreSQL - **pg_dump**, которая создаёт целостные копии, даже при параллельном использовании базы данных. Программа **pg_dump** не препятствует доступу других пользователей к базе данных (ни для чтения, ни для записи).

Программа **pg_dump** выгружает только одну базу данных. Чтобы сохранить глобальные объекты, относящиеся ко всем базам в кластере, например роли и табличные пространства, воспользуйтесь программой **pg_dumpall**.

Выгружаемые данные могут быть сохранены в виде скрипта, либо в одном из архивных форматов. Скрипты представляют собой текстовые файлы, содержащие SQL-команды, необходимые для воссоздания базы данных до состояния на момент создания скрипта. Для восстановления из скрипта его содержимое можно передать *psql*. Скрипты можно использовать для восстановления базы на других машинах, в том числе с иной архитектурой, а с некоторыми коррективами даже в других СУБД.

Для восстановления из архивных форматов файлов используется утилита **pg_restore**. Эти форматы позволяют указывать **pg_restore** какие объекты базы данных восстановить, а также позволяют изменить порядок следования восстанавливаемых объектов. Архивные форматы файлов спроектированы так, чтобы их можно были переносить на другие платформы с другой архитектурой.

Применение архивных форматов в сочетании утилит **pg_restore** и **pg_dump** позволяет организовывать эффективный механизм архивации и переноса данных. **pg_dump** можно использовать для резервирования всей базы данных, а затем при применении **pg_restore** выбрать нужные объекты для восстановления. Наиболее гибкие форматы выходных файлов это «custom» (-Fc) и «directory» (-Fd). Они позволяют выбрать и изменить порядок объектов, поддерживают восстановление в несколько потоков, а также сжимаются по умолчанию. При этом только формат `directory` поддерживает выгрузку данных в несколько потоков.

Во время работы **pg_dump** следует обращать внимание на предупреждения, которые печатаются в стандартный поток ошибок, особенно ввиду рассмотренных далее ограничений.

4.1 Плановое резервное копирование информации

Администратор резервного копирования настраивает процедуры резервного копирования информационных ресурсов Витрины с периодичностью их выполнения не реже, чем 1 раз в сутки.

Состав копируемой информации, используемые устройства хранения резервных копий, выбор методов резервного копирования и программных средств для выполнения данных операций, ротация резервных копий определяются администратором резервного копирования самостоятельно и могут быть изменены им с целью оптимизации процесса резервного копирования и использования устройств хранения резервных копий.

Администратор резервного копирования вправе менять периодичность проведения операций резервного копирования в сторону уменьшения временных интервалов между процедурами резервного копирования.

Полное копирование должно применяться не реже, чем один раз в неделю.

4.2 Рекомендации по выполнению резервного копирования

1. Определить период времени для резервного копирования, когда в программу не выполняется загрузка данных.
2. Убедиться, что нет активных загрузок, для этого выполнить команду:

```
get_delta_hot(),
```

Пример ответа:

```
0
```

где, `0` – активных загрузок нет.

1. Остановить следующие docker-контейнеры:
   - CSV-uploader;
   - Сервис исполнения запросов.
4. Сделать резервную копию *volume* для Apache ZooKeeper:

```
sudo tar -czvf ~/zokeeper.tgz /var/lib/docker/volumes/zookeeper/_data
```

5. Сделать резервную копию PostgreSQL через *pg_dump* (внутри docker-контейнера), для этого выполните команду:

```
docker exec -it postgres pg_dump -U dtm -d test -Fc -f /var/lib/postgresql/data/db.dump
```

где, */var/lib/docker/volumes/postgres/_data/db.dump* – директория, в которую будет выгружена резервная копия.

6. Запустить *Сервис исполнения запросов*.
7. Запустить CSV-uploader.

4.3 Контроль результатов резервного копирования

Контроль результатов выполнения процедур резервного копирования осуществляется администратором резервного копирования в срок до 12:00 рабочего дня, следующего за установленной датой выполнения этих процедур. Контроль результатов резервного копирования производится путем просмотра журналов событий операционной системы и специализированного программного обеспечения. В случае обнаружения сбоя планового резервного копирования администратор резервного копирования должен выполнить процедуру внепланового резервного копирования информационных ресурсов Витрины.

4.4 Внеплановое резервное копирование информационных ресурсов Системы

Внеплановое резервное копирование информационных выполняется администратором резервного копирования в следующих случаях:

- сбоя планового резервного копирования информационных ресурсов Системы;
- установки на сервер нового программного обеспечения, модификации установленного программного обеспечения.

Действия администратора резервного копирования при внеплановом сохранении информационных ресурсов аналогичны действиям при плановом резервном копировании, при этом если внеплановое резервное копирование производится по причине сбоя планового, то администратор должен выявить и устранить причину возникновения сбоя, выполнив следующие действия:

- Проверить функционирование аппаратной части сервера (дисковые массивы, оперативная память, сетевая карта). Устранить причины сбоя путем замены вышедших из строя компонентов;
- Проверить функционирование операционной системы сервера. Устранить причины сбоя путем перенастройки, установки патчей, переустановки операционной системы;
- Проверить доступности сервера из локальной вычислительной сети. Проверить функционирование сетевых сервисов операционной системы, сетевых настроек. Устранить причины сбоя путем переустановки и перенастройки сетевых сервисов.

4.5 Восстановление информации из резервных копий

Восстановление данных из резервных копий производится администратором резервного копирования самостоятельно в случаях неработоспособности серверов или сервисов либо на основании служебной записки, ответственного за наполнение и работоспособность ПО, если содержимое информационных ресурсов повреждено или нарушена работоспособность Системы.

5 ДОПОЛНИТЕЛЬНЫЕ ВОЗМОЖНОСТИ

Необходимость выполнения действий данного раздела определяется в процессе эксплуатации программы.

5.1 Логирование

Сбор лог-файлов программы, с записями о событиях производится с помощью *Graylog*, через утилиту полнотекстового поиска и аналитики **Elasticsearch**, которая позволяет в режиме реального времени хранить, искать и анализировать большие объемы данных.

При запуске **Graylog** автоматически конфигурирует **Elasticsearch**.

Для передачи сообщений в **Graylog** используется **Filebeat**.

Просмотр записей лог-файлов доступен через web-интерфейс *Graylog* (см. Рисунок - 5.1) по адресу http://0.0.0.0:9010/ (авторизация: admin/somepasswordpepper).

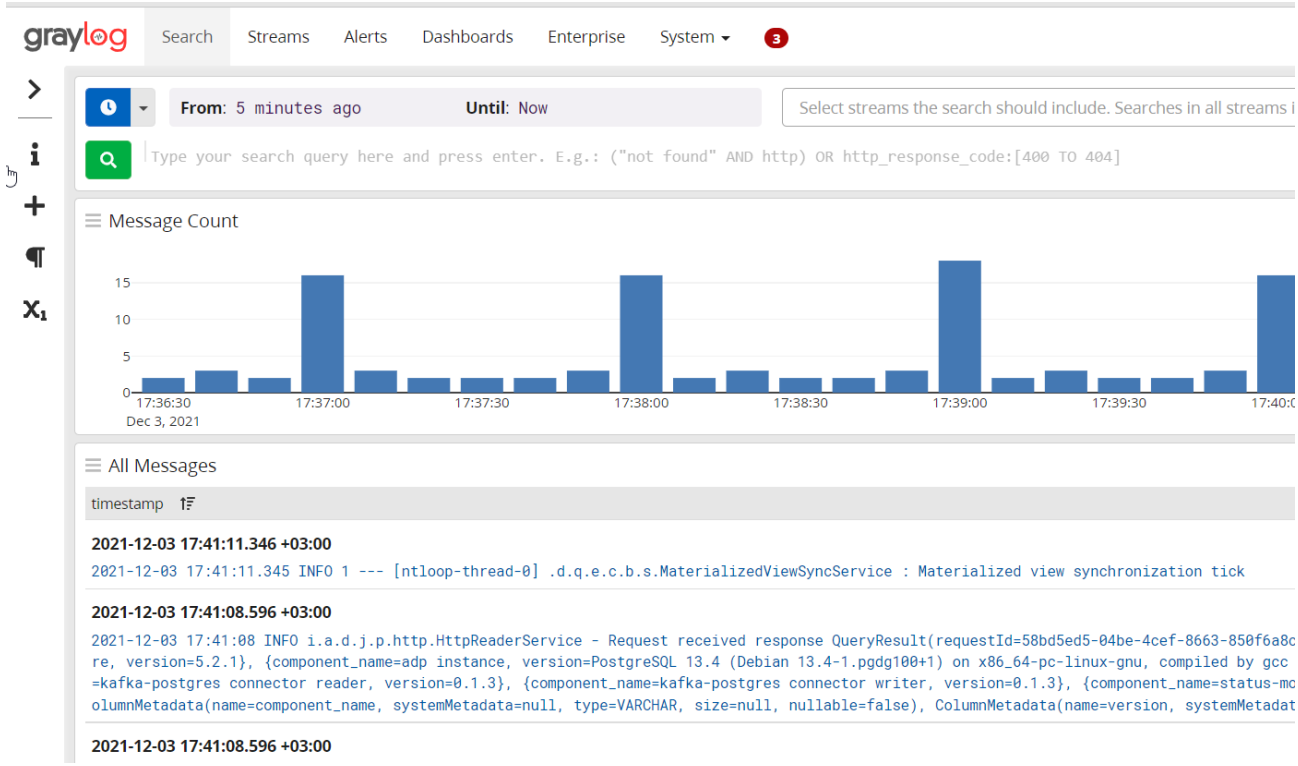

Рисунок - 5.1 Просмотр записей лог-файлов в Graylog

Каждая запись в таблице содержит следующую информацию:

1. Уровень логирования;
2. Дата и время события в формате `yyyy-mm-dd hh:mm:ss`;
3. Имя узла, на котором произошло событие.

5.2 Проверка версии компонентов

Версии используемых компонентов программы можно проверить с помощью запроса `CHECK_VERSIONS`.

6 СООБЩЕНИЯ АДМИНИСТРАТОРУ

6.1 Сообщения в ходе установки программы с помощью Ansible

Выполнение установки и настройки программы представляет собой процесс автоматизированного формирования конфигурационных файлов программы с помощью *Ansible*, в частности указания сетевых адресов и идентификаторов компонентов для взаимосвязи между ними, задания путей на дисковых пространствах для обработки полезных и служебных данных, а также метаданных.

Описание типичных ошибок при работе **Ansible** можно просмотреть на официальном сайте разработчика приложения.

Внесенные изменения в дистрибутив приложения и конфигурационные файлы влияют на результаты установки и работы программы. Компоненты программы в ходе выполнения настройки формируют сообщения и выводят их в стандартный порт вывода, перенаправленный в соответствующие лог-файлы. Просмотреть лог-файлы можно с помощью приложения Grafana.

6.2 Сообщения при эксплуатации

В ходе эксплуатации компоненты программы формируют сообщения и выводят их в стандартный порт вывода, перенаправленный в соответствующие лог-файлы.

Дополнительно, система может формировать сообщения приведенные в Таблица 6.1.

Таблица 6.1 Сообщения

Сообщение	Описание
DATAMART-17473	Запрос не прошел валидацию. Если в запросе тип данных параметра не поддерживается и/или формат значения недопустимый и/или значения недопустимый и/или набор параметров (или их значения, или их сочетание) некорректные.
DATAMART-17001	Внутренняя ошибка Витрины, возникает в процессе генерации файлов (в случае успешного считывания параметров).

Генерация сообщений администратору в ходе эксплуатации программы подчиняются следующей блок-схеме (см. Рисунок - 6.1).

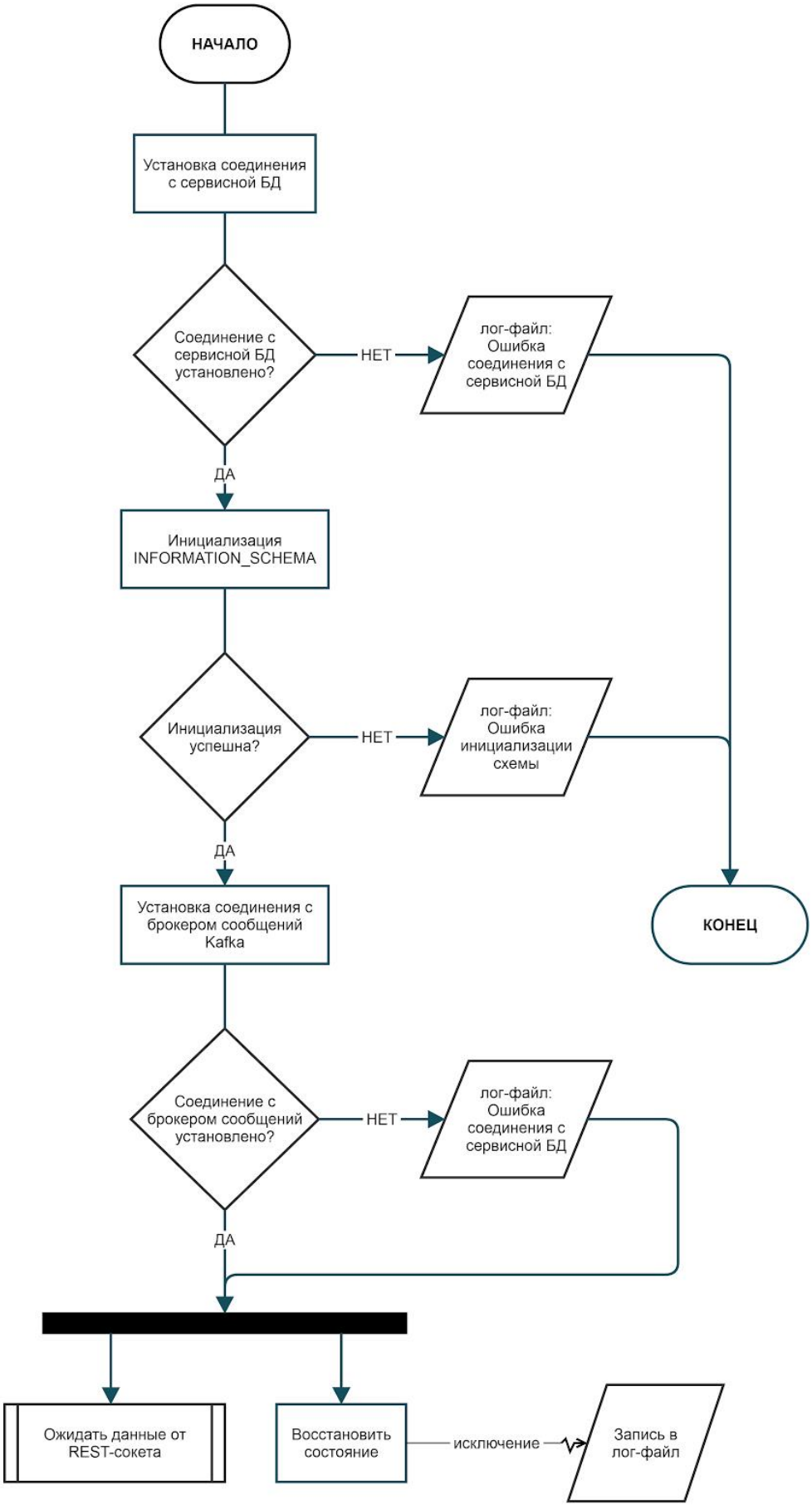

Рисунок - 6.1 Блок-схема журналирования сообщений в лог-файлы при запуске программы

7 REST API

7.1 Получение данных

URL	protocol://name[:port]/api/csv/[table]; protocol - HTTP; name - IP или FQDN HTTP-сервера, на котором установлен CSV-Uploader; port - номер TCP-порта; api/csv - постоянный путь; table – наименование витрины и таблицы, данной витрины, из которой необходимо получать данные; После указания таблицы может быть указан id – первичный ключ таблицы, по которому необходимо получить данные.
Метод	GET
Описание	Метод предназначен для получения данных из таблицы или какой-либо строки таблицы витрины.
Аутентификция	Не используется
Протокол	HTTP
Входные параметры	Идентификатор конкретного ресурса (числовой или строковый).
Выходные параметры	В случае успешного выполнения HTTP-код ответа равен 200 (OK); Возвращается стандартный HTTP ответ с хедерами content-type,content-disposition(должен содержать имя CSV-файла), и content-length.
HTTP-статус коды в ответе	200 - в случае успешного выполнения; 400 - в случае пользовательской ошибки (сервер не смог обработать запрос, отправленный клиентом из-за неверного синтаксиса/входных данных); 404 - в случае, если не найден ресурс из URL; 500 - внутренняя ошибка загрузчика.

Пример:

```
http://t5-csv-uploader-01.ru-central1.internal:8080/api/csv/demo_view.passenger/1
```

В представленном примере:

- `http` – протокол передачи данных;
- `t5-csv-uploader-01.ru-central1.internal` - имя сервера;
- `8080` – порт подключения;
- `api/csv/demo_view.passenger` - постоянная часть пути до таблицы `passengers`, в витрине `demo_view`;
- `1` - первичный ключ пассажира из таблицы `passengers`, информацию по которому предполагается получить (в случае если не указан, вернутся все данные из таблицы `passengers`).

7.2 Добавление/изменение данных (1 файл)

URL	http/name[:port]/api/csv/[table];

	protocol - HTTP; name - ip или FQDN HTTP-сервера, на котором установлен CSV-Uploader; port - номер TCP-порта; api/csv - постоянный путь; table – наименование витрины и таблицы, данной витрины, которую необходимо наполнить данными или внести изменения.
Метод	POST
Описание	Метод предназначен для заполнения или внесения изменений в данные существующей таблицы витрины.
Header	Content-Type: application/octet-stream
Аутентификация	Не используется
Протокол	HTTP
Входные параметры	mimeType: application/octet-stream; файл в формате CSV, содержащий данные таблицы.
Выходные параметры	В случае успешного выполнения HTTP-код ответа равен 200 (OK).
HTTP-статус Код ответа	200 - в случае успешного выполнения; 400 - в случае пользовательской ошибки (сервер не смог обработать запрос, отправленный клиентом из-за неверного синтаксиса/входных данных); 500 - внутренняя ошибка загрузчика.

Пример:

```
http://t5-csv-uploader-01.ru-central1.internal:8080/api/csv/demo_view.passenger
```

В представленном примере:

- `http` – протокол передачи данных;
- `t5-csv-uploader-01.ru-central1.internal` - имя сервера;
- `8080` – порт подключения;
- `api/csv/demo_view.passenger` - постоянная часть пути до таблицы `passengers`, в витрине `demo_view` в которую необходимо внести изменения.

Файл, содержащий данные, которые требуется добавить/изменить указывается в теле сообщения (тип: `file`) содержимое csv-файла - `demo_view_passenger.csv`.

7.3 Добавление/изменение данных (несколько файлов)

URL	protocol://name[:port]/multipart/csv; protocol - HTTP; name - ip или FQDN HTTP-сервера, на котором установлен CSV-Uploader; port - номер TCP-порта; multipart/csv – постоянный путь для использования нескольких файлов.
Метод	POST
Описание	Метод предназначен для заполнения или внесения изменений в данные нескольких, существующих таблиц витрины.
Header	Content-Type: multipart/form-data
Аутентификация	Не используется

Протокол	HTTP
Входные параметры	mimeType: multipart/form-data Наименование - файл в формате CSV, содержащий данные таблицы.
Выходные параметры	В случае успешного выполнения HTTP-код ответа равен 200 (OK).
HTTP-статус Код ответа	200 - в случае успешного выполнения; 400 - в случае пользовательской ошибки (сервер не смог обработать запрос, отправленный клиентом из-за неверного синтаксиса/входных данных); 500 - внутренняя ошибка загрузчика.

Пример:

```
http://t5-csv-uploader-01.ru-central1.internal:8080/api/multipart/csv
```

В представленном примере:

- `http` – протокол передачи данных;
- `t5-csv-uploader-01.ru-central1.internal` - имя сервера;
- `8080` – порт подключения.

Файлы, содержащие данные, которые требуется добавить/изменить указывается в теле сообщения (тип: `multipart`) содержимое:

```
file1 - demo_view_passenger(1).csv
file2 - demo_view_passenger(2).csv
```

7.4 Удаление данных (1 файл)

URL	protocol://name[:port]/api/csv/[table]; protocol - HTTP; name - ip или FQDN HTTP-сервера, на котором установлен CSV-Uploader; port - номер TCP-порта; api/csv - постоянный путь; table – наименование витрины и таблицы, этой витрины данные которой необходимо удалить.
Метод	DELETE
Описание	Метод предназначен для удаления данных из существующей таблицы витрины.
Header	Content-Type: application/octet-stream
Аутентификция	Не используется
Протокол	HTTP
Входные параметры	mimeType: application/octet-stream; Путь, по которому находится файл в формате CSV, содержащий данные подлежащие удалению.
Выходные параметры	В случае успешного выполнения HTTP-код ответа равен 200 (OK).
HTTP-статус Код ответа	200 - в случае успешного выполнения; 400 - в случае пользовательской ошибки (сервер не смог обработать запрос, отправленный клиентом из-за неверного синтаксиса/входных данных); 500 - внутренняя ошибка загрузчика.

Пример:

```
http://t5-csv-uploader-01.ru-central1.internal:8080/api/csv/demo_view.passenger/demo.v
```

В представленном примере:

- `http` – протокол передачи данных;
- `t5-csv-uploader-01.ru-central1.internal` - имя сервера;
- `8080` – порт подключения;
- `api/csv/demo_view.passenger` - постоянная часть пути до таблицы `passengers`, в витрине `demo_view`.

Файл, содержащий данные, которые требуется удалить, указывается в теле сообщения (тип: `file`) содержимое csv-файла - `demo_view_passenger.csv`.

7.5 Удаление данных (несколько файлов)

URL	protocol://name[:port]/multipart/csv; protocol - HTTP; name - ip или FQDN HTTP-сервера, на котором установлен CSV-Uploader; port - номер TCP-порта; multipart/csv – постоянный путь для использования нескольких файлов.
Метод	DELETE
Описание	Метод предназначен для удаления данных нескольких, существующих таблиц витрины.
Header	Content-Type: multipart/form-data
Аутентификция	Не используется
Протокол	HTTP
Входные параметры	mimeType: multipart/form-data; Наименование - файл в формате csv, содержащий данные таблицы.
Выходные параметры	В случае успешного выполнения HTTP-код ответа равен 200 (OK).
HTTP-статус Код ответа	200 - в случае успешного выполнения; 400 - в случае пользовательской ошибки (сервер не смог обработать запрос, отправленный клиентом из-за неверного синтаксиса/входных данных); 500 - внутренняя ошибка загрузчика.

Пример:

```
http://t5-csv-uploader-01.ru-central1.internal:8080/api/multipart/csv
```

В представленном примере:

- `http` – протокол передачи данных;
- `t5-csv-uploader-01.ru-central1.internal` - имя сервера;
- `8080` – порт подключения.

Файлы, содержащие данные, которые требуется добавить/изменить указывается в теле сообщения (тип: `multipart`) содержимое:

```
file1 - demo_view_passenger(1).csv
file2 - demo_view_passenger(2).csv
```

7.6 Использование *Curl* для загрузки данных

URL	protocol://name[:port]/ api/csv/; protocol - HTTP; name - ip или FQDN HTTP-сервера, на котором установлен CSV-Uploader; port - номер TCP-порта; api/csv/– постоянный путь для использования Curl для загрузки данных.
Метод	POST
Описание	Метод предназначен для загрузки данных нескольких, существующих таблиц витрины.
Header	Content-Type: text/plain
Аутентификация	Не используется
Протокол	HTTP
Входные параметры	Содержимое для загрузки data-raw
Выходные параметры	В случае успешного выполнения HTTP-код ответа равен 200 (OK).
HTTP-статус Код ответа	200 - в случае успешного выполнения; 400 - в случае пользовательской ошибки (сервер не смог обработать запрос, отправленный клиентом из-за неверного синтаксиса/входных данных); 500 - внутренняя ошибка загрузчика.

Пример:

```
curl --location --request POST
'http://localhost:8080/api/csv/test_para.passenger'

--header 'Content-Type: text/plain'

--data-raw 'code;id;firstname;middlename;lastname;birthday;passport

1;8255bcce-fa66-4915-b805-c06e003bc7fb;33;Васильевич;Кротов;03.12.1992;/test/test.zip

10;7888b9d9-1ae2-4b88-954b-
677cd59a76bf;Станислав;Васильевич;Гуськов;15.03.1980;/test/image/Picture_2.jpg

6;41414e39-dd3d-4b29-a1bc-
afb4095900c6;Григорий;Антонович;Давыдов;03.07.1991;/test/image/Picture_3.jpg

5;e990203d-be40-4923-8075-
c621897bc305;Иван5;Иванов5;Иванович5;13.09.1995;/test/image/Picture24.jpg

8;229c9856-41d5-4a2c-a38c-fa3dd9d0f1a5;Иван6;Иванов6;Иванович6;23.01.1990;

7;85f66365-86da-4032-852d-
b0d9ecf34ae9;Иван7;Иванов7;Иванович7;13.05.1998;/test/image/Picture_6.jpg

4;0fe26963-6cdf-4db6-b632-
03825a408d35;Иван8;Иванов8;Иванович8;23.08.1994;/test/passengers.csv

3;c1060f8e-14e5-46f7-8414-
fe1431a997e6;Иван9;Иванов9;Иванович9;03.02.1981;/test/trips.csv

9;d34bc498-f036-49eb-a465-
19887d926fdd;Иван10;Иванов10;Иванович10;07.02.1988;/test/image/Picture_9.jpg
```

```
2;2bd4c379-8fa8-40a1-9eb1-
1dde12bd6998;Иван11;Иванов11;Иванович11;03.10.1990;/test/image/Picture_10.jpg'
```

В представленном примере:

- `http` – протокол передачи данных;
- `localhost` - имя сервера;
- `8080` – порт подключения;
- `api/csv/test_para.passenger` - постоянная часть пути до таблицы `passengers`, в Витрине `test_para`.

8 ПРИЛОЖЕНИЕ 1

8.1 Настройка firewall (Iptables)

Утилита **iptables** - это межсетевой экран для операционных систем Linux. Настройка **iptables** производится в командной строке. С помощью правил **iptables** можно разрешать или блокировать прохождение трафика.

Для выполнения настройки межсетевого экрана необходимо создать конфигурационный файл `iptables.conf` в папке */etc/*:

```
/etc/iptables.conf
```

Далее, необходимо скопировать в файл следующие настройки:

```
*filter
:INPUT ACCEPT [0:0]
:FORWARD DROP [0:0]
:OUTPUT ACCEPT [0:0]
:FILTERS - [0:0]
:DOCKER-USER - [0:0]

-F INPUT
-F DOCKER-USER
-F FILTERS

-A INPUT -i lo -j ACCEPT
-A INPUT -p icmp --icmp-type any -j ACCEPT
-A INPUT -j FILTERS

-A DOCKER-USER -o docker0 -j FILTERS

-A FILTERS -m state --state ESTABLISHED,RELATED -j ACCEPT
-A FILTERS -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A FILTERS -m state --state NEW -m tcp -p tcp --dport 3000 -j ACCEPT
-A FILTERS -m state --state NEW -m tcp -p tcp --dport 8080 -j ACCEPT
-A FILTERS -m state --state NEW -m tcp -p tcp --dport 9000 -j ACCEPT
-A FILTERS -j REJECT --reject-with icmp-host-prohibited

COMMIT
```

1. Выполнить команду

```
iptables-restore -n /etc/iptables.conf
```

2. Создать файл

```
/etc/systemd/system/iptables.service
```

3. Сохранить в файл

```
[Unit]
Description=Restore iptables firewall rules
Before=network-pre.target

[Service]
Type=oneshot
ExecStart=/sbin/iptables-restore -n /etc/iptables.conf

[Install]
WantedBy=multi-user.target
```

4. Включить **iptables**, выполнив команду:

```
sudo systemctl enable --now iptables
```

или выполнить следующие две команды:

```
sudo systemctl enable iptables
sudo systemctl start iptables
```

После обновления правил в файле */etc/iptables.conf*, выполнить следующую команду:

```
sudo systemctl restart iptables
```

8.1.1 Инструкция по эксплуатации CSV-uploader

8.1.1.1 Загрузка структуры Витрины

Внимание:
XML-файл со структурой Витрины может быть загружен только один раз после установки «Витрина данных Лайт».

Для передачи xml-файла со структурой Витрины, выполните следующие действия:

1. Откройте программный интерфейс CSV-uploader.
2. Выберите вкладку **Загрузка структуры**.
3. В открывшемся окне *Загрузка структуры Витрины* нажмите кнопку **Выберите файл**, выберите XML-файла для загрузки и нажмите кнопку **Загрузить**. (см. Рисунок - 8.1)

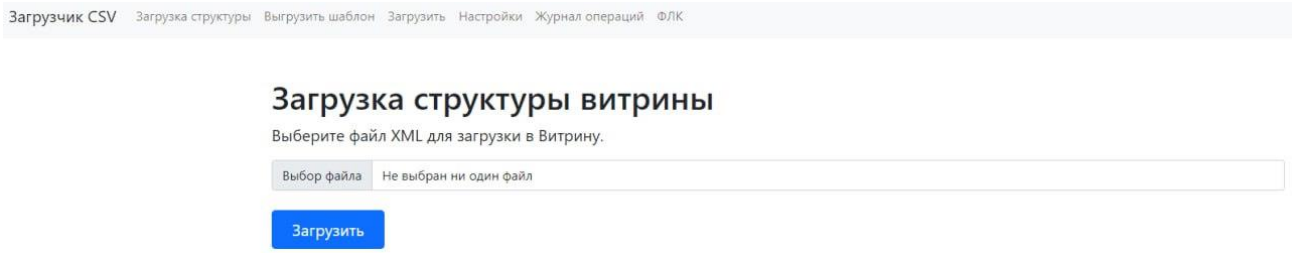

Рисунок - 8.1 Загрузка структуры Витрины

В случае успешного применения настроек отобразится информационное сообщение: *Список таблиц загружен*.

8.1.1.2 Выгрузка шаблона CSV

Для выгрузки существующего CSV-файла со структурой Витрины, выполните следующие действия:

1. Откройте программный интерфейс CSV-uploader.

2. Выберите вкладку **Выгрузка шаблона CSV**.
3. Выберите таблицу для выгрузки, например, **demo_view_podd.all_types_table**, для выгрузки примера CSV-таблиц для ПОДД (см. Рисунок - 8.2).

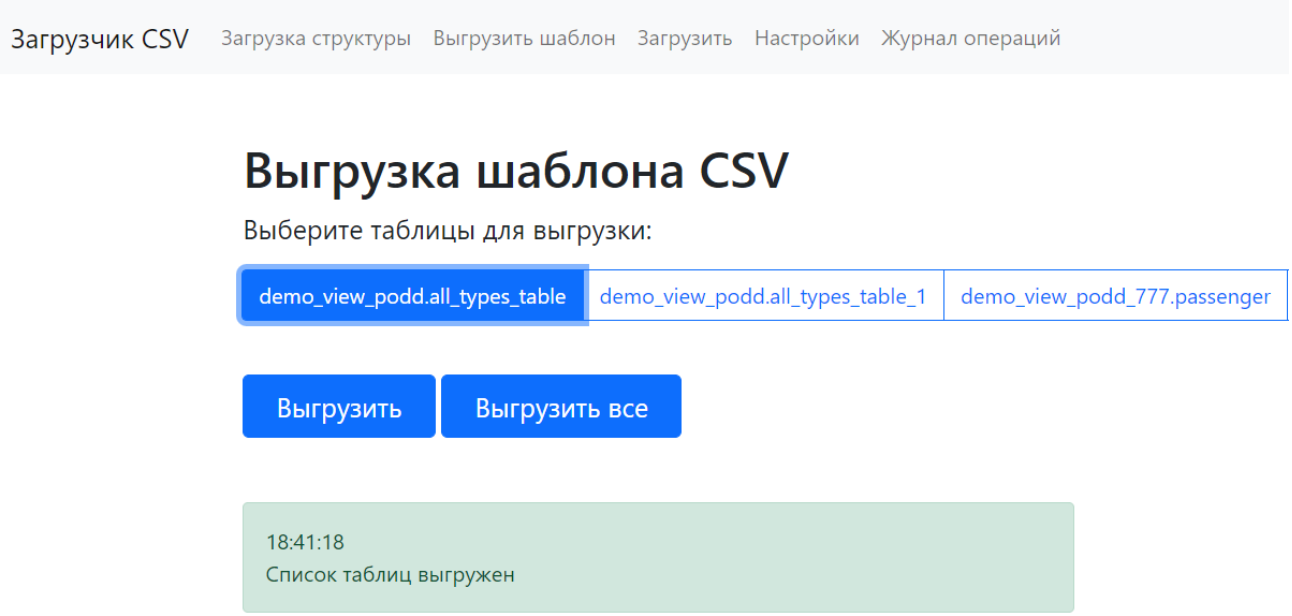

Рисунок - 8.2 Выгрузка шаблона CSV

4. Нажмите кнопку **Выгрузить**. Файл будет загружен на локальный компьютер. Если требуется выгрузить все таблицы, нажмите кнопку **Выгрузить все**.

В случае успешной выгрузки на экране монитора отобразится информационное сообщение: *Список таблиц выгружен*.

8.1.1.3 Загрузка CSV-файла

Для загрузки CSV-файла, выполните следующие действия:

1. Откройте программный интерфейс CSV-uploader.
2. Выберите вкладку **Загрузчик CSV**.
3. В открывшемся окне *Загрузка файла* выберите **Режим** загрузки:
   - **Вставка** - параметр определяет, что данные будут добавлены.
   - **Удаление** - параметр определяет, что данные будут удалены.

В случае, если в настройках модуля CSV-Uploader включен ФЛК и прописан адрес модуля REST-Uploader, на странице отображается переключатель с текстом «Выполнять проверку форматно-логического контроля».

1. Для автоматического определения типа таблиц включите переключатель **Автоматическое определение таблицы**, если автоматическое определение таблиц не требуется, выключите переключатель и выберите таблицу, в которую требуется внести

изменения, например, **demo_view_podd.all_types_table** (см. Рисунок - 8.3).

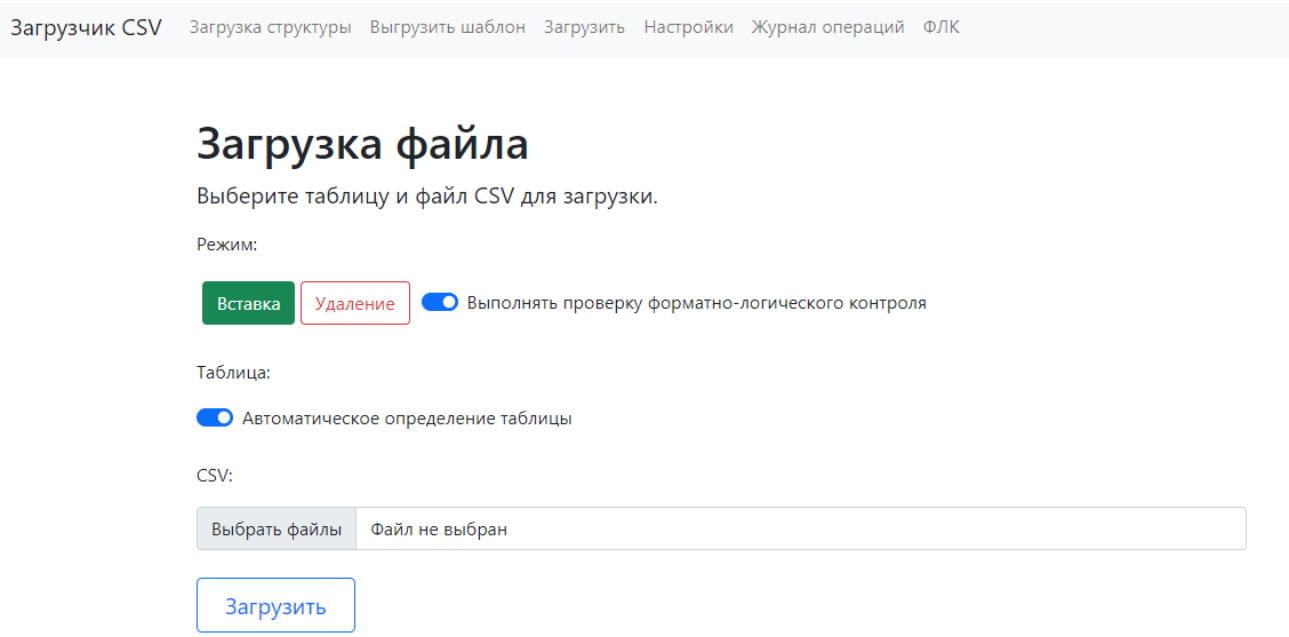

Рисунок - 8.3 Загрузка CSV-файла

5. Нажмите кнопку **Выберите файл** чтобы выбрать файл для загрузки.
6. Нажмите кнопку **Загрузить**.
7. Убедитесь, что файл с таблицами был загружен.

При включенной настройке определения таблиц после выбора и загрузки файла:

- модулем CSV-Uploader определяется таблица загрузки:
  - в случае, если активен переключатель «Автоматическое определение таблицы» по метаданным csv файла;
  - в случае если выбрана конкретная таблица, в соответствии с выбором пользователя;
- модуль CSV-Uploader обогащает URL запроса на загрузку именем датамарта и таблицы;
- модуль CSV-Uploader выполняет запрос /v2/datamarts/{datamart_name}/tables/{table_name}/upload к модулю REST-Uploader;
- модуль CSV-Uploader отображает текст ответа на странице загрузки в формате:
  - время ответа;
  - код ответа;
  - body ответа:
- в случае успешного ответа, модуль CSV-Uploader сохраняет requestId файла в топик flk_logs в внутренней Kafka.

8.1.1.4 Загрузка CSV-файла с предварительным форматно-логическим контролем

В случае, если в настройках модуля CSV-Uploader включена настройка `VALIDATION_ENABLE: true` и прописан адрес модуля REST-Uploader (`REST_UPLOADER_URL`) при активном режиме «Вставка» на странице отображается переключатель с текстом «Выполнять проверку форматно-логического контроля» , по умолчанию значение `вкл` (`true`) см. Рисунок - 8.4.

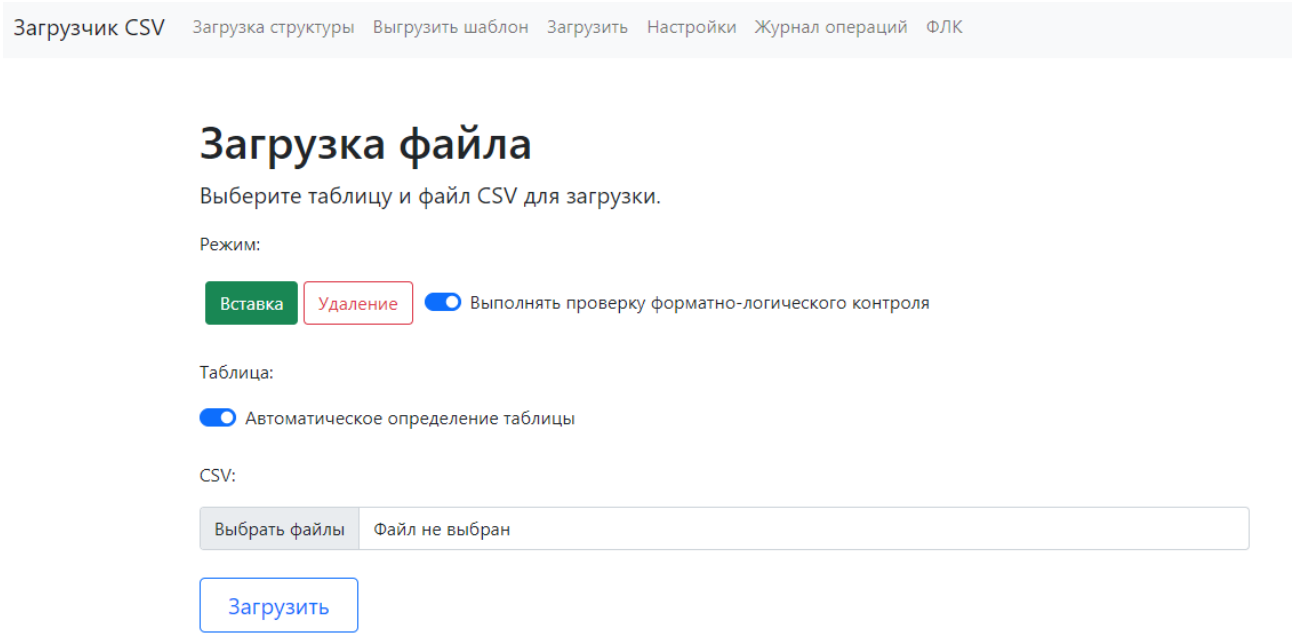

Рисунок - 8.4 Переключатель выполнения ФЛК

1. При включенном переключателе «Выполнять проверку форматно-логического контроля» после выбора файла и нажатия кнопки **Загрузить**:
   - модулем CSV-Uploader определяется таблица загрузки:
     - в случае, если включен переключатель «автоопределение таблицы» по метаданным CSV файла;
     - в случае если выбрана конкретная таблица, в соответствии с выбором пользователя;
   - модуль CSV-Uploader обогащает URL запроса на загрузку именем датамарта и таблицы;
   - модуль CSV-Uploader выполняет запрос `/v2/datamarts/{datamart_name}/tables/{table_name}/upload` к модулю REST-Uploader;
   - модуль CSV-Uploader отображает текст синхронного ответа на странице загрузки в формате:
     - время ответа;
     - код ответа;

- body ответа.

2. При выключенном переключателе «Выполнять проверку форматно-логического контроля» загрузка выполняется стандартным способом через модуль CSV-Uploader.

8.1.1.5 Обязательная загрузка данных с предварительным форматно-логическим контролем

При включении настройки `VALIDATION_MANDATOR: true`, переключатель «Выполнять проверку форматно-логического контроля» неактивен и находится во включенном положении. В данном режиме загрузка данных в ручном режиме с использованием CSV-Uploader невозможна, для всех загружаемых данных будут проводиться проверки форматно-логического контроля в модуле REST-Uploader см. Рисунок - 8.5.

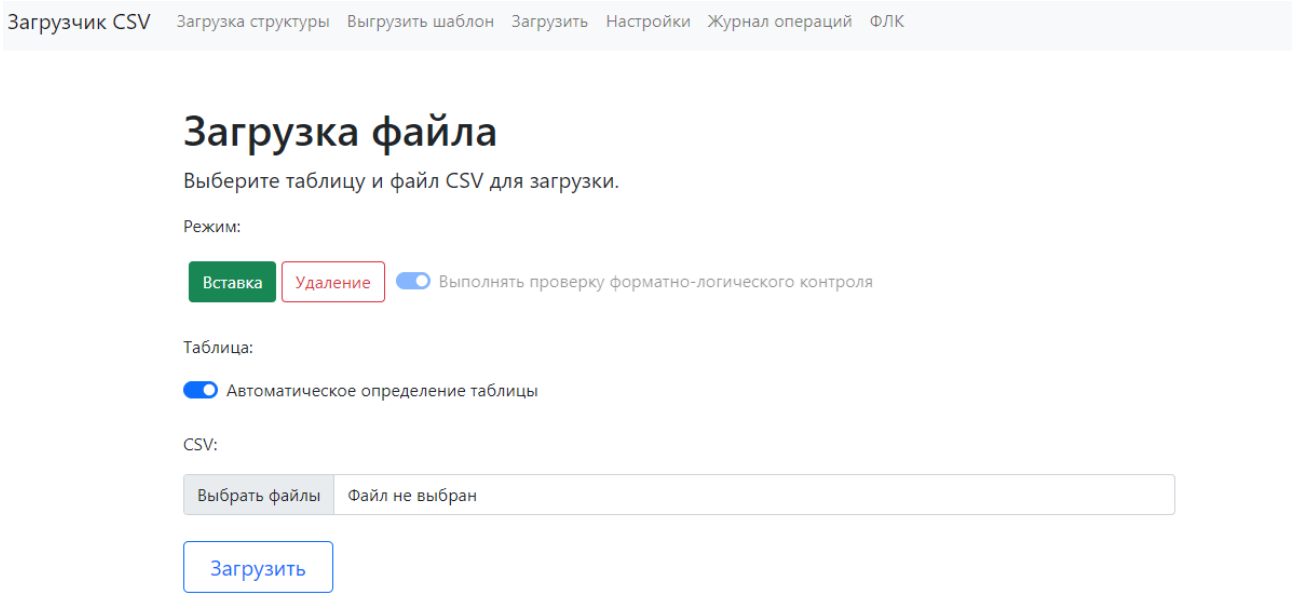

Рисунок - 8.5 Обязательная загрузка данных с предварительным форматно-логическим контролем

8.1.1.6 Аутентификация с использованием jwt-токена при включенной аутентификации в модуле REST-Uploader

Для использования jwt-токена при загрузке данных с предварительным ФЛК в случае, если в REST-Uploader включена аутентификация, необходимо включить следующие настройки в модуле CSV-Uploader:

- `VALIDATION_ENABLE:true`;
- `JWT_AUTH:true`.

В случае, если обе настройки имеют значение true, при открытии загрузчика CSV-uploader отображается модальное окно ввода токена пользователя, а на странице загрузки данных в витрину отображается поле «Изменить JWT» (см. Рисунок - 8.6).

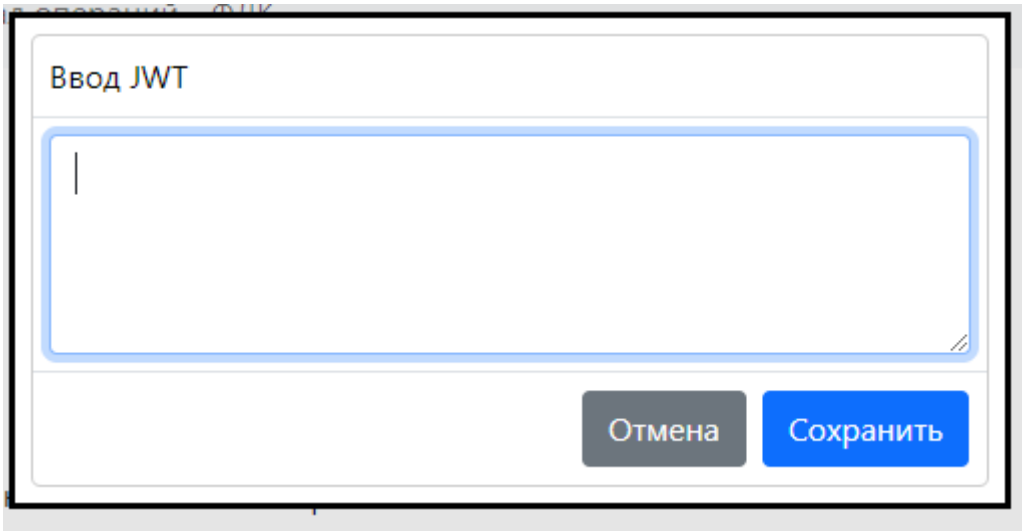

Рисунок - 8.6 Модальное окно ввода токена

Значение внесенного JWT-токена используется как барьерный токен при обращении к REST-Uploader.

Внесенное значение токена сохраняется в сессии пользователя и автоматически подставляется при включении переключателя выполнения ФЛК проверок. Для того, чтобы изменить JWT-токен для аутентификации, необходимо нажать кнопку **Изменить JWT** (см. Рисунок - 8.7).

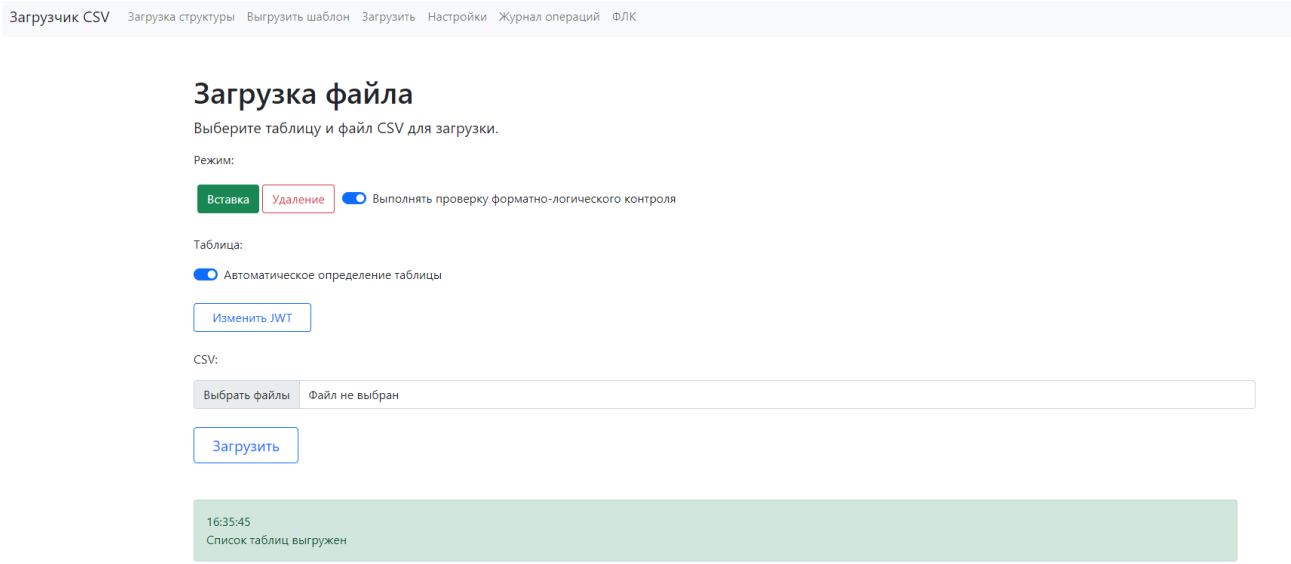

Рисунок - 8.7 Отображение кнопки **Изменить JWT**

8.1.1.7 Настройки CSV-uploader

Для CSV-uploader можно настроить следующие параметры:

- автоматический запуск загрузки CSV-файлов по расписанию;
- количество отображаемых записей для *Журнала операций*.

8.1.1.8 Автоматический запуск загрузки CSV-файлов по расписанию

Для настройки автоматического запуска загрузки CSV-файлов по расписанию, выполните следующие действия:

1. Откройте программный интерфейс CSV-uploader.
2. Выберите вкладку **Настройки**.
3. В открывшемся окне **Настройки** в поле **Запуск по расписанию**, укажите время в `Cron` формате (например, `0 15 10? * *` - загрузка файлов будет происходить каждый день в 10.15) и путь к каталогу с CSV-файлами (см. Рисунок - 8.8) .

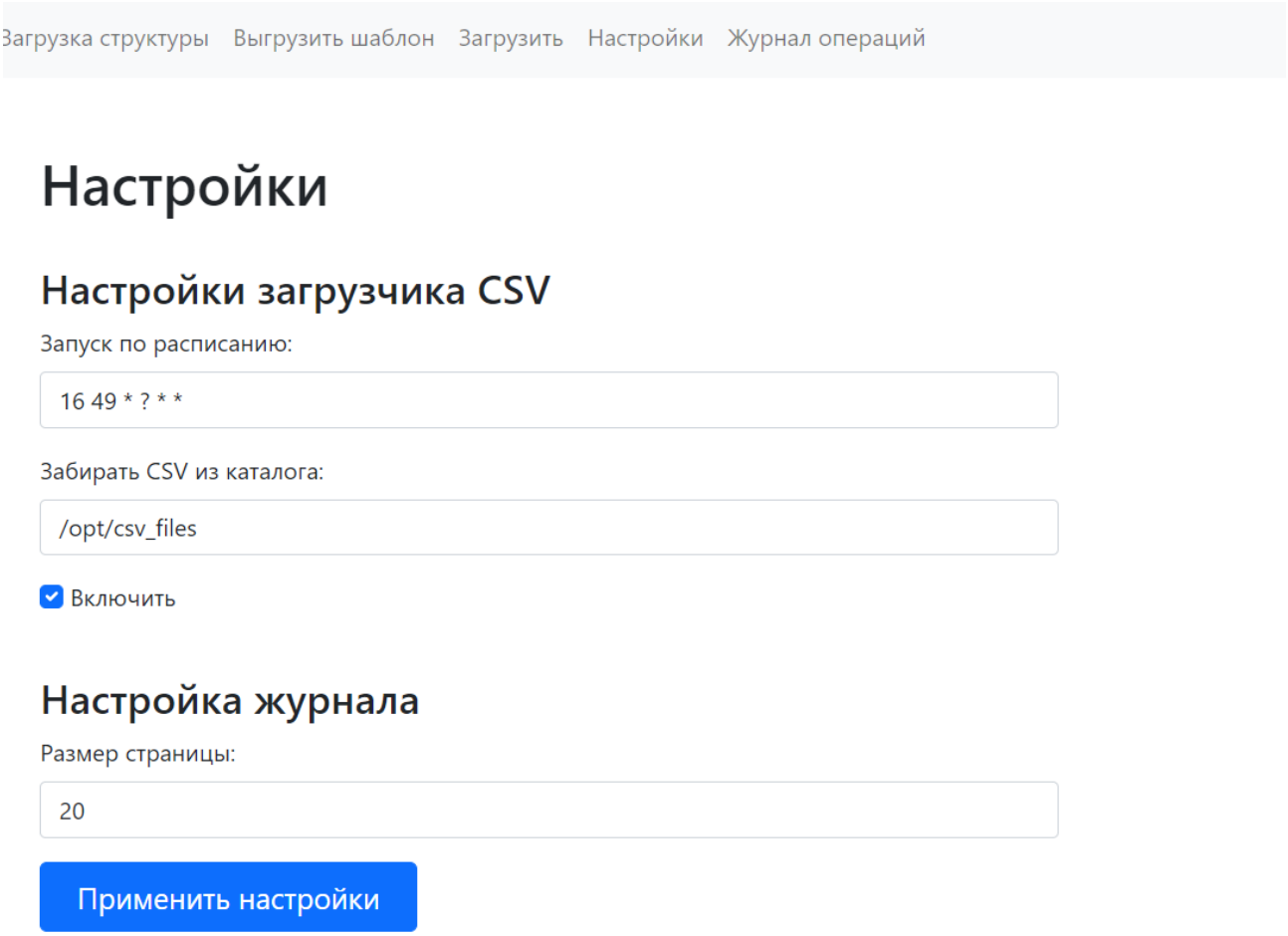

Рисунок - 8.8 Автоматический запуск загрузки CSV-файлов по расписанию

4. Установите маркер в поле **Включить**, для активации автоматического запуска загрузки.
5. Нажмите кнопку **Применить настройки**.

В случае успешного применения настроек отобразится информационное сообщение: *Конфигурация успешно получена*.

8.1.1.9 Настройка *Журнала операций*

Для настройки *Журнала операций*, выполните следующие действия:

1. Откройте программный интерфейс CSV-uploader.
2. Выберите вкладку **Настройки**.
3. В открывшемся окне **Настройки** в поле **Размер страницы**, укажите количество записей на страницу, например, `20` (см. Рисунок - 8.9).

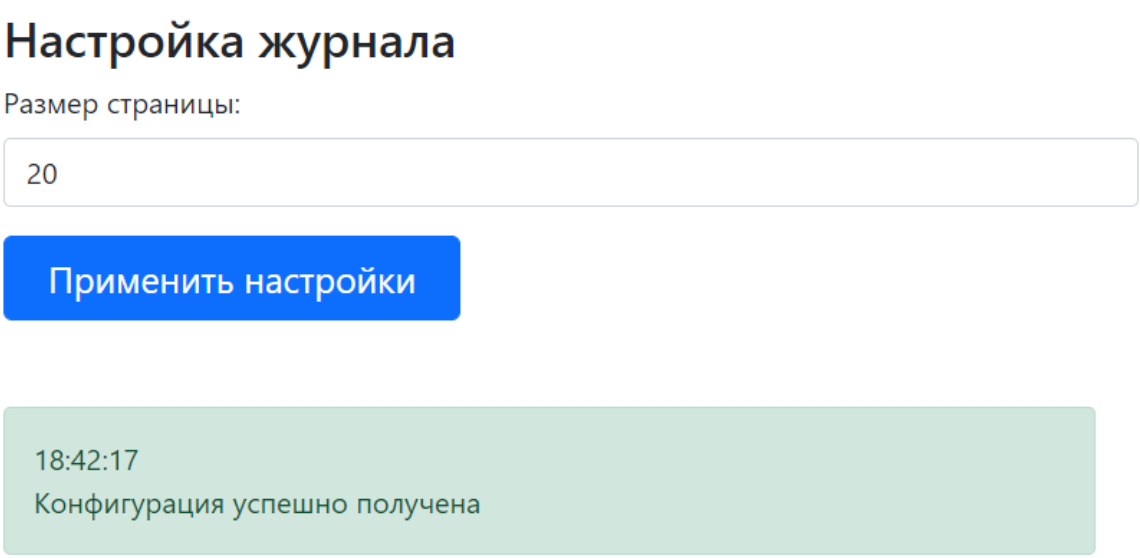

Рисунок - 8.9 Настройка *Журнала операций*

4. Нажмите кнопку **Применить настройки**.

В случае успешного применения настроек отобразится информационное сообщение: *Конфигурация успешно получена*.

8.1.1.10 Просмотр *Журнала операций*

В *Журнале операций* можно просмотреть действия выполненные в CSV-uploader:

- Время - время, когда операция была выполнена.
- Уровень - статус операции.
- `ERROR` - ошибка загрузки;
- `INFO` - описание операции.
- Сообщение - краткое информационное сообщение об операции.

Для просмотра *Журнала операций*, выполните следующие действия:

1. Откройте программный интерфейс CSV-uploader.
2. Выберите вкладку **Журнал операций**.
3. В открывшемся окне просмотрите операции, которые были выполнены в CSV-uploader (см. Рисунок - 8.10).

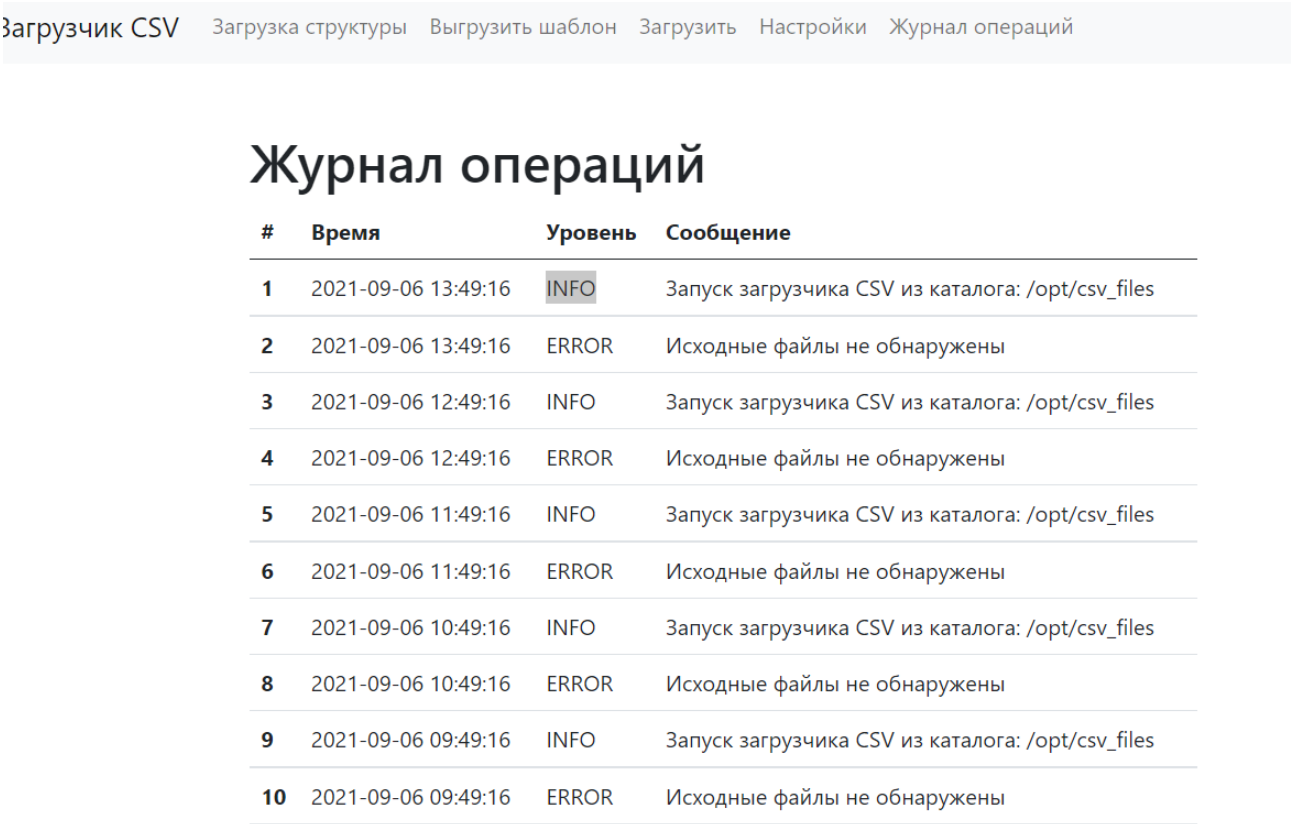

Рисунок - 8.10 Просмотр *Журнала операций*

4. Нажмите кнопку **Применить настройки**.

8.1.1.11 Интерфейс Форматно-логического контроля

На вкладке **Форматно-логический контроль** (см. Рисунок - 8.11) отображается:

- список последних отправленных файлов, определяемых настройками модуля CSV-Uploader - значение по умолчанию 20:
  - список requestId;
  - время записи в кафку;
  - статус загрузки файла;
- управляющий элемент для запроса отчета об ошибках для файла (кнопка отображается активной только в случае финальных статусов):
  - статус 3;
  - статус 4;
  - статус 7;
- элементы пагинации списка requestId.

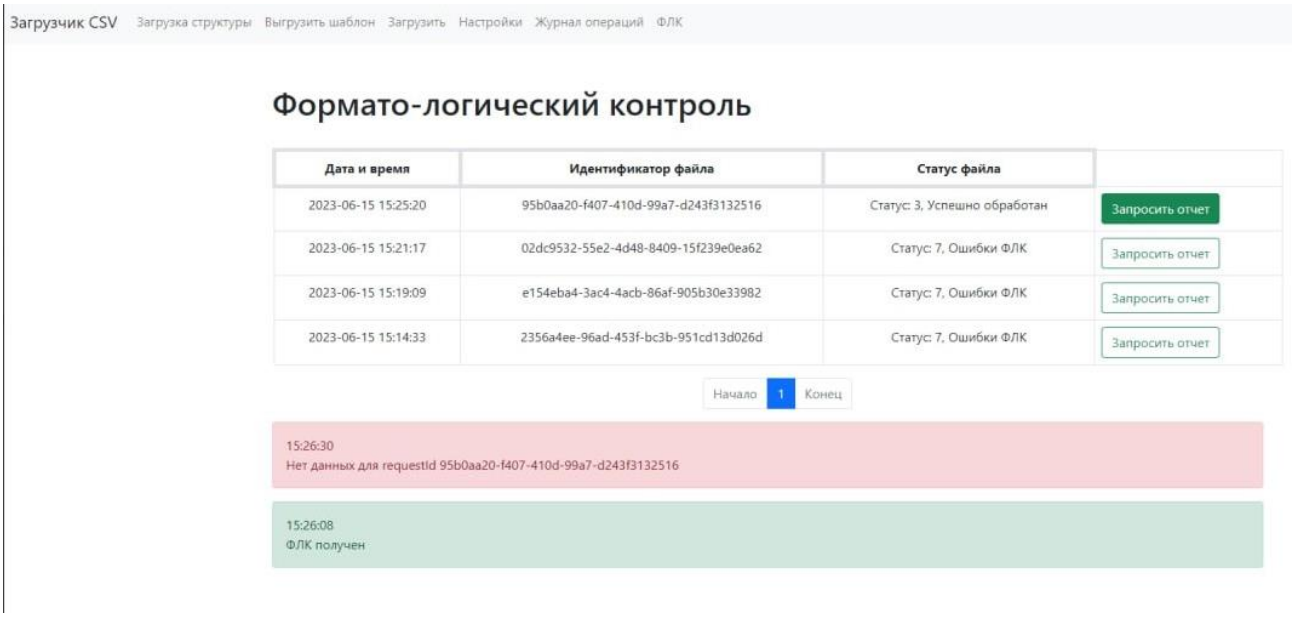

Рисунок - 8.11 Форматно-логический контроль

При нажатии на кнопку запроса отчета об ошибках для файла, модуль CSV-Uploader:

- вызывает метод `/v2/requests/{request_id}/report/`;
- получает CSV файл с именем `report_requestId.csv`;
- в зависимости от ответа:
  - если **response 200 ok**: скачивает файл на ПК пользователя автоматически или при нажатии на название отчета с выводом сообщения о загрузке файла;
  - если **response 400** - выводит сообщение Нет данных.

7 ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

ADCM

Arenadata Cluster Manager (ADCM) — Универсальный оркестратор гибридного ландшафта. Он позволяет быстро устанавливать, настраивать все data-сервисы компании и управлять ими. Наиболее ярко преимущества ADCM раскрываются при работе с гетерогенной инфраструктурой, при которой появляется возможность размещать data-сервисы на различных типах инфраструктур: в облаке, on-premise или в качестве PaaS-сервисов.

ADS

Arenadata Streaming (ADS) - Масштабируемая отказоустойчивая система для потоковой обработки данных в режиме реального времени на базе Apache Kafka и Apache Nifi.

Airflow

открытое программное обеспечение для создания, выполнения, мониторинга и оркестровки потоков операций по обработке данных.

Apache

Организация-фонд, способствующая развитию проектов программного обеспечения Apache.

Apache Airflow

Платформа для программного создания, планирования и мониторинга рабочих процессов.

Apache Hadoop

Свободно распространяемый набор утилит, библиотек и фреймворк для разработки и выполнения распределённых программ, работающих на кластерах из сотен и тысяч узлов.

Apache Spark

Фреймворк с открытым исходным кодом для реализации распределённой обработки неструктурированных и слабоструктурированных данных.

Apache Kafka

Распределённый программный брокер сообщений, проект с открытым исходным кодом, разрабатываемый в рамках фонда Apache.

Apache Avro

Система сериализации данных, разработанная в рамках проекта Hadoop.

API

Application programming interface (англ.) – описание сервисов взаимодействия компьютерной программы с другими программами.

Avro

(Object Container File) Линейно-ориентированный формат хранения файлов Big Data

BLOB-адаптер

Информационно-технологический компонент Витрины, обеспечивающий чтение бинарных файлов из Хранилище BLOB-объектов ведомства.

ClickHouse

Колоночная аналитическая СУБД с открытым кодом, позволяющая выполнять аналитические запросы в режиме реального времени на структурированных больших данных, разрабатываемая компанией Яндекс.

Docker

Программное обеспечение для автоматизации развёртывания и управления приложениями в средах с поддержкой контейнеризации, контейнеризатор приложений.

Docker Compose

Платформа контейнеризации, предназначена для конфигурирования многоконтейнерных приложений. В Docker Compose можно управлять несколькими контейнерами Docker.

DATA-uploader

Модуль исполнения асинхронных заданий.

Counter-Provider

Сервис генерации уникального номера.

CSV

Comma-Separated Values (англ.) – значения, разделённые запятыми) — текстовый формат, предназначенный для представления табличных данных.

CSV-extractor

Специализированное программное обеспечение, которое извлекает данные из csv-файлов в собственную БД-хранилища сервиса (Tarantool)

CSV-Uploader

Программный модуль Витрины данных, который предназначен для загрузки csv-файлов в Витрину данных.

Grafana

Веб-приложение для аналитики и интерактивной визуализации показателей мониторинга с открытым исходным кодом.

Greenplum

Массово-параллельная СУБД для хранилищ данных на основе PostgreSQL.

DAG

Файл, содержащий блок данных.

DBeaver

Клиентское приложение для управления базами данных (БД), которое использует программный интерфейс JDBC для взаимодействия с реляционными БД через драйвер JDBC-драйвер.

DDL

Data definition language (англ.) – семейство компьютерных языков, используемых в компьютерных программах для описания структуры баз данных.

DNS

Domain Name System «система доменных имён» — компьютерная распределённая система для получения информации о доменах. Чаще всего используется для получения IP-адреса по имени хоста (компьютера или устройства), получения информации о маршрутизации почты и/или обслуживающих узлах для протоколов в домене.

Endpoint

Шлюз (в переводе с англ. — конечная точка), который соединяет серверные процессы приложения с внешним интерфейсом. Простыми словами, это адрес, на который отправляются сообщения (работает с API)

ETL

Extract, transform, load (англ.) – решение, используемое при выгрузке данных из различных источников ведомств и дальнейшего хранения их в Витрине ProStore для чтения, использования и взаимодействия с другими ведомствами.

FileZilla

FTP-клиент

Greennplum

Система управления данными из мира big data.

HikariCP

Hikari Connection Pool.

HTTP

HyperText Transfer Protocol (англ.) – протокол прикладного уровня передачи данных, в настоящий момент используется для передачи произвольных данных.

IAM

Сервисы управления идентификацией и контролем доступа (Identity&AccessManagement)

JDBC

Java DataBase connectivity (англ.) – платформенно-независимый промышленный стандарт взаимодействия Java-приложений с различными СУБД.

JDBC-драйвер

Специализированное программное обеспечение, которое размещается на стороне системы, использующей ADTM (клиента ADTM). Драйвер предоставляет JDBC-интерфейс подключения из этой системы к ADTM и взаимодействует с сервисом исполнения запросов по REST API, предоставляемым сервисом исполнения запросов.

JDBC-extractor

Специализированное программное обеспечение, которое извлекает данные из jdbc-источника (ведомства) в собственную БД-хранилища сервиса (Tarantool).

JDBC-CSV-transformer

Специализированное программное обеспечение, которое предназначено для подключения к БД по JDBC, с последующим сохранением в csv-файлы.

Kafka

Распределённый программный брокер сообщений, проект с открытым исходным кодом, разрабатываемый в рамках фонда Apache.

Kafka-loader

Специализированное программное обеспечение, которое загружает данные, извлеченные и приведенные в соответствие логической структуре данных Витрины, собственно в Витрину.

Loki

Приложение для агрегирования log-файлов, используется совместно с Prometheus.

MD5

128-битный алгоритм хеширования. Предназначен для создания «отпечатков» или дайджестов сообщения произвольной длины и последующей проверки их подлинности.

MPP

Массово-параллельная архитектура (англ. massive parallel processing, MPP, также «массивно-параллельная архитектура»).

NTP

Network Time Protocol — сетевой протокол для синхронизации внутренних часов компьютера с использованием сетей с переменной латентностью.

OpenAPI

The OpenAPI Specification (англ.) – формализованная спецификация и экосистема множества инструментов, предоставляющая интерфейс между front-end системами, кодом библиотек низкого уровня и коммерческими решениями в виде API.

ProStore

Интеграционная система, обеспечивающая единый интерфейс к хранилищу разнородных данных. Определяет структуры данных, запись и чтение данных Витрины. Позволяет работать со входящими в состав хранилища СУБД одинаковым образом, используя единый синтаксис запросов SQL и единую логическую схему данных.

Prostore

Ядро интеграционной системы ProStore, состоящее из сервиса исполнения запросов и сервиса мониторинга.

Prometheus

Программное приложение, используемое для мониторинга событий и оповещения, которое записывает метрики в реальном времени в базу данных временных рядов, построенную с использованием модели HTTP-запроса, с гибкими запросами и оповещениями в режиме реального времени.

Proxy API

Проксирование запросов через Datamart Studio к инсталляциям приложений Витрин данных

PSQL

Терминальный клиент для работы с PostgreSQL

PuTTY

Свободно распространяемый клиент для различных протоколов удалённого доступа, включая SSH, Telnet, rlogin.

PXF

Фреймворк, позволяющий ADB (Greenplum) параллельно обмениваться данными со сторонними системами.

REST

Representational state transfer (англ.) – архитектурный стиль взаимодействия компонентов распределенного приложения в сети.

REST-адаптер

Сервис, реализующий публикацию конечных точек API для обработки запросов с использованием спецификации OpenAPI версии 3. Используется для сохранения обратной совместимости получения данных из ведомства по REST.

REST API

Набор правил, по которым различные программы могут взаимодействовать между собой и обмениваться данными с помощью протокола HTTP

REST-Uploader

Модуль асинхронной загрузки данных из сторонних источников.

SOAP

(от англ. Simple Object Access Protocol — простой протокол доступа к объектам) — протокол обмена структурированными сообщениями в распределённой вычислительной среде.

SQL

Structured query language (англ.) – язык структурированных запросов. Декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных.

SQL-запрос

Запрос к Базе данных.

SSH

Secure Shell (англ.) – «безопасная оболочка». Сетевой протокол прикладного уровня, позволяющий производить удалённое управление операционной системой и туннелирование TCP-соединений.

Tarantool

Платформа in-memory вычислений с гибкой схемой данных для создания высоконагруженных приложений. Включает в себя базу данных и сервер приложений на Lua.

UDP

Протокол передачи данных. С UDP компьютерные приложения могут посылать сообщения другим хостам по IP-сети без необходимости предварительного сообщения для установки специальных каналов передачи или путей данных.

URI

Унифицированный идентификатор ресурса. URI — последовательность символов, идентифицирующая абстрактный или физический ресурс.

UUID

Стандарт идентификации, используемый в создании программного обеспечения, стандартизированный Open Software Foundation как часть DCE — среды распределённых вычислений. Основное назначение UUID — это позволить распределённым системам уникально идентифицировать информацию без центра координации.

Vert.x

Библиотека для разработки асинхронных приложений, основанная на событиях.

VipNet

программное обеспечение (далее - ПО) для защиты сетевого трафика на рабочих местах пользователей.

XML

eXtensibe Markup Language (англ.) – универсальный текстовый формат для хранения и передачи структурированных данных.

XML-extractor

Специализированное программное обеспечение, для копирования данных из xml-файлов в собственную БД-хранилища сервиса (Tarantool).

ZooKeeper

Сервер с открытым исходным кодом для высоконадежной распределенной координации облачных приложений.

Агент ПОДД

Типовое программное обеспечение, устанавливаемое на стороне УВ и обеспечивающее сопряжение Витрин, хранилищ реплик, ИС Участника взаимодействия с ПОДД. В частности, чтение данных из Витрины, запись данных в реплику, обработка промежуточных/временных массивов данных, порождаемых в процессе выполнения распределённых запросов.

База данных

Совокупность данных, хранимых в соответствии со схемой данных, манипулирование которыми выполняют в соответствии с правилами средств моделирования данных.

Брокер сообщений

Архитектурный паттерн в распределённых системах; приложение, которое преобразует сообщение по одному протоколу от приложения-источника в сообщение протокола приложения-приёмника, тем самым выступая между ними посредником.

Витрина данных

Комплекс программных и технических средств в составе информационно-телекоммуникационной инфраструктуры участника НСУД, предназначенный для формирования и (или) получения данных с использованием среды взаимодействия НСУД.

ВС

Вид сведений

ГОСТ

Нормативно-правовой документ, в соответствии требованиями которого производится стандартизация производственных процессов

Дельта

Логически целостная совокупность изменений информации об объектах. Каждой дельте поставлено в соответствие целое число из монотонно возрастающей последовательности целых чисел начиная с 0, отражающее ее место в общей последовательности дельт и дата-время ее исполнения.

ЕИП

Единая информационная платформа

ИС

Информационная система.

КриптоПро

Разработанная одноименной компанией линейка криптографических утилит (вспомогательных программ) — так называемых криптопровайдеров. Они используются в других программах для генерации электронной подписи (ЭП), работы с сертификатами, организации структуры PKI и т.д.

Логическая модель данных

Схема базы данных, выраженная в понятиях бизнес-требований.

набор данных

Совокупность данных (датасетов), систематизированных в определённом формате, представляющих собой базовый элемент для работы с данными во многих отраслях

НСУД

Национальная система управления данными.

ОГРН

Основной государственный регистрационный номер, который налоговая служба присваивает юридическим лицам сразу же после регистрации

ПО

Программное обеспечение.

ПОДД

Подсистемы обеспечения доступа к данным

ПОДД-адаптер

Программно-технический продукт, обеспечивающий взаимодействие витрины и ПОДД СМЭВ.

ПОДД-адаптер - Модуль исполнения запросов

Логический модуль ПОДД-адаптера, предназначен для исполнения запросов ПОДД СМЭВ (через протокол коммуникации Агент ПОДД).

ПОДД-адаптер - Модуль MPPR

Логический модуль ПОДД-адаптера, предназначен для чтения данных в многопоточном режиме (massively parallel processing, MPP).

ПОДД-адаптер - Модуль MPPW

Логический модуль ПОДД-адаптера выполняет загрузку данных в многопоточном режиме.

Поставщик данных

Организация, осуществляющая передачу государственных данных в НСУД.

Потребитель данных

Организация, осуществляющая использование государственных данных, содержащихся в НСУД.

Реплика

СУБД, хранящая реплицируемые наборы данных, полученные от Поставщика данных.

Сервис Формирования документов

Модуль витрины, предназначенный для работы с формируемыми документами.

СМЭВ3

Система межведомственного электронного взаимодействия.

СМЭВ3-адаптер

Информационно-технологический компонент СМЭВ, устанавливаемый на стороне Участника взаимодействия. СМЭВ3-адаптер обеспечивает информационное взаимодействие через единый электронный сервис единой системы межведомственного электронного взаимодействия (СМЭВ).

Сообщение

Сведения в виде законченного блока данных, передаваемые при функционировании информационной системы.

СУБД

Система управления базами данных

Токен

Ключ безопасности (Цифровой сертификат)

Участник НСУД

Федеральный орган исполнительной власти, иной орган государственной власти, государственный внебюджетный фонд, орган местного самоуправления, иное юридическое лицо, являющиеся стороной действующего соглашения о присоединении к Национальной системе управления данными.

ФЛК

Форматно-логический контроль загружаемых в Витрину данных.

Хранилище BLOB-объектов

Место для хранения BLOB-объектов (бинарных данных). Располагается на стороне ведомства и не является частью Витрины данных. Взаимодействие с Хранилищем BLOB-объектов осуществляется через BLOB-адаптер.

Хранилище S3 (объектное хранилище S3)

Хранилище бинарных объектов, позволяющее хранить файлы любого типа и объема. Доступ к хранилищу предоставляется через API.