ИНФРАСТРУКТУРА ЭЛЕКТРОННОГО ПРАВИТЕЛЬСТВА

# ВЫПОЛНЕНИЕ РАБОТ ПО РАЗВИТИЮ ТИПОВОГО ТИРАЖИРУЕМОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ВИТРИН ДАННЫХ

Техническое описание Типового ПО «Витрина данных НСУД»

Версия 1.16.3

Листов 178

Москва, 2024

**СОДЕРЖАНИЕ**

## Аннотация

Настоящий документ является общим описанием модернизированного программного обеспечения «Витрина данных НСУД» (далее – Программа).

Перед началом работы рекомендуется ознакомиться с документом 83219291.62.01.11.А003.ПД.01.1 «Описание применения» на ПО «Витрина данных НСУД», разработанным в рамках выполнения государственного контракта № 0173100007520000024_144316 от 13 ноября 2020 года, на выполнение работ по модернизации федеральной государственной информационной системы «Единая информационная платформа Национальной системы управления данными» в части выделения программного обеспечения компонента «Витрина данных» и его доработки, необходимой для подготовки к публикации на условиях открытой лицензии.

В разделе «Общие сведения» указаны назначение и возможности Программы, основные характеристики Программы, технические и программные средства, режим работы, требования к персоналу.

В разделе «Структура программы» приведены сведения о составных частях Программы, ее модулях, составе компонентов в дистрибутиве, связях между составными частями и другими программами, а также карта портов.

В разделе «Архитектура программы» приведены архитектурная и компонентная схемы и алгоритм работы Программы.

В разделе «Входные и выходные данные» приведено описание организации используемой входной и выходной информации.

В разделе «Вызов и загрузка» приведены описания действий по ручному запуску и остановке программы.

В разделе «Описание технических решений» приведены описания задач реализованных технических решений, а также их описание.

В приложениях к документу «Техническое описание программы «Витрина данных НСУД»» приведены описания спецификаций модулей Программы.

Оформление программного документа «Техническое описание программы «Витрина данных НСУД»» произведено по требованиям ЕСПД (ГОСТ 19.101-77, ГОСТ 19.103-77, ГОСТ 19.104-78, ГОСТ 19.105-78, ГОСТ 19.106-78, ГОСТ 19.503-79, ГОСТ 19.604-78).

# 1 ОБЩИЕ СВЕДЕНИЯ

## 1.1 Обозначение и наименование программы

Полное наименование: Типовое тиражируемое программное обеспечение «Витрина данных НСУД».

Условное обозначение: ПО «Витрина данных НСУД».

## 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.

Для формирования и (или) для получения данных с использованием среды взаимодействия НСУД необходим комплекс программных и технических средств в составе информационно-телекоммуникационной инфраструктуры участника НСУД, описываемое в данном документе «Витрина данных НСУД», но возможно и применение «Витрина данных НСУД». Данный документ описывает применение именно ПО среды взаимодействия НСУД.

Программа «Витрина данных НСУД» 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/потребителей данных.

## 1.3 Возможности программы

В настоящий момент реализовано две конфигурации Программы:

- Стандарт;
- Лайт.

## Возможности конфигурации Стандарт

Программа обеспечивает выполнение следующих задач:

- описание логической модели данных;
- настройка программы и структуры таблиц в ее БД для хранения публикуемых данных;
- загрузка и хранение публикуемых данных в БД программы;
- извлечение данных из внешних систем (внешних ИС по отношению к Витрине данных НСУД);
- выполнение запросов в соответствии с протоколом ПОДД через механизмы ПОДД СМЭВ:
  - поддержка протокола коммуникации Агента СМЭВ4;
  - предоставление публикуемых данных (в т. ч. BLOB-объектов и/или с использованием табличных параметров);
  - генерация формируемых документов на основании публикуемых данных;
  - репликация публикуемых данных (в качестве витрины-поставщика);
  - получение реплицируемых данных (в качестве витрины-получателя).
- обмен в соответствии с протоколом СМЭВ3:
  - подключение к СМЭВ3 как информационной системы участника взаимодействия;
  - обработку запросов на предоставление публикуемых данных (видов сведений), в т.ч. BLOB-объектов;
  - инициативная рассылка оповещений об обновлении публикуемых данных.
- публикация конечных точек API для обработки запросов с использованием спецификации OpenAPI версии 3;
- предоставление публикуемых данных информационным системам с использованием интерфейса REST-запросов;
- восстановление данных в непротиворечивое состояние после сбоев;
- поддержка языка SQL (см. Приложение 2 «Поддержка функций SQL», раздел «1 SQL-синтаксис»);
- журналирование событий функциональных блоков;
- мониторинг информации о работоспособности экземпляра Программы.

## Возможности конфигурации Лайт

Программа обеспечивает выполнение следующих задач:

- 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 Показатели назначения программы

Наименования и значения параметров, характеризующих показатели назначения программы, приведены в Таблица 1.1.

Таблица 1.1 Показатели назначения

| № | Показатель | Значение |
|---|---|---|
| 1 | Хранение данных, доступных для запроса, включая исторические | Не менее 10 ТБ или 1 млрд. записей |
| 2 | Загрузка данных в ПО «Витрина данных НСУД» | Не менее 3 ТБ / час или 300 млн. записей / час |
| 3 | Выборка данных из ПО «Витрина данных НСУД» в рамках одного критерия поиска | Не менее 3 ТБ / час или 300 млн. записей / час |
| 4 | Время от отправки одной записи в витрину до появления данных в результатах запросов | Не более 5 сек для 95% записей |
| 5 | Поиск одной записи по предопределённому критерию поиска (предполагается предварительная индексация) | Не более 0,1 сек в 95% запросов при не менее 3 млн. запросов / час |
| 6 | Поиск одной записи по произвольному критерию (предполагается сканирование всего объёма записей) | Не более 5 сек в 95% запросов при не менее 50 тыс. запросов / час |

Требования к техническим средствам, при которых достигаются указанные показатели назначения описаны в Таблица 1.4 и Таблица 1.5 Руководства по установке Типового ПО «Витрина данных НСУД» в разделе Рекомендуемые технические и программные средства.

Количественные значения показателей надежности для ПО «Витрина данных НСУД» представлены в таблице Таблица 1.2.

Таблица 1.2 Значения показателей надежности

| № | Показатель | Значение |
|---|---|---|
| 1 | Безотказность | 10000 часов |
| 2 | Коэффициент готовности | 0,9995 |
| 3 | Доступность | 99,95% |
| 4 | Допустимая потеря данных | 1 час |

Тестирование программы на соответствие показателям назначения и надежности проводилось на тестовом стенде с указанными требованиями к техническим средствам в Таблица 1.4 и Таблица 1.5 Руководства по установке Типового ПО «Витрина данных НСУД» в разделе Рекомендуемые технические и программные средства.

## 1.5 Требования к каналам связи

Требования к каналам связи сервера, на котором будет установлена программа:

– пропускная способность 100 Гбит/сек с протоколами TCP/IP и UDP;
– отсутствие ПО, блокирующего или замедляющего трафик.

Требования к каналам связи, используемым для взаимодействия с клиентскими сетями Ведомства:

– пропускная способность 10 Гбит/сек с протоколами TCP/IP и UDP.

Необходимо учитывать процент свободных ресурсов оборудования и пиковые нагрузки.

## 1.6 Режим работы программы

Программа предназначена для круглосуточного функционирования, 24 часа в сутки, 7 дней в неделю, с перерывами на плановое техническое обслуживание, восстановление работоспособности.

Программа поддерживает функционирование в следующих режимах:

- штатный;
- сервисный;
- аварийный;
- остановка.

### 1.6.1 Штатный режим

Штатный режим является основным режимом функционирования Программы. В этом режиме функционирования обеспечивается круглосуточная работа Программы без потери функциональности и работоспособности ее программно-технических средств.

### 1.6.2 Сервисный режим

Режим функционирования, при котором Программа выполняет часть основных функций, но параллельно с этим проводятся работы по техническому обслуживанию, накладывающие ограничения на функциональность Программы. В сервисном режиме функционирования обеспечивается возможность круглосуточного функционирования Программы.

### 1.6.3 Аварийный режим

Режим функционирования, при котором в Программе доступны только базовые функциональные возможности, необходимые для восстановления работоспособности.

### 1.6.4 Остановка

Режим работы, когда Программа не выполняет ни одну из своих функций.

## 1.7 Используемые языки программирования

Таблица 1.3 Языки программирования

| **Название** | **Версия** | **Описание** |
|---|---|---|
| Kotlin | 1.9.24 | Язык программирования |
| Java SE | 17 | Язык программирования |
| SQL | SQL:2016 | Язык программирования, применяемый для создания модификации и управления данными в реляционной базе данных |
| HTML | 5 | Язык разметки |
| CSS | 3 | Язык разметки. Описание внешнего вида документов |
| Python | 3.0 | Язык программирования |
| JavaScript | ES2017 (ES8) | Язык программирования |

## 1.8 Требования к персоналу

### 1.8.1 Требования к системному администратору

Системный администратор обеспечивает штатную работу программы. Выполняет установку, настройку и мониторинг работоспособности программного и аппаратного обеспечения.

В основные обязанности системного администратора входит:

- установка и настройка программы;
- установка обновлений;
- резервное копирование;
- настройка логирования;
- настройка системы мониторинга программы, аппаратного обеспечения и др.

Необходимые навыки для работы:

- знание TCP/IP;
- знание Docker (развертывание, установка библиотек, администрирование);
- знание Ansible;
- умение работать с веб-серверами (Apache);
- знание SSH;
- резервное копирование;
- логирование;
- обновление программы;
- понимание модели баз данных;
- знание операционных систем CentOS на уровне администрирования;
- опыт работы по управлению и администрированию баз данных;
- навык проводить диагностику и анализ проблемных мест;
- умение распознать следствие/причины некорректной работы ПО или техники;
- умение анализировать сетевой трафик.

### 1.8.2 Требования к программисту

Программист решает задачи связанные непосредственно с сопровождением и доработкой программы под требования пользователей. Определяет схемы и алгоритмы обработки данных программой.

Программист должен обладать знаниями следующих языков программирования:

- Java;
- Kotlin;
- Javascript.

Программист должен уметь использовать в своей работе следующее программное обеспечение и технологии:

- Apache Zookeeper;
- Apache Kafka;
- Docker;
- Vertx;
- Spring (Spring-boot);
- Web (http, настройка SSH).

В перечень задач, выполняемых программистом, входят:

- поддержание работоспособности программы;

– доработка программы под требованиям пользователей системы.

# 2 СТРУКТУРА ВИТРИНЫ ДАННЫХ

## 2.1 Составные части Витрины данных

Программа имеет модульную архитектуру и построена на базе отдельных компонентов (включая разработки сторонних производителей).

Общую схему взаимосвязей компонентов можно просмотреть в разделе Архитектура Витрины данных.

### Составные части конфигурации Стандарт

**Основные компоненты**

- **ProStore** - основной компонент программы с открытым исходным кодом, обеспечивает 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 включает следующие компоненты:
  - **Сервис исполнения запросов** — анализирует и исполняет SQL-запросы; предоставляет REST API для JDBC-драйвера и взаимодействует с сервисом мониторинга статусов Kafka по REST API. В свою очередь состоит из следующих компонентов:
    - Коннектор **Kafka-Postgres reader** - считывает данные из PostgreSQL и передает их в брокер сообщений Kafka;
    - Коннектор **Kafka-Postgres writer** - записывает данные из брокера сообщений Kafka в PostgreSQL;
- **Слой адаптеров** - общее название логических модулей программы, которые обеспечивают подключение к СМЭВ4, как информационной системы участника взаимодействия. В зависимости от предназначения логические модули обеспечивают загрузку запросов из очереди ИС УВ в СМЭВ4 , формирование и отправку ответов в СМЭВ4, инициативное формирование уведомлений об изменении данных в экземпляре ПО «Витрина данных НСУД», отправку уведомлений в СМЭВ4, регистрацию реплики данных ИС УВ, подписки на репликацию и поддержку реплики в актуальном состоянии.

**Компоненты, входящие в состав слоя адаптеров**

- **СМЭВ3-адаптер** - обеспечивает информационное взаимодействие через единый электронный сервис единой системы межведомственного электронного взаимодействия (далее – СМЭВ).
- **Сервис формирования документов** - предназначен для обеспечения возможности формирования документов, в формате XML и PDF, на основе предварительно подготовленных pebble-шаблонов, с возможностью добавления к сформированным документам электронной подписи.
- **CSV-Uploader** - программный модуль Витрины данных, предназначен для загрузки и выгрузки csv-файлов со структурой Витрины и csv-шаблонов с демо-шаблонами структуры Витрины, а также просмотра Журнала операций.
- **СМЭВ4-адаптер - Модуль исполнения запросов** - предназначен для исполнения запросов СМЭВ4 (через протокол коммуникации Агент СМЭВ4).

- **СМЭВ4-адаптер – Модуль MPPR** - предназначен для чтения данных в многопоточном режиме.
- **СМЭВ4-адаптер - Модуль MPPW** - исполняет запросы в многопоточном режиме, записывающие данные в Prostore.
- **Модуль группировки чанков репликации** - группирует фрагменты данных подписки, полученные из топика `delta.in.rq` и размещает их во временные топики с именем `mppw.data.[hash (requestId+subscriptionId)].deltaNum.streamNum`, отправляет команду в топик `subscription.in` модулю подписок при получении `lastChunk` на загрузку сгруппированных фрагментов (по каждой дельте каждого стрима).
- **СМЭВ4-адаптер - Модуль группировки данных табличных параметров** - предназначен для группировки данных при выполнении запросов с табличными параметрами.
- **СМЭВ4-адаптер - Модуль импорта табличных параметров** - предназначен для создания временных таблиц при выполнении временных запросов с табличными параметрами.
- **СМЭВ4-адаптер - Модуль дефрагментации чанков табличных параметров** - образует пакеты с данными табличных параметров, поступающие от Агента СМЭВ4 в брокер сообщений Kafka, к формату, позволяющему обрабатывать их в многопоточном режиме.
- **СМЭВ4-адаптер - Модуль подписок** - предназначен для управления подписками между Получателем данных (consumer) и Поставщиком данных (producer).
- **BLOB-адаптер** - программный модуль Витрины данных, предназначен для получения доступа из Витрины данных к BLOB-объектам ведомства (BLOB-объект - это специальный тип двоичных данных, предназначенный для хранения бинарных файлов: изображений, скан-копий документов, текстовых файлов и т.д.).
- **DATA-Uploader** - Модуль исполнения асинхронных заданий обеспечивает обработку очереди файлов, используя следующие функциональные особенности:
  - обработка очереди файлов производится циклами;
  - очередь файлов работает в режиме упорядочения процесса по принципу «первым пришел – первым обслужен»;
  - каждый элемент в очереди файлов содержит UUID задания, имя витрины и таблицы, содержимое CSV-файла;
  - файлы в очереди могут относится к разным витринам и/или разным таблицам одной витрины.
- **REST-Uploader** - Модуль асинхронной загрузки данных из сторонних источников реализован для обеспечения параллельной загрузки данных с независимым масштабированием REST интерфейса.
- **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.
- **Counter-provider** - Сервис генерации уникального номера (Counter-Provider) позволяет создавать неповторяющиеся уникальные порядковые номера для сквозной нумерации файлов в сервисе формирования документов Типового ПО

Витрины данных конфигурации установки Стандарт.

- **СМЭВ QL Сервер** - СМЭВ QL Сервер — приложение для конфигурирования и запуска типового API извлечения данных из хранилищ под управлением типового ПО «Витрина данных» от Минцифры (Ядро Prostore 6.1+). СМЭВ QL Сервер обладает характеристиками:
  - поддержка множественных источников данных Prostore;
  - выбор источника по условиям запроса;
  - защита атрибутов модели данных по их предоставлению;
  - автоматический параллелизм запросов к Prostore.
- **dtm-calcite-core** - библиотека Calcite осуществляет парсинг классов, которые используются в библиотеке podd-adapter-calcite.

**Дополнительные компоненты**

Рекомендуемое программное обеспечение для администрирования и мониторинга (не входит в поставку программы):

- **Агент СМЭВ4** - программное обеспечение, обеспечивающее сопряжение Типового ПО «Витрина данных» и ИС УВ с Ядром СМЭВ4.
- **Брокер сообщений Kafka** - используется для непрерывной передачи сообщений между СМЭВ4-адаптером - Модуль исполнения запросов и Агентом СМЭВ4.
- **Apache ZooKeeper** - необходим для поддержки информации о конфигурации и распределенной координации между компонентами Витрины, также используется как сервисная база данных ProStore, для хранения технической информации (метаданных) от поступающих в Витрину данных запросах.
- **Docker** - программное обеспечение для автоматизации развёртывания и управления программы в виртуальных средах с поддержкой контейнеризации. Используется для установки Arenadata Cluster Manager (ADCM). Официальный сайт разработчика приложения: https://www.docker.com/.
- **Elasticsearch** - система поиска и аналитики, позволяет быстро в режиме реального времени хранить, искать и анализировать большие объемы данных и сохраняет их для Graylog. Для передачи сообщений в Graylog использует Filebeat. Официальный сайт разработчика приложения: https://www.elastic.co/elasticsearch/.
- **Filebeat** - агент на сервере для отправки различных типов оперативных данных в Elasticsearch. Официальный сайт разработчика приложения: https://www.elastic.co/elasticsearch/.
- **Grafana** - инструмент реализован в виде панели управления и мониторинга и позволяет визуализировать системные события программы на базе собираемых метрик. Официальный сайт разработчика приложения: https://grafana.com/docs/.
- **Graylog** - программное обеспечение для управления лог-файлами. Официальный сайт разработчика приложения: https://www.graylog.org/;
- **MongoDB** - база данных Graylog. Официальный сайт разработчика приложения: https://www.mongodb.com/.
- **Node_exporter** - процессы, обеспечивающие сбор и передачу системных метрик серверу Prometheus. Также, используется для сбора метрик модулей СМЭВ4-адаптера и CSV-uploader см. https://github.com/prometheus/node_exporter.
- **Prometheus** - используется как система мониторинга системных ресурсов программы. Связь компонентов реализована через HTTP. Данные хранятся локально, в собственной TSBD базе, индексы хранятся в LevelDB. Метрики

представляют собой time-series данные. Каждая метрика состоит из имени метрики, временной метки и пары «ключ – значение». Визуализация осуществляется через подключение к Grafana. Официальный сайт разработчика приложения: https://prometheus.io/.

– **Redis** - резидентная система управления базами данных класса NoSQL, работающая со структурами данных типа «ключ — значение».

## Составные части конфигурации Лайт

### Основные компоненты

– **ProStore** - основной компонент программы с открытым исходным кодом, обеспечивает 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 включает следующие компоненты:
  - **Сервис исполнения запросов** — анализирует и исполняет SQL-запросы; предоставляет REST API для JDBC-драйвера и взаимодействует с сервисом мониторинга статусов Kafka по REST API. В свою очередь состоит из следующих компонентов:
    - Коннектор **Kafka-Postgres reader** - считывает данные из PostgreSQL и передает их в брокер сообщений Kafka;
    - Коннектор **Kafka-Postgres writer** - записывает данные из брокера сообщений Kafka в PostgreSQL;
– **Слой адаптеров** - общее название логических модулей программы, которые обеспечивают подключение к СМЭВ4, как информационной системы участника взаимодействия. В зависимости от предназначения логические модули обеспечивают загрузку запросов из очереди ИС УВ в СМЭВ4 , формирование и отправку ответов в СМЭВ4, инициативное формирование уведомлений об изменении данных в экземпляре ПО «Витрина данных НСУД», отправку уведомлений в СМЭВ4, регистрацию реплики данных ИС УВ, подписки на репликацию и поддержку реплики в актуальном состоянии.

### Компоненты, входящие в состав слоя адаптеров

– **CSV-Uploader** - программный модуль Витрины данных, предназначен для загрузки и выгрузки csv-файлов со структурой Витрины и csv-шаблонов с демо-шаблонами структуры Витрины, а также просмотра Журнала операций.
– **СМЭВ4-адаптер - Модуль исполнения запросов** - предназначен для исполнения запросов СМЭВ4 (через протокол коммуникации Агент СМЭВ4).
– **СМЭВ4-адаптер – Модуль MPPR** - предназначен для чтения данных в многопоточном режиме.
– **СМЭВ4-адаптер - Модуль MPPW** - исполняет запросы в многопоточном режиме, записывающие данные в Prostore.
– **Модуль группировки чанков репликации** - группирует фрагменты данных подписки, полученные из топика `delta.in.rq` и размещает их во временные топики с именем `mppw.data.[hash (requestId+subscriptionId)].deltaNum.streamNum`, отправляет команду в топик `subscription.in` модулю подписок при получении `lastChunk` на загрузку

сгруппированных фрагментов (по каждой дельте каждого стрима).

- **СМЭВ4-адаптер - Модуль подписок** - предназначен для управления подписками между Получателем данных (consumer) и Поставщиком данных (producer).
- **СМЭВ4-адаптер - Модуль группировки данных табличных параметров** - предназначен для группировки данных при выполнении запросов с табличными параметрами.
- **СМЭВ4-адаптер - Модуль импорта табличных параметров** - предназначен для создания временных таблиц при выполнении временных запросов с табличными параметрами.
- **СМЭВ4-адаптер - Модуль дефрагментации чанков табличных параметров** - образует пакеты с данными табличных параметров, поступающие от Агента СМЭВ4 в брокер сообщений Kafka, к формату, позволяющему обрабатывать их в многопоточном режиме.

**Дополнительные компоненты**

- **Apache ZooKeeper** — необходим для поддержки информации о конфигурации и распределенной координации между компонентами Витрины, также используется как сервисная база данных ProStore, для хранения технической информации (метаданных) от поступающих в Витрину данных запросах.
- **Брокер сообщений Kafka** — используется для непрерывной передачи сообщений между:
  - **CSV-uploader** и **ProStore**;
  - **СМЭВ4-адаптером – Модулем исполнения запросов** и **Агентом СМЭВ4**.
- **Ansible** — платформа удалённого управления конфигурациями программного обеспечения, предназначенная для упрощения развёртывания «Витрина данных Лайт» через создание специальных сценариев. Официальный сайт разработчика приложения: https://www.ansible.com/.
- **Docker** — программное обеспечение для автоматизации развёртывания и управления программы в виртуальных средах с поддержкой контейнеризации. Контейнер позволяет производить изолированный запуск ОС с подключённой файловой системой из образа, изолированно разворачивать приложения и реализовывать микросервисы. Настройки среды хранятся в GitHub, обеспечивая единую точку управления конфигурациями. Может быть использован для для развёртывания тестового окружения «Витрина данных Лайт», без прерывания работы сервисов в продуктовой среде. Официальный сайт разработчика приложения: https://www.docker.com/.
- **Elasticsearch** — утилита полнотекстового поиска и аналитики, которая позволяет быстро в режиме реального времени хранить, искать и анализировать большие объемы данных и сохраняет их для Graylog. Для передачи сообщений в Graylog использует Filebeat. Официальный сайт разработчика приложения: https://www.elastic.co/elasticsearch/.
- **Filebeat** — агент на сервере для отправки различных типов оперативных данных в Elasticsearch. Официальный сайт разработчика приложения: https://www.elastic.co/elasticsearch/.
- **Grafana** — инструмент реализован в виде панели управления и мониторинга и позволяет визуализировать системные события программы на базе собираемых

метрик. Официальный сайт разработчика приложения: https://grafana.com/docs/.

- **Graylog** — программное обеспечение для управления лог-файлами. Официальный сайт разработчика приложения: https://www.graylog.org/.
- **MongoDB** — база данных Graylog. Официальный сайт разработчика приложения: https://www.mongodb.com/.
- **Node_exporter** — процессы, обеспечивающие сбор и передачу системных метрик серверу Prometheus. Также, используется для сбора метрик СМЭВ4-адаптера и CSV-uploader см. https://github.com/prometheus/node_exporter.
- **Portainer** — web-приложение для управления docker-контейнерами. Официальный сайт разработчика приложения: https://www.portainer.io/.
- **Prometheus** — используется как система мониторинга системных ресурсов «Витрина данных Лайт». Связь компонентов реализована через HTTP. Данные хранятся локально, в собственной TSBD базе, индексы хранятся в LevelDB. Метрики представляют собой time-series данные. Каждая метрика состоит из имени метрики, временной метки и пары «ключ – значение». Визуализация осуществляется через подключение к Grafana. Официальный сайт разработчика приложения: https://prometheus.io/.
- **PostrgeSQL** — база данных ProStore.

## 2.2 Состав компонентов в дистрибутиве

### Состав компонентов в дистрибутиве конфигурации Стандарт

Состав компонентов в дистрибутиве конфигурации Стандарт версии 1.16.3 приведен в таблице ниже (см. Таблица 2.1)

Таблица 2.1 Состав компонентов в дистрибутиве программы

| Наименование компонента | Версия | Техническое наименование |
|---|---|---|
| query-execution | 6.12.1 | query-execution:6.12.1 |
| backup-manager | 1.16.3 | backup-manager: 1.16.3 |
| blob-adapter | 1.16.3 | blob-adapter:1.16.3 |
| counter-provider | 1.16.3 | counter-provider:1.16.3 |
| csv-uploader | 1.16.3 | csv-uploader:1.16.3 |
| data-uploader | 1.16.3 | data-uploader:1.16.3 |
| dtm-uploader | 1.16.3 | dtm-uploader:1.16.3 |
| podd-adapter-group-repl | 1.16.3 | podd-adapter-group-repl:1.16.3 |
| podd-adapter-group-tp | 1.16.3 | podd-adapter-group-tp:1.16.3 |
| podd-adapter-import-tp | 1.16.3 | podd-adapter-import-tp:1.16.3 |
| podd-adapter-mppr | 1.16.3 | podd-adapter-mppr:1.16.3 |
| podd-adapter-mppw | 1.16.3 | podd-adapter-mppw:1.16.3 |
| podd-adapter-query | 1.16.3 | podd-adapter-query:1.16.3 |
| podd-adapter-replicator | 1.16.3 | podd-adapter-replicator:1.16.3 |
| podd-avro-defragmentator | 1.16.3 | podd-avro-defragmentator:1.16.3 |
| printable-form-service | 1.16.3 | printable-form-service:1.16.3 |
| rest-adapter | 1.16.3 | rest-adapter:1.16.3 |
| rest-uploader | 1.16.3 | rest-uploader:1.16.3 |

| | | |
|---|---|---|
| smevql-server | 1.1.0 | smevql-server:1.1.0 |
| СМЭВ3-адаптер | 1.16.3 | smev3-adapter:1.16.3 |
| dtm-tools | 1.17.0 | dtm-tools:1.17.0 |
| greenplum-kafka PXF connector reader | 1.2.0 | pxf-kafka:1.2.0 |
| kafka-greenplum PXF connector writer | 1.2.0 | pxf-kafka-greenplum:1.2.0 |
| kafka-clickhouse-reader | 0.9.0 | kafka-clickhouse-reader:0.9.0 |
| kafka-clickhouse-writer | 0.9.0 | kafka-clickhouse-writer:0.9.0 |
| kafka-postgres-writer | 0.11.0 | kafka-postgres-writer:0.11.0 |
| kafka-postgres-reader | 0.11.0 | kafka-postgres-reader:0.11.0 |
| kafka-jet-writer | 1.5.0 | kafka-jet-writer:1.5.0 |

Дистрибутив программы версии 1.16.3 тестировался с компонентами, приведенными в Таблица 2.2

Таблица 2.2 компоненты для тестирования программы

| **Наименование компонента** | **Версия** | **Техническое наименование** |
|---|---|---|
| Агент СМЭВ4 | 3.18.0 | Агент СМЭВ4:3.18.0 |
| kafka | 2.6.0 | kafka:2.6.0 |
| PostgreSQL | 13.4 | postgreSQL:13.4 |
| redis | 7.0.11 | redis:7.0.11 |

Примечание:

Дополнительные компоненты не входят в дистрибутив Программы

Конкретная конфигурация Витрины данных определяется Участником взаимодействия на этапе реализации Витрины данных в составе ИТ-инфраструктуры Участника взаимодействия.

## Состав компонентов в дистрибутиве конфигурации Лайт

Состав компонентов в дистрибутиве конфигурации Лайт версии 1.16.3 приведен в таблице ниже (см. Таблица 2.3)

Таблица 2.3 Состав компонентов в дистрибутиве программы

| **Наименование компонента** | **Версия** | **Техническое наименование** |
|---|---|---|
| graylog | 4.3.15 | graylog:4.3.15 |
| prometheus | 2.34.0 | prometheus:2.34.0 |
| query-execution | 6.12.1 | query-execution:6.12.1 |
| podd-adapter-replicator | 1.16.3 | podd-adapter-replicator:1.16.3 |
| podd-adapter-mppw | 1.16.3 | podd-adapter-mppw:1.16.3 |
| podd-adapter-mppr | 1.16.3 | podd-adapter-mppr:1.16.3 |
| podd-adapter-group-repl | 1.16.3 | podd-adapter-group-repl:1.16.3 |
| podd-adapter-query | 1.16.3 | podd-adapter:1.16.3 |
| podd-adapter-group-tp | 1.16.3 | podd-adapter-group-tp:1.16.3 |
| podd-adapter-import-tp | 1.16.3 | podd-adapter-import-tp:1.16.3 |
| podd-avro-defragmentator | 1.16.3 | podd-avro-defragmentator:1.16.3 |
| csv-uploader | 1.16.3 | csv-uploader:1.16.3 |
| kafka-jet-writer | 1.5.0 | kafka-jet-writer:1.5.0 |

| | | |
|---|---|---|
| grafana | 9.5.2 | grafana:9.5.2 |
| node_exporter | 1.3.1 | node_exporter:1.3.1 |
| filebeat | 7.10.2 | filebeat:7.10.2 |
| mongo | 4.4 | mongo:4.4 |
| opensearch | 1.3.14 | opensearch:1.3.14 |
| kafka-postgres-writer | 0.11.0 | kafka-postgres-writer:0.11.0 |
| kafka-postgres-reader | 0.11.0 | kafka-postgres-reader:0.11.0 |
| postgres | 13.4 | postgres:13.4 |
| kafka | 2.13 | kafka:2.13-2.6.0-alt-p10-r3 |
| zookeeper | 3.5.7 | zookeeper:3.5.7-alt-p10-r3 |
| portainer | 2.14.0 | portainer:2.14.0 |

## 2.3 Связи между компонентами

### Связи между компонентами конфигурации Стандарт

Связи между компонентами конфигурации Стандарт приведены в Таблица 2.4.

Таблица 2.4 Взаимодействие между составными частями

| **Клиент** | **Сервер** | **Способ взаимодействия** | **Описание** |
|---|---|---|---|
| Сервисная база данных Prostore (PostgreSQL) | Prostore | TCP | Хранение/получение метаданных |
| | СУБД хранилища | | |
| | Apache Kafka | | |
| | СМЭВ4 Адаптер | | |
| | ETL | | |
| Prostore | Хранилище СУБД | REST API<br>в зависимости<br>от типа СУБД хранилища данных | Перенаправление запросов в конкретную СУБД |
| REST-адаптер | ProStore | REST API | Исполнение запросов |
| ETL | ProStore | Kafka (Диспетчер сообщений)<br>JDBC | Исполнение запросов |
| Сервис формирования документов | ProStore | REST API | Запрашивает данные для формирования ПФ |
| СМЭВ3-адаптер | ProStore | Kafka (Диспетчер сообщений)<br>REST API | Исполнение запросов |
| | BLOB-адаптер | HTTP | Запрашивает бинарное содержимое BLOB. |

| Клиент | Сервер | Способ взаимодействия | Описание |
|---|---|---|---|
| СМЭВ4-адаптер - Модуль исполнения запросов | ProStore | REST API | – исполняет полученный запрос (LLR);<br>– запрашивает структуру таблиц при подписании на репликацию (источник данных);<br>– создает структуру таблиц при подписании на репликацию (потребитель данных); |
| | Диспетчер сообщений Kafka | Kafka (Диспетчер сообщений) | – получает сообщение о закрытии новой дельты в ProStore (для репликации, источник данных);<br>– обменивается управляющей информацией с модулем импорта ТП;<br>– отправляет запросы, перенаправленные в Модуль MPPR. |
| СМЭВ4-адаптер - Модуль MPPR | ProStore | REST API | Подает запрос MPPR. |
| | Диспетчер сообщений Kafka | Kafka | – получает запросы данных СМЭВ4, перенаправленные СМЭВ4 Адаптером;<br>– получает запрошенные данные из ProStore (в формате MPPR);<br>– передает запрошенные данные в СМЭВ4 в формате СМЭВ4;<br>– обменивается управляющей информацией с модулем импорта ТП. |
| СМЭВ4-адаптер - Модуль MPPW | ProStore | REST API | Подает запрос MPPW. |
| | Диспетчер сообщений Kafka | Kafka | – получает записываемые данные, подготовленные модулем Группировки данных ТП;<br>– передает записываемые данные в ProStore (в формате MPPW);<br>– обменивается управляющей информацией с модулем импорта ТП. |
| СМЭВ4-адаптер - Модуль импорта ТП | ProStore | REST API | – создает табличный параметр (перед выполнением запроса);<br>– удаляет табличный параметр (после выполнения запроса). |
| | Диспетчер сообщений Kafka | Kafka | Обменивается управляющей информацией с модулями:<br>– MPPW;<br>– MPPR;<br>– СМЭВ4 Адаптер;<br>– Группировки данных ТП. |
| СМЭВ4-адаптер - Модуль Группировки данных ТП | Диспетчер сообщений Kafka | Kafka | – Получает сообщения с ТП, подготовленные Модуль дефрагментации чанков табличных параметров (из единого топика);<br>– Создает отдельные топики для каждого ТП и передает в них сообщения с данными этого ТП. |
| СМЭВ4-адаптер - Модуль дефрагментации чанков табличных параметров | Диспетчер сообщений Kafka | Kafka | Передает сообщения с ТП, в формате пригодном для параллельной обработки |

| Клиент | Сервер | Способ взаимодействия | Описание |
|---|---|---|---|
| | Сервисная СУБД Zookeeper | TCP | Сохраняет список обрабатываемых сообщений ТП<br>Хранение метаданных о подписках на репликацию (источник данных). |

### Связи между компонентами конфигурации Лайт

Связи между компонентами конфигурации Лайт приведены в Таблица 2.5.

Таблица 2.5 Взаимодействие между составными частями

| Клиент | Сервер | Способ взаимодействия | Описание |
|---|---|---|---|
| СМЭВ4-адаптер — Модуль исполнения запросов | ProStore | JDBC<br>Брокер сообщений Kafka | Исполнение запросов. |
| CSV-Uploader | ProStore | JDBC<br>Брокер сообщений Kafka | Управление логической структурой таблиц.<br>Загрузка публикуемых данных в Витрину. |
| ProStore | СУБД PostgreSQL | JDBC | Управление логической структурой таблиц.<br>Исполнение запросов.<br>Управление загрузкой публикуемых данных в Витрину. |
| СМЭВ4 - адаптер — Модуль MPPR | Агент СМЭВ4 | Через брокера сообщений Kafka | Предоставляет Результат запрос/подзапрос на получение публикуемых данных<br>(в т.ч. с использованием ТП), делегированного СМЭВ4-адаптером. |

## 2.4 Модули Витрины данных

| Примечание: |
|---|
| Описание настроек модулей, процессы запуска и остановки модуля см. «Руководстве администратора». |

### 2.4.1 СМЭВ4-адаптер - Модуль исполнения запросов

| Примечание: |
|---|
| Модуль входит в состав конфигурации Стандарт и Лайт |

#### 2.4.1.1 Общее описание

Логический модуль **СМЭВ4-адаптер - Модуль исполнения запросов** предназначен для исполнения запросов СМЭВ4 (через протокол коммуникации Агент СМЭВ4).

Установка опциональна модуля опциональна.

Обмен сообщениями между **Модулем исполнения запросов** и **Агентом СМЭВ4** происходит через заранее согласованные топики брокера сообщений **Kafka**.

Формат обмена электронными сообщениями описан в разделе Спецификация Модуля исполнения запросов Приложения 1.

**2.4.1.1.1 Общая схема взаимодействия через СМЭВ4-адаптер- Модуль исполнения запросов**

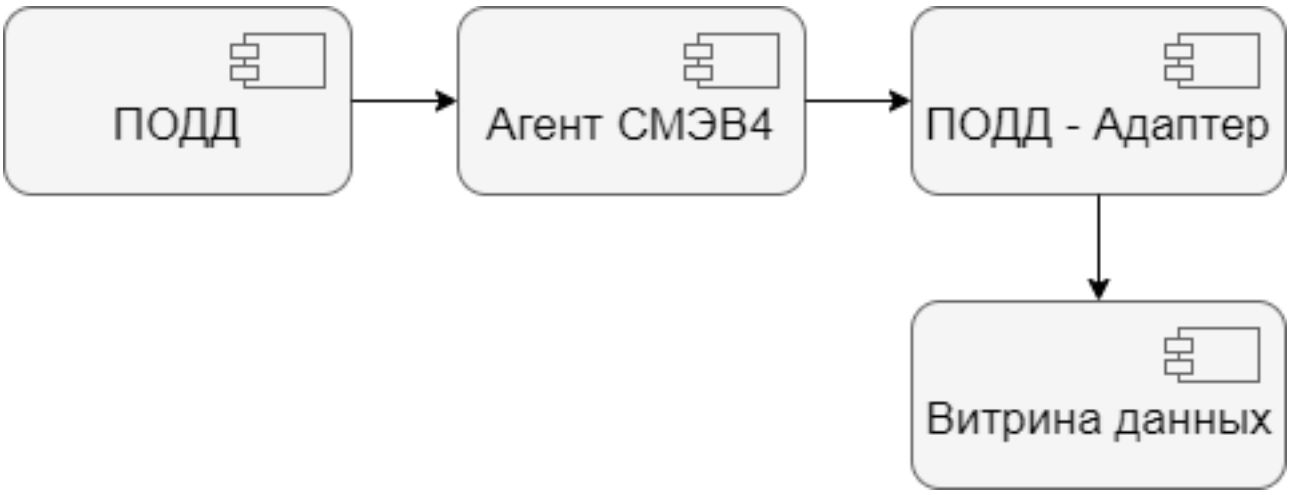


Рисунок - 2.1 Взаимодействие программы с СМЭВ4

**2.4.1.1.2 Процесс обработки запроса через СМЭВ4-адаптер- Модуль исполнения запросов**

1. Получатель данных отправляет через СМЭВ4 запрос к Витрине данных.
2. Запрос поступает в **Агент СМЭВ4**.
3. **Модуль исполнения запросов** (через заранее согласованные топики брокера сообщений Kafka) получает запрос от **Агента СМЭВ4** на предоставление данных.
4. **Модуль исполнения запросов** обрабатывает запрос и отправляет его в Витрину данных.
5. Витрина данных обрабатывает запрос и формирует на него ответ в СМЭВ4-адаптер.
6. **Модуль исполнения запросов** обрабатывает ответ, записывает результат в заранее согласованные топик обмена сообщениями и предоставляет ответ **Агенту СМЭВ4**.
7. **Агент СМЭВ4** отправляет полученный ответ через СМЭВ4 Получателю данных.

Процесс получения BLOB-объектов через **Модуль исполнения запросов** описан в разделе Взаимодействие через СМЭВ-адаптер.

## 2.4.2 СМЭВ4-адаптер - Модуль подписок

| Примечание: |
| --- |
| Модуль входит в состав конфигурации Стандарт и Лайт |

### 2.4.2.1 Общее описание

**СМЭВ4-адаптер - Модуль подписок** предназначен для управления подписками между Получателем данных (consumer) и Поставщиком данных (producer).

Модуль используется для получения результатов комплексных запросов из нескольких Витрин источников. Подписка позволяет автоматически загрузить и поддерживать в актуальном состоянии данные из Витрины Поставщика в специальном хранилище на стороне Потребителя - **Хранилище данных по подписке**. Потребитель посылает запросы напрямую в своё **Хранилище данных по подписке**, в результате чего сокращается продолжительность сеансов обмена и необходимость «склейки» запросов на стороне **СМЭВ4**.

Обмен между Витринами осуществляется по предварительно созданной подписке на уведомления об изменениях или репликацию.

Модуль решает следующие задачи:

- запрос создания подписки (Поставщик данных);
- запрос отмены подписки (Поставщик данных);
- запрос дельты (Поставщик данных);
- запрос создания структуры по подписке (Получатель данных);
- запрос применения дельты (Получатель данных).

Формат обмена электронными сообщениями описан в разделе Спецификация Модуля исполнения запросов Приложения 1.

Потребители данных могут получать сведения из Витрин Поставщиков данных путем:

1. отправки регламентированных запросов;
2. подписки на изменения сведений.

Подписка позволяет автоматически загрузить и поддерживать в актуальном состоянии данные из Витрины Поставщика в специальном хранилище на стороне Потребителя (Хранилище данных по подписке) и Потребитель посылает запросы напрямую в своё Хранилище, в результате чего сокращается продолжительность сеансов обмена и необходимость «склейки» запросов на стороне **СМЭВ4**.

Информационный обмен по подписке состоит из следующих этапов:

1. Регистрация подписки в Витрине Поставщика данных и создание структуры данных в Хранилище Потребителя данных.
2. Передача снапшота из Витрины Поставщика данных в Хранилище Потребителя данных (только для подписки на репликацию). В текущей реализации снапшот не содержит историчность.
3. Актуализация данных посредством передачи пакета дельт от Витрины Поставщика данных в Хранилище Потребителя данных в одном из режимов:
   - по расписанию (если оно указано в подписке);
   - по событию об изменении данных (если расписание не указано в подписке).

   Подписка определяется следующими параметрами:
   - уникальный идентификатор подписки;
   - источник данных по подписке – мнемоника Витрины Поставщика данных;
   - адресат данных по подписке – мнемоника Витрины Потребителя данных;
   - набор SQL-выражений, каждое из которых описывает подмножество данных Витрины;
   - расписание синхронизации (может отсутствовать).

   Виды подписок:
   - Подписка на репликацию - снэпшот текущего состояния витрины;
   - Подписка на уведомление - выгружаем данные только по дельте.

   Реализованы два варианта подписки:
   - одиночная;
   - распределенная.

   Ключевые особенности одиночных подписок:
   - подписка только на один датамарт;
   - в одиночных подписках можно создать подписку с множественными SQL-запросами к разным таблицам одной витрины.

   Ключевые особенности распределенных подписок:
   - количество витрин-источников больше 1;
   - одной подписке соответствует один SQL-запрос;
   - один датамарт может фигурировать в нескольких подписках витрины потребителя.

В случае необходимости отключить подписку, осуществляется отмена подписки через ВС «Отмена подписки на репликацию или уведомлений в изменении данных».

### 2.4.3 СМЭВ4-адаптер – Модуль MPPR

| Примечание: |
| --- |
| Модуль входит в состав конфигурации Стандарт и Лайт |

#### 2.4.3.1 Общее описание

Логический модуль **СМЭВ4-адаптер - Модуль MPPR** является частью **СМЭВ4-адаптера** и предназначен для чтения данных в многопоточном режиме (*MPP - massively parallel processing*).

**Модуль MPPR** предназначен для следующих задач:

1. Многопоточное параллельное чтение данных.
2. Отправка ответа с результатом запроса в **Агент СМЭВ4**.
3. Удаление временных таблиц, созданных на основе табличных параметров.

Обмен сообщениями между **СМЭВ4-адаптером** и **Модулем MPPR** происходит через топик `mppr.query`.

Формат обмена электронными сообщениями описан в разделе Спецификация Модуля исполнения запросов Приложения 1.

##### 2.4.3.1.1 Общая схема взаимодействия

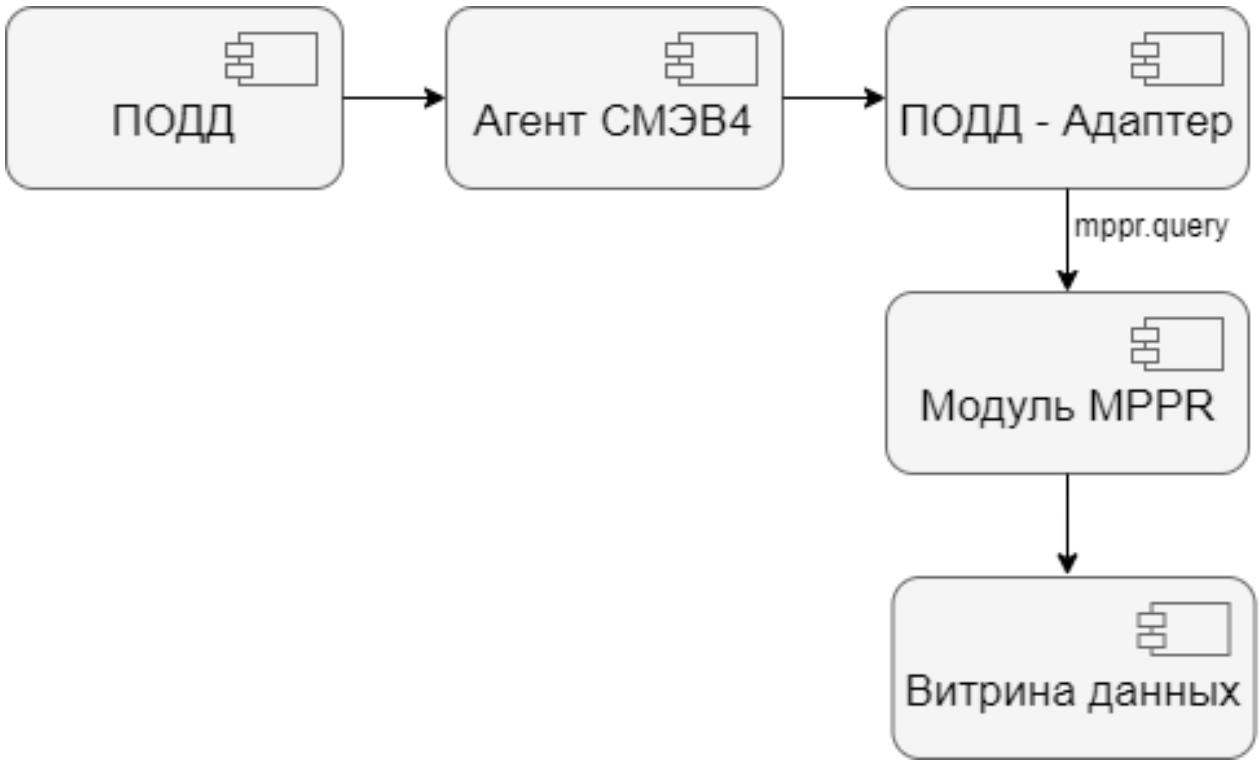


Рисунок - 2.2 Взаимодействие через Модуль MPPR

##### 2.4.3.1.2 Процесс обработки запроса через Модуль MPPR

1. Получатель данных отправляет через СМЭВ4 запрос к Витрине данных.
2. Запрос поступает через **Агент СМЭВ4** в **СМЭВ4-адаптер**.
3. Если формат обработки данных предполагает *MPP*, то **СМЭВ4-адаптер** отправляет запрос через топик `mppr.query` в **Модуль MPPR**.

4. **Модуль MPPR** создает временную таблицу (по результатам запроса) и временный топик с запросом для Витрины.
5. Витрина считывает топик, обрабатывает запрос, формирует на него ответ.
6. **Модуль MPPR** получает ответ и выкладывает полученные данные во временную таблицу.
7. **СМЭВ4-адаптер** считывает ответ из временной таблицы и отправляет данные в **Агент СМЭВ4**.
8. **Агент СМЭВ4** отправляет полученный ответ через **СМЭВ4** Получателю данных.
9. **Модуль MPPR** удаляет временный топик и таблицу.

### 2.4.4 СМЭВ4-адаптер - Модуль MPPW

| Примечание: |
| --- |
| Модуль входит в состав конфигурации Стандарт и Лайт |

#### 2.4.4.1 Общее описание

Модуль **СМЭВ4-адаптер-Модуль MPPW** исполняет запросы в многопоточном режиме, записывающий данные в **Prostore**.

Модуль предназначен для следующих задач:

1. Записывать данные в базу данных **Prostore** при получении команд от других модулей программы.
2. Оповещать другие модули об успешной и/или неуспешной записи данных в базу данных **Prostore**.

Формат обмена электронными сообщениями описан в разделе Спецификация Модуля исполнения запросов «Приложения 1.

### 2.4.5 СМЭВ4-адаптер - Модуль группировки чанков репликации

| Примечание: |
| --- |
| Модуль входит в состав конфигурации Стандарт и Лайт |

#### 2.4.5.1 Общее описание

**Модуль группировки чанков репликации** на стороне Витрины потребителя при обмене по подписке группирует фрагменты данных подписки, полученные из топика `delta.in.rq` и размещает их во временные топики с именем `mppw.data.[hash (requestId+subscriptionId)].deltaNum.streamNum`, отправляет команду в топик `subscription.in` модулю подписок при получении `lastChunk` на загрузку сгруппированных фрагментов (по каждой дельте каждого стрима).

##### 2.4.5.1.1 Интерфейсы модуля

Входящие топики

- `delta.in.rq`

Исходящие топики

- `subscription.in`
- `mppw.data.[hash (requestId+subscriptionId)].deltaNum.streamNum`

##### 2.4.5.1.2 Процесс обработки запроса через Модуль MPPR

1. **Модуль группировки чанков репликации** считывает сообщение с фрагментом какой-то таблицы (в рамках какой-то дельты) из `delta.in.rq`.

2. **Модуль группировки чанков репликации** отправляет полученный фрагмент в динамический топик с именем, содержащим `[hash (requestId+subscriptionId)]`, `synId (номер дельты)` и `streamNum` - топик `mppw.data.X`
3. Если полученный фрагмент является последним ( `isLastChunk: true`), то **Модуль группировки чанков репликации** отправляет сообщение (`subscriptionId`, `synId` (номер дельты), `tableId`) в топик `subscription.in`.
4. **Модуль группировки чанков репликации** подтверждает обработку (`committing an offset`) сообщения с фрагментом в `delta.in.rq`.

### 2.4.6 CSV-Uploader

| Примечание: |
| --- |
| Модуль входит в состав конфигурации Стандарт и Лайт |

#### 2.4.6.1 Общее описание

**CSV-uploader** - программный модуль Витрины данных, который предназначен для загрузки CSV-файлов в Витрину данных.

**CSV-uploader** предназначен для следующих задач:

- загрузка CSV-файлов;
- загрузка CSV-файлов со структурой Витрины;
- выгрузка CSV-шаблонов с демо-шаблонами структуры Витрины;
- автоматический запуск загрузки CSV-файлов по расписанию из выбранного каталога;
- просмотр *Журнала операций*.

| Внимание: |
| --- |
| Загружаемые файлы обязательно должны быть в кодировке UTF-8 |

#### 2.4.6.1.1 Общая схема взаимодействия через CSV-uploader

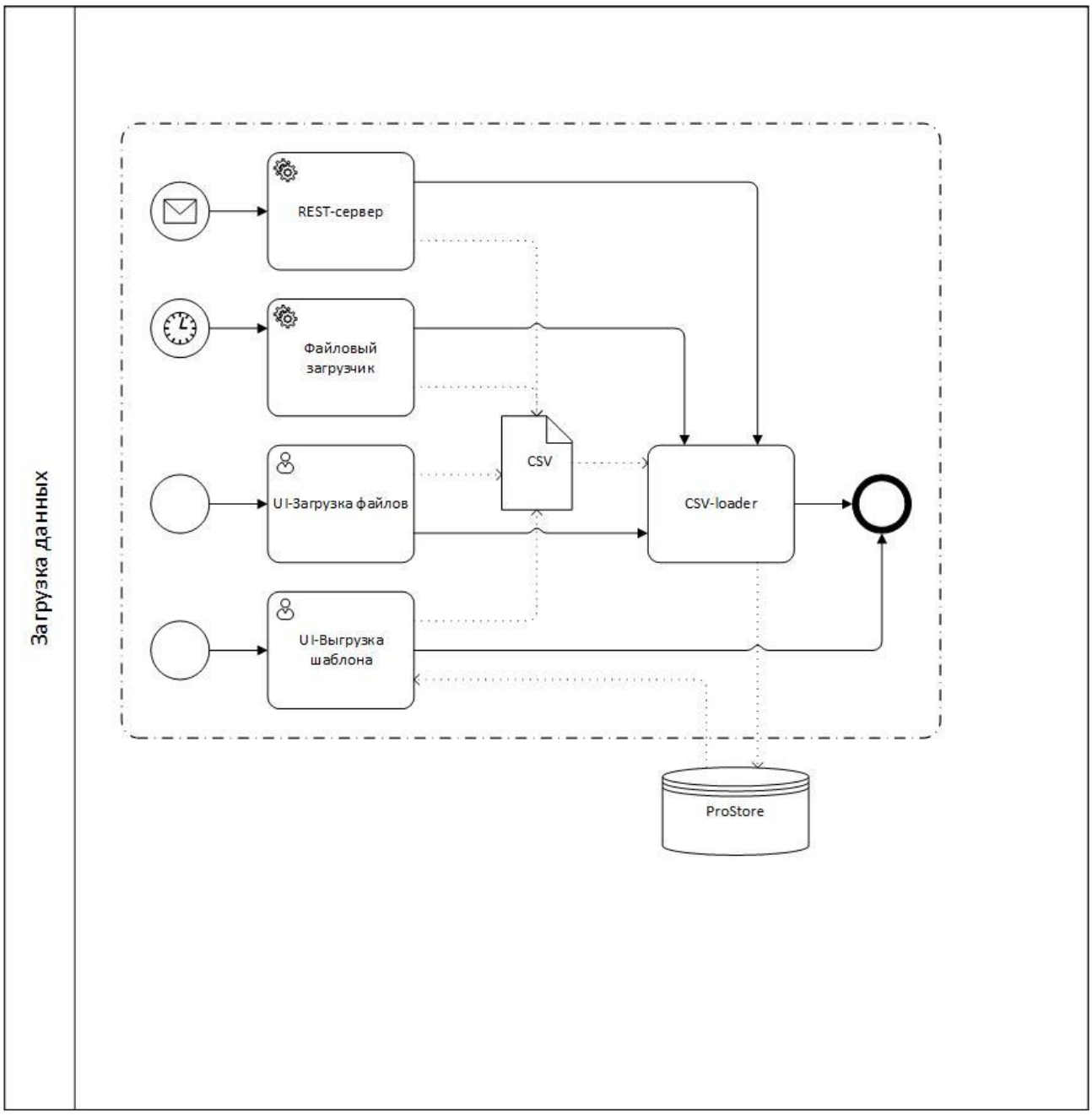


Рисунок - 2.3 Общая схема взаимодействия через CSV-uploader

## 2.4.7 СМЭВ4-адаптер - Модуль импорта табличных параметров

| Примечание: |
| --- |
| Модуль входит в состав конфигурации Стандарт и Лайт |

### 2.4.7.1 Общее описание

Модуль **СМЭВ4-адаптер - Модуль импорта данных табличных параметров** предназначен для создания временных таблиц при выполнении временных запросов с ТП и выполняет следующие задачи:

- создает временные таблицы в **Prostore** для хранения табличных параметров перед выполнением запроса;
- удаляет временные таблицы после обработки запроса.

### 2.4.8 СМЭВ4-адаптер - Модуль группировки данных табличных параметров

| Примечание: |
| --- |
| Модуль входит в состав конфигурации Стандарт и Лайт |

#### 2.4.8.1 Общее описание

Модуль **СМЭВ4-адаптер - Модуль группировки данных табличных параметров** предназначен для группировки данных при выполнении запросов с табличными параметрами, и выполняет следующие задачи:

- группирует поступающие пакеты каждого табличного параметра в отдельные топики;
- подготавливает сгруппированные пакеты данных для последующей записи в **Prostore** (с помощью **Модуля MPPW**);
- выдает команду **СМЭВ4-адаптеру - Модуль импорта данных табличных параметров** на запись данных во временные
- таблицы, созданные в **Prostore**.

Формат обмена электронными сообщениями описан в разделе Спецификация Модуля исполнения запросов Приложения 1.

### 2.4.9 СМЭВ4-адаптер - Модуль дефрагментации чанков табличных параметров

| Примечание: |
| --- |
| Модуль входит в состав конфигурации Стандарт и Лайт |

#### 2.4.9.1 Общее описание

Модуль **СМЭВ4-адаптер - Модуль дефрагментации чанков табличных параметров** преобразует пакеты с данными табличных параметров, поступающие от Агента СМЭВ4 в *брокер сообщений Kafka*, к формату, позволяющему обрабатывать их в многопоточном режиме.

Модуль дефрагментации чанков табличных параметров выполняет следующие функции:

- считывает все чанки из общего топика query.tp.bin;
- группирует их по сочетанию {requestId, subRequestId, tableParamId, streamNum};
- накапливает чанки на локальном жестком диске;
- при получении всех чанков одного стрима:
  - собирает из чанков одного стрима единый AVRO-объект;
  - десериализует единый AVRO-объект;
  - разделяет полученные строки на подмножества строк;
  - каждое подмножество строк сериализует в отдельный avro-объект;
  - полученные avro-объекты записывает в топик query.TP

### 2.4.10 DATA-Uploader - Модуль исполнения асинхронных заданий

| Примечание: |
| --- |
| Модуль входит в состав конфигурации Стандарт |

### 2.4.10.1 Общее описание

Модуль исполнения асинхронных заданий обеспечивает обработку очереди файлов, используя следующие функциональные особенности:

- обработка очереди файлов производится циклами;
- очередь файлов работает в режиме упорядочения процесса по принципу «первым пришел – первым обслужен»;
- каждый элемент в очереди файлов содержит UUID задания, имя витрины и таблицы, содержимое CSV-файла;
- файлы в очереди могут относится к разным витринам и/или разным таблицам одной витрины.

Примечание:

Заливка данных через через модуль DATA-Uploader не предусматривают параллельную заливку в датамарты вместе с другими инструментами. Параллельная заливка данных в те же датамарты вручную или средствами ETL приведет к конфликту в работе с дельтами и к ошибкам соответственно.

## 2.4.11 REST-Uploader - Модуль асинхронной загрузки данных из сторонних источников

Примечание:

Модуль входит в состав конфигурации Стандарт

### 2.4.11.1 Общее описание

Модуль асинхронной загрузки данных из сторонних источников реализован для обеспечения параллельной загрузки данных с независимым масштабированием REST интерфейса.

Обеспечена буферизация поступающих на загрузку данных. Буферизированные данные направляются в базу менеджером дельт с группировкой по датамартам.

Обеспечены следующие функциональные особенности:

- идентификатор генерируется по стандарту UUID;
- метаданные от сервера витрины кешируются механизмом, и проверяются на соответствие по количеству и по типам полей (при несоответствии загружаемых данных метаданным целевой таблицы сервис для передачи / загрузки данных возвращает статус запроса с ошибкой, без размещения данных в очереди на загрузку);
- загруженные данные размещаются вместе с UUID в очереди с именем «queue»;
- формируется запись с ключом «status.[UUID запроса]» и статусом 0 в очереди;
- клиенту, отправившему запрос, возвращается успешный статус запроса вместе с UUID;
- в логе приложения формироваться запись события получения запроса на загрузку с указанием идентификатора запроса, идентификатора ВУЗа, времени обработки и размера загруженных данных.

Внимание:

Загружаемые файлы обязательно должны быть в кодировке UTF-8

Примечание:

Заливка данных через через модуль REST-Uploader не предусматривают параллельную заливку в

датамарты вместе с другими инструментами. Параллельная заливка данных в те же датамарты вручную или средствами ETL приведет к конфликту в работе с дельтами и к ошибкам соответственно.

Общие правила формата загружаемых CSV-файлов приведены в таблице Таблица 2.6.

Таблица 2.6 Общие правила формата загружаемых CSV-файлов

| **Параметр** | **Значение** |
|---|---|
| Разделитель строк | Любой вариант из: CR/LF (0x0D0A), CR (0x0D), LF (0x0A) |
| Разделитель полей | по настройке csv-parser/separator (Параметры конфигурации) |
| Строка заголовка | да (обязательно) |
| Порядок полей в строке | определяется строкой заголовка |
| Ограничитель текстового поля | по настройке csv-parser/quote-char (Параметры конфигурации) |
| Символ маскировки в текстовом поле | по настройке csv-parser/escape-char (Параметры конфигурации) |
| Обнаружение значения null | До релиза 1.5.0 (включительно): по настройке csv-parser/field-as-null (Параметры конфигурации)<br>начиная с релиза 1.10.0: в текущей реализации парсера данная настройка не поддерживается |
| Кодирование символов | UTF-8 |
| Десятичный разделитель | символ `.` (0x2E), может не указываться для целых значений |
| Формат даты | любой из: `dd.MM.yyyy`, `yyyy-MM-dd` |
| Формат времени | любой из: `HH:mm:ss`, `H:mm:ss` |
| Формат даты-времени | до релиза 1.5.0(включительно) любой из: `yyyy-MM-dd HH:mm:ss`, `dd.MM.yyyy HH:mm:ss`<br>начиная с релиза 1.10.0 любой из: `yyyy-MM-dd HH:mm:ss.000000`, `dd.MM.yyyy HH:mm:ss.000000` |

### 2.4.11.2 Проверка форматно-логического контроля

Проверка форматно-логического контроля включает в себя:

- обязательные проверки, выполняющиеся вне зависимости от настроек модуля в синхронном режиме;
- необязательные проверки, индивидуальные для каждой таблицы, которыми управляет администратор Системы, выполняющиеся в асинхронном режиме.

Таблица 2.7 Список реализованных проверок

| **Наименование проверки** | **Код ошибки** | **Кирилическое описание** |
|---|---|---|
| Проверка уникальности | `dublicate` | Дубликат файла/группы |
| Проверка парсинга файла | `parsingErr` | Ошибка парсинга: *текст ошибки* |
| Проверка кодирования | `encodingErr` | Кодировка файла не соответствует кодировке UTF-8 |
| Проверка превышения предельного размера файла (больше 512 Мб) | `tooLargeFile` | Слишком большой файл |
| Проверка наличия данных в файле | `emptyFile` | Пустой файл |
| Проверка соответствия | `wrongMetadata` | Структура файла не соответствует схеме |

| Наименование проверки | Код ошибки | Кирилическое описание |
|---|---|---|
| заголовков инфосхеме | | |
| Проверка соответствия числа столбцов в строке | `wrongFieldsCount` | Некорректное число столбцов в строке |
| Проверка соответствия типам полей | `wrongFieldType` | Значение не соответствует типу *требуемый тип* |
| Проверка уникальности полей | `nonUniq` | Значение не отвечает требованиям уникальности |
| Проверка регулярных выражений | `nonMatchRegex` | Значение не соответствует регулярному выражению *регулярное выражение* |
| Проверка соответствия условию | `nonMatchConstant` | Значение не соответствует условию *условие* |
| Таймаут валидации | `validationTimeout` | Истек таймаут валидации файла |

#### 2.4.11.2.1 Синхронная проверка ФЛК

Примечание:

- синхронные проверки выполняются вне зависимости от настроек модуля REST-Uploader;
- синхронные проверки являются блокирующими;
- ошибки синхронных проверок возвращаются в теле ответа по REST-API.

К синхронным проверкам относятся:

- проверка соответствия инфосхеме:
  - проверка соответствия имен и количества полей в заголовках;
  - проверка типа данных;
  - проверка экранирования данных: проверка соответствия числа столбцов по каждой строке;
- проверка соответствия файла кодировке UTF-8 , отсутствие BOM (при наличии BOM при загрузке удаляются начальные байты `ef bb bf`);
- проверка размера файла и наличия данных:
  - проверка предельного размера загружаемого файла 512Мб;
  - проверка наличия данных в файле.

#### 2.4.11.2.2 Асинхронная проверка

Примечание:

- асинхронные проверки выполняются в зависимости от настроек модуля;
- проверки не являются блокирующими (поведение при их наличии определяется конфигурацией модуля);
- список проверок уникален для каждой таблицы и хранится в Zookeeper в виде отдельного YAML файла.

К асинхронным проверкам относятся:

- проверка уникальности полей:
  - по сочетанию атрибутов (для комплексных ключей);
  - по заданному атрибуту;
- сравнение значения с константой;
- соответствие регулярному выражению.

Для одного поля возможно создать не более одной проверки одного типа, при этом у каждого поля может быть несколько проверок разных типов.

#### 2.4.11.2.2.1 Проверка уникальности по одному или по сочетанию полей

Проверка уникальности проводится:

- в рамках группы файлов, если заданы **headers** - для проверки в рамках группы обязательно заполнения всех полей:
  - `group_id`;
  - `group_file_num`;
  - `group_file_count`.
- при проверке в рамках группы значения ключей для проверки уникальности в рамках группы хранится в **Redis**:
  - по всем файлам в рамках группы при наличии групповых атрибутов
  - по одному файлу, при отсутствии групповых атрибутов

Пример запроса для проверки уникальности по группе файлов:

```
--проверка по сочетанию полей
fields:
  id:
    uniq: true
    uniq-with: [type,region]
--проверка уникальности по одному полю
  snils:
    match: "/^[-¥s¥d]{11}$/"
    uniq: true
```

#### 2.4.11.2.2.2 Проверка соответствия заданному значению

- проверка соответствия заданному значению проводится для каждого файла вне зависимости от наличия group_id;
- проверка осуществляется для значений каждого поля в соответствии с заданным правилом;
- проверка соответствия заданному значению включает в себя:
-
  - проверку сравнения с константой (`>`, `<`, `>=`, `<=`, `=`, `!=`);
  - проверку соответствия регулярному выражению (должна выполняться на основе Java Util Regexp https://docs.oracle.com/javase/7/docs/api/java/util/regex/package-summary.html )

#### 2.4.11.2.2.3 Поведение в случае таймаута валидации

Период выполнения асинхронных проверок определяется конфигурационным параметром `validation-timeout` и по умолчанию составляет 60 минут.

В случае, если за указанное в настройках время асинхронные проверки не были выполнены, файл удаляется из очереди с обогащением отчета о найденных ошибках ошибкой `validationTimeout`.

В случае возникновения подобной ошибки рекомендуется:

- проверить регулярные выражения, по которым происходит проверка, так как неверно заданное регулярное выражение кратно увеличивает скорость проверки;
- увеличить значение `validation-timeout` и повторить загрузку данных.

### 2.4.11.3 Статусная модель

Статус запроса к модулю REST-Uploder можно получить, выполнив запрос с передачей идентификатора запроса `GET '/v2/requests/:requestId/status'`

В ответ возвращается три поля:

– code - код статуса;
– errorMessage - сообщение об ошибке, заполняется лишь в случае ошибочного статуса;
– description - описание ошибки, заполняется лишь в случае ошибочного статуса.

Пример ответа на запрос статуса


```
{"code":2,"description":null,"errorMessage":null}
{"code":3,"description":"Успешно обработан","errorMessage":null}
{"code":4,"description":"Ошибка обработки
запроса","errorMessage":"ru.itone.dtm.data.uploader.upload.UploadException:
ru.itone.dtm.prostore.rest.api.ProstoreRestException: Error executing query [insert
into univer.slots select
resource_id,slot_id,tag_type,tag_age,available_date,duration,slot_create_ts,slot_update
_ts,slot_status,sys_op from univer.slots_upload_ext]: All of the connectors are
failed¥n¥tat"}
{"code":5,"description":"Идентификатор запроса не обнаружен","errorMessage":null}
{"code":7,"description":"Ошибки ФЛК","errorMessage":null}
```


В свою очередь, идентификатор запроса ранее был получен в ответ от REST-Uploader:
– POST“/v2/datamarts/{datamart_name}/tables/{table_name}/upload;
– POST“/v2/datamarts/{datamart_name}/tables/{table_name}/delete.

Таблица 2.8 Статусная модель

| **Код статуса** | **Описание статуса** | **Действия при получении данного статуса** |
|---|---|---|
| -1 | Загрузка данных в буффер | Данный статус клиентскому приложению не возвращается, ответ вернется после того как загружаемые данные будут сохранены приложением REST-Uploader для дальнейшей загрузки. |
| 0 | Запрос бufферизирован | Выполнить повторный запрос статуса с некоторой задержкой. Рекомендуемая задержка 30сек. |
| 1 | Ожидает открытия дельты | Выполнить повторный запрос статуса с некоторой задержкой. Рекомендуемая задержка 30сек. |
| 2 | В обработке (модулем DATA-Uploader) | Выполнить повторный запрос статуса с некоторой задержкой. Рекомендуемая задержка 30сек. |
| 3 | Успешно обработан | Дополнительных действий не требуется |
| 4 | Ошибка обработки запроса | Необходимо:<br>– изучить содержимое вернувшегося поля description<br>– если суть проблемы не ясна из предыдущего пункта, изучить содержимое логов приложений на предмет наличия ошибок:<br>○ REST-Uploader<br>○ DATA-Uploader<br>○ podd-adapter-mppw<br>○ query-execution-core<br>○ используемого коннектора (kafka-postgres-writer или kafka-jet-writer) |
| 5 | Идентификатор запроса не обнаружен | Использовать действующий идентификатор запроса |
| 6 | Форматно-логический контроль | Выполнить повторный запрос статуса с некоторой задержкой. Рекомендуемая задержка 30сек. |
| 7 | Ошибки ФЛК | В процессе ФЛК выявлены ошибки, необходимо запросить отчет ФЛК, обратившись к REST-Uploader с запросом `GET '/v2/requests/{request_id}/report/'`<br>Далее проанализировать и устранить выявленные недочеты в загружаемых данных или скорректировать проверки ФЛК. |

### 2.4.12 BLOB-адаптер

| Примечание: |
| --- |
| Модуль входит в состав конфигурации Стандарт |

#### 2.4.12.1 Общее описание

**BLOB-адаптер**, программный модуль Витрины данных, предназначен для получения доступа из Витрины данных к BLOB-объектам ведомства.

**BLOB-объект** - это специальный тип двоичных данных, предназначенный для хранения бинарных файлов: изображений, скан-копий документов, текстовых файлов и т.д.

**BLOB-адаптер** предоставляет возможность настроить доступ к BLOB-объектам расположенным в **Хранилище BLOB-объектов**.

| Примечание: |
| --- |
| Хранилище BLOB-объектов располагается на стороне ведомства и не является частью Витрины данных. |

BLOB-адаптер предназначен для следующих задач:

- настройка доступа в **Хранилище BLOB-объектов**;
- предоставление регламентированного доступа к BLOB-объектам;
- получение и отправка запросов на получение BLOB-объектов;
- чтение BLOB-объектов;
- сохранение BLOB-объектов на FTP-сервере (через СМЭВ3-адаптер).

Взаимодействие с BLOB-объектами возможно через запросы из **СМЭВ3** (через СМЭВ3-адаптер) или **СМЭВ** (через СМЭВ4-адаптер). Доступ к считыванию BLOB-объектов производится методом `GET` по протоколу **HTTP/HTTPS** (указывается в конфигурации, параметр `host`) .

Формат обмена электронными сообщениями через **BLOB-адаптер** описан в разделе Спецификация модуля «BLOB-адаптер» Приложения 1.

### 2.4.12.2 Общая схема взаимодействия через BLOB-адаптер

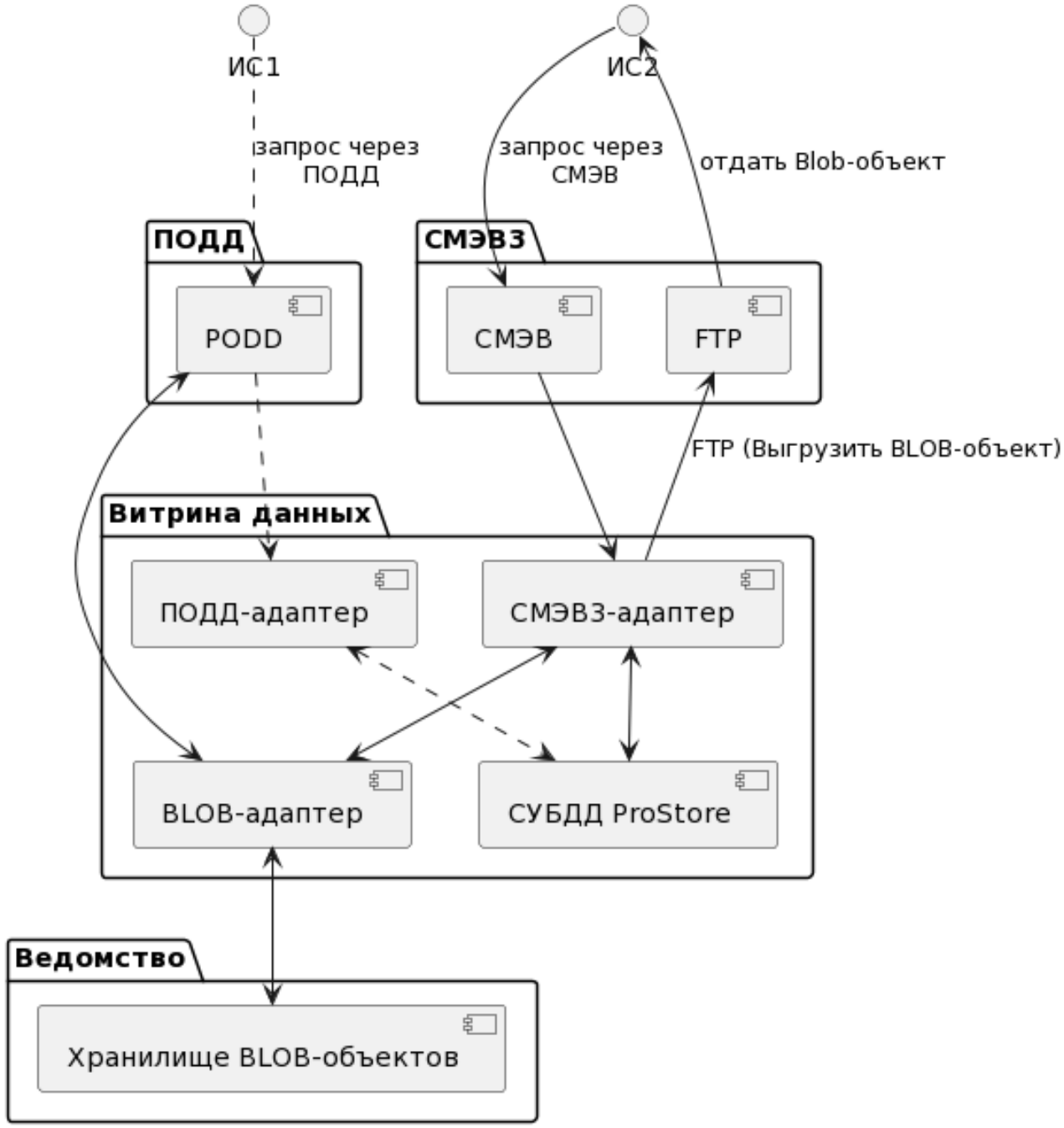


Рисунок - 2.4 Общая схема взаимодействия через BLOB-адаптер

### 2.4.12.3 Взаимодействие через СМЭВ-адаптер

#### 2.4.12.3.1 Схема взаимодействия через СМЭВ-адаптер

Общая схема получения BLOB-объектов через **СМЭВ-адаптер** выглядит следующим образом:

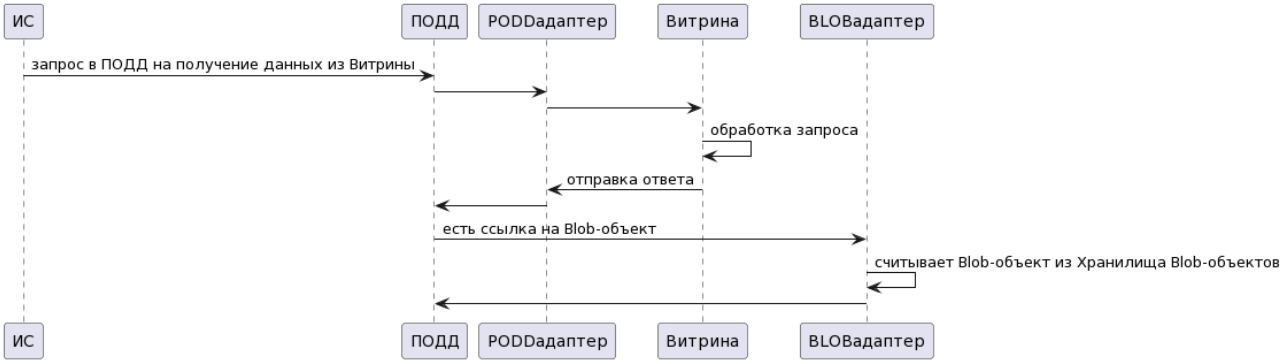


Рисунок - 2.5 Взаимодействие BLOB-адаптера через СМЭВ4-адаптер

**2.4.12.3.2 Процесс обработки запроса на получение BLOB-объекта (СМЭВ4-адаптер)**

1. В **СМЭВ4** поступает запрос на получение данных из Витрины.
2. **СМЭВ4** отправляет запрос через **СМЭВ4-адаптер** в Витрину.
3. Витрина данных обрабатывает запрос.
4. **СМЭВ4-адаптер** считывает данные полученные от Витрины и отправляет ответ в Kafka, на стороне **СМЭВ4**. В случае, если в теле запроса содержится ссылка на **BLOB-объект** (например, изображение) Kafka, на стороне **СМЭВ4** , отправляет запрос в **BLOB-адаптер** на получение этого **BLOB-объекта**. BLOB-адаптер, считывает ссылку на **BLOB-объект** и обращается в Хранилище **BLOB-объектов** на стороне ведомства. После получения **BLOB-объекта**, возвращает его в **СМЭВ4**.

## 2.4.12.4 Взаимодействие через СМЭВ3-адаптер

### 2.4.12.4.1 Схема взаимодействия через СМЭВ3-адаптер

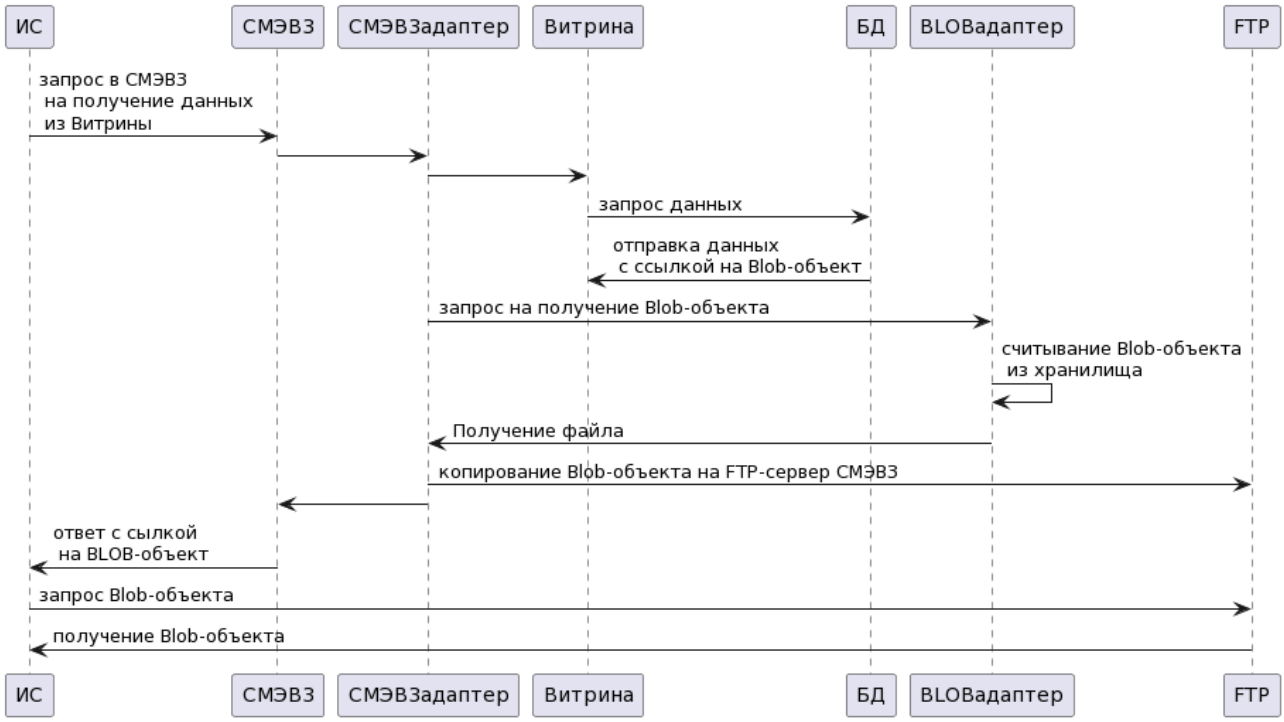


Рисунок - 2.6 Взаимодействие BLOB-адаптера через СМЭВ4

**2.4.12.4.2 Процесс обработки запроса на получение BLOB-объекта (через СМЭВ3-адаптер)**

1. Из внешней ИС в **СМЭВ4** поступает запрос на получение данных из Витрины.
2. **СМЭВ3-адаптер** получает запрос и отправляет его в Витрину.
3. Витрина обращается к БД, подготавливает ответ на запрос. Если в запросе есть ссылка на **BLOB-объект**, **BLOB-адаптер** обращается к **Хранилищу BLOB-объектов**, копирует **BLOB-объект**, выкладывает его на FTP-сервер СМЭВ4 и отправляет ссылку на **BLOB-объект** в Витрину.
4. После того как **BLOB-объект** загружен на FTP-сервер **СМЭВ4**, Витрина отправляет ответ на запрос в **СМЭВ4**, в котором содержится ссылка на **BLOB-объект** (сохраненный на FTP-сервере).
5. После получения ответа внешняя ИС может обратиться по ссылке и скачать **BLOB-объект** с FTP-сервера СМЭВ4.

### 2.4.12.5 Требования к серверу BLOB-адаптера

1. Следующие файлы не должны контролироваться системой управления конфигурации (при ее наличии), поскольку они должны быть доступны процессу установки для создания и модификации:
   - `/etc/hosts`
   - `/etc/selinux/config`
   - `/etc/sysctl.conf`
   - файлы директории `/usr/lib/systemd/system/`
   - `/etc/sysconfig/iptables¥*`
   - `/etc/firewalld/¥*`
   - `/etc/docker/¥*`
1. Снаружи сервер должен быть доступен по следующим портам:
   - 22 (SSH).

### 2.4.12.6 Требования к Хранилищу BLOB-объектов

**Хранилище BLOB-объектов** должно поддерживать регламентированный витриной интерфейс доступа к BLOB-объектам по запросу (см. раздел Спецификация модуля «BLOB-адаптер» Приложения 1).

Для корректной работы с Хранилищем BLOB-объектов, необходимо выполнить следующие условия:

- Предоставить доступ:
  - к таблицам с метаданными по документам, хранилища ведомства;
  - к ссылкам на BLOB-объекты (изображения, архивы, pdf-файлы и т.д.) в хранилище ведомства, соответствующие метаданным.
- Предоставить связь между ссылками и метаданными, если они разнесены по разным хранилищам.

| Примечание: |
| --- |
| Все обновления ссылок на документы происходят только через ETL, т.е. от Поставщика данных (ведомства) к Витрине. Обновление самих BLOB через витрину, с последующей проливкой в ведомства, не предусмотрено. |

#### 2.4.12.7 Требования к предоставляемому интерфейсу Хранилища BLOB-объектов (API-интерфейс)

**API-интерфейс** реализованный соответственно данной спецификации должен предоставляться Хранилищем BLOB-объектов для возможности успешного взаимодействия с Витриной.

**API-интерфейс** Хранилища BLOB-объектов должен использовать метод `GET`, который поддерживается Витриной для получения тела BLOB-объекта из Хранилища BLOB-объектов.

С помощью **API-интерфейса** отправляется запрос на получение BLOB-объектов.

Считывание BLOB-объекта производится по протоколу HTTP (допустимо HTTPS, указывается в конфигурации файла `application.yml`, параметр `host` (см. раздел Конфигурация BLOB-адаптера (application.yml) в документе «Руководство администратора ПО «Витрина данных НСУД»)).

### 2.4.13 Сервис формирования документов

| Примечание: |
| --- |
| Модуль входит в состав конфигурации Стандарт |

#### 2.4.13.1 Общее описание

**Сервис Формирования документов** предназначен для обеспечения возможности формирования документов, в формате XML и PDF, на основе предварительно подготовленных pebble-шаблонов, с возможностью добавления к сформированным документам электронной подписи.

| Внимание: |
| --- |
| Максимальный размер формируемого отчета не должен превышать 2Гб! |

**Сервис Формирования документов** предназначен для решения следующих задач:

- формировать документ, на основе поступившего в Витрину запроса, в формате:
  - XML;
  - PDF.
- отправлять сформированные документы на подпись в сервис **Notarius**;
- отправлять сформированные и подписанные документы в **СМЭВ4** в виде ответа на пришедший запрос.

Формат обмена электронными сообщениями описан в разделе Спецификация модуля «Сервис Формирования документов» Приложения 1.

##### 2.4.13.1.1 Процесс обработки запроса через модуль «Сервис Формирования документов»

1. Запрос на предоставление сформированного документа поступает через Агента СМЭВ4, в топик procedure.query.rq.
2. СМЭВ4-адаптер считывает запрос из топика и передает его в Сервис Формирования документов. Сервис Формирования документов запускает соответствующий пришедшему типу документа pebble-шаблон, собирает данные из Витрины (Prostore) и формирует на основании этих данных JSON-файл.
3. Из содержащейся в JSON-файле информации формируется итоговая форма документа. Для этого используется Шаблон generate_xml.peb и Шаблон generate_pdf.peb, предназначенные для генерации документов.

4. В случае, если электронная подпись не требуется, то PDF-документ сразу пересылается в топик report.rs.
5. Если требуется электронная подпись:
   – PDF-документ и публичная часть SSH-ключа (pub) будут отправлены в Сервис электронной подписи. Сервис электронной подписи сформирует для этого PDF-документа файл подписи (p7s) и вернет его в Сервис Формирования документов.
   – Полученный XML или PDF-файл Сервис Формирования документов отправляет в **СМЭВ4-адаптер**, в топик report.rs **Kafka СМЭВ4-адаптер**.
6. **Агент СМЭВ4** проверяет топик и забирает сформированные документы для передачи в **СМЭВ4**.

| Внимание: |
| --- |
| Следует обратить внимание, что при формировании XML-документов используется - *присоединенная подпись* (подпись содержится в самом XML-документе).<br>Для PDF-документов - *отсоединенная подпись* (подпись документа формируется отдельным файлом), т.е. при формировании PDF-документа сгенерируется два файла: PDF-документ и файл электронной подписи для этого документа. |

#### 2.4.13.1.2 Компоненты

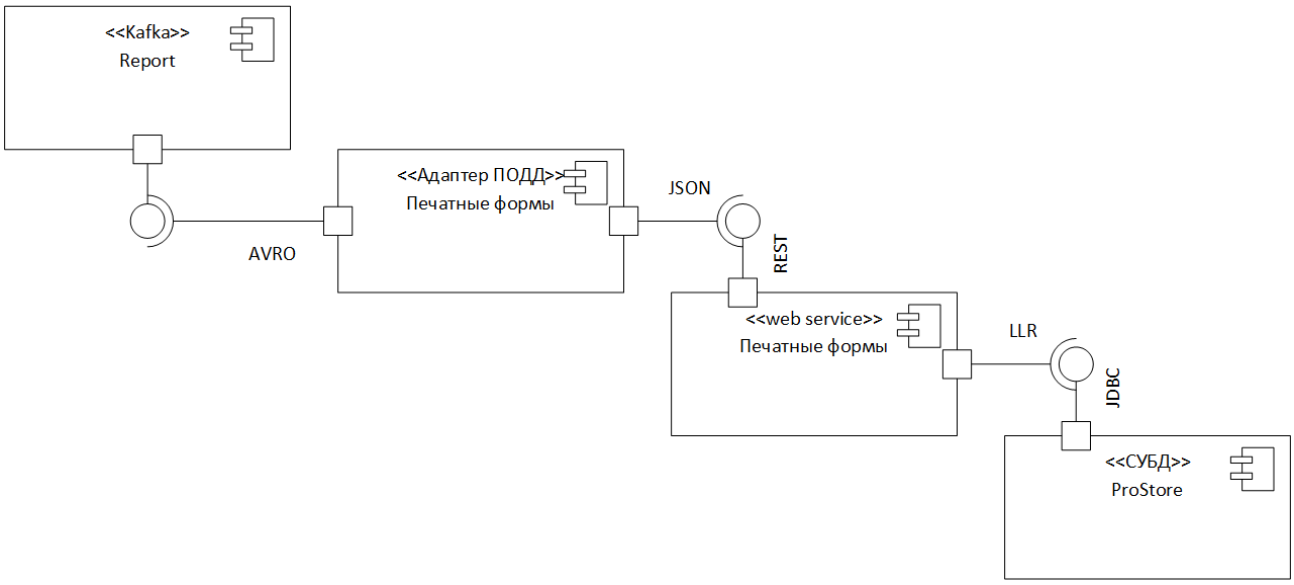


Рисунок - 2.7 Компоненты модуля «Сервис Формирования документов»

### 2.4.13.1.3 Общая архитектура решения

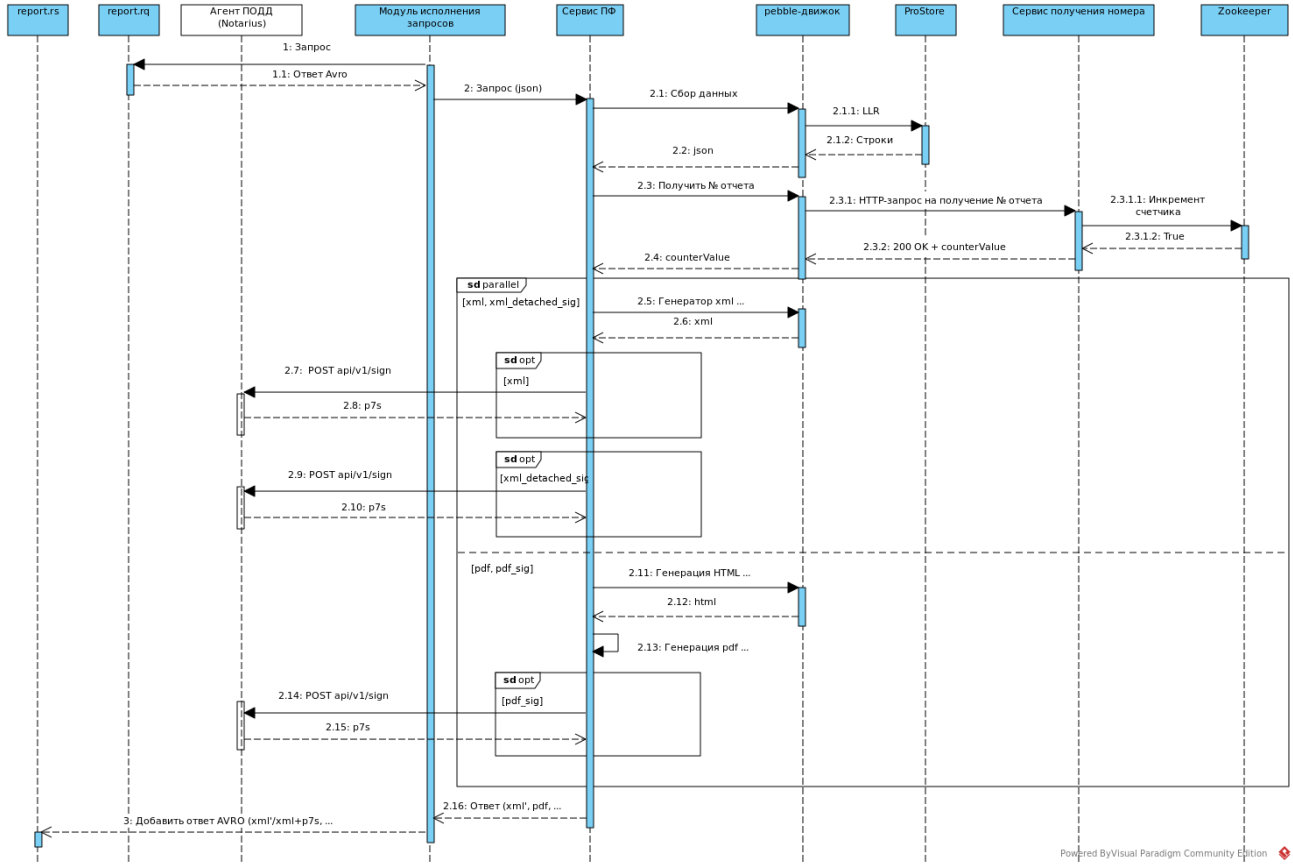


Рисунок - 2.8 Общая схема взаимодействия модуля «Сервис Формирования документов»

### 2.4.13.1.4 Назначение параметров QueryRequest.parameters[]

Таблица 2.9 Назначение параметров QueryRequest.parameters[]

| **Индекс параметра в QueryRequest.parameters[]** | **Тип QueryRequest.parameters[].type** | **Значение QueryRequest.parameters[].value** |
|---|---|---|
| 0 | string | публичная часть сертификата (формат PKCS#7) в кодировке BASE64 |
| 1 | string | Список запрошенных форматов ПФ (через запятую, без учета регистра).<br>Допустимые значения (без кавычек):<br>– «XML» - файл xml;<br>– «PDF» - pdf без ЭП (содержимое «штампика об ЭП» - на усмотрение разработчика шаблона для pdf-файла);<br>«PDF_SIG» - pdf с откреплённой ЭП (содержимое «штампика об ЭП» - на ответственности разработчика шаблона для pdf-файла). |
| 2 | | параметры для формирования запрашиваемой ПФ, зависят от логики формирования ПФ. |

#### 2.4.13.1.5 Примеры шаблонов

##### 2.4.13.1.5.1 Шаблон extract_data.peb

```
{#формируем sql запрос в переменную passengersquery#}
{% var passengersquery %}
    {% if _0 is empty %}
        select * from auto_db.passenger limit 10
    {% else %}
        select * from auto_db.passenger limit {{ _0 }}
    {% endif %}
{% endvar %}
{# выполняем sql запрос и помещаем результат выполнения в переменную
rows.searchpassenger #}
{{ sql("searchpassenger", passengersquery) }}

{% var json_data %}
{
    "passengers": [
    {% for p in rows.searchpassenger %}
    {#  формируем json динамически  #}
        {% if loop.first %}
        {% else %}
            ,
        {% endif %}
        {
            "id": "{{ p.id }}",
            "firstname": "{{ p.firstname }}",
            "middlename": "{{ p.middlename }}",
            "lastname": "{{ p.lastname }}",
            "birthday": "{{ p.birthday }}"
        }
    {% endfor %}
    ]
}
{% endvar %}

{#выведем полученный json в неэкранированной форме#}
{{ json_data | raw }}
```

##### 2.4.13.1.5.2 Шаблон generate_xml.peb

```
{#соберем xml документ#}
<passengers>
{% for p in _0.passengers %}
    <passenger id="{{ p.id }}">
        <firstname>{{ p.firstname }}</firstname>
        <middlename>{{ p.middlename }}</middlename>
        <lastname>{{ p.lastname }}</lastname>
        <birthday>{{ p.birthday }}</birthday>
    </passenger>
{% endfor %}
</passengers>
```

##### 2.4.13.1.5.3 Шаблон generate_pdf.peb

```
{#соберем html документ#}
<html>
    <head>
    <style>
        table, th, td {
            border: 1px solid black;
        }
```

```
    </style>
</head>
    <body>
        <h3>Passengers</h3>
        <table>
            <tr>
                <th>id</th>
                <th>firstname</th>
                <th>middlename</th>
                <th>lastname</th>
                <th>birthday</th>
            </tr>
{% for p in _0.passengers %}
            <tr>
                <td>{{ p.id }}</td>
                <td>{{ p.firstname }}</td>
                <td>{{ p.middlename }}</td>
                <td>{{ p.lastname }}</td>
                <td>{{ p.birthday }}</td>
            </tr>
{% endfor %}
        </table>
    <body>
</html>
```

#### 2.4.13.1.6 REST запрос к сервису

##### 2.4.13.1.6.1 Запрос

**Endpoint:** /report

**Request type:** GET

**Headers:**

- Content-type: application/json

**content schema**

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "title": "docGenRequest",
  "type": "object",
  "properties": {
    "docName": {
      "type": "string"
    },
    "customerId": {
      "type": [
      "null",
      "string"]
    },
    "customerOgrn": {
      "type": [
      "null",
      "string"]
    },
    "queryMnemonic": {
      "type": [
      "null",
      "string"]
    },
    "parameters": {
      "type": "array",
      "items": {
        "type": "object",
```

```
        "properties": {
        "name": {
            "type": "string"
        },
        "value": {
            "type": [
            "string",
            "number",
            "boolean",
            "null"
            ]
        },
        "mandatory": {
            "type": [
            "boolean",
            "null"
            ]
        },
        "regExp": {
            "type": [
            "string",
            "null"
            ]
        }
      },
      "required": [
        "name",
        "value"
        ]
      }
    }
  },
  "required": [
    "docName",
    "parameters"
  ]
}
```

#### 2.4.13.1.6.2 Ответ

Ответ должен содержать заголовок с HTTP-кодом ответа и n частей:

1. Ответ на запрос:
   **Код ошибки:** (200 OK)
   **Content-type:** multipart/mixed
2. Часть, содержащая сгенерированный XML
   **Content-Disposition:** attachment; name=»xml»; filename=»result.xml»
   **Content-type:** application/xml
3. Часть, содержащая metadata для XML
   **Content-Disposition:** attachment; name=»xml»; filename=»metadata»
   **Content-type:** text/plain; charset=utf-8
4. Часть, содержащая сгенерированный PDF
   **Content-Disposition:** attachment; name=»pdf»; filename=»result.pdf»
   **Content-type:** application/pdf

## 2.4.14 СМЭВ QL Сервер

| Примечание: |
| --- |
| Модуль входит в состав конфигурации Стандарт |

### 2.4.14.1 Назначение СМЭВ QL сервера

#### 2.4.14.1.1 О системе

СМЭВ QL сервер - это компонент взаимодействия с витриной данных, который реализует её типовое API согласно внутренней спецификации СМЭВ QL.

Основным назначением СМЭВ QL сервера является обработка REST-запросов на получение данных от Потребителей и формирование оптимальных (простых) SQL-запросов к витрине для получения запрашиваемых данных.

Для Потребителя взаимодействие со СМЭВ QL сервера равнозначно виду информационного обмена - Обмен с использованием регламентированных запросов типа «Rest-сервис».

Основной принцип работы СМЭВ QL сервера заключается в следующем:

1. СМЭВ QL принимает на вход REST-запрос, который содержит перечень запрашиваемых ресурсов, условий выбора данных, требуемые атрибуты ответа и пр. После чего определяет данные каких объектов и из каких источников необходимо извлечь. При этом определение источников происходит на основании заранее описанных моделей данных, которые хранятся на сервере в файлах вида `model.yaml` (что извлекать) и `source.yaml` (откуда извлекать).
2. Далее формирует, в определенной последовательности (на основании плана выполнения запросов), столько SQL-запросов к источникам данных, сколько ресурсов было запрошено в исходном запросе от клиента. При этом запросы могут выполняться как последовательно, в случае если в запросе ресурсы имеют вложенность (иерархию) или параллельно, если ресурсы указаны на одном уровне. Это позволяет значительно упростить sql-выражения и уменьшить сложность запроса к БД.
3. Обрабатывает результаты выполнения SQL-запросов по мере их поступления от источников.
4. Затем формирует и передает комплексный ответ клиенту.

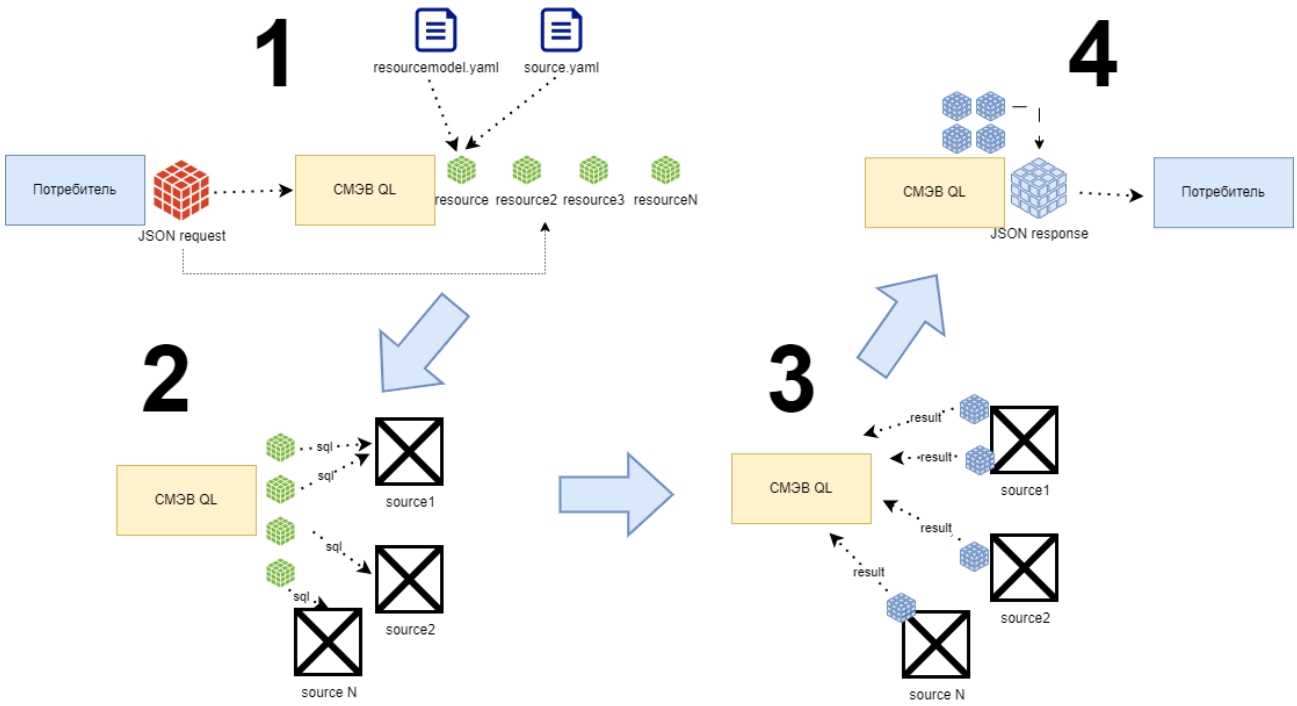


Рисунок - 2.9 Схема СМЭВ QL Сервера

#### 2.4.14.1.2 Цели СМЭВ QL сервера

Целями создания СМЭВ QL сервера являются:

1. Повышение скорости предоставления ответов от витрины данных Поставщика по сравнению с типовым видом взаимодействия Агент-Витрина.
2. Защита витрины данных Поставщика от неоптимальных запросов.
3. Сокращение объёма ответов.
4. Сокращение количества передаваемых запросов от Потребителей.
5. Повышение скорости развития услуг ЕПГУ.

#### 2.4.14.1.3 Задачи СМЭВ QL сервера

Основные задачи СМЭВ QL сервера:

1. Формирование API и модели данных витрины .
2. Приём REST-запросов от Потребителей через Агент СМЭВ4.
3. Формирование простых SQL-запросов к витрине.
4. Формирование распределенных запросов к нескольким витринам.
5. Формирование и передача ответа Потребителю.
6. Проверка и формирование цифровых подписей ответов.
7. Описание и исполнение при вызове модели машины состояний.
8. Нотификация подписчиков при изменении данных витрины.
9. Предоставление внешним клиентам OpenAPI для управления.

#### 2.4.14.1.4 Место СМЭВ QL сервера в ИТ-ландшафте

СМЭВ QL сервер взаимодействует со следующими компонентами СМЭВ4:

1. Агент СМЭВ4.
2. Сервис исполнения запросов ядра витрины данных.
3. Сервер криптографии (Notarius).
4. Сервис формирования документов.

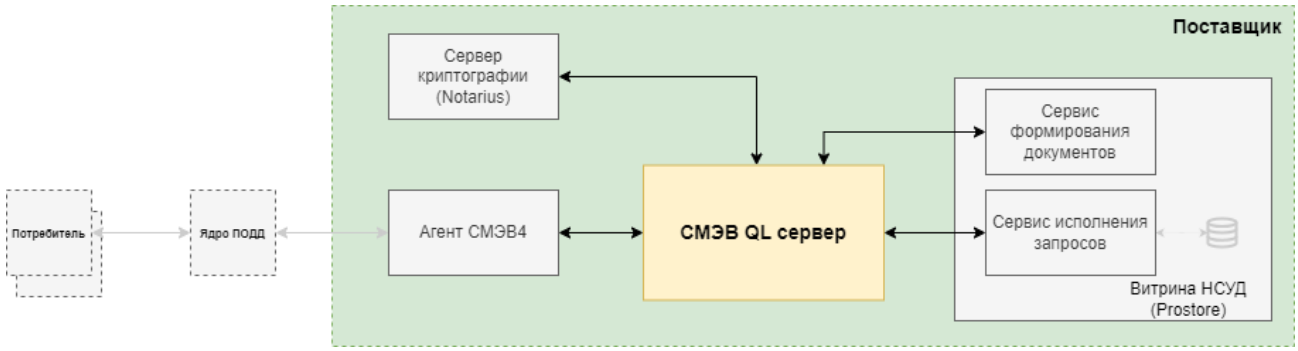


Рисунок - 2.10 Схема взаимодействия СМЭВ QL Сервера с компонентами СМЭВ4

#### 2.4.14.1.5 Язык и синтаксис

##### 2.4.14.1.5.1 Моделирование

Для моделирования документного слоя данных в спецификации выбран язык разметки YAML.

##### 2.4.14.1.5.2 Запросы и ответы

Для написания запросов, а также в качестве сериализатора ответов, спецификация определяет использование JSON.

#### 2.4.14.1.6 Типизация

Фактические типы данных наследуют типы данных JSON (включая NULL):

- string;
- number;
- object;
- array;
- boolean;
- null.

##### 2.4.14.1.6.1 Типы данных в модели и приведение типов

В описании модели допускается указание фактического типа данных атрибута ресурса **вторым элементом массива type**. Указание является опциональным, по умолчанию подразумевается неограниченный **STRING**.

Пример из описания модели:

```
fields:
  id:
    name: Идентификатор записи
    type:
      - number
      - SHORT
    length: 20
    nullable: not NULL
    key: PRIMARY
```

В качестве второго уточняющего типа следует применять типы НСУД:

- STRING;
- DOUBLE;
- FLOAT;
- BOOLEAN;
- BYTE (не поддерживается на витрине);

- BINARY;
- BIG_DECIMAL (не поддерживается на витрине);
- LONG;
- INTEGER;
- SHORT;
- DATE;
- TIME;
- TIMESTAMP.

#### 2.4.14.1.7 Моделирование данных

Модели данных описываются в формате `YAML` в папке проекта `models` согласно спецификации СМЭВ QL.

Структура базовой модели приведена в Базовая модель данных.

Структура базовой модели приведена в Модель данных.

> Примечание:
>
> Заливка данных через через модуль RESt-Uploader и DATA-Uploader не предусматривают параллельную заливку в датамарты вместе с другими инструментами. Параллельная заливка данных в те же датамарты вручную или средствами ETL приведет к конфликту в работе с дельтами и к ошибкам соответственно.

#### 2.4.14.1.8 Метрики

Для обеспечения возможности сбора информации о работе СМЭВ QL Сервера реализован набор метрик, обеспечивающий формирование показателей:

- время исполнения входящих запросов;
- количество успешных / не успешных выполнений входящих запросов;
- время исполнения исходящих запросов или обращений к СПО;
- количество успешных / не успешных выполнений исходящих запросов или обращений к СПО.

## 2.4.15 СМЭВ3-адаптер

> Примечание:
>
> Модуль входит в состав конфигурации Стандарт

### 2.4.15.1 Общее описание

Модуль **СМЭВ3-адаптер** обеспечивает информационное взаимодействие через единый электронный сервис единой системы межведомственного электронного взаимодействия (далее – **СМЭВ**).

С помощью **СМЭВ3-адаптер** Витрина данных выступает участником взаимодействия в роли Поставщика данных, а именно:

- получает запросы из очереди СМЭВ;
- отправляет ответы на запросы из очереди СМЭВ;
- формирует и отправляет уведомления в СМЭВ об изменении данных в экземпляре Витрины данных.

> Внимание:
>
> Отправляемые через СМЭВ3-адаптер файлы, не должны быть нулевыми (не содержать никаких данных)!

### 2.4.15.2 Схема взаимодействия

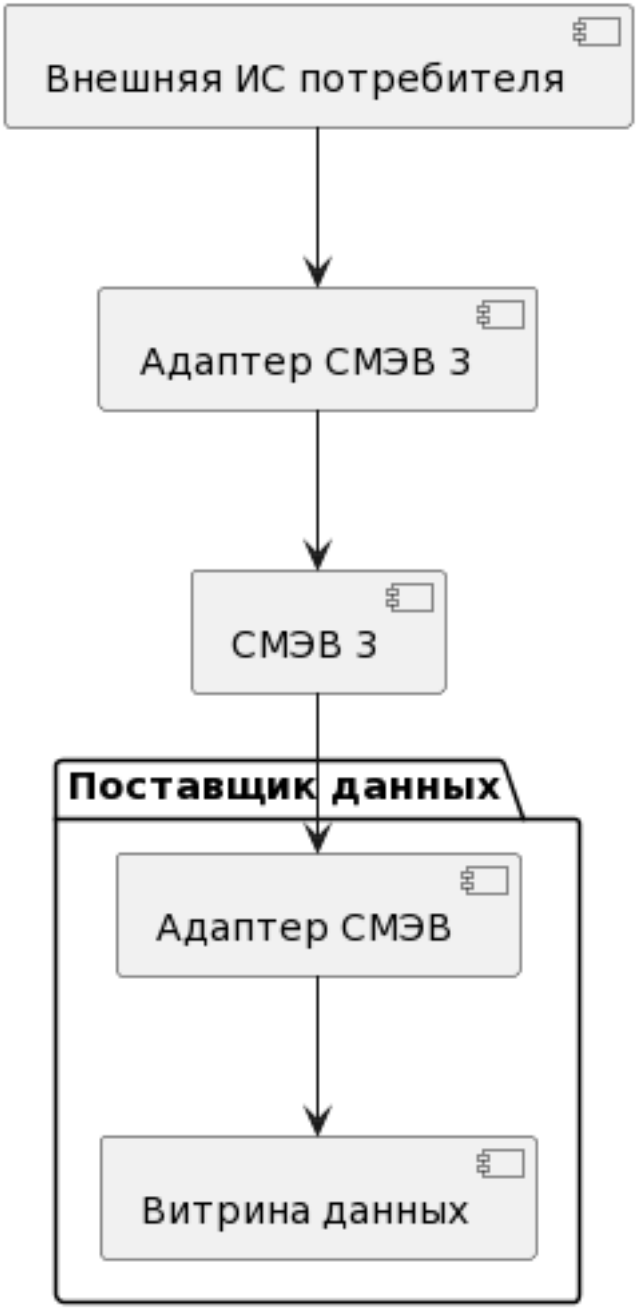


Рисунок - 2.11 Подключение к СМЭВ

1. Внешняя ИС выполняет запрос к СМЭВ 3 на получение данных от Поставщика данных.
2. Запрос через **СМЭВ3-адаптер** отправляется в **СМЭВ**.
3. Адаптер СМЭВ 3 на стороне Поставщика данных принимает запрос из СМЭВ и отправляет его в Витрину данных поставщика.
4. В Витрине Поставщика данных формируется ответ на поступивший запрос.

## 2.4.16 REST-адаптер

| Примечание: |
| --- |
| Модуль входит в состав конфигурации Стандарт |

### 2.4.16.1 Схема взаимодействия через СМЭВ3-адаптер

REST-адаптер представляет возможность подключения Внешней ИС к Витрине данных через REST-адаптер (см. Рисунок - 2.12).


Рисунок - 2.12 Взаимодействие Внешней ИС через REST-адаптер

Внешняя ИС формирует REST-запрос и отправляет его в REST-адаптер.
REST-адаптер:

- На основании своих настроек преобразует полученный REST-запрос в SQL-запрос.
- Отправляет сформированный SQL-запрос в Витрину данных.
- Преобразует полученный от Витрины данных ответ на SQL-запрос в REST-ответ.
- Отправляет сформированный REST-ответ Внешней ИС.

При взаимодействии через REST-адаптер Программа выполняет следующие основные операции по обработке данных:

- предоставляет программный интерфейс к конечным точкам API по протоколу HTTP;
- конечная точка доступа поддерживает конфигурирование, которое позволяет: - с использованием атрибутов HTTP-запроса построить и выполнить SQL-запросы из Внешней ИС к программе; - с использованием атрибутов HTTP-запроса и результатов SQL-запросов построить и отправить ответ на HTTP-запрос из программы к Внешней ИС; - документировать сконфигурированный API с использованием спецификации OpenAPI версии 3.

## 2.4.17 Сервис генерации уникального номера (Counter-Provider)

| Примечание: |
| --- |
| Модуль входит в состав конфигурации Стандарт |

### 2.4.17.1 Общее описание

**Сервис генерации уникального номера** позволяет создавать неповторяющиеся уникальные порядковые номера для сквозной нумерации файлов в сервисе формирования документов Типового ПО Витрины данных конфигурации Стандарт.

В сервисе реализованы функции:

- долговременного хранения неограниченного списка счетчиков;
- атомарного изменения счетчика при параллельном использовании этой функции.

## 2.4.18 ETL - Модуль загрузки/ удаления данных

| Примечание: |
| --- |
| Модуль входит в состав конфигурации Стандарт |

Базовый сервис загрузки данных предоставляет возможность асинхронного приёма данных из сторонних источников с целью последующей загрузки их в Витрину данных. Загрузка/обновление данных осуществляется в соответствии с заранее подготовленными Avro-схемами.

Сервис загрузки данных реализуется компонентом **ETL**, предоставляющей REST API. Доступ к REST API должен осуществляться через Proxy API Datamart Studio. Перед загрузкой необходимо получить токен Proxy API, который в дальнейшем используется для авторизации при операциях, описанных в разделах ниже. При получении ошибки авторизации необходимо повторить авторизацию, получить новый токен и использовать его для дальнейшей загрузки.

Для взаимодействия с REST API (через Proxy API) в продуктивной среде на стороне источника данных требуется использовать сертифицированную версию ОС, а также механизмы, соответствующие требованиям безопасности, установленным для эксплуатации системы (например, через программный продукт Postman).

На стороне источника данных должны соблюдаться следующие требования к механизму взаимодействия:

1. Запрещается хранение логина и пароля в открытом виде на диске. Логин и пароль должны выгружаться из безопасного хранилища в память ВМ (или контейнера) при старте взаимодействия с Proxy API для дальнейшего использования, сама ВМ должна находиться в закрытом контуре ИС или ведомства.
2. Рекомендуется реализовать механизм стирания логина и пароля из памяти после успешной аутентификации через Proxy API.

В целях тестирования взаимодействия на тестовой среде может использоваться утилита curl.

## 2.4.19 Backup manager - утилита резервного копирования

| Примечание: |
| --- |
| Утилита входит в состав конфигурации Стандарт |

### 2.4.19.1 Общее описание

Утилита **Backup manager** разработана для реализации механизма резервного копирования Витрины данных и восстановления из резервной копии.

**Backup manager** используется для оркестрации процесса резервного копирования слоя адаптеров и утилиты DTM-tools, осуществляющей резервное копирование логической модели

базы данных и физических данных СУБД, входящих в инсталляцию.

Подробное описание утилиты и работы с ней приведено в разделе Бекапирование Витрины данных НСУД Руководства администратора ПО «Витрина данных НСУД».

## 2.5 Связи с другими программами

### Связи с другими программами конфигурации Стандарт

Взаимодействие с другими программами происходит путем вызова соответствующих модулей программы:

Связи программы с другими программами приведены в Таблица 2.10 .

Таблица 2.10 Связи с другими программами

| **Клиент** | **Сервер** | **Способ взаимодействия** | **Описание** |
|---|---|---|---|
| Сервис формрования документов | Агент СМЭВ4 | REST | Передает сформированные документы для формирования ЭП. |
| СМЭВ4-адаптер - Модуль исполнения запросов | Агент СМЭВ4 | Через брокера сообщений Kafka | Получает запросы и предоставляет ответы на них согласно протоколу СМЭВ4:<br>– запрос/подзапрос на получение публикуемых данных (в т. ч. с использованием ТП или в режиме оценки запроса);<br>– предоставление структуры таблиц при подписании на репликацию (источник данных);<br>– создание структуры таблиц при подписании на репликацию (потребитель данных);<br>– предоставление реплицируемых данных (источник данных);<br>– получение реплицируемых данных (потребитель данных);<br>– запрос генерации ПФ. |
| СМЭВ4-адаптер - Модуль MPPR | Агент СМЭВ4 | Через брокера сообщений Kafka | Предоставляет Результат запрос/подзапрос на получение публикуемых данных<br>(в т. ч. с использованием ТП), делегированного СМЭВ4-адаптером. |
| СМЭВ4-адаптер - Модуль дефрагментации чанков табличных параметров | Агент СМЭВ4 | Через брокера сообщений Kafka | Получает сообщения с ТП, подготовленные Агентом СМЭВ4 |
| BLOB-адаптер | Агент СМЭВ4 | Через брокера сообщений Kafka | – Получает запрос на предоставление содержимого BLOB-объекта;<br>– Передает бинарное содержимое запрошенного BLOB-объекта. |
| | Хранилище BLOB | HTTP | Запрашивает бинарное содержимое BLOB. |
| СМЭВ3-адаптер | СМЭВ3 | SOAP | – Получает запрос вида сведений от СМЭВ3;<br>– Передает Результат запроса вида сведений;<br>– Инициативно рассылает сведения об изменении публикуемых данных. |
| | FTP-сервер СМЭВ3 | FTP | Загружает на сервер бинарное содержимое запрошенных BLOB-объектов. |
| | VipNet | REST | – Проверяет ЭП получаемых от СМЭВ3 сообщений;<br>– Формирует ЭП для отправляемых в СМЭВ |

| Клиент | Сервер | Способ взаимодействия | Описание |
|---|---|---|---|
| | | | сообщений. |
| ETL | Файловое хранилище ведомства | S3, FTP и т.п. | Считывание CSV файлов с данными, импортируемыми в витрину |
| | БД ведомства | JDBC Driver | Считывание данных, импортируемых в витрину |
| Внутренняя ИС Ведомств | Сервер конечных точек | REST API | Имитация поведения, существующего REST API |
| Внутренняя ИС Ведомств | Ядро витрины | JDBC Driver | Доступ к БД ведомства |

### Связи с другими программами конфигурации Лайт

Взаимодействие с другими программами происходит путем вызова соответствующих модулей программы:

- Внутренняя ИС Ведомства взаимодействует с ProStore через JDBC-driver.
- СМЭВ4-адаптер - Модуль исполнения запросов для взаимодействия с ИС участников взаимодействия через Агента СМЭВ4.
- CSV-uploader для взаимодействия с ИС участников взаимодействия для передачи файлов в формате XML и CSV.

Связи программы со сторонними программами приведены в см. Таблица 2.11.

Таблица 2.11 Связи с другими программами

| Клиент | Сервер | Способ взаимодействия | Описание |
|---|---|---|---|
| Внутренняя ИС Ведомств | CSV-uploader | Файловый обмен (CSV)<br>REST | Загрузка публикуемых данных в Витрину |
| | ProStore | JDBC<br>Брокер сообщений Kafka | Управление логической структурой таблиц.<br>Исполнение запросов.<br>Загрузка публикуемых данных в Витрину. |
| СМЭВ4-адаптер — Модуль исполнения запросов | Агент СМЭВ4 | Брокер сообщений Kafka<br>Kafka | Исполнение запросов. |

## 2.6 Карта портов

Карта портов компонентов программы представлена в Таблица 2.12.

Таблица 2.12 Карта портов

| Компонент | Описание |
|---|---|
| podd-adapter-query | Порт: 8083<br>Протокол: HTTP<br>Описание: Взаимодействие с СМЭВ4-адаптером |
| query-execution | Порт: 8080<br>Протокол: HTTP<br>Описание: номер порта сервиса метрик<br>Порт: 9090<br>Протокол: TCP<br>Описание: номер порта сервиса исполнения запросов |
| prometheus | Порт: 9090<br>Протокол: HTTP<br>Описание: Подключение к Prometheus WEB UI |
| grafana | Порт: 3000<br>Протокол: HTTP<br>Описание: WEB-интерфейс для работы с Grafana |

| Компонент | Описание |
|---|---|
| node_exporter | Порт: 9100<br>Протокол: HTTP<br>Описание: Порт для загрузки метрик |
| filebeat | Порт: нет открытых портов<br>Протокол: -<br>Описание: - |
| mongodb | Порт: 27017<br>Протокол:TCP<br>Описание: Подключение к MongoDB. Порт по умолчанию для экземпляров mongod и mongos.<br>Вы можете изменить этот порт с помощью port или `--port`.<br>Порт: 27018<br>Протокол: TCP<br>Описание: Подключение к MongoDB. Порт по умолчанию для mongod при запуске с<br>параметром командной строки `--shardsvr` или значением `shardsvr` для параметра `clusterRole` в файле конфигурации |
| elasticsearch | Порт: 9200<br>Протокол: HTTP<br>Описание: Подключение к Elasticsearch. |
| kafka_postgres_writer | Порт: 8096<br>Протокол: HTTP<br>Описание: Порт используется для записи топиков Kafka в ProStore |
| kafka_postgres_reader | Порт: 8094<br>Протокол: HTTP<br>Описание: Порт используется для чтения топиков Kafka из ProStore |
| postgres | Порт: 5432<br>Протокол: TCP PostgresSQL Protocol<br>Описание: Источник данных SQL |
| kafka | Порт: 9092<br>Протокол: Порт используется для<br>Описание: TCP |
| zookeeper | Порт: 2181<br>Протокол: TCP<br>Описание: Порт используется для доступа к Zookeeper |
| portainer | Порт: 9000<br>Протокол: HTTP<br>Описание: Web-интерфейс для работы с Portainer |

# 3 АРХИТЕКТУРА ВИТРИНЫ ДАННЫХ

## 3.1 Общая архитектурная схема

### Общая архитектурная схема конфигурации Стандарт

Схематичное отображение общей архитектуры Витрины данных приведено на рисунке (см. Рисунок - 3.1).

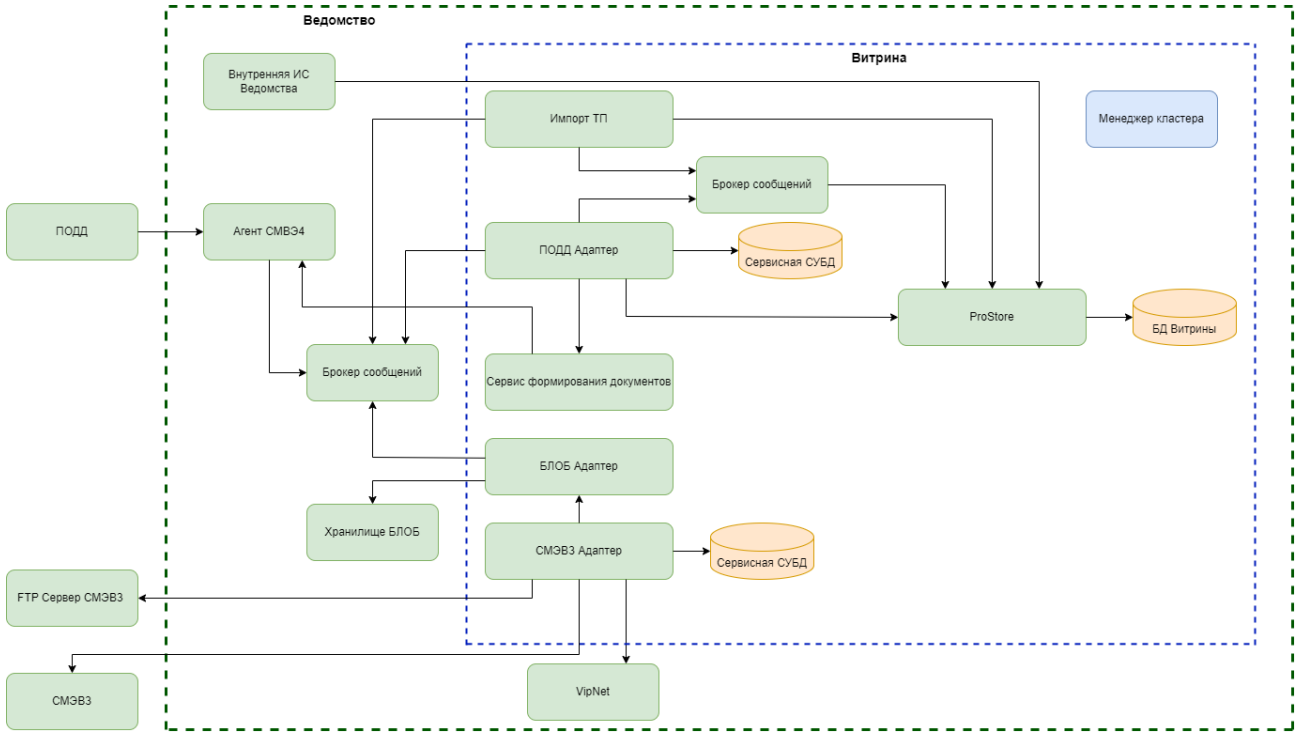


Рисунок - 3.1 Общая архитектура Витрины данных

### Общая архитектурная схема конфигурации Лайт

Схематичное отображение общей архитектуры Витрины данных приведено на рисунке ниже (см. Рисунок - 3.2).

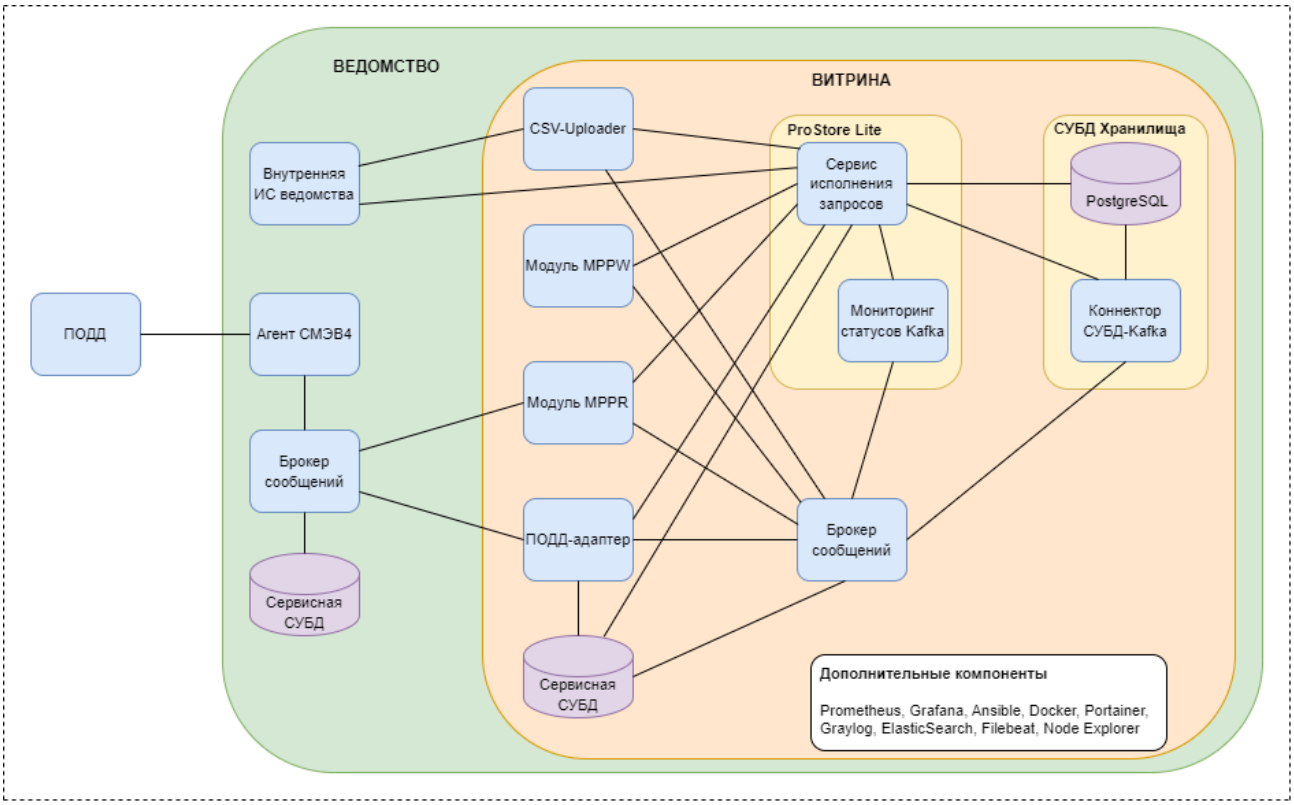


Рисунок - 3.2 Общая архитектура Витрины данных

## 3.2 Общая компонентная схема

### Общая компонентная схема конфигурации Стандарт

Схема компонентов конфигурации Стандарт представлена на рисунке ниже (см. Рисунок - 3.3).

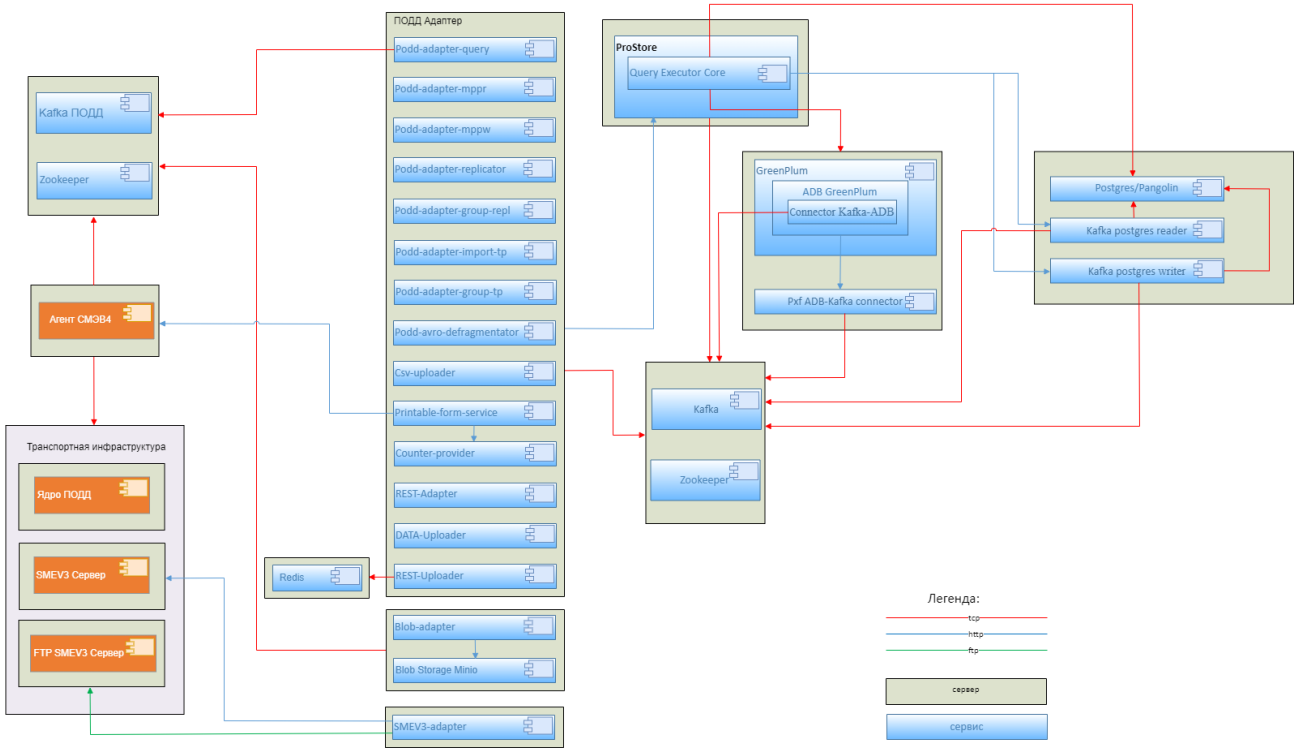


Рисунок - 3.3 Схема компонентов

### Общая компонентная схема конфигурации Лайт

Схема компонентов конфигурации Лайт представлена на рисунке ниже (см. Рисунок - 3.4).

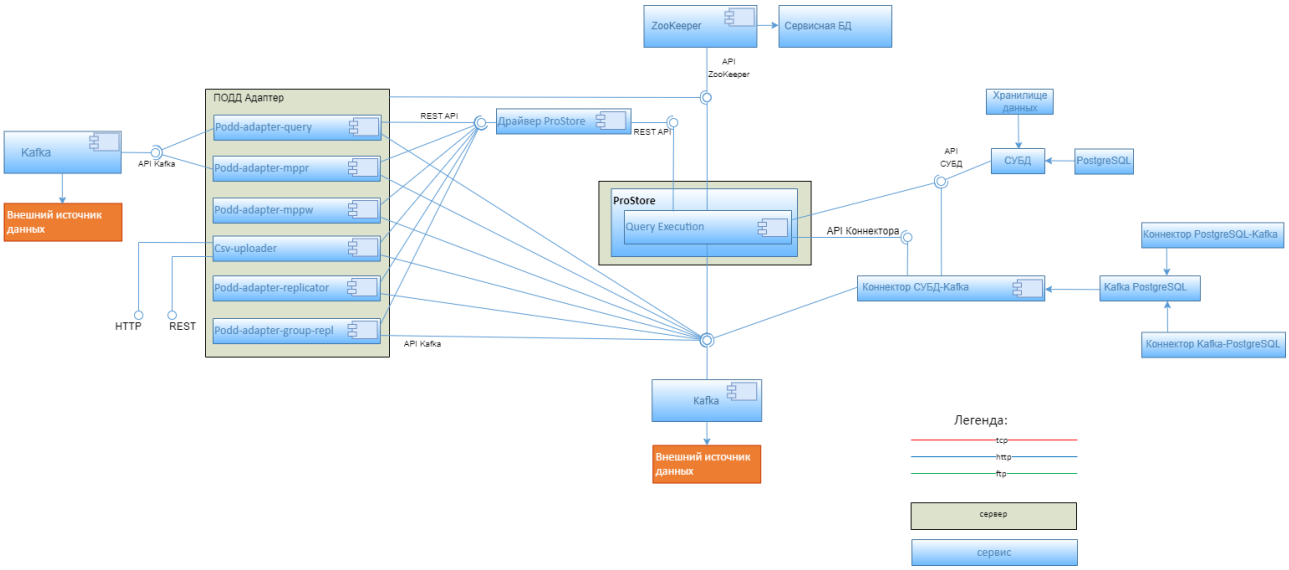


Рисунок - 3.4 Схема компонентов

## 3.3 Схема развертывания конфигурации Лайт

На Рисунок - 3.5 приведена схема развертывания программы.

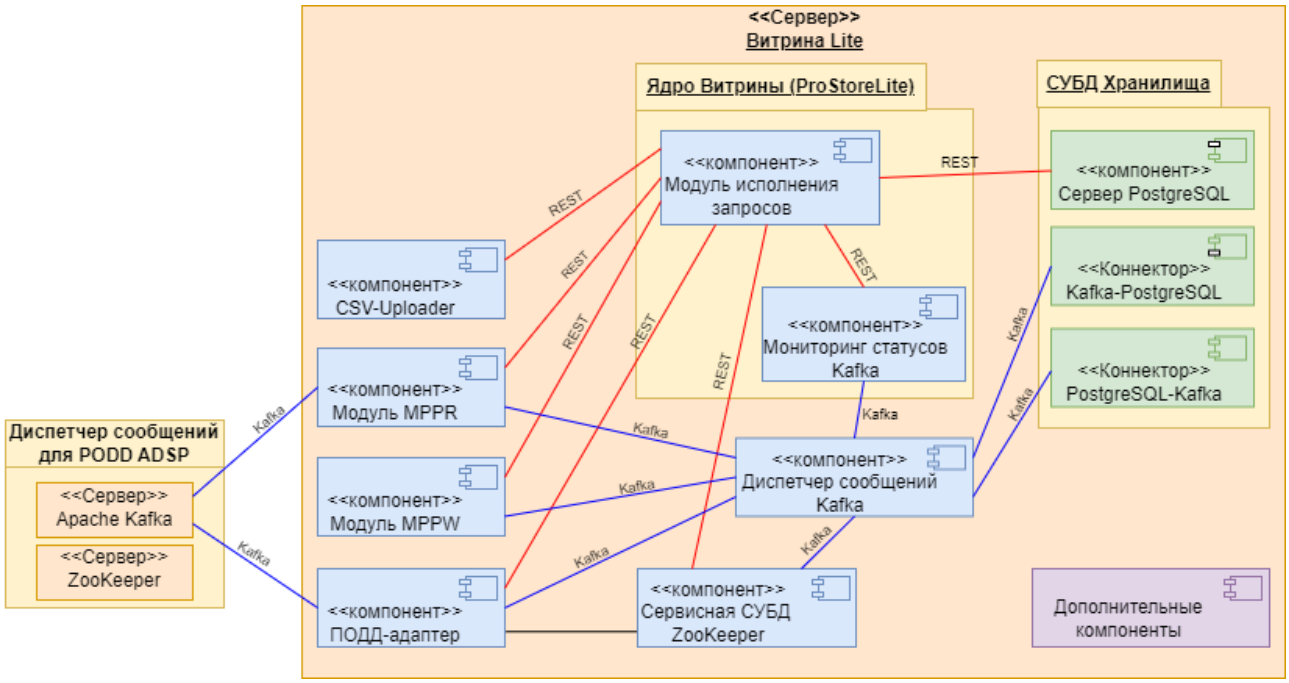


Рисунок - 3.5 Схема развертывания

## 3.4 Алгоритм работы Витрины данных конфигурации Стандарт

1. После развертывания Витрины, необходимо создать в ней логическую модель данных (для хранения данных в качестве Поставщика данных).
2. После того, как в Витрине была создана логическая модель данных, можно выполнять следующие действия:
   – загружать данные в Витрину (средствами сервиса загрузки данных (**ETL**) или

самостоятельно через **REST-сервис**);
- посылать запросы для получения данных из Витрины (через **СМЭВ4**, **СМЭВ3** или **REST-сервис**);
- подписываться (в качестве Потребителя данных) на репликацию данных из другой Витрины (**СМЭВ4** пришлет метаданные для создания логических таблиц для хранения реплики и первоначальный снапшот текущих данных из Витрины Поставщика);
- обрабатывать (в качестве Поставщика данных) подписки других Витрин на репликацию данных этой Витрины (передавать им метаданные для создания логической структуры для хранения реплики и снапшот текущих данных);
- при загрузке данных в Витрину, если есть подписчики на репликацию данных, им отправляется дельта с обновленными данными;
- если эта Витрина подписана на репликацию данных из другой Витрины, то при изменении данных в Витрине-Поставщике **СМЭВ4** доставит дельту с обновленными данными.

## 3.5 Описание логической структуры конфигурации Лайт

1. Установка программы осуществляется с помощью *Ansible* на предварительно сконфигурированный ПК (см. «Руководство по установке»).
2. После установки программа не содержит никакой логической модели данных. Необходимо загрузить структуру витрины через web-интерфейс программы (для хранения данных в качестве Поставщика данных). За загрузку структуры витрины отвечает модуль CSV-uploader. Описание загрузки структуры приведено в документе «Руководство администратора» в разделе «Инструкция по эксплуатации CSV-uploader». При работе с ПОДД структура таблиц настраивается в ЕИП НСУД и передаются в Витрину через ПОДД.
3. После того, как логическая модель данных в Витрине настроена можно:
   - обрабатывать SQL-запросы в качестве Поставщика данных;
   - выгружать шаблон через web-интерфейс;
   - загружать данные в Витрину через:
     - web-интерфейс;
     - файловый обмен;
     - REST.

# 4 ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ

## 4.1 Входные данные

### Конфигурация Стандарт

Входные данные программы конфигурации Стандарт приведены в таблице ниже (см. Таблица 4.1).

Таблица 4.1 Входные данные

| № | Описание |
|---|---|
| 1 | Данные для витрины от внутренних ИС ведомства |
| 2 | Данные, реплицированные из других витрин |
| 3 | Запрос от СМЭВ4 на предоставление данных для ИС-потребителя |
| 4 | Файлы конфигурации программы |
| 5 | Команды, вводимые администратором посредством интерфейса командной строки |

### Конфигурация Лайт

Входные данные программы приведены в таблице ниже (см. Таблица 4.2).

Таблица 4.2 Входные данные

| № | Описание |
|---|---|
| 1 | Данные для Витрины от внутренних ИС ведомства |
| 2 | Данные, реплицированные из других витрин |
| 3 | Запрос от СМЭВ4 на предоставление данных для ИС-потребителя |
| 4 | CSV и XML-файлы загруженные через модуль CSV-Uploader |
| 5 | Файлы конфигурации программы |
| 6 | Команды, вводимые администратором посредством интерфейса командной строки |

## 4.2 Выходные данные

### Конфигурация Стандарт

Выходные данные ПО «Витрина данных НСУД» приведены в см. Таблица 4.3.

Таблица 4.3 Выходные данные

| № | Описание |
|---|---|
| 1 | Данные витрины, предоставляемые системам-потребителям в ответ на запрос |
| 2 | Реплицированные из других витрин данные, предоставляемые внутренним ИС ведомства |
| 3 | Сообщения, передаваемые администратору посредством интерфейса командной |

Взаимодействие **Модуля исполнения запросов** с **Агентом СМЭВ4** производится через список топиков брокера сообщений Kafka.

Назначение и формат сообщений, последовательность обмена ими соответствуют документу «Методические рекомендации по работе с подсистемой обеспечения доступа к данным федеральной государственной информационной системы «Единая система межведомственного электронного взаимодействия»» (версия 3.11.0).

Формат взаимодействия с **Модулем исполнения запросов** (название топика, формат сообщений, схема взаимодействия) описан в разделе Спецификация Модуля исполнения

запросов Приложения 1.

Формат взаимодействия с **BLOB-адаптером** (название топика, формат сообщений, схема взаимодействия) описан в разделе Спецификация модуля «BLOB-адаптер» Приложения 1.

Формат взаимодействия с **Сервисом формирования документов** (название топика, формат сообщений, схема взаимодействия) описан в разделе Спецификация модуля «Сервис Формирования документов» Приложения 1.

### Конфигурация Лайт

Выходные данные ПО «Витрина данных НСУД» приведены в таблице ниже (см. Таблица 4.4).

Таблица 4.4 Выходные данные

| № | Описание |
|---|---|
| 1 | Данные Витрины, предоставляемые системам-потребителям в ответ на запрос |
| 2 | Реплицированные из других витрин данные, предоставляемые внутренним ИС ведомства |
| 3 | CSV и XML-файлы выгруженные через модуль CSV-Uploader |
| 4 | Сообщения, передаваемые администратору посредством интерфейса командной |

Взаимодействие СМЭВ4-адаптера - Модуль исполнения запросов с Агентом СМЭВ4 производится через список топиков Брокера сообщений Kafka. Назначение и формат сообщений, последовательность обмена ими соответствуют документу «Методические рекомендации по работе с подсистемой обеспечения доступа к данным федеральной государственной информационной системы «Единая система межведомственного электронного взаимодействия»» (версия 1.0.0).

Описание формата взаимодействия (название топика, формат сообщений, схема взаимодействия) см. «Руководство администратора».

# 5 ВЫЗОВ И ЗАГРУЗКА

## Конфигурация Стандарт

Программа не имеет графического интерфейса и запускается автоматически после установки.

При необходимости любой из сервисов/модулей можно остановить и запустить заново.

Описание запуска и остановки модулей приведено в разделе Запуск и остановка Программы Руководства администратора ПО «Витрина данных НСУД».

## Конфигурация Лайт

Программа не имеет графического интерфейса и запускается с жесткого диска сервера как `systemd` сервис.

При необходимости любой из сервисов/модулей можно остановить и запустить заново.

Для ручной остановки и запуска необходимо подключиться по SSH на сервер и с правами `sudo` использовать штатный функционал команд `systemctl`, которая является инструментом центрального управления для контроля системы инициализации.

Например

```
sudo systemctl stop query-execution
sudo systemctl start query-execution
```

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

# ПРИЛОЖЕНИЕ 1. ОПИСАНИЕ СПЕЦИФИКАЦИИ

## 1 Спецификация Модуля исполнения запросов

## 1.1 Запрос данных из Витрины

Данная спецификация описывает возможность запроса данных к Витрине, получения успешного ответа на запрос или ошибки, в случае невозможности выполнения запроса, с описанием причины ошибки.

### 1.1.1 query.rq

`query.rq` - Топик sql запросов на исполнение

**Структура сообщения**

```yaml
datamartExecuteQueryRequestMessage:
  description: Исполнение sql запроса на витрине
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      MESSAGE_TYPE:
        description: Тип сообщения
        type: string
        const: DatamartExecuteQueryRequest:0.1
      REQUEST_ID:
        description: Идентификатор запроса
        type: string
      QUERY_DEADLINE:
        description: Время в миллисекундах от эпохи, до которого запрос должен быть выполнен
        type: string
        format: int64
      AGENT_CONSUMER_ID:
        description: Мнемоника потребителя (мнемоника агента)
        type: string
      QUERY_MNEMONIC:
        description: '<Полная мнемоника РЗ>.<версия РЗ>'
        type: string
  payload:
    $ref: '#/components/schemas/datamartExecuteQueryRequest'
  examples:
    - name: simple
      summary: Простой запрос на исполнение без параметров
      headers:
        MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
        replyTo: agent-fias
        datamartMnemonic: fias
        sql: select * from v1_addrobj
        parameters: [ ]
        namedParams: [ ]
```

```
      tableParams: [ ]
      isForEstimation: false
      rowCountThreshold: -1
      customerId: aaa
      customerOgrn: ""
      queryMnemonic: fias.selectAllAddrobj.1.0
  - name: estimation
    summary: Запрос на оценку
    headers:
      MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
    payload:
      requestId: 403eada5-05f6-480c-bca9-03328091efeb
      subRequestId: 451000b8-dff2-4a1b-ab1b-42500a70d232
      replyTo: agent-fias
      datamartMnemonic: fias
      sql: select * from v1_addrobj
      parameters: [ ]
      namedParams: [ ]
      tableParams: [ ]
      isForEstimation: true
      rowCountThreshold: 1000
      customerId:
        string: agent-fias
      customerOgrn:
        string: "1053600591197"
      queryMnemonic:
        string: fias.selectAllAddrobj.1.0
  - name: complex
    summary: Запрос с параметрами
    headers:
      MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
    payload:
      requestId: 68758a92-0027-4258-bf17-aa3d24f85094
      subRequestId: 96e6eb99-7ff1-4efa-abae-ef1c5744b723
      replyTo: agent-fias
      datamartMnemonic: fias
      sql: select * from v1_addrobj where oktmo = ? and name = @tbl.fullname
      parameters:
        - type: STRING
          value:
            string: asdasdasd
        - type: LONG
          value: null
      namedParams: [ ]
      tableParams: [ ]
      isForEstimation: false
      rowCountThreshold: -1
      customerId:
        string: agent-fias
      customerOgrn:
        string: "1053600591197"
      queryMnemonic:
        string: fias.selectAddrobjWithParams.1.0
  - name: complex_named
    summary: Запрос с именованными параметрами
    headers:
      MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
    payload:
      requestId: 12358a92-0027-4258-bf17-aa3d24f85094
      subRequestId: 56e6eb99-7ff1-4efa-abae-ef1c5744b723
      replyTo: agent-fias
      datamartMnemonic: fias
```

```
      sql: select * from @tbl.fullname el LEFT JOIN v1_addrobj where oktmo = @p1 and
kod = @p2
      parameters: [ ]
      namedParams:
        - name: p1
          type: STRING
          value:
            string: asdasdasd
        - name: p2
          type: LONG
          value: null
      tableParams: [ ]
      isForEstimation: false
      rowCountThreshold: -1
      customerId:
        string: agent-fias
      customerOgrn:
        string: "1053600591197"
      queryMnemonic:
        string: fias.selectAddrobjWithParams.1.0
  - name: deadline
    summary: Простой запрос на исполнение без параметров
    headers:
      MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
      QUERY_DEADLINE: 1629289006904
    payload:
      requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
      replyTo: agent-fias
      datamartMnemonic: fias
      sql: select * from v1_addrobj
      parameters: [ ]
      namedParams: [ ]
      tableParams: [ ]
      isForEstimation: false
      rowCountThreshold: -1
      customerId: agent-fias
      customerOgrn: "1053600591197"
      queryMnemonic: fias.selectAllWithDeadline.1.0
```

**Avro-схема сообщения**

```
datamartExecuteQueryRequest:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: QueryRequest
  namespace: datamart.query
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      type:
        type: string
        logicalType: uuid
    - name: replyTo
      description: Служебная информация маршрутизации сообщения. Ответ, формируемый
витриной, обязан содержать переданное значение без каких либо искажений
      type: string
```

```
    - name: datamartMnemonic
      description: Мнемоника витрины, к которой выполняется запрос
      type: string
    - name: sql
      description: SQL запрос на исполнение, либо имя хранимой процедуры для
регламентированных запросов
      type: string
    - name: parameters
      description: Параметры к SQL запросу
      default: [ ]
      type:
        type: array
        items:
          type: record
          name: QueryParameter
          description: Описание параметра
          fields:
            - name: type
              type: string
              description: Тип параметра
              enum:
                - BIG_DECIMAL
                - BINARY
                - BOOLEAN
                - DATE
                - DOUBLE
                - FLOAT
                - INTEGER
                - LONG
                - SHORT
                - STRING
                - TIME
                - TIMESTAMP
            - name: value
              description: Значение параметра
              type:
                - string
                - 'null'
    - name: namedParams
      description: Именованные параметры запроса
      default: [ ]
      type:
        type: array
        items:
          type: record
          name: NamedParam
          description: Описание именованного параметра
          fields:
            - name: name
              description: Имя (мнемоника) параметра
              type: string
            - name: type
              type: string
              description: Тип параметра
              enum:
                - BIG_DECIMAL
                - BINARY
                - BOOLEAN
                - DATE
                - DOUBLE
                - FLOAT
                - INTEGER
```

```
              - LONG
              - SHORT
              - STRING
              - TIME
              - TIMESTAMP
          - name: value
            description: Значение параметра
            type:
              - string
              - 'null'
    - name: tableParams
      description: Табличные параметры запроса
      default: [ ]
      type:
        type: array
        items:
          type: record
          name: TableParam
          fields:
            - name: id
              description: Уникальный идентификатор
              type:
                type: string
                logicalType: uuid
            - name: name
              description: Имя параметра
              type: string
            - name: columns
              description: Описание колонок таблицы
              type:
                type: array
                items:
                  type: record
                  description: Описание колонки
                  name: TableParamColumnInfo
                  fields:
                    - name: name
                      type: string
                      description: Имя колонки
                    - name: type
                      type: string
                      description: Тип атрибута
                      enum:
                        - BIG_DECIMAL
                        - BINARY
                        - BOOLEAN
                        - DATE
                        - DOUBLE
                        - FLOAT
                        - INTEGER
                        - LONG
                        - SHORT
                        - STRING
                        - TIME
                        - TIMESTAMP
    - name: isForEstimation
      description: Признак необходимости вернуть статистику по запросу в качестве
результата. В случае, если оценка по результату исполнения sql запроса не превышает
rowCountThreshold записей, должен сразу отдаваться результат без отправки оценки в ядро
      type: boolean
      default: false
    - name: rowCountThreshold
```

```
      description: Максимальное оценочное количество строк результата, при превышении которого возвращается статистика по запросу. Если оценка по запросу не превышет данный параметр, витрина сразу возвращает ответ с результатом. Заполняется в случае isForEstimation = true
      type: long
      default: -1
    - name: customerId
      description: Мнемоника ИС Потребителя
      type:
        - 'null'
        - string
      default: null
    - name: customerOgrn
      description: ОГРН ИС Потребителя
      type:
        - 'null'
        - string
      default: null
    - name: queryMnemonic
      description: 'Мнемоника РЗ, сформированная по правилу: <мнемоника витрины>.<мнемоника РЗ>.<версия РЗ> Если запрос распределенный, то формируется по правилу: podd.<мнемоника РЗ>.<версия РЗ>'
      type:
        - 'null'
        - string
      default: null
```

### 1.1.2 query.rs

`query.rs` - Топик с чанками данных исполнения запросов

**Структура сообщения**

```
datamartExecuteQueryResultChunkMessage:
  description: Чанк с данными по исполнению запроса
  contentType: 'application/octet-stream'
  bindings:
    kafka:
      key:
        $ref: '#/components/schemas/datamartExecuteQueryResultChunk'
  headers:
    type: object
    properties:
      MESSAGE_TYPE:
        description: Тип сообщения
        type: string
        const: DatamartExecuteQueryResultChunk:0.1
  payload:
    description: Бинарные данные чанка
  examples:
    - name: base64
      headers:
        MESSAGE_TYPE: DatamartExecuteQueryResultChunk:0.1
      payload:
        value: JEEJNodyLO7p1pgsRHG9pEiXeYGvHW4YCl4FgrgBmu5C92iVX1PV2GZdcqsb66bx8sk=
```

**Avro-схема сообщения**

```
datamartExecuteQueryResultChunk:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: QueryResultChunk
  namespace: datamart.query
```

```
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      type:
        type: string
        logicalType: uuid
    - name: replyTo
      description: Служебная информация маршрутизации сообщения. Заполняется
соответсвующим значением из запроса
      type: string
    - name: chunkNumber
      description: Номер порции по порядку
      type: int
      minimum: 1
    - name: isLastChunk
      description: Признак последнего сообщения
      type: boolean
    - name: streamNumber
      description: Номер стрима дланных
      minimum: 1
      type:
        - int
        - "null"
    - name: streamTotal
      description: Общее количество стримов
      minimum: 1
      type:
        - int
        - "null"
    - name: isFragmented
      description: Признак присутствия в чанке неполных строк (строк, которые были
разбиты на несколько чанков)
      type: boolean
    - name: uncompressedSize
      description: Признак присутствия в чанке неполных строк (строк, которые были
разбиты на несколько чанков)
      type: int
      minimum: 0
  examples:
    - requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
      replyTo: agent-fias
      chunkNumber: 1
      isLastChunk: true
      streamNumber:
        int: 1
      streamTotal:
        int: 1
      isFragmented: false
      uncompressedSize: 10
```

### 1.1.3 query.err

`query.err` - Топик с ошибками исполнения sql запросов на витрине

**Структура сообщения**

```
datamartExecuteQueryErrorMessage:
```

```yaml
  description: Ошибка исполнения запроса
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      MESSAGE_TYPE:
        description: Тип сообщения
        type: string
        const: DatamartExecuteQueryError:0.1
  payload:
    $ref: '#/components/schemas/datamartExecuteQueryError'
  examples:
    - name: error
      summary: Сообщение с ошибкой исполенния запроса на витрине без header
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
        replyTo: agent-fias
        errorCode: DATAMART-001
        message: Непредвиденная ошибка
    - name: errorWithHeader
      summary: Сообщение с ошибкой исполенния запроса на витрине
      headers:
        MESSAGE_TYPE: DatamartExecuteQueryError:0.1
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
        replyTo: agent-fias
        errorCode: DATAMART-001
        message: Непредвиденная ошибка
```

**Avro-схема сообщения**

```yaml
datamartExecuteQueryError:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: QueryError
  namespace: datamart.query
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      type:
        type: string
        logicalType: uuid
    - name: replyTo
      description: Служебная информация маршрутизации сообщения. Заполняется
соответсвующим значением из запроса
      type: string
    - name: errorCode
      description: Код возникшей ошибки
      type: string
```

```
    - name: message
      description: Сообщение с ошибкой исполнения
      type: string
  examples:
    - requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
      replyTo: agent-fias
      errorCode: DATAMART-001
      message: Непредвиденная ошибка
```

### 1.1.4 query.estimation.rs

Топик `QUERY.ESTIMATION.RS` предоставляет возможность произвести предварительную оценку объема получаемых данных при выполнении запроса к Витрине данных, а также, ограничить выгрузку данных в случае, если количество получаемых данных превысит заданное количество строк (параметр `rowCountThreshold`). В этом случае, ответом на запрос будет предварительная оценка объема.

Например, если вам нужна информация из какой-либо таблицы контактов, то, возможно, следует предварительно узнать, какой объем данных вы можете получить на такой запрос т.к ответ может содержать несколько гигабайт информации и выполнение запроса может занять много времени. Вы сможете установить ограничение на получение данных, например, не более 10 контактов из таблицы. В этом случае, если ответом на запрос будет 5 контактов, то Витрина предоставит ответ полностью. Если ответом будет 1000 контактов, то в качестве ответа будет сформирована предварительная оценка такого ответа, а именно, что данный ответ будет содержать 1000 строк и содержать информацию, например, на 15000 байт. Используя топик `query.estimation.rs` можно прогнозировать объем получаемых данных, в соответствии с которыми оптимизировать запросы к Витрине.

В случае использования топика `query.estimation.rs` запрашивается не конечный ответ на запрос, а приблизительная оценка объема (байт) и количество строк в ответе.

> Примечание:
>
> Данное требование не распространяется на механизм подписок Потребителей данных ПОДД.

**Алгоритм работы query.estimation.rs**

1. Витрина получает запрос query.rq с признаком `isForEstimation` оценивает объем результата по этому запросу (в байтах и количестве строк).
2. Витрина сравнивает результаты оценки объема запроса со значением предельного числа строк в параметре `rowCountThreshold` (топик query.rq).
3. Если значение в оценке меньше, чем предельное значение в `rowCountThreshold`, то Витрина возвращает результат запроса в качестве ответа в топик query.rs.
4. Если значение оценки превышает предельное значение, возвращает предварительную оценку объема в качестве ответа.

**Структура сообщения**

```
datamartQueryEstimationMessage:
  description: Оценка по запросу
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
```

```
  payload:
    $ref: '#/components/schemas/datamartQueryEstimation'
  examples:
    - name: estimation
      summary: Сообщение с оценкой по исполнению запроса
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
        estimatedRowCount: 100
        estimatedSize: 1000
        estimatedTime: 50
```

**Avro-схема сообщения**

```
datamartQueryEstimation:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: Estimation
  namespace: datamart.query
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      type:
        type: string
        logicalType: uuid
    - name: estimatedRowCount
      description: Оценка количества строк результата выполнения запроса
      type: long
    - name: estimatedSize
      description: Оценка объема результата выполнения запроса, в байтах
      type: long
    - name: estimatedTime
      description: Оценка времени выполнения запроса в миллисекундах
      type: long
  examples:
    - requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
      estimatedRowCount: 100
      estimatedSize: 1000
      estimatedTime: 50
```

## 1.2 Отмена запроса данных

Данная спецификация описывает возможность отмены ранее отправленного запроса к Витрине, получения ответа об успешной отмене запроса или ошибки, с описанием возможной причины.

### 1.2.1 cancel.rq

`cancel.rq` - Топик с сообщениями об отмене исполнения запроса

**Структура сообщения**

```
datamartQueryCancellationRequestMessage:
  description: Запрос на отмену исполнения
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
```

```
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      REQUEST_ID:
        description: Идентификатор запроса
        type: string
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    $ref: '#/components/schemas/datamartQueryCancellationRequest'
  examples:
    - name: request
      summary: Пример запроса на отмену
      headers:
        REQUEST_ID: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        AGENT_CONSUMER_ID: agent-fias
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
```

**Avro-схема сообщения**

```
datamartQueryCancellationRequest:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: AgentQueryCancellationRequest
  namespace: ru.rtlabs.common.query.cancel
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
```

### 1.2.2 cancel.rs

`cancel.rs` - Топик с ответами на отмену запроса

**Структура сообщения**

```
datamartCancelQuerySuccessMessage:
  description: Ответ об успешной отмене запроса
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      REQUEST_ID:
        description: Идентификатор запроса
        type: string
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
```

```
    $ref: '#/components/schemas/datamartCancelQuerySuccess'
  examples:
    - name: success
      summary: Пример запроса на отмену
      headers:
        REQUEST_ID: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        AGENT_CONSUMER_ID: agent-fias
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        isSuccess: true
```

**Avro-схема сообщения**

```
datamartCancelQuerySuccess:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: DatamartCancelQuerySuccess
  namespace: datamart.query
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: isSuccess
      description: Признак успешного выполнения операции
      type: boolean
```

### 1.2.3 cancel.err

`cancel.err` - Топик с ошибками по отмене запроса

**Структура сообщения**

```
datamartCancelQueryErrorMessage:
  description: Ответ об успешной отмене запроса
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      REQUEST_ID:
        description: Идентификатор запроса
        type: string
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    $ref: '#/components/schemas/datamartCancelQueryError'
  examples:
    - name: success
      summary: Пример запроса на отмену
      headers:
        REQUEST_ID: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        AGENT_CONSUMER_ID: agent-fias
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        errorCode: DATAMART-001
```

```
      message: Непредвиденная ошибка
```

**Avro-схема сообщения**

```
datamartCancelQueryError:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: DatamartCancelQueryError
  namespace: datamart.query.cancel
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: errorCode
      description: Код ошибки выполнения
      type: string
    - name: message
      description: Сообщение об ошибке
      type: string
```

## 1.3 Запрос оценки выполнения запроса на Витрине

Данная спецификация описывает возможность получения оценки выполнения запросов на Витрине.

### 1.3.1 query.rq

`query.rq` - Топик sql запросов на исполнение

**Структура сообщения**

```
datamartExecuteQueryRequestMessage:
  description: Исполнение sql запроса на витрине
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      MESSAGE_TYPE:
        description: Тип сообщения
        type: string
        const: DatamartExecuteQueryRequest:0.1
      REQUEST_ID:
        description: Идентификатор запроса
        type: string
      QUERY_DEADLINE:
        description: Время в миллисекундах от эпохи, до которого запрос должен быть
выполнен
        type: string
        format: int64
      AGENT_CONSUMER_ID:
        description: Мнемоника потребителя (мнемоника агента)
        type: string
      QUERY_MNEMONIC:
        description: '<Полная мнемоника РЗ>.<версия РЗ>'
        type: string
```

```yaml
payload:
  $ref: '#/components/schemas/datamartExecuteQueryRequest'
examples:
  - name: simple
    summary: Простой запрос на исполнение без параметров
    headers:
      MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
    payload:
      requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
      replyTo: agent-fias
      datamartMnemonic: fias
      sql: select * from v1_addrobj
      parameters: [ ]
      namedParams: [ ]
      tableParams: [ ]
      isForEstimation: false
      rowCountThreshold: -1
      customerId: aaa
      customerOgrn: ""
      queryMnemonic: fias.selectAllAddrobj.1.0
  - name: estimation
    summary: Запрос на оценку
    headers:
      MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
    payload:
      requestId: 403eada5-05f6-480c-bca9-03328091efeb
      subRequestId: 451000b8-dff2-4a1b-ab1b-42500a70d232
      replyTo: agent-fias
      datamartMnemonic: fias
      sql: select * from v1_addrobj
      parameters: [ ]
      namedParams: [ ]
      tableParams: [ ]
      isForEstimation: true
      rowCountThreshold: 1000
      customerId:
        string: agent-fias
      customerOgrn:
        string: "1053600591197"
      queryMnemonic:
        string: fias.selectAllAddrobj.1.0
  - name: complex
    summary: Запрос с параметрами
    headers:
      MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
    payload:
      requestId: 68758a92-0027-4258-bf17-aa3d24f85094
      subRequestId: 96e6eb99-7ff1-4efa-abae-ef1c5744b723
      replyTo: agent-fias
      datamartMnemonic: fias
      sql: select * from v1_addrobj where oktmo = ? and name = @tbl.fullname
      parameters:
        - type: STRING
          value:
            string: asdasdasd
        - type: LONG
          value: null
      namedParams: [ ]
      tableParams: [ ]
      isForEstimation: false
      rowCountThreshold: -1
```

```
        customerId:
          string: agent-fias
        customerOgrn:
          string: "1053600591197"
        queryMnemonic:
          string: fias.selectAddrobjWithParams.1.0
    - name: complex_named
      summary: Запрос с именованными параметрами
      headers:
        MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
      payload:
        requestId: 12358a92-0027-4258-bf17-aa3d24f85094
        subRequestId: 56e6eb99-7ff1-4efa-abae-ef1c5744b723
        replyTo: agent-fias
        datamartMnemonic: fias
        sql: select * from @tbl.fullname el LEFT JOIN v1_addrobj where oktmo = @p1 and
kod = @p2
        parameters: [ ]
        namedParams:
          - name: p1
            type: STRING
            value:
              string: asdasdasd
          - name: p2
            type: LONG
            value: null
        tableParams: [ ]
        isForEstimation: false
        rowCountThreshold: -1
        customerId:
          string: agent-fias
        customerOgrn:
          string: "1053600591197"
        queryMnemonic:
          string: fias.selectAddrobjWithParams.1.0
    - name: deadline
      summary: Простой запрос на исполнение без параметров
      headers:
        MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
        QUERY_DEADLINE: 1629289006904
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
        replyTo: agent-fias
        datamartMnemonic: fias
        sql: select * from v1_addrobj
        parameters: [ ]
        namedParams: [ ]
        tableParams: [ ]
        isForEstimation: false
        rowCountThreshold: -1
        customerId: agent-fias
        customerOgrn: "1053600591197"
        queryMnemonic: fias.selectAllWithDeadline.1.0
```

**Avro-схема сообщения**

```
datamartExecuteQueryRequest:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: QueryRequest
  namespace: datamart.query
```

```
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      type:
        type: string
        logicalType: uuid
    - name: replyTo
      description: Служебная информация маршрутизации сообщения. Ответ, формируемый витриной, обязан содержать переданное значение без каких либо искажений
      type: string
    - name: datamartMnemonic
      description: Мнемоника витрины, к которой выполняется запрос
      type: string
    - name: sql
      description: SQL запрос на исполнение, либо имя хранимой процедуры для регламентированных запросов
      type: string
    - name: parameters
      description: Параметры к SQL запросу
      default: [ ]
      type:
        type: array
        items:
          type: record
          name: QueryParameter
          description: Описание параметра
          fields:
            - name: type
              type: string
              description: Тип параметра
              enum:
                - BIG_DECIMAL
                - BINARY
                - BOOLEAN
                - DATE
                - DOUBLE
                - FLOAT
                - INTEGER
                - LONG
                - SHORT
                - STRING
                - TIME
                - TIMESTAMP
            - name: value
              description: Значение параметра
              type:
                - string
                - 'null'
    - name: namedParams
      description: Именованные параметры запроса
      default: [ ]
      type:
        type: array
        items:
          type: record
          name: NamedParam
          description: Описание именованного параметра
```

```
        fields:
          - name: name
            description: Имя (мнемоника) параметра
            type: string
          - name: type
            type: string
            description: Тип параметра
            enum:
              - BIG_DECIMAL
              - BINARY
              - BOOLEAN
              - DATE
              - DOUBLE
              - FLOAT
              - INTEGER
              - LONG
              - SHORT
              - STRING
              - TIME
              - TIMESTAMP
          - name: value
            description: Значение параметра
            type:
              - string
              - 'null'
    - name: tableParams
      description: Табличные параметры запроса
      default: [ ]
      type:
        type: array
        items:
          type: record
          name: TableParam
          fields:
            - name: id
              description: Уникальный идентификатор
              type:
                type: string
                logicalType: uuid
            - name: name
              description: Имя параметра
              type: string
            - name: columns
              description: Описание колонок таблицы
              type:
                type: array
                items:
                  type: record
                  description: Описание колонки
                  name: TableParamColumnInfo
                  fields:
                    - name: name
                      type: string
                      description: Имя колонки
                    - name: type
                      type: string
                      description: Тип атрибута
                      enum:
                        - BIG_DECIMAL
                        - BINARY
                        - BOOLEAN
                        - DATE
```

```
                    - DOUBLE
                    - FLOAT
                    - INTEGER
                    - LONG
                    - SHORT
                    - STRING
                    - TIME
                    - TIMESTAMP
    - name: isForEstimation
      description: Признак необходимости вернуть статистику по запросу в качестве
результата. В случае, если оценка по результату исполнения sql запроса не превышает
rowCountThreshold записей, должен сразу отдаваться результат без отправки оценки в ядро
      type: boolean
      default: false
    - name: rowCountThreshold
      description: Максимальное оценочное количество строк результата, при превышении
которого возвращается статистика по запросу. Если оценка по запросу не превышет данный
параметр, витрина сразу возвращает ответ с результатом. Заполняется в случае
isForEstimation = true
      type: long
      default: -1
    - name: customerId
      description: Мнемоника ИС Потребителя
      type:
        - 'null'
        - string
      default: null
    - name: customerOgrn
      description: ОГРН ИС Потребителя
      type:
        - 'null'
        - string
      default: null
    - name: queryMnemonic
      description: 'Мнемоника РЗ, сформированная по правилу: <мнемоника
витрины>.<мнемоника РЗ>.<версия РЗ> Если запрос распределенный, то формируется по
правилу: podd.<мнемоника РЗ>.<версия РЗ>'
      type:
        - 'null'
        - string
      default: null
```

### 1.3.2 query.estimation.rs

Топик `QUERY.ESTIMATION.RS` предоставляет возможность произвести предварительную оценку объема получаемых данных при выполнении запроса к Витрине данных, а также, ограничить выгрузку данных в случае, если количество получаемых данных превысит заданное количество строк (параметр `rowCountThreshold`). В этом случае, ответом на запрос будет предварительная оценка объема.

Например, если вам нужна информация из какой-либо таблицы контактов, то, возможно, следует предварительно узнать, какой объем данных вы можете получить на такой запрос т.к ответ может содержать несколько гигабайт информации и выполнение запроса может занять много времени. Вы сможете установить ограничение на получение данных, например, не более 10 контактов из таблицы. В этом случае, если ответом на запрос будет 5 контактов, то Витрина предоставит ответ полностью. Если ответом будет 1000 контактов, то в качестве ответа будет сформирована предварительная оценка такого ответа, а именно, что данный ответ будет содержать 1000 строк и содержать информацию, например, на 15000 байт. Используя топик

query.estimation.rs можно прогнозировать объем получаемых данных, в соответствии с которыми оптимизировать запросы к Витрине.

В случае использования топика query.estimation.rs запрашивается не конечный ответ на запрос, а приблизительная оценка объема (байт) и количество строк в ответе.

> Примечание:
>
> Данное требование не распространяется на механизм подписок Потребителей данных ПОДД.

**Алгоритм работы query.estimation.rs**

1. Витрина получает запрос query.rq с признаком isForEstimation оценивает объем результата по этому запросу (в байтах и количестве строк).
2. Витрина сравнивает результаты оценки объема запроса со значением предельного числа строк в параметре rowCountThreshold (топик query.rq).
3. Если значение в оценке меньше, чем предельное значение в rowCountThreshold, то Витрина возвращает результат запроса в качестве ответа в топик query.rs.
4. Если значение оценки превышает предельное значение, возвращает предварительную оценку объема в качестве ответа.

**Структура сообщения**

```yaml
datamartQueryEstimationMessage:
  description: Оценка по запросу
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  payload:
    $ref: '#/components/schemas/datamartQueryEstimation'
  examples:
    - name: estimation
      summary: Сообщение с оценкой по исполнению запроса
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
        estimatedRowCount: 100
        estimatedSize: 1000
        estimatedTime: 50
```

**Avro-схема сообщения**

```yaml
datamartQueryEstimation:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: Estimation
  namespace: datamart.query
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      type:
        type: string
        logicalType: uuid
```

```
    - name: estimatedRowCount
      description: Оценка количества строк результата выполнения запроса
      type: long
    - name: estimatedSize
      description: Оценка объема результата выполнения запроса, в байтах
      type: long
    - name: estimatedTime
      description: Оценка времени выполнения запроса в миллисекундах
      type: long
  examples:
    - requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
      estimatedRowCount: 100
      estimatedSize: 1000
      estimatedTime: 50
```

### 1.3.3 Запрос статистики

Данная спецификация описывает возможность запроса статистики Витрины.

### 1.3.4 statistics.rq

`statistics.rq` - Топик запросов статистики витрины

**Структура сообщения**

```
datamartStatisticRequestMessage:
  description: Запрос статистики витрины
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        $ref: '#/components/schemas/datamartStatisticRequestKey'
  headers:
    type: object
    properties:
      REQUEST_ID:
        description: Идентификатор запроса
        type: string
  payload:
    $ref: '#/components/schemas/datamartStatisticRequest'
  examples:
    - name: simple
      headers:
        REQUEST_ID: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      payload:
        protocol: read.statistic.protocol.v.1
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        datamart:
          mnemonic: fias
          version:
            major: 1
            minor: 0
```

**Avro-схема сообщения**

```
datamartStatisticRequest:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: DatamartStatisticRequest
  namespace: ru.rtlabs.common.statistic
  fields:
    - name: protocol
      description: Версия протокола. Указывается константа read.statistic.protocol.v.1
```

```
      type: string
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: datamart
      description: Витрина
      type:
        type: record
        name: DatamartInfo
        fields:
          - name: mnemonic
            description: Мнемоника витрины
            type: string
          - name: version
            description: Версия
            type:
              type: record
              name: SemanticVersion
              namespace: ru.rtlabs.common.model.metadata
              fields:
                - name: major
                  type: int
                  minimum: 1
                - name: minor
                  type: int
                  minimum: 0
```

### 1.3.5 statistics.rs

`statistics.rs` - Топик со статистикой витрины

**Структура сообщения**

```
datamartStatisticResponseMessage:
  description: Статистика витрины
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор запроса
  headers:
    type: object
    properties:
      REQUEST_ID:
        description: Идентификатор запроса
        type: string
  payload:
    $ref: '#/components/schemas/datamartStatisticResponse'
  examples:
    - name: simple
      headers:
        REQUEST_ID: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      payload:
        protocol: read.statistic.protocol.v.1
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        datamart:
          mnemonic: fias
          version:
            major: 1
```

```
          minor: 0
        tables:
          - mnemonic: addrobj
            columns:
              - mnemonic: oktmo
                notGreater10: 10.0
                inRange11And100: 50.0
                inRange101And1000: 30.0
                moreThan1000: 10.0
```

**Avro-схема сообщения**

```
datamartStatisticResponse:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: DatamartStatisticResponse
  namespace: ru.rtlabs.common.datamart.profile
  fields:
    - name: protocol
      description: Версия протокола. Указывается константа read.statistic.protocol.v.1
      type: string
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: datamart
      description: Статистика по витрине
      type:
        type: record
        name: DatamartStatistic
        fields:
          - name: mnemonic
            description: Мнемоника витрины
            type: string
          - name: version
            description: Версия
            type:
              type: record
              name: SemanticVersion
              namespace: ru.rtlabs.common.model.metadata
              fields:
                - name: major
                  type: int
                  minimum: 1
                - name: minor
                  type: int
                  minimum: 0
          - name: tables
            type:
              type: array
              items:
                type: record
                name: TableStatistic
                fields:
                  - name: mnemonic
                    description: Мнемоника витрины
                    type: string
                  - name: columns
                    description: Колонки
                    type:
                      type: array
```

```
                    items:
                      type: record
                      name: ColumnStatistic
                      description: Статистика по колонке
                      fields:
                        - name: mnemonic
                          type: string
                        - name: notGreater10
                          type: double
                        - name: inRange11And100
                          type: double
                        - name: inRange101And1000
                          type: double
                        - name: moreThan1000
                          type: double
```

### 1.3.6 statistics.err

`statistics.err` - Топик с ошибками получения статистики витрины

**Структура сообщения**

```
datamartStatisticErrorMessage:
  description: Неуспешный результат обработки запроса на получение статистики
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор запроса
  headers:
    type: object
    properties:
      REQUEST_ID:
        description: Идентификатор запроса
        type: string
  payload:
    $ref: '#/components/schemas/datamartStatisticError'
  examples:
    - name: simple
      headers:
        REQUEST_ID: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      payload:
        protocol: read.statistic.protocol.v.1
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        errorCode: DATAMART-001
        message: Непредвиденная ошибка
```

**Avro-схема сообщения**

```
datamartStatisticError:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: DatamartStatisticError
  namespace: ru.rtlabs.common.statistic
  fields:
    - name: protocol
      type: string
      description: Версия протокола. Указывается константа read.statistic.protocol.v.1
      conts: read.statistic.protocol.v.1
    - name: requestId
      description: Уникальный идентификатор запроса
```

```
      type:
        type: string
        logicalType: uuid
    - name: errorCode
      description: Код ошибки
      type: string
    - name: message
      description: Сообщение об ошибке
      type: string
```

### 1.4 Запрос данных по регламентированным запросам

Данная спецификация описывает возможность запроса данных по регламентированным запросам

#### 1.4.1 procedure.query.rq

`procedure.query.rq` - Топик регламентированных запросов на исполнение

**Структура сообщения**

```
examples:
  - name: simple
    summary: Простой запрос на исполнение без параметров
    headers:
      MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
    payload:
      requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
      replyTo: agent-fias
      datamartMnemonic: fias
      sql: select * from v1_addrobj
      parameters: [ ]
      namedParams: [ ]
      tableParams: [ ]
      isForEstimation: false
      rowCountThreshold: -1
      customerId: aaa
      customerOgrn: ""
      queryMnemonic: fias.selectAllAddrobj.1.0
  - name: estimation
    summary: Запрос на оценку
    headers:
      MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
    payload:
      requestId: 403eada5-05f6-480c-bca9-03328091efeb
      subRequestId: 451000b8-dff2-4a1b-ab1b-42500a70d232
      replyTo: agent-fias
      datamartMnemonic: fias
      sql: select * from v1_addrobj
      parameters: [ ]
      namedParams: [ ]
      tableParams: [ ]
      isForEstimation: true
      rowCountThreshold: 1000
      customerId:
        string: agent-fias
      customerOgrn:
        string: "1053600591197"
      queryMnemonic:
        string: fias.selectAllAddrobj.1.0
  - name: complex
    summary: Запрос с параметрами и табличными параметрами
```

```
    headers:
      MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
    payload:
      requestId: 68758a92-0027-4258-bf17-aa3d24f85094
      subRequestId: 96e6eb99-7ff1-4efa-abae-ef1c5744b723
      replyTo: agent-fias
      datamartMnemonic: fias
      sql: select * from v1_addrobj where oktmo = ? and name = @tbl.fullname
      parameters:
        - type: STRING
          value:
            string: asdasdasd
        - type: LONG
          value: null
      namedParams: [ ]
      tableParams: [ ]
      isForEstimation: false
      rowCountThreshold: -1
      customerId:
        string: agent-fias
      customerOgrn:
        string: "1053600591197"
      queryMnemonic:
        string: fias.selectAddrobjWithParams.1.0
  - name: complex_named
    summary: Запрос с именованными параметрами
    headers:
      MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
    payload:
      requestId: 12358a92-0027-4258-bf17-aa3d24f85094
      subRequestId: 56e6eb99-7ff1-4efa-abae-ef1c5744b723
      replyTo: agent-fias
      datamartMnemonic: fias
      sql: select * from @tbl.fullname el LEFT JOIN v1_addrobj where oktmo = @p1 and kod
= @p2
      parameters: [ ]
      namedParams:
        - name: p1
          type: STRING
          value:
            string: asdasdasd
        - name: p2
          type: LONG
          value: null
      tableParams: [ ]
      isForEstimation: false
      rowCountThreshold: -1
      customerId:
        string: agent-fias
      customerOgrn:
        string: "1053600591197"
      queryMnemonic:
        string: fias.selectAddrobjWithParams.1.0
  - name: deadline
    summary: Простой запрос на исполнение без параметров
    headers:
      MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
      QUERY_DEADLINE: 1629289006904
    payload:
      requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
      replyTo: agent-fias
```

```
    datamartMnemonic: fias
    sql: select * from v1_addrobj
    parameters: [ ]
    namedParams: [ ]
    tableParams: [ ]
    isForEstimation: false
    rowCountThreshold: -1
    customerId: agent-fias
    customerOgrn: "1053600591197"
    queryMnemonic: fias.selectAllWithDeadline.1.0
```

**Avro-схема сообщения**

```
datamartExecuteQueryRequest:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: QueryRequest
  namespace: datamart.query
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      type:
        type: string
        logicalType: uuid
    - name: replyTo
      description: Служебная информация маршрутизации сообщения. Ответ, формируемый
витриной, обязан содержать переданное значение без каких либо искажений
      type: string
    - name: datamartMnemonic
      description: Мнемоника витрины, к которой выполняется запрос
      type: string
    - name: sql
      description: SQL запрос на исполнение, либо имя хранимой процедуры для
регламентированных запросов
      type: string
    - name: parameters
      description: Параметры к SQL запросу
      default: [ ]
      type:
        type: array
        items:
          type: record
          name: QueryParameter
          description: Описание параметра
          fields:
            - name: type
              type: string
              description: Тип параметра
              enum:
                - BIG_DECIMAL
                - BINARY
                - BOOLEAN
                - DATE
                - DOUBLE
                - FLOAT
                - INTEGER
                - LONG
```

```
            - SHORT
            - STRING
            - TIME
            - TIMESTAMP
        - name: value
          description: Значение параметра
          type:
            - string
            - 'null'
  - name: namedParams
    description: Именованные параметры запроса
    default: [ ]
    type:
      type: array
      items:
        type: record
        name: NamedParam
        description: Описание именованного параметра
        fields:
          - name: name
            description: Имя (мнемоника) параметра
            type: string
          - name: type
            type: string
            description: Тип параметра
            enum:
              - BIG_DECIMAL
              - BINARY
              - BOOLEAN
              - DATE
              - DOUBLE
              - FLOAT
              - INTEGER
              - LONG
              - SHORT
              - STRING
              - TIME
              - TIMESTAMP
          - name: value
            description: Значение параметра
            type:
              - string
              - 'null'
  - name: tableParams
    description: ¥use only Datamart¥ Табличные параметры запроса
    default: [ ]
    type:
      type: array
      items:
        type: record
        name: TableParam
        fields:
          - name: id
            description: Уникальный идентификатор
            type:
              type: string
              logicalType: uuid
          - name: name
            description: Имя параметра
            type: string
          - name: columns
            description: Описание колонок таблицы
```

```
            type:
              type: array
              items:
                type: record
                description: Описание колонки
                name: TableParamColumnInfo
                fields:
                  - name: name
                    type: string
                    description: Имя колонки
                  - name: type
                    type: string
                    description: Тип атрибута
                    enum:
                      - BIG_DECIMAL
                      - BINARY
                      - BOOLEAN
                      - DATE
                      - DOUBLE
                      - FLOAT
                      - INTEGER
                      - LONG
                      - SHORT
                      - STRING
                      - TIME
                      - TIMESTAMP
    - name: isForEstimation
      description: Признак необходимости вернуть статистику по запросу в качестве
результата. В случае, если оценка по результату исполнения sql запроса не превышает
rowCountThreshold записей, должен сразу отдаваться результат без отправки оценки в ядро
      type: boolean
      default: false
    - name: rowCountThreshold
      description: Максимальное оценочное количество строк результата, при превышении
которого возвращается статистика по запросу. Если оценка по запросу не превышет данный
параметр, витрина сразу возвращает ответ с результатом. Заполняется в случае
isForEstimation = true
      type: long
      default: -1
    - name: customerId
      description: Мнемоника ИС Потребителя
      type:
        - 'null'
        - string
      default: null
    - name: customerOgrn
      description: ОГРН ИС Потребителя
      type:
        - 'null'
        - string
      default: null
    - name: queryMnemonic
      description: 'Мнемоника РЗ, сформированная по правилу: <мнемоника
витрины>.<мнемоника РЗ>.<версия РЗ> Если запрос распределенный, то формируется по
правилу: podd.<мнемоника РЗ>.<версия РЗ>'
      type:
        - 'null'
        - string
      default: null
```

### 1.4.2 procedure.query.rs

`procedure.query.rs` - Топик с чанками данных исполнения запросов

**Структура сообщения**

```
datamartExecuteQueryResultChunk:
schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
type: record
name: QueryResultChunk
namespace: datamart.query
fields:
    - name: requestId
    description: Уникальный идентификатор запроса
    type:
        type: string
        logicalType: uuid
    - name: subRequestId
    description: Уникальный идентификатор подзапроса
    type:
        type: string
        logicalType: uuid
    - name: replyTo
    description: Служебная информация маршрутизации сообщения. Заполняется
соответсвующим значением из запроса
    type: string
    - name: chunkNumber
    description: Номер порции по порядку
    type: int
    minimum: 1
    - name: isLastChunk
    description: Признак последнего сообщения
    type: boolean
    - name: streamNumber
    description: Номер стрима данных
    minimum: 1
    type:
        - int
        - "null"
    - name: streamTotal
    description: Общее количество стримов
    minimum: 1
    type:
        - int
        - "null"
    - name: isFragmented
    description: Признак присутствия в чанке неполных строк (строк, которые были
разбиты на несколько чанков)
    type: boolean
    - name: uncompressedSize
    description: Оригинальный размер чанка в байтах
    type: int
    minimum: 0
```

**Пример key query.rs**

```
examples:
- requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
    subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
    replyTo: agent-fias
    chunkNumber: 1
    isLastChunk: true
    streamNumber:
    int: 1
    streamTotal:
    int: 1
    isFragmented: false
    uncompressedSize: 10
```

### 1.4.3 procedure.query.err

`procedure.query.err` - Топик с ошибками исполнения sql запросов на витрине

**Структура сообщения**

```
datamartExecuteQueryError:
schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
type: record
name: QueryError
namespace: datamart.query
fields:
    - name: requestId
    description: Уникальный идентификатор запроса
    type:
        type: string
        logicalType: uuid
    - name: subRequestId
    description: Уникальный идентификатор подзапроса
    type:
        type: string
        logicalType: uuid
    - name: replyTo
    description: Служебная информация маршрутизации сообщения. Заполняется
соответсвующим значением из запроса
    type: string
    - name: errorCode
    description: Код возникшей ошибки
    type: string
    - name: message
    description: Сообщение с ошибкой исполнения
    type: string
```

**Пример query.err**

```
examples:
- name: error
    summary: Сообщение с ошибкой исполнения запроса на витрине без header
    payload:
    requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
    subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
    replyTo: agent-fias
    errorCode: DATAMART-001
    message: Непредвиденная ошибка
- name: errorWithHeader
    summary: Сообщение с ошибкой исполнения запроса на витрине
    headers:
    MESSAGE_TYPE: DatamartExecuteQueryError:0.1
    payload:
    requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
    subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
    replyTo: agent-fias
    errorCode: DATAMART-001
    message: Непредвиденная ошибка
```

## 1.5 Запрос метаданных

Данная спецификация описывает возможность запроса метаданных Витрины

### 1.5.1 metadata.rq

Передача Агентом ПОДД запроса метаданных в Витрину.

**Формат сообщения**

| | |
|---|---|
| Header | Не используется |
| Key | текст, содержит requestId, не используется. |
| Value | Сериализация: в json (см. схему ниже.) |

**Схема**

```
{
"$schema": "http://json-schema.org/draft-04/schema#",
"type": "object",
"properties": {
        "requestId": {
        "type": "string"
        },
        "datamartMnemonic": {
        "type": "string"
        }
},
"required": [
        "requestId",
        "datamartMnemonic"
]
}
```

, где:

- requestId - UUID запроса;
- datamartMnemonic - мнемоника Витрины данных, к которой адресован запрос.

### 1.5.2 metadata.rs

Передача Агенту ПОДД ответа на запрос метаданных витрины из Витрины.

**Формат сообщения**

| | |
|---|---|
| Header | Не используется |
| Key | текст, содержит requestId |
| Value | Сериализация: в json (см. схему ниже.) |

**Схема**

```
{
"$schema": "http://json-schema.org/draft-04/schema#",
"type": "object",
"properties": {
        "requestId": {
        "type": "string"
        },
        "metadata": {
        "type": "array",
        "items": [
                {
                "type": "object",
                "properties": {
                        "datamart": {
                        "type": "object",
                        "properties": {
                                "id": {
                                "type": "string"
                                },
                                "mnemonic": {
                                "type": "string"
                                },
                                "datamartClassess": {
                                "type": "array",
                                "items": [
                                        {
                                        "type": "object",
                                        "properties": {
                                                "id": {
                                                "type": "string"
                                                },
                                                "mnemonic": {
                                                "type": "string"
                                                },
                                                "label": {
                                                "type": "string"
                                                },
                                                "classAttributes": {
                                                "type": "array",
                                                "items": [
                                                        {
                                                        "type": "object",
                                                        "properties": {
                                                                "id": {
                                                                "type": "string"
                                                                },
                                                                "mnemonic": {
                                                                "type": "string"
                                                                },
                                                                "type": {
                                                                "type": "object",
                                                                "properties": {
                                                                        "id": {
                                                                        "type": "string"
                                                                        },
```

```
                                        "value": {
                                        "type": "string"
                                        }
                                },
                                "required": [
                                        "id",
                                        "value"
                                ]
                                }
                        },
                        "required": [
                                "id",
                                "mnemonic",
                                "type"
                        ]
                        }
                ]
                },
                "primaryKey": {
                "type": "array",
                "items": [
                        {
                        "type": "object",
                        "properties": {
                                "id": {
                                "type": "string"
                                },
                                "mnemonic": {
                                "type": "string"
                                },
                                "type": {
                                "type": "object",
                                "properties": {
                                        "id": {
                                        "type": "string"
                                        },
                                        "value": {
                                        "type": "string"
                                        }
                                },
                                "required": [
                                        "id",
                                        "value"
                                ]
                                }
                        },
                        "required": [
                                "id",
                                "mnemonic",
                                "type"
                        ]
                        }
                ]
                }
        },
        "required": [
                "id",
                "mnemonic",
                "label",
                "classAttributes",
                "primaryKey"
        ]
```

```
                                        }
                                ]
                                }
                        },
                        "required": [
                                "id",
                                "mnemonic",
                                "datamartClassess"
                        ]
                        }
                },
                "required": [
                        "datamart"
                ]
                }
        ]
        }
},
"required": [
        "requestId",
        "metadata"
]
}
```

, где:

- `requestId` - UUID запроса;
- `metadata` – описание структуры данных;
- `datamart` – описание витрины;
- `id` – UUID витрины (не используется);
- `mnemonic` – имя витрины;
- `datamartClassess` – список таблиц витрины;
- `id` – UUID таблицы (не используется);
- `mnemonic` – имя таблицы;
- `label`– не используется;
- `classAttributes` – список полей таблицы;
- `id` – UUID поля (не используется);
- `mnemonic` – имя поля;
- `type` – тип данных поля;
- `id` – UUID типа данных (не используется);
- `value` – название типа данных;
- `primaryKey` – список полей, составляющих PK таблицы;
- `id` – UUID поля (не используется);
- `mnemonic` – имя поля;
- `type` – тип данных поля;
- `id` – UUID типа данных (не используется);
- `value` – название типа данных.

### 1.5.3 metadata.err

Получение Агентом ПОДД ошибки при обработке запроса метаданных от Витрины.

**Формат сообщения**

| Header | Не используется |
|---|---|
| Key | текст, содержит requestId |

| | |
|---|---|
| Value | Сериализация: в json (см. схему ниже.) |

**Схема**

```
{
"$schema": "http://json-schema.org/draft-04/schema#",
"type": "object",
"properties": {
        "requestId": {
        "type": "string"
        },
        "errorCode": {
        "type": "string"
        },
        "msg": {
        "type": "string"
        }
},
"required": [
        "requestId",
        "errorCode",
        "msg"
]
}
```

, где:

- `requestId` - UUID запроса;
- `errorCode` – содержит константу `INTERNAL`;
- `msg` – описание ошибки.

# 2 Спецификация модуля «BLOB-адаптер»

## 2.1 Запрос на считывание BLOB

Настоящая спецификация определяет формат обмена электронными сообщениями через BLOB-адаптер. Описывает возможность запроса на считывание BLOB-объект по полученной ссылке, получения успешного ответа на чтение содержимого BLOB или ошибки, в случае невозможности считывания, с описанием причины ошибки.

| Топик | Назначение |
|---|---|
| blob.rq | Запросы на считывание BLOB'а по ссылке. |
| blob.rs | Содержимое BLOB'а (ответ на запрос). |
| blob.err | Сообщения об ошибке считывания. |

Для строковых параметров используется кодировка UTF-8.

### 2.2.1 blob.rq

`blob.rq` - Топик запросов на получение бинарных данных по полученной ранее ссылке.

**Структура сообщения**

```
datamartBlobRequestMessage:
  description: Запрос бинарных данных по ссылке
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
```

```
      description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      REQUEST_ID:
        description: Идентификатор запроса
        type: string
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
      MESSAGE_TYPE:
        description: Тип сообщения
        type: string
        const: DatamartBlobRequest:0.1
  payload:
    $ref: '#/components/schemas/datamartBlobRequest'
  examples:
    - name: getBlobDataRequest
      summary: Запрос бинарных данных по ссылке
      headers:
        REQUEST_ID: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        AGENT_CONSUMER_ID: agent-fias
        MESSAGE_TYPE: DatamartBlobRequest:0.1
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        queryRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
        reference:
          subRequestId: 4cbb11d6-47de-4928-953f-47dfa6c6b310
          path: reference
```

**Avro-схема сообщения**

```
datamartBlobRequest:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: BlobRequest
  namespace: datamart.blob
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: queryRequestId
      description: Идентификатор исходного запроса, в рамках которого была получена
ссылка
      type:
        type: string
        logicalType: uuid
    - name: reference
      description: Ссылка на данные
      type:
        type: record
        name: BinaryReference
        namespace: query.result
        fields:
          - name: subRequestId
            description: Идентификатор подзапроса
            type:
              type: string
              logicalType: uuid
          - name: path
```

```
          description: Ссылка
          type: string
```

### 2.2.2 blob.rs

`blob.rs` - Топик с бинарными данным блобов

**Структура сообщения**

```
datamartBlobChunkMessage:
  description: Чанки бинарных данных
  contentType: 'application/octet-stream'
  bindings:
    kafka:
      key:
        $ref: '#/components/schemas/datamartBlobChunkInfo'
  headers:
    type: object
    properties:
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
      MESSAGE_TYPE:
        description: Тип сообщения
        type: string
        const: DatamartBlobChunkInfo:0.1
  payload:
    description: Бинарные данные
  examples:
    - name: base64
      headers:
        MESSAGE_TYPE: DatamartBlobChunkInfo:0.1
      payload:
        value: JEEJNodyLO7p1pgsRHG9pEiXeYGvHW4YCl4FgrgBmu5C92iVX1PV2GZdcqsb66bx8sk=
```

**Avro-схема сообщения**

```
datamartBlobChunkInfo:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: BlobChunk
  namespace: datamart.blob
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: queryRequestId
      description: Идентификатор исходного запроса, в рамках которого была получена
ссылка
      type:
        type: string
        logicalType: uuid
    - name: chunkNum
      description: Номер чанка
      type: int
      minimum: 1
    - name: isLast
      description: Признак последнего чанка
      type: boolean
  examples:
    - requestId: 3546e40b-47fe-41b6-9c06-a2e915eb4181
```

```
      queryRequestId: a8e9f47b-38cd-4db6-a245-0fbd6e78c195
      chunkNum: 1
      isLast: true
```

### 2.2.3 blob.err

`blob.err` - Топик с ошибками получения бинарных данных по ссылке.

**Структура сообщения**

```
datamartBlobErrorResponseMessage:
  description: Ошибка получения бинарных данных по ссылке
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
      MESSAGE_TYPE:
        description: Тип сообщения
        type: string
        const: DatamartBlobErrorResponse:0.1
  payload:
    $ref: '#/components/schemas/datamartBlobErrorResponse'
  examples:
    - name: blobError
      summary: Пример ошибки получения бинарных данных по ссылке
      headers:
        AGENT_CONSUMER_ID: agent-fias
        MESSAGE_TYPE: DatamartBlobErrorResponse:0.1
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        queryRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
        errorCode: DATAMART-001
        errorMessage: Непредвиденная ошибка обработки
```

**Avro-схема сообщения**

```
datamartBlobErrorResponse:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: BlobError
  namespace: datamart.blob
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: queryRequestId
      description: Идентификатор исходного запроса, в рамках которого была получена
ссылка
      type:
        type: string
        logicalType: uuid
    - name: errorCode
```

```
      description: Код ошибки
      type: string
    - name: errorMessage
      description: Сообщение с ошибкой
      type: string
```

# 3 Спецификация модуля «Сервис Формирования документов»

## 3.1 Запрос формирования документов

Данная спецификация описывает возможность запроса на генерацию формирования файлов, получения успешного ответа (сгенерированных файлов и их метаданных) или ошибки, в случае невозможности сгенерировать файлы, с описанием причины ошибки.

| Топик | Назначение |
|---|---|
| Report.rq | Запросы на генерацию файлов. |
| Report.rs | Содержимое сгенерированных файлов (ответ на запрос). |
| Report.err | Сообщения об ошибке генерации. |

### 3.1.1 report.rq

> Внимание:
>
> Название топика может быть изменено на этапе внедрения!

Запрос на генерацию файлов. Одно сообщение - один запрос. Один запрос - один набор параметров (в наборе м.б. много параметров, в параметрах м.б. указано «сгенерируй много форматов файлов на основании одной и той же выборки данных»).

**Структура сообщения**

```
datamartExecuteQueryRequestMessage:
description: Исполнение sql запроса на витрине
schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
bindings:
    kafka:
    key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
headers:
    type: object
    properties:
    MESSAGE_TYPE:
        description: Тип сообщения
        type: string
        const: DatamartExecuteQueryRequest:0.1
    REQUEST_ID:
        description: Идентификатор запроса
        type: string
    QUERY_DEADLINE:
        description: Время в миллисекундах от эпохи, до которого запрос должен быть
выполнен
        type: string
        format: int64
    AGENT_CONSUMER_ID:
        description: Мнемоника потребителя (мнемоника агента)
        type: string
    QUERY_MNEMONIC:
        description: '<Полная мнемоника РЗ>.<версия РЗ>'
```

```
        type: string
payload:
    $ref: '#/components/schemas/datamartExecuteQueryRequest'
examples:
    - name: report
    summary: Запрос к Сервису формирования документов
    headers:
        MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
    payload:
        requestId: 68758a92-0027-4258-bf17-aa3d24f85094
        subRequestId: 96e6eb99-7ff1-4efa-abae-ef1c5744b723
        replyTo: agent-fias
        datamartMnemonic: fias
        sql: v1_printable_form_address
        parameters:
        - type: STRING
            value:
            string:
MIIB9wYJKoZIhvcNAQcCoIIB6DCCAeQCAQExADALBgkqhkiG9w0BBwGgggHMMIIByDCCAXOgAwIBAgIEV/dqTjA
MBggqhQMHAQEDAgUAMDUxCzAJBgNVBAYTAlJVMQswCQYDVQQKEwJSVDEZMBcGA1UEAwwQYmxhc3RvZmZfY2FfdG
VzdDAeFw0yMjAzMDQwNzQ5MzBaFw0zMjAzMDEwNzQ5MzBaMC0xETAPBgNVBAMMCGl0b25lZGV2MQswCQYDVQQKD
AJSVDELMAk
        - type: STRING
            value:
            string: xml
        namedParams: [ ]
        tableParams: [ ]
        isForEstimation: false
        rowCountThreshold: -1
        customerId:
        string: agent-fias
        customerOgrn:
        string: "1053600591197"
        queryMnemonic:
        string: fias.selectAddrobjWithParams.1.0
```

**Avro-схема сообщения**

```
datamartExecuteQueryRequest:
schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
type: record
name: QueryRequest
namespace: datamart.query
fields:
    - name: requestId
    description: Уникальный идентификатор запроса
    type:
        type: string
        logicalType: uuid
    - name: subRequestId
    description: Уникальный идентификатор подзапроса
    type:
        type: string
        logicalType: uuid
    - name: replyTo
    description: Служебная информация маршрутизации сообщения. Ответ, формируемый
витриной, обязан содержать переданное значение без каких либо искажений
    type: string
    - name: datamartMnemonic
    description: Мнемоника витрины, к которой выполняется запрос
    type: string
    - name: sql
```

```
    description: тип отчета
    type: string
    - name: parameters
    description: Параметры к SQL запросу (при запросе к сервису формирования
документов, первым параметром ВСЕГДА идет сертификат, а вторым формат файла (xml или
pdf))
    default: [ ]
    type:
        type: array
        items:
        type: record
        name: QueryParameter
        description: Описание параметра
        fields:
            - name: type
            type: string
            description: Тип параметра
            enum:
                - BIG_DECIMAL
                - BINARY
                - BOOLEAN
                - DATE
                - DOUBLE
                - FLOAT
                - INTEGER
                - LONG
                - SHORT
                - STRING
                - TIME
                - TIMESTAMP
            - name: value
            description: Значение параметра
            type:
                - string
                - 'null'
    - name: namedParams
    description: Именованные параметры запроса
    default: [ ]
    type:
        type: array
        items:
        type: record
        name: NamedParam
        description: Описание именованного параметра
        fields:
            - name: name
            description: Имя (мнемоника) параметра
            type: string
            - name: type
            type: string
            description: Тип параметра
            enum:
                - BIG_DECIMAL
                - BINARY
                - BOOLEAN
                - DATE
                - DOUBLE
                - FLOAT
                - INTEGER
                - LONG
                - SHORT
                - STRING
```

```
                - TIME
                - TIMESTAMP
            - name: value
            description: Значение параметра
            type:
                - string
                - 'null'
    - name: tableParams
    description: Табличные параметры запроса
    default: [ ]
    type:
        type: array
        items:
        type: record
        name: TableParam
        fields:
            - name: id
            description: Уникальный идентификатор
            type:
                type: string
                logicalType: uuid
            - name: name
            description: Имя параметра
            type: string
            - name: columns
            description: Описание колонок таблицы
            type:
                type: array
                items:
                type: record
                description: Описание колонки
                name: TableParamColumnInfo
                fields:
                    - name: name
                    type: string
                    description: Имя колонки
                    - name: type
                    type: string
                    description: Тип атрибута
                    enum:
                        - BIG_DECIMAL
                        - BINARY
                        - BOOLEAN
                        - DATE
                        - DOUBLE
                        - FLOAT
                        - INTEGER
                        - LONG
                        - SHORT
                        - STRING
                        - TIME
                        - TIMESTAMP
    - name: isForEstimation
    description: Признак необходимости вернуть статистику по запросу в качестве
результата. В случае, если оценка по результату исполнения sql запроса не превышает
rowCountThreshold записей, должен сразу отдаваться результат без отправки оценки в ядро
    type: boolean
    default: false
    - name: rowCountThreshold
    description: Максимальное оценочное количество строк результата, при превышении
которого возвращается статистика по запросу. Если оценка по запросу не превышет данный
параметр, витрина сразу возвращает ответ с результатом. Заполняется в случае
```

```
isForEstimation = true
    type: long
    default: -1
    - name: customerId
    description: Мнемоника ИС Потребителя
    type:
        - 'null'
        - string
    default: null
    - name: customerOgrn
    description: ОГРН ИС Потребителя
    type:
        - 'null'
        - string
    default: null
    - name: queryMnemonic
    description: 'Мнемоника РЗ, сформированная по правилу: <мнемоника
витрины>.<мнемоника РЗ>.<версия РЗ> Если запрос распределенный, то формируется по
правилу: podd.<мнемоника РЗ>.<версия РЗ>'
    type:
        - 'null'
        - string
    default: null
```

### 3.1.2 report.rs

> Внимание:
>
> Название топика может быть изменено на этапе внедрения!

Позитивный ответ на запрос - содержимое сгенерированных файлов и метаданные для них, передается только в случае успешного выполнения генерации. Один запрос - один ответ. Один ответ - несколько сообщений. Одно сообщение - один chunk.

**Структура сообщения**

```
datamartExecuteQueryResultChunkMessage:
  description: Чанк с данными по исполнению запроса
  contentType: 'application/octet-stream'
  bindings:
    kafka:
      key:
        $ref: '#/components/schemas/datamartExecuteQueryResultChunk'
  headers:
    type: object
    properties:
      MESSAGE_TYPE:
        description: Тип сообщения
        type: string
        const: DatamartExecuteQueryResultChunk:0.1
  payload:
    description: Бинарные данные чанка
  examples:
    - name: base64
      headers:
        MESSAGE_TYPE: DatamartExecuteQueryResultChunk:0.1
      payload:
        value: JEEJNodyLO7p1pgsRHG9pEiXeYGvHW4YCl4FgrgBmu5C92iVX1PV2GZdcqsb66bx8sk=
```

**Avro-схема Key сообщения**

```
datamartExecuteQueryResultChunk:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
```

```yaml
  type: record
  name: QueryResultChunk
  namespace: datamart.query
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      type:
        type: string
        logicalType: uuid
    - name: replyTo
      description: Служебная информация маршрутизации сообщения. Заполняется
соответсвующим значением из запроса
      type: string
    - name: chunkNumber
      description: Номер порции по порядку
      type: int
      minimum: 1
    - name: isLastChunk
      description: Признак последнего сообщения
      type: boolean
    - name: streamNumber
      description: Номер стрима данных
      minimum: 1
      type:
        - int
        - "null"
    - name: streamTotal
      description: Общее количество стримов
      minimum: 1
      type:
        - int
        - "null"
    - name: isFragmented
      description: Признак присутствия в чанке неполных строк (строк, которые были
разбиты на несколько чанков)
      type: boolean
    - name: uncompressedSize
      description: Оригинальный размер чанка в байтах
      type: int
      minimum: 0
  examples:
    - requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
      replyTo: agent-fias
      chunkNumber: 1
      isLastChunk: true
      streamNumber:
        int: 1
      streamTotal:
        int: 1
      isFragmented: false
      uncompressedSize: 10
```

**Avro-схема Value (для report.rs она универсальная)**

```yaml
{
  "type": "record",
```

```
  "name": "QueryResultRow",
  "namespace": "datamart.query",
  "fields": [
    {
      "name": "DocType",
      "type": "string"
    },
    {
      "name": "FileName",
      "type": "string"
    },
    {
      "name": "Content",
      "type": "bytes"
    },
    {
      "name": "Meta",
      "type": ["string", "null"]
    }
  ]
}
```

### 3.1.3 report.err

> Внимание:
>
> Название топика может быть изменено на этапе внедрения!

Негативный ответ на запрос - описание причины ошибки, передается только в случае невозможности выполнения запроса (если для одного из форматов в запросе не настроена генерация, то возвращаются настроенные форматы и это не считается ошибкой). Один запрос - один ответ (об ошибке). Один ответ - одно сообщение.

**Структура сообщения**

```
datamartExecuteQueryErrorMessage:
  description: Ошибка исполнения запроса
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      MESSAGE_TYPE:
        description: Тип сообщения
        type: string
        const: DatamartExecuteQueryError:0.1
  payload:
    $ref: '#/components/schemas/datamartExecuteQueryError'
  examples:
    - name: error
      summary: Сообщение с ошибкой исполнения запроса на витрине без header
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
        replyTo: agent-fias
        errorCode: DATAMART-001
        message: Непредвиденная ошибка
```

```
    - name: errorWithHeader
      summary: Сообщение с ошибкой исполнения запроса на витрине
      headers:
        MESSAGE_TYPE: DatamartExecuteQueryError:0.1
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
        replyTo: agent-fias
        errorCode: DATAMART-001
        message: Непредвиденная ошибка
```

**Avro-схема сообщения**

```
datamartExecuteQueryError:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: QueryError
  namespace: datamart.query
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      type:
        type: string
        logicalType: uuid
    - name: replyTo
      description: Служебная информация маршрутизации сообщения. Заполняется
соответсвующим значением из запроса
      type: string
    - name: errorCode
      description: Код возникшей ошибки
      type: string
    - name: message
      description: Сообщение с ошибкой исполнения
      type: string
  examples:
    - requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
      replyTo: agent-fias
      errorCode: DATAMART-001
      message: Непредвиденная ошибка
```

# ПРИЛОЖЕНИЕ 2. ПОДДЕРЖКА ФУНКЦИЙ SQL

## 1 SQL-синтаксис

Регламентированные SQL-запросы, регистрируемые в СМЭВ4 должны соответствовать следующему синтаксису.

Таблица 1 SQL-синтаксис

| № | Описание | Пример запроса |
|---|---|---|
| | **Числовые типы данных** | |
| 1 | Типы данных INTEGER и SMALLINT | `SELECT CAST(1 AS INT)` |
| 2 | Типы данных REAL, DOUBLE PRECISION и FLOAT | `SELECT CAST(1 AS FLOAT)` |
| 3 | Типы данных DECIMAL и NUMERIC | `SELECT CAST(1 AS NUMERIC)` |
| 4 | Арифметические операторы | `SELECT 10+1, 9-2, 8*3, 7/2` |
| 5 | Числовые сравнения | `SELECT 1 WHERE 1 < 2` |
| 6 | Неявные преобразования между числовыми типами данных | `SELECT int_column FROM t WHERE int_column = 1.00` |
| **Символьные типы данных** | | |
| 7 | Тип данных CHARACTER<br>Длина по умолчанию 30 | `SELECT CAST('1111111111111111111111111111111111111111' AS CHAR)` |
| 8 | Тип данных CHARACTER VARYING<br>Длина по умолчанию 30 | `SELECT CAST('1111111111111111111111111111111111111111' AS VARCHAR)` |
| 9 | Символьные строки | `SELECT ''` |
| 10 | Функция CHARACTER_LENGTH убирает завершающие пробелы из значений CHARACTER перед подсчётом символов | `SELECT character_length(char_column) FROM t` |
| 11 | Функция OCTET_LENGTH | `SELECT octet_length(char_column) FROM t` |
| 12 | Функция SUBSTRING | `SELECT substring(char_column FROM 1 FOR 1) FROM t` |
| 13 | Конкатенация символьных строк | `SELECT 'a' \|\| 'b' FROM t` |
| 14 | Функции UPPER и LOWER | `SELECT upper('a'),lower('B') FROM t`<br>`SELECT int_column FROM t WHERE int_column > (SELECT DISTINCT (int_column) FROM t)` |
| 15 | Функция TRIM | `SELECT trim('a ') FROM t` |
| 16 | Неявные преобразования между типами символьных строк | `SELECT char_column FROM t WHERE char_column > varchar_column` |
| 17 | Функция POSITION | `SELECT position('A' IN char_column) FROM t` |
| 18 | Сравнения символов | `SELECT char_column FROM t WHERE char_column > 'a'` |
| **Идентификаторы** | | |
| 19 | Идентификаторы с разделителями | `SELECT 1 AS «t47»` |
| 20 | Идентификаторы в нижнем регистре | `SELECT 1 AS t48` |
| 21 | Завершающее подчёркивание | `SELECT 1 AS t49_` |
| **Базовое определение запросов** | | |
| 22 | SELECT DISTINCT | `SELECT DISTINCT int_column FROM t` |
| 23 | Предложение GROUP BY | `SELECT DISTINCT int_column FROM t GROUP BY int_column` |
| 24 | GROUP BY может содержать колонки не из <списка выборки> | `SELECT DISTINCT char_column FROM t GROUP BY lower(char_column)` |
| 25 | Элементы списка выборки могут переименовываться | `SELECT int_column AS K FROM t ORDER BY K` |
| 26 | Предложение HAVING | `SELECT count(*) FROM t HAVING count(*) > 0` |
| 27 | Корреляционные имена в предложении FROM | `SELECT K.column FROM t AS K` |
| 28 | Переименование колонок в | `SELECT column FROM t AS x(q, c)` |

| № | Описание | Пример запроса |
|---|---|---|
| | предложении FROM | |
| **Базовые предикаты и условия поиска** | | |
| 29 | Предикат сравнения | `SELECT column FROM t WHERE 0 = 0` |
| 30 | Предикат BETWEEN | `SELECT column FROM t WHERE ‘ ‘ BETWEEN ‘’ AND ‘’` |
| 31 | Предикат IN со списком значений | `SELECT column FROM t WHERE char_column IN (‘a’, upper(‘a’))` |
| 32 | Предикат LIKE | `SELECT column FROM t WHERE char_column LIKE ‘_’` |
| 33 | Предложение ESCAPE в предикате LIKE | `SELECT column FROM t WHERE ‘abc’ LIKE ‘abcX_’ ESCAPE ‘X’` |
| 34 | Предикат NULL | `SELECT column FROM t WHERE char_column IS NOT NULL` |
| 35 | Предикаты количественного сравнения | `SELECT column FROM t WHERE char_column = ANY (SELECT char_column FROM t)` |
| 36 | Предикат EXISTS | `SELECT column FROM t WHERE NOT EXISTS (SELECT char_column FROM t)` |
| 37 | Подзапросы в предикате сравнения | `SELECT column FROM t WHERE int_column > (SELECT max (int_column) FROM t)` |
| 38 | Подзапросы в предикате IN | `SELECT column FROM t WHERE char_column IN (SELECT char_column FROM t)` |
| 39 | Подзапросы в предикате количественного сравнения | `SELECT column FROM t WHERE char_column >= ALL(SELECT char_column FROM t)` |
| 40 | Коррелирующие подзапросы | `SELECT column FROM t WHERE int_column = (SELECT int_column FROM t2 WHERE t2.char_ column = t.char_column)` |
| 41 | Условие поиска | `SELECT column FROM t WHERE 0 <> 0 OR ‘a’ < ‘b’ AND int_column IS NOT NULL` |
| **Простые выражения с запросами** | | |
| 42 | Табличный оператор UNION DISTINCT | `SELECT column FROM t UNION DISTINCT SELECT column1 FROM t` |
| 43 | Табличный оператор UNION ALL | `SELECT column FROM t UNION ALL SELECT column1 FROM t` |
| 44 | Табличный оператор EXCEPT DISTINCT | `SELECT column FROM t EXCEPT DISTINCT SELECT column1 FROM t` |
| 45 | Колонки, объединяемые табличными операторами, могут иметь разные типы данных | `SELECT char_column FROM t UNION SELECT 5` |
| 46 | Табличные операторы в подзапросах | `SELECT column FROM t WHERE ‘a’ IN (SELECT char_column FROM t UNION SELECT char_column FROM t)` |
| **Функции множеств** | | |
| 47 | AVG | `SELECT avg(int_column) FROM t` |
| 48 | COUNT | `SELECT count(int_column) FROM t` |
| 49 | MAX | `SELECT max(int_column) FROM t` |
| 50 | MIN | `SELECT min(int_column) FROM t` |
| 51 | SUM | `SELECT sum(int_column) FROM t` |
| 52 | Дополнение ALL | `SELECT sum(ALL int_column) FROM t` |
| 53 | Дополнение DISTINCT | `SELECT sum(DISTINCT int_column) FROM t` |
| 54 | Оператор SELECT, возвращающий одну строку | `SELECT count(*) FROM t` |
| **Базовая поддержка курсоров** | | |
| 55 | Колонки ORDER BY, отсутствующие в списке выборки | `SELECT int_column FROM t ORDER BY char_column` |
| 56 | Выражения значений в предложении ORDER BY | `SELECT int_column FROM t ORDER BY -int_column` |
| 57 | Поддержка NULL (NULL вместо значений) | `SELECT int_column FROM t WHERE int_column IS NULL` |
| **Базовое соединение таблиц** | | |
| 58 | Внутреннее соединение (но не обязательно с ключевым словом INNER) | `SELECT a.int_column FROM t a JOIN t b ON a.int_column = b.int_column` |
| 59 | Ключевое слово INNER | `SELECT a.int_column FROM t a JOIN t b ON a.int_column = b.int_column` |

| № | Описание | Пример запроса |
|---|---|---|
| 60 | LEFT OUTER JOIN | `SELECT a.int_column, b.int_column FROM t a LEFT OUTER JOIN t b ON a.int_column = b.int_column` |
| 61 | RIGHT OUTER JOIN | `SELECT a.int_column, b.int_column FROM t a RIGHT OUTER JOIN t b ON a.int_column = b.int_column` |
| 62 | Внешние соединения могут быть вложенными | `SELECT a.int_column FROM t a LEFT OUTER JOIN t b ON a.int_column = b.int_column LEFT OUTER JOIN t2 c ON a.int_column = c.int_column` |
| 63 | Внутренняя таблица с левой или правой стороны внешнего соединения может также участвовать во внутреннем соединении | `SELECT t2.int_column FROM (t2 LEFT OUTER JOIN t ON t.int_column = t2.int_column) j INNER JOIN t2 ON j.int_column = t2.int_column` |
| 64 | Поддерживаются все операторы сравнения (а не только =) | `SELECT column FROM t WHERE 0 = 1 OR 0 > 1 OR 0 < 1 OR 0 <> 1` |
| **Базовая поддержка даты и времени** | | |
| 65 | Тип данных DATE (включая поддержку строк DATE) | `SELECT ‘2012-07-12’ AS "DATE"` |
| 66 | Тип данных TIME (включая поддержку строк TIME) с точностью до секунд как минимум с 0 знаков после запятой | `SELECT ‘1:2:3’ AS "TIME"` |
| 67 | Тип данных TIMESTAMP (включая поддержку строк TIMESTAMP) с точностью до секунд как минимум с 0 и 6 знаками после запятой | `SELECT ‘2012-07-12’ AS "TIMESTAMP"` |
| 68 | Предикаты сравнения с типами данных DATE, TIME и TIMESTAMP | `SELECT column FROM t3 WHERE date_column = date_column AND time_column = time_column AND timestamp_column = timestamp_column` |
| 69 | Явное приведение (CAST) между типами даты/времени и типами символьных строк | `SELECT cast(date_column AS VARCHAR(10)) FROM t3` |
| 70 | CURRENT_DATE | `SELECT current_date FROM t` |
| 71 | LOCALTIME | `SELECT localtime FROM t` |
| 72 | LOCALTIMESTAMP | `SELECT localtimestamp FROM t` |
| **Расширенная поддержка даты и времени** | | |
| 73 | FOR SYSTEM_TIME (запрос данных, актуальных на указанную дату и время) | `SELECT column FROM t FOR SYSTEM_TIME AS OF 'YYYY-MM-DD hh:mm:ss'`<br>`SELECT column FROM t FOR SYSTEM_TIME AS OF TIMESTAMP 'YYYY-MM-DD hh:mm:ss'` |
| 74 | Указание «TIMESTAMP» опционально Вычисление интервала (с указанием единиц времени: YEAR, MONTH, DAY, HOUR, MINUTE, SECOND) | `select ('YYYY-MM-DD hh:mm:ss' - 'YYYY-MM-DD hh:mm:ss') MONTH` |
| 75 | Функция CAST | `SELECT cast(int_column AS INT) FROM t` |
| **Выражение CASE** | | |
| 76 | Простой оператор CASE | `SELECT CASE WHEN 1 = 0 THEN 5 ELSE 7 END FROM t` |
| 77 | Оператор CASE с условиями | `SELECT CASE 1 WHEN 0 THEN 5 ELSE 7 END FROM t` |
| 78 | NULLIF | `SELECT nullif(int_column, 7) FROM t` |
| 79 | COALESCE | `SELECT coalesce(int_column,7) FROM t` |
| 80 | Длинные идентификаторы | `SELECT 1 AS A123456789012345678901234567890123456789` |
| 81 | Спецсимволы Unicode в идентификаторах | `SELECT 1 AS Я123456789012345678901234567890123456789` |
| 82 | Спецсимволы Unicode в текстовых строках | `SELECT U&'\6553'` |
| 83 | Национальные символы | `SELECT 'Я'` |
| 84 | Скалярные значения подзапросов | `SELECT int_column  FROM t WHERE int_column = (SELECT` |

| № | Описание | Пример запроса |
|---|---|---|
| | | `count(*) FROM t)` |
| 85 | Расширенный предикат NULL | `SELECT column FROM t WHERE row(int_column, int_column) IS NOT NULL` |
| **Функции по управлению текстовым поиском** | | |
| 86 | Функция TO_TSVECTOR подготовки текстовых значения, выбранных запросом | `SELECT TO_TSVECTOR('russian', 'иванов иван')` |
| 87 | Функция PLAINTO_TSQUERY подготовки неформатированного текста без сохранения порядка слов | `SELECT PLAINTO_TSQUERY('russian', 'иванов иван')` |
| 88 | Функция TO_TSQUERY подготовки слов разделенных операторорами & (AND), \| (OR), ! (NOT), и (или) <-> (FOLLOWED BY) | `SELECT TO_TSQUERY('russian', 'иванов & иван');` |
| 89 | Функция PHRASETO_TSQUERY подготавки неформатированного текста с сохранением порядка слов | `SELECT PHRASETO_TSQUERY('russian', 'иванов иван')` |
| 90 | Функция WEBSEARCH_TO_TSQUERY подготавки неформатированного текста с сохранением порядка слов или без него. Поддерживает операторы OR и - (NOT) | `SELECT WEBSEARCH_TO_TSQUERY('russian', 'иванов иван OR петр')` |
| 91 | Оператор @@ сравнения текстового вектора с подготовленным или неподготовленным текстом | `SELECT column FROM t WHERE (TO_TSVECTOR('russian', lastname) @@ PLAINTO_TSQUERY('russian', 'иванов иван')` `SELECT column FROM t WHERE (PLAINTO_TSQUERY('russian', 'иванов иван' @@ TO_TSVECTOR('russian', lastname))` `SELECT column FROM t WHERE (lastname @@ PLAINTO_TSQUERY('russian', 'иванов иван')` |

# 2 Поддержка функции LISTAGG

## 2.1 LISTAGG

### 2.1.1 Описание

Функция LISTAGG объединяет значения measure_column для каждой группы на основе order_by_clause.

### 2.1.2 Поддержка в модулях

- Сервис исполнения запросов;
- ПОДД-адаптер – Модуль MPPR.

### 2.1.3 Синтаксис

**Функция LISTAGG в модуле «ПОДД-адаптер – Модуль MPPR»**

Для ПОДД-адаптер – Модуль MPPR реализована поддержка LISTAGG (expression, separator) [WITHIN GROUP (order_by_clause)].

Пример запроса:

```
{"requestId":"4ec61462-0cf5-4b41-84a8-70f215f4109c","subRequestId":"567fbc0a-04b9-43c8-819b-882d3414c2b1","datamartMnemonic":"demo_view","replyTo":"","sql":"SELECT LISTAGG(firstname, ', ') WITHIN GROUP (ORDER BY firstname) AS ¥"firstname_Listing¥" FROM v1_passenger","parameters":[],"tableParams":[],"isForEstimation":false,"rowCountThreshold": 0}
```


Пример ответа:


```
key = [{"requestId": "4ec61462-0cf5-4b41-84a8-70f215f4109c", "subRequestId": "567fbc0a-04b9-43c8-819b-882d3414c2b1", "replyTo": "", "chunkNumber": 1, "isLastChunk": true, "streamNumber": 1, "streamTotal": 1, "uncompressedSize": 0, "isFragmented": false}],

value = [{"firstname_listing": "Григорий, Иван10, Иван11, Иван5, Иван6, Иван7, Иван8, Иван9, Станислав, Станислав"}]
```


**Функция LISTAGG в «Сервисе исполнения запросов»**

Для Сервис исполнения запросов реализована поддержка функции LISTAGG для работы с множественными атрибутами. А(10) Р(10) Д(1) = 21 Простор: А(1) Р(3) Т(2) =6

Пример запроса:


```
{"requestId":"d58b1fa4-e674-4698-857f-f2fb779c9245","subRequestId":"6861b2a8-6821-424b-8b1f-ced06fba75a0","datamartMnemonic":"demo_view","replyTo":"","sql":"SELECT LISTAGG(firstname, ', ') WITHIN GROUP (ORDER BY firstname) AS ¥"firstname_Listing¥" FROM v1_passenger limit 10","parameters":[],"tableParams":[],"isForEstimation":false,"rowCountThreshold": 0}
```


Пример ответа:


```
key = [{"requestId": "d58b1fa4-e674-4698-857f-f2fb779c9245", "subRequestId": "6861b2a8-6821-424b-8b1f-ced06fba75a0", "replyTo": "", "chunkNumber": 1, "isLastChunk": true, "streamNumber": 1, "streamTotal": 1, "uncompressedSize": 0, "isFragmented": false}],

value = [{"firstname_listing": "Григорий, Иван10, Иван11, Иван5, Иван6, Иван7, Иван8, Иван9, Станислав, Станислав"}]
```

# ПРИЛОЖЕНИЕ 3. ПРИМЕР XML-ФАЙЛА СО СТРУКТУРОЙ ВИТРИНЫ

```
<?xml version='1.0' encoding='utf-8'?>
<ns:PODDMetadataRequest
       xmlns:ns="urn://x-artefacts-podd-gosuslugi-local/metadata/datamart/2/1.6.0"
       xmlns:ns1="urn://x-artefacts-podd-gosuslugi-local/metadata/types/1.3">
   <ns:requestId>00000000-0000-0000-0000-000000000001</ns:requestId>
   <ns:metadata>
       <ns1:datamart>
           <ns1:id>1806436d-437a-400d-b32e-aa15c1a2d4bc</ns1:id>
           <ns1:mnemonic>demo_view</ns1:mnemonic>
           <ns1:description>demo_view</ns1:description>
           <ns1:tenantId>c52f062e-af97-4a44-a33f-d1a94024d0cf</ns1:tenantId>
           <ns1:version>
               <ns1:major>1</ns1:major>
               <ns1:minor>0</ns1:minor>
           </ns1:version>
           <ns1:supportedFrom>2021-01-01T00:00:00</ns1:supportedFrom>
           <ns1:datamartClass>
               <ns1:id>4c4ff97b-938b-4db6-9f4d-ae21046e4d20</ns1:id>
               <ns1:mnemonic>Passenger</ns1:mnemonic>
               <ns1:description>Passenger</ns1:description>
               <ns1:classAttribute>
                   <ns1:id>6fe29bdb-7db1-405a-a05c-b49c541c92bd</ns1:id>
                   <ns1:mnemonic>Code</ns1:mnemonic>
                   <ns1:description>Code</ns1:description>
                   <ns1:type>LONG</ns1:type>
               </ns1:classAttribute>
               <ns1:classAttribute>
                   <ns1:id>e590e7b3-b611-4891-bbd1-a5e256105e73</ns1:id>
                   <ns1:mnemonic>Id</ns1:mnemonic>
                   <ns1:description>Id</ns1:description>
                   <ns1:type>STRING</ns1:type>
               </ns1:classAttribute>
               <ns1:classAttribute>
                   <ns1:id>c97a8102-6ad0-4dbd-934d-c82b83a4d83f </ns1:id>
                   <ns1:mnemonic>FirstName</ns1:mnemonic>
                   <ns1:description>FirstName</ns1:description>
                   <ns1:type>STRING</ns1:type>
               </ns1:classAttribute>
               <ns1:classAttribute>
                   <ns1:id>d2312bfb-7ec0-4c95-9026-0f6dea48c5d9</ns1:id>
                   <ns1:mnemonic>MiddleName</ns1:mnemonic>
                   <ns1:description>MiddleName</ns1:description>
                   <ns1:type>STRING</ns1:type>
               </ns1:classAttribute>
               <ns1:classAttribute>
                   <ns1:id>7b63db89-bd0e-4c92-8bc0-e609175937b9</ns1:id>
                   <ns1:mnemonic>LastName</ns1:mnemonic>
                   <ns1:description>LastName</ns1:description>
                   <ns1:type>STRING</ns1:type>
               </ns1:classAttribute>
               <ns1:classAttribute>
                   <ns1:id>8f3e7f95-f66b-4d4a-b2eb-55a3e6134c3e</ns1:id>
                   <ns1:mnemonic>Birthday</ns1:mnemonic>
                   <ns1:description>Birthday</ns1:description>
                   <ns1:type>DATE</ns1:type>
               </ns1:classAttribute>
```

```
            <ns1:classAttribute>
                <ns1:id>e3658240-b405-4838-99af-d32cd063c463</ns1:id>
                <ns1:mnemonic>Passport</ns1:mnemonic>
                <ns1:description>Passport</ns1:description>
                <ns1:type>STRING</ns1:type>
            </ns1:classAttribute>
            <ns1:primaryKey>
                <ns1:id>6fe29bdb-7db1-405a-a05c-b49c541c92bd</ns1:id>
                <ns1:mnemonic>Code</ns1:mnemonic>
                <ns1:description>Code</ns1:description>
                <ns1:type>
                    <ns1:id>00000000-0000-0000-0000-000000000001</ns1:id>
                    <ns1:value>LONG</ns1:value>
                </ns1:type>
            </ns1:primaryKey>
        </ns1:datamartClass>
        <ns1:datamartClass>
            <ns1:id>cafe41db-3878-4796-ba60-cbd54f042c63</ns1:id>
            <ns1:mnemonic>Ticket</ns1:mnemonic>
            <ns1:description>Ticket</ns1:description>
            <ns1:classAttribute>
                <ns1:id>bc90563b-168a-4faa-9394-7b7390dd0d92</ns1:id>
                <ns1:mnemonic>Id</ns1:mnemonic>
                <ns1:description>Id</ns1:description>
                <ns1:type>STRING</ns1:type>
            </ns1:classAttribute>
            <ns1:classAttribute>
                <ns1:id>ac93618f-752b-44d5-a77c-23a3c9eb069b</ns1:id>
                <ns1:mnemonic>PassengerId</ns1:mnemonic>
                <ns1:description>PassengerId</ns1:description>
                <ns1:type>STRING</ns1:type>
            </ns1:classAttribute>
            <ns1:classAttribute>
                <ns1:id>51355519-2d59-426e-b199-9589930acaaa</ns1:id>
                <ns1:mnemonic>TripId</ns1:mnemonic>
                <ns1:description>TripId</ns1:description>
                <ns1:type>STRING</ns1:type>
            </ns1:classAttribute>
            <ns1:classAttribute>
                <ns1:id>fe92c245-929e-4684-b9c9-22bda6939c09</ns1:id>
                <ns1:mnemonic>Number</ns1:mnemonic>
                <ns1:description>Number</ns1:description>
                <ns1:type>LONG</ns1:type>
            </ns1:classAttribute>
            <ns1:classAttribute>
                <ns1:id>4a32ded4-c970-4874-b0b1-2e3eed8b6483</ns1:id>
                <ns1:mnemonic>ByCard</ns1:mnemonic>
                <ns1:description>ByCard</ns1:description>
                <ns1:type>BOOLEAN</ns1:type>
            </ns1:classAttribute>
            <ns1:classAttribute>
                <ns1:id>35f59c80-fcc3-483c-9cd3-dc3afb606d66</ns1:id>
                <ns1:mnemonic>Price</ns1:mnemonic>
                <ns1:description>Price</ns1:description>
                <ns1:type>DOUBLE</ns1:type>
            </ns1:classAttribute>
            <ns1:classAttribute>
                <ns1:id>8b46ff55-6853-458c-851d-6e1666da918b</ns1:id>
                <ns1:mnemonic>Sold</ns1:mnemonic>
                <ns1:description>Sold</ns1:description>
                <ns1:type>TIMESTAMP</ns1:type>
            </ns1:classAttribute>
```

```
                <ns1:primaryKey>
                    <ns1:id>fe92c245-929e-4684-b9c9-22bda6939c09</ns1:id>
                    <ns1:mnemonic>Number</ns1:mnemonic>
                    <ns1:description>Number</ns1:description>
                    <ns1:type>
                        <ns1:id>00000000-0000-0000-0000-000000000001</ns1:id>
                        <ns1:value>LONG</ns1:value>
                    </ns1:type>
                </ns1:primaryKey>
            </ns1:datamartClass>
            <ns1:datamartClass>
                <ns1:id>76268090-60ee-4960-8268-1b91f4186e87</ns1:id>
                <ns1:mnemonic>Trip</ns1:mnemonic>
                <ns1:description>Trip</ns1:description>
                <ns1:classAttribute>
                    <ns1:id>bd173e24-ea7e-4869-9d43-9f57f5b0a82f</ns1:id>
                    <ns1:mnemonic>Id</ns1:mnemonic>
                    <ns1:description>Id</ns1:description>
                    <ns1:type>STRING</ns1:type>
                </ns1:classAttribute>
                <ns1:classAttribute>
                    <ns1:id>1ed32816-8bdb-4d35-9f66-8c08df13ad28</ns1:id>
                    <ns1:mnemonic>Number</ns1:mnemonic>
                    <ns1:description>Number</ns1:description>
                    <ns1:type>INTEGER</ns1:type>
                </ns1:classAttribute>
                <ns1:classAttribute>
                    <ns1:id>78f587fa-b53e-4912-b631-0c4a249d20b6</ns1:id>
                    <ns1:mnemonic>Duration</ns1:mnemonic>
                    <ns1:description>Duration</ns1:description>
                    <ns1:type>STRING</ns1:type>
                </ns1:classAttribute>
                <ns1:classAttribute>
                    <ns1:id>1750c564-20a7-4e07-988a-b382227123e4</ns1:id>
                    <ns1:mnemonic>Length</ns1:mnemonic>
                    <ns1:description>Length</ns1:description>
                    <ns1:type>FLOAT</ns1:type>
                </ns1:classAttribute>
                <ns1:primaryKey>
                    <ns1:id>1ed32816-8bdb-4d35-9f66-8c08df13ad28</ns1:id>
                    <ns1:mnemonic>Number</ns1:mnemonic>
                    <ns1:description>Number</ns1:description>
                    <ns1:type>
                        <ns1:id>00000000-0000-0000-0000-000000000002</ns1:id>
                        <ns1:value>INTEGER</ns1:value>
                    </ns1:type>
                </ns1:primaryKey>
            </ns1:datamartClass>
        </ns1:datamart>
    </ns:metadata>
</ns:PODDMetadataRequest>
```

# ПРИЛОЖЕНИЕ 4 ОПИСАНИЕ ТОПИКОВ

## 1 blob.err

`blob.err` - Топик с ошибками получения бинарных данных по ссылке.

**Структура сообщения**

```yaml
datamartBlobErrorResponseMessage:
  description: Ошибка получения бинарных данных по ссылке
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
      MESSAGE_TYPE:
        description: Тип сообщения
        type: string
        const: DatamartBlobErrorResponse:0.1
  payload:
    $ref: '#/components/schemas/datamartBlobErrorResponse'
  examples:
    - name: blobError
      summary: Пример ошибки получения бинарных данных по ссылке
      headers:
        AGENT_CONSUMER_ID: agent-fias
        MESSAGE_TYPE: DatamartBlobErrorResponse:0.1
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        queryRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
        errorCode: DATAMART-001
        errorMessage: Непредвиденная ошибка обработки
```

**Avro-схема сообщения**

```yaml
datamartBlobErrorResponse:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: BlobError
  namespace: datamart.blob
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: queryRequestId
      description: Идентификатор исходного запроса, в рамках которого была получена
ссылка
      type:
        type: string
        logicalType: uuid
    - name: errorCode
      description: Код ошибки
```

```
      type: string
    - name: errorMessage
      description: Сообщение с ошибкой
      type: string
```

### 2 blob.rq

`blob.rq` - Топик запросов на получение бинарных данных по полученной ранее ссылке.

**Структура сообщения**

```
datamartBlobRequestMessage:
  description: Запрос бинарных данных по ссылке
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      REQUEST_ID:
        description: Идентификатор запроса
        type: string
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
      MESSAGE_TYPE:
        description: Тип сообщения
        type: string
        const: DatamartBlobRequest:0.1
  payload:
    $ref: '#/components/schemas/datamartBlobRequest'
  examples:
    - name: getBlobDataRequest
      summary: Запрос бинарных данных по ссылке
      headers:
        REQUEST_ID: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        AGENT_CONSUMER_ID: agent-fias
        MESSAGE_TYPE: DatamartBlobRequest:0.1
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        queryRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
        reference:
          subRequestId: 4cbb11d6-47de-4928-953f-47dfa6c6b310
          path: reference
```

**Avro-схема сообщения**

```
datamartBlobRequest:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: BlobRequest
  namespace: datamart.blob
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: queryRequestId
```

```
      description: Идентификатор исходного запроса, в рамках которого была получена ссылка
      type:
        type: string
        logicalType: uuid
    - name: reference
      description: Ссылка на данные
      type:
        type: record
        name: BinaryReference
        namespace: query.result
        fields:
          - name: subRequestId
            description: Идентификатор подзапроса
            type:
              type: string
              logicalType: uuid
          - name: path
            description: Ссылка
            type: string
```

### 3 blob.rs

`blob.rs` - Топик с бинарными данным блобов

**Структура сообщения**

```
datamartBlobChunkMessage:
  description: Чанки бинарных данных
  contentType: 'application/octet-stream'
  bindings:
    kafka:
      key:
        $ref: '#/components/schemas/datamartBlobChunkInfo'
  headers:
    type: object
    properties:
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
      MESSAGE_TYPE:
        description: Тип сообщения
        type: string
        const: DatamartBlobChunkInfo:0.1
  payload:
    description: Бинарные данные
  examples:
    - name: base64
      headers:
        MESSAGE_TYPE: DatamartBlobChunkInfo:0.1
      payload:
        value: JEEJNodyLO7p1pgsRHG9pEiXeYGvHW4YCl4FgrgBmu5C92iVX1PV2GZdcqsb66bx8sk=
```

**Avro-схема сообщения**

```
datamartBlobChunkInfo:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: BlobChunk
  namespace: datamart.blob
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
```

```
      type:
        type: string
        logicalType: uuid
    - name: queryRequestId
      description: Идентификатор исходного запроса, в рамках которого была получена ссылка
      type:
        type: string
        logicalType: uuid
    - name: chunkNum
      description: Номер чанка
      type: int
      minimum: 1
    - name: isLast
      description: Признак последнего чанка
      type: boolean
  examples:
    - requestId: 3546e40b-47fe-41b6-9c06-a2e915eb4181
      queryRequestId: a8e9f47b-38cd-4db6-a245-0fbd6e78c195
      chunkNum: 1
      isLast: true
```

## 4 cancel.err

`cancel.err` - Топик с ошибками по отмене запроса

**Структура сообщения**

```
datamartCancelQueryErrorMessage:
  description: Ответ об успешной отмене запроса
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      REQUEST_ID:
        description: Идентификатор запроса
        type: string
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    $ref: '#/components/schemas/datamartCancelQueryError'
  examples:
    - name: success
      summary: Пример запроса на отмену
      headers:
        REQUEST_ID: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        AGENT_CONSUMER_ID: agent-fias
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        errorCode: DATAMART-001
        message: Непредвиденная ошибка
```

**Avro-схема сообщения**

```yaml
datamartCancelQueryError:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: DatamartCancelQueryError
  namespace: datamart.query.cancel
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: errorCode
      description: Код ошибки выполнения
      type: string
    - name: message
      description: Сообщение об ошибке
      type: string
```

## 5 cancel.rq

`cancel.rq` - Топик с сообщениями об отмене исполнения запроса

**Структура сообщения**

```yaml
datamartQueryCancellationRequestMessage:
  description: Запрос на отмену исполнения
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      REQUEST_ID:
        description: Идентификатор запроса
        type: string
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    $ref: '#/components/schemas/datamartQueryCancellationRequest'
  examples:
    - name: request
      summary: Пример запроса на отмену
      headers:
        REQUEST_ID: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        AGENT_CONSUMER_ID: agent-fias
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
```

**Avro-схема сообщения**

```yaml
datamartQueryCancellationRequest:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: AgentQueryCancellationRequest
  namespace: ru.rtlabs.common.query.cancel
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
```

### 6 cancel.rs

`cancel.rs` - Топик с ответами на отмену запроса

**Структура сообщения**

```yaml
datamartCancelQuerySuccessMessage:
  description: Ответ об успешной отмене запроса
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      REQUEST_ID:
        description: Идентификатор запроса
        type: string
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    $ref: '#/components/schemas/datamartCancelQuerySuccess'
  examples:
    - name: success
      summary: Пример запроса на отмену
      headers:
        REQUEST_ID: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        AGENT_CONSUMER_ID: agent-fias
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        isSuccess: true
```

**Avro-схема сообщения**

```yaml
datamartCancelQuerySuccess:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: DatamartCancelQuerySuccess
  namespace: datamart.query
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: isSuccess
      description: Признак успешного выполнения операции
      type: boolean
```

## 7 delta.err

`delta.err` - Топик с ошибками получения дельт у поставщика.

**Структура сообщения**

```yaml
deltaErrorMessage:
  description: Ответ с ошибкой получения дельты у поставщика
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    $ref: '#/components/schemas/deltaError'
  examples:
    - name: simple
      headers:
        AGENT_CONSUMER_ID: agent-oktmo
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId:
          string: 00000000-0000-0000-0000-000000000000
        subscriptionId: dcf43fc7-e152-459b-8af5-48d91d4b6a21
        errorCode: DATAMART-001
        message: Непредвиденная ошибка
```

**Avro-схема сообщения**

```yaml
deltaError:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: DeltaError
  namespace: ru.rtlabs.common.replication.delta
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
```

```
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      default: null
      type:
        - 'null'
        - type: string
          logicalType: uuid
    - name: subscriptionId
      description: Уникальный идентификатор подписки
      type:
        type: string
        logicalType: uuid
    - name: errorCode
      description: Код ошибки выполнения
      type: string
    - name: message
      description: Сообщение об ошибке
      type: string
```

## 8 delta.in.err

`delta.in.err` - Ответ с ошибкой применения дельты. Один ответ на дельту с множеством чанков.

### Структура сообщения

```
deltaApplyErrorMessage:
  description: Ответ с ошибкой применения дельты у потребителя
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    $ref: '#/components/schemas/deltaApplyError'
  examples:
    - name: simple
      headers:
        AGENT_CONSUMER_ID: agent-oktmo
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId:
          string: 00000000-0000-0000-0000-000000000000
        subscriptionId: dcf43fc7-e152-459b-8af5-48d91d4b6a21
        synId: 1
        errorCode: DATAMART-001
        message: Непредвиденная ошибка применения дельты
```

### Avro-схема сообщения

```
deltaApplyError:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: DeltaApplyError
  namespace: ru.rtlabs.common.replication.delta
```

```
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      default: null
      type:
        - 'null'
        - type: string
          logicalType: uuid
    - name: subscriptionId
      description: Уникальный идентификатор подписки
      type:
        type: string
        logicalType: uuid
    - name: synId
      description: ¥use in replicator version 1.1¥Номер дельты, на которой возникла
ошибка
      default: null
      type:
        - 'null'
        - int
    - name: errorCode
      description: Код ошибки выполнения
      type: string
    - name: message
      description: Сообщение об ошибке
      type: string
```

## 9 delta.in.rq

`delta.in` - Топик чанков дельт репликации на применение у поставщика.

**Структура сообщения**

```
deltaResultChunkMessage:
  description: Чанк с данными дельты репликации
  contentType: 'application/octet-stream'
  bindings:
    kafka:
      key:
        $ref: '#/components/schemas/deltaResultChunkKey'
  headers:
    type: object
    properties:
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    description: Бинарные данные чанка
  examples:
    - name: base64
      headers:
        AGENT_CONSUMER_ID: agent_fias
      payload:
        value: JEEJNodyLO7p1pgsRHG9pEiXeYGvHW4YCl4FgrgBmu5C92iVX1PV2GZdcqsb66bx8sk=
```

**Avro-схема сообщения**

```
deltaResultChunkKey:
```

```yaml
schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
type: record
name: DeltaResultChunk
namespace: ru.rtlabs.common.replication.delta
fields:
  - name: requestId
    description: Уникальный идентификатор распределенного
    type:
      type: string
      logicalType: uuid
  - name: subRequestId
    description: Уникальный идентификатор подзапроса
    default: null
    type:
      - 'null'
      - type: string
        logicalType: uuid
  - name: sourceDatamart
    description: Наименование датамарта источника
    default: null
    type:
      - 'null'
      - string
  - name: subscriptionId
    description: Уникальный идентификатор подписки
    type:
      type: string
      logicalType: uuid
  - name: synId
    description: Идентификатор синхронизации витрины
    type: int
  - name: sql
    description:sql, для которого возвращается дельта
    default: null
    type:
      - 'null'
      - string
  - name: minSynId
    description: Номер первой передаваемой в этом пакете дельты
    default: null
    type:
      - 'null'
      - int
  - name: maxSynId
    description: Номер последней передаваемой в этом пакете дельты
    default: null
    type:
      - 'null'
      - int
  - name: synTime
    description:Время дельты
    default: 0
    type: long
  - name: streamNumber
    description: Номер стрима данных
    type:
      - int
      - 'null'
  - name: streamTotal
    description: Общее количество стримов
    default: 1
    type:
```

```
      - int
      - 'null'
  - name: chunkNumber
    description: Номер порции по порядку
    type: int
  - name: isLastChunk
    description: Признак последнего сообщения
    type: boolean
  - name: replicaHash
    description: Чек-сумма реплики после применения дельты с данным
    default: null
    type:
      - 'null'
      - string
examples:
  - requestId: 00000000-0000-0000-0000-000000000000
    subRequestId:
      string: 00000000-0000-0000-0000-000000000000
    subscriptionId: 00000000-0000-0000-0000-000000000000
    sourceDatamart:
      string: dm
    sql:
      string: 'select * from v1_addrobj'
    minSynId:
      int: 10
    maxSynId:
      int: 10
    synTime: 1000
    streamNumber:
      int: 1
    streamTotal:
      int: 1
    chunkNumber: 1
    isLastChunk: true
    replicaHash:
      string: '123456789'
```

## 10 delta.in.rs

`delta.in.rs` - Топик с ответами с подтверждением применения дельты у потребителя.

**Структура сообщения**

```
deltaApplyResultMessage:
  description: Ответ с подтверждением применения дельты
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    $ref: '#/components/schemas/deltaApplyResult'
  examples:
    - name: simple
      headers:
```

```
      AGENT_CONSUMER_ID: agent-oktmo
    payload:
      requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId:
        string: 00000000-0000-0000-0000-000000000000
      subscriptionId: dcf43fc7-e152-459b-8af5-48d91d4b6a21
      synId: 100
```

**Avro-схема сообщения**

```
deltaApplyResult:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: DeltaApplyResult
  namespace: ru.rtlabs.common.replication.delta
  fields:
    - name: requestId
      description: Уникальный идентификатор распределенного запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      default: null
      type:
        - 'null'
        - type: string
          logicalType: uuid
    - name: subscriptionId
      description: Уникальный идентификатор подписки
      type:
        type: string
        logicalType: uuid
    - name: synId
      description: Id дельты витрины-потребителя
      type: int
```

## 11 delta.notification

`delta.notification` - Топик нотификаций ядра о наличии новых дельт.

**Структура сообщения**

```
deltaNotificationMessage:
  description: Ответ с ошибкой получения дельты
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    $ref: '#/components/schemas/deltaNotification'
  examples:
    - name: simple
      headers:
```

```
      AGENT_CONSUMER_ID: agent-oktmo
    payload:
      requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId:
        string: 00000000-0000-0000-0000-000000000000
      datamartMnemonic:
        string: dm
      synId:
        int: 1
      subscriptions:
        - subscriptionId: dcf43fc7-e152-459b-8af5-48d91d4b6a21
          fromId: 100
          deltaKeySize: 1024
          snapshotKeySize: 1024
```

**Avro-схема сообщения**

```
deltaNotification:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: DeltaNotificationEvent
  namespace: ru.rtlabs.common.replication.delta
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      default: null
      type:
        - 'null'
        - type: string
          logicalType: uuid
    - name: datamartMnemonic
      description: Мнемоника витрины-источника, в которой обновились данные
      default: null
      type:
        - 'null'
        - string
    - name: synId
      description: ДЛЯ РАСПРЕДПОДПИСОК. Id новой дельты
      default: null
      type:
        - 'null'
        - int
    - name: subscriptions
      description: Список всех подписок, для которых есть новые данные
      type:
        type: array
        items:
          type: record
          name: DeltaNotificationSubscription
          fields:
            - name: subscriptionId
              description: Идентификатор подписки, для которой есть новые данные
              type:
                type: string
                logicalType: uuid
            - name: fromId
              description: ¥use in replicator version 1.1¥ Id новой дельты
```

```
            default: null
            type:
              - 'null'
              - int
          - name: deltaKeySize
            description: размер ключей дельты в байтах
            default: 0
            type: long
          - name: snapshotKeySize
            description: Размер ключей снапшота в байтах
            default: 0
            type: long
```

## 12 delta.rq

`delta.rq` - Топик запросов на получение дельты у поставщика.

**Структура сообщения**

```
deltaRequestMessage:
  description: Запрос из ПОДД на получение дельты у поставщика
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    $ref: '#/components/schemas/deltaRequest'
  examples:
    - name: snapshot
      headers:
        AGENT_CONSUMER_ID: agent-oktmo
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId:
          string: 00000000-0000-0000-0000-000000000000
        subscriptionId: dcf43fc7-e152-459b-8af5-48d91d4b6a21
        fromId: null
    - name: delta
      headers:
        AGENT_CONSUMER_ID: agent-oktmo
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId:
          string: 00000000-0000-0000-0000-000000000000
        subscriptionId: dcf43fc7-e152-459b-8af5-48d91d4b6a21
        fromId:
          int: 100
```

**Avro-схема сообщения**

```
deltaRequest:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: DeltaRequest
```

```
  namespace: ru.rtlabs.common.replication.delta
  fields:
    - name: requestId
      description: Уникальный идентификатор операции, все запрос-ответы в рамках одной операции
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Идентификатор подзапроса, идентифицирует одно взаимодействие типа запрос-ответ
      default: null
      type:
      - 'null'
      - type: string
        logicalType: uuid
    - name: subscriptionId
      description: Уникальный идентификатор подписки
      type:
        type: string
        logicalType: uuid
    - name: fromId
      description: Идентификатор запрашиваемой дельты.
      default: null
      type:
        - int
        - 'null'
```

## 13 delta.rs

`delta.rs` - Топик чанков данных дельт репликации от поставщика.

**Структура сообщения**

```
deltaResultChunkMessage:
  description: Чанк с данными дельты репликации
  contentType: 'application/octet-stream'
  bindings:
    kafka:
      key:
        $ref: '#/components/schemas/deltaResultChunkKey'
  headers:
    type: object
    properties:
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    description: Бинарные данные чанка
  examples:
    - name: base64
      headers:
        AGENT_CONSUMER_ID: agent_fias
      payload:
        value: JEEJNodyLO7p1pgsRHG9pEiXeYGvHW4YCl4FgrgBmu5C92iVX1PV2GZdcqsb66bx8sk=
```

**Avro-схема сообщения**

```
deltaResultChunkKey:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: DeltaResultChunk
  namespace: ru.rtlabs.common.replication.delta
```

```yaml
fields:
  - name: requestId
    description: Уникальный идентификатор запроса
    type:
      type: string
      logicalType: uuid
  - name: subscriptionId
    description: Уникальный идентификатор подписки
    type:
      type: string
      logicalType: uuid
  - name: sql
    description: sql, для которого возвращается дельта
    type: string
  - name: synId
    description: Идентификатор синхронизации витрины
    type: int
  - name: minSynId
    description: Номер первой передаваемой в этом пакете дельты
    type: int
  - name: maxSynId
    description: Номер последней передаваемой в этом пакете дельты
    type: int
  - name: synTime
    description: Время дельты
    type: long
  - name: streamNumber
    description: Номер стрима дланных
    type:
      - int
      - 'null'
  - name: streamTotal
    description: Общее количество стримов
    type:
      - int
      - 'null'
  - name: chunkNumber
    description: Номер порции по порядку
    type: int
  - name: isLastChunk
    description: Признак последнего сообщения
    type: boolean
  - name: replicaHash
    description: Чек-сумма реплики после применения дельты с данным
    type:
      - string
      - 'null'
examples:
  - requestId: 74e43a84-c6c6-4e25-bac4-9d39c59b3da5
    subscriptionId: a1ba39be-962b-4ac0-b3f3-893151883e59
    sql: 'select * from v1_addrobj'
    synId: 10
    minSynId: 10
    maxSynId: 10
    synTime: 1000
    streamNumber:
      int: 1
    streamTotal:
      int: 1
    chunkNumber: 1
    isLastChunk: true
    replicaHash:
```

```
      string: '123456789'
  - requestId: 74e43a84-c6c6-4e25-bac4-9d39c59b3da5
    subscriptionId: a1ba39be-962b-4ac0-b3f3-893151883e59
    sql: 'select * from v1_addrobj'
    synId: 10
    minSynId: 10
    maxSynId: 10
    synTime: 1000
    streamNumber:
      int: 1
    streamTotal:
      int: 1
    chunkNumber: 1
    isLastChunk: false
    replicaHash: null
```

## 14 procedure.query.rq

`procedure.query.rq` - Топик регламентированных запросов на исполнение

**Структура сообщения**

```
examples:
  - name: simple
    summary: Простой запрос на исполнение без параметров
    headers:
      MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
    payload:
      requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
      replyTo: agent-fias
      datamartMnemonic: fias
      sql: select * from v1_addrobj
      parameters: [ ]
      namedParams: [ ]
      tableParams: [ ]
      isForEstimation: false
      rowCountThreshold: -1
      customerId: aaa
      customerOgrn: ""
      queryMnemonic: fias.selectAllAddrobj.1.0
  - name: estimation
    summary: Запрос на оценку
    headers:
      MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
    payload:
      requestId: 403eada5-05f6-480c-bca9-03328091efeb
      subRequestId: 451000b8-dff2-4a1b-ab1b-42500a70d232
      replyTo: agent-fias
      datamartMnemonic: fias
      sql: select * from v1_addrobj
      parameters: [ ]
      namedParams: [ ]
      tableParams: [ ]
      isForEstimation: true
      rowCountThreshold: 1000
      customerId:
        string: agent-fias
      customerOgrn:
        string: "1053600591197"
      queryMnemonic:
        string: fias.selectAllAddrobj.1.0
  - name: complex
```

```
    summary: Запрос с параметрами и табличными параметрами
    headers:
      MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
    payload:
      requestId: 68758a92-0027-4258-bf17-aa3d24f85094
      subRequestId: 96e6eb99-7ff1-4efa-abae-ef1c5744b723
      replyTo: agent-fias
      datamartMnemonic: fias
      sql: select * from v1_addrobj where oktmo = ? and name = @tbl.fullname
      parameters:
        - type: STRING
          value:
            string: asdasdasd
        - type: LONG
          value: null
      namedParams: [ ]
      tableParams: [ ]
      isForEstimation: false
      rowCountThreshold: -1
      customerId:
        string: agent-fias
      customerOgrn:
        string: "1053600591197"
      queryMnemonic:
        string: fias.selectAddrobjWithParams.1.0
  - name: complex_named
    summary: Запрос с именованными параметрами
    headers:
      MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
    payload:
      requestId: 12358a92-0027-4258-bf17-aa3d24f85094
      subRequestId: 56e6eb99-7ff1-4efa-abae-ef1c5744b723
      replyTo: agent-fias
      datamartMnemonic: fias
      sql: select * from @tbl.fullname el LEFT JOIN v1_addrobj where oktmo = @p1 and kod
= @p2
      parameters: [ ]
      namedParams:
        - name: p1
          type: STRING
          value:
            string: asdasdasd
        - name: p2
          type: LONG
          value: null
      tableParams: [ ]
      isForEstimation: false
      rowCountThreshold: -1
      customerId:
        string: agent-fias
      customerOgrn:
        string: "1053600591197"
      queryMnemonic:
        string: fias.selectAddrobjWithParams.1.0
  - name: deadline
    summary: Простой запрос на исполнение без параметров
    headers:
      MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
      QUERY_DEADLINE: 1629289006904
    payload:
      requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
```

```
      replyTo: agent-fias
      datamartMnemonic: fias
      sql: select * from v1_addrobj
      parameters: [ ]
      namedParams: [ ]
      tableParams: [ ]
      isForEstimation: false
      rowCountThreshold: -1
      customerId: agent-fias
      customerOgrn: "1053600591197"
      queryMnemonic: fias.selectAllWithDeadline.1.0
```

**Avro-схема сообщения**

```
datamartExecuteQueryRequest:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: QueryRequest
  namespace: datamart.query
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      type:
        type: string
        logicalType: uuid
    - name: replyTo
      description: Служебная информация маршрутизации сообщения. Ответ, формируемый
витриной, обязан содержать переданное значение без каких либо искажений
      type: string
    - name: datamartMnemonic
      description: Мнемоника витрины, к которой выполняется запрос
      type: string
    - name: sql
      description: SQL запрос на исполнение, либо имя хранимой процедуры для
регламентированных запросов
      type: string
    - name: parameters
      description: Параметры к SQL запросу
      default: [ ]
      type:
        type: array
        items:
          type: record
          name: QueryParameter
          description: Описание параметра
          fields:
            - name: type
              type: string
              description: Тип параметра
              enum:
                - BIG_DECIMAL
                - BINARY
                - BOOLEAN
                - DATE
                - DOUBLE
                - FLOAT
                - INTEGER
```

```
            - LONG
            - SHORT
            - STRING
            - TIME
            - TIMESTAMP
        - name: value
          description: Значение параметра
          type:
            - string
            - 'null'
  - name: namedParams
    description: Именованные параметры запроса
    default: [ ]
    type:
      type: array
      items:
        type: record
        name: NamedParam
        description: Описание именованного параметра
        fields:
          - name: name
            description: Имя (мнемоника) параметра
            type: string
          - name: type
            type: string
            description: Тип параметра
            enum:
              - BIG_DECIMAL
              - BINARY
              - BOOLEAN
              - DATE
              - DOUBLE
              - FLOAT
              - INTEGER
              - LONG
              - SHORT
              - STRING
              - TIME
              - TIMESTAMP
          - name: value
            description: Значение параметра
            type:
              - string
              - 'null'
  - name: tableParams
    description: ¥use only Datamart¥ Табличные параметры запроса
    default: [ ]
    type:
      type: array
      items:
        type: record
        name: TableParam
        fields:
          - name: id
            description: Уникальный идентификатор
            type:
              type: string
              logicalType: uuid
          - name: name
            description: Имя параметра
            type: string
          - name: columns
```

```
            description: Описание колонок таблицы
            type:
              type: array
              items:
                type: record
                description: Описание колонки
                name: TableParamColumnInfo
                fields:
                  - name: name
                    type: string
                    description: Имя колонки
                  - name: type
                    type: string
                    description: Тип атрибута
                    enum:
                      - BIG_DECIMAL
                      - BINARY
                      - BOOLEAN
                      - DATE
                      - DOUBLE
                      - FLOAT
                      - INTEGER
                      - LONG
                      - SHORT
                      - STRING
                      - TIME
                      - TIMESTAMP
    - name: isForEstimation
      description: Признак необходимости вернуть статистику по запросу в качестве результата. В случае, если оценка по результату исполнения sql запроса не превышает rowCountThreshold записей, должен сразу отдаваться результат без отправки оценки в ядро
      type: boolean
      default: false
    - name: rowCountThreshold
      description: Максимальное оценочное количество строк результата, при превышении которого возвращается статистика по запросу. Если оценка по запросу не превышет данный параметр, витрина сразу возвращает ответ с результатом. Заполняется в случае isForEstimation = true
      type: long
      default: -1
    - name: customerId
      description: Мнемоника ИС Потребителя
      type:
        - 'null'
        - string
      default: null
    - name: customerOgrn
      description: ОГРН ИС Потребителя
      type:
        - 'null'
        - string
      default: null
    - name: queryMnemonic
      description: 'Мнемоника РЗ, сформированная по правилу: <мнемоника витрины>.<мнемоника РЗ>.<версия РЗ> Если запрос распределенный, то формируется по правилу: podd.<мнемоника РЗ>.<версия РЗ>'
      type:
        - 'null'
        - string
      default: null
```

## 15 procedure.query.rs

`procedure.query.rs` - Топик с чанками данных исполнения запросов

**Структура сообщения**

```
datamartExecuteQueryResultChunk:
schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
type: record
name: QueryResultChunk
namespace: datamart.query
fields:
    - name: requestId
    description: Уникальный идентификатор запроса
    type:
        type: string
        logicalType: uuid
    - name: subRequestId
    description: Уникальный идентификатор подзапроса
    type:
        type: string
        logicalType: uuid
    - name: replyTo
    description: Служебная информация маршрутизации сообщения. Заполняется
соответсвующим значением из запроса
    type: string
    - name: chunkNumber
    description: Номер порции по порядку
    type: int
    minimum: 1
    - name: isLastChunk
    description: Признак последнего сообщения
    type: boolean
    - name: streamNumber
    description: Номер стрима данных
    minimum: 1
    type:
        - int
        - "null"
    - name: streamTotal
    description: Общее количество стримов
    minimum: 1
    type:
        - int
        - "null"
    - name: isFragmented
    description: Признак присутствия в чанке неполных строк (строк, которые были
разбиты на несколько чанков)
    type: boolean
    - name: uncompressedSize
    description: Оригинальный размер чанка в байтах
    type: int
    minimum: 0
```

**Пример key query.rs**

```
examples:
- requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
    subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
    replyTo: agent-fias
    chunkNumber: 1
    isLastChunk: true
    streamNumber:
    int: 1
    streamTotal:
    int: 1
    isFragmented: false
    uncompressedSize: 10
```

## 16 procedure.query.err

`procedure.query.err` - Топик с ошибками исполнения sql запросов на витрине

**Структура сообщения**

```
datamartExecuteQueryError:
schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
type: record
name: QueryError
namespace: datamart.query
fields:
    - name: requestId
    description: Уникальный идентификатор запроса
    type:
        type: string
        logicalType: uuid
    - name: subRequestId
    description: Уникальный идентификатор подзапроса
    type:
        type: string
        logicalType: uuid
    - name: replyTo
    description: Служебная информация маршрутизации сообщения. Заполняется
соответсвующим значением из запроса
    type: string
    - name: errorCode
    description: Код возникшей ошибки
    type: string
    - name: message
    description: Сообщение с ошибкой исполнения
    type: string
```

**Пример query.err**

```yaml
examples:
- name: error
    summary: Сообщение с ошибкой исполнения запроса на витрине без header
    payload:
    requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
    subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
    replyTo: agent-fias
    errorCode: DATAMART-001
    message: Непредвиденная ошибка
- name: errorWithHeader
    summary: Сообщение с ошибкой исполнения запроса на витрине
    headers:
    MESSAGE_TYPE: DatamartExecuteQueryError:0.1
    payload:
    requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
    subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
    replyTo: agent-fias
    errorCode: DATAMART-001
    message: Непредвиденная ошибка
```

## 17 query.err

`query.err` - Топик с ошибками исполнения sql запросов на витрине

**Структура сообщения**

```yaml
datamartExecuteQueryErrorMessage:
  description: Ошибка исполнения запроса
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      MESSAGE_TYPE:
        description: Тип сообщения
        type: string
        const: DatamartExecuteQueryError:0.1
  payload:
    $ref: '#/components/schemas/datamartExecuteQueryError'
  examples:
    - name: error
      summary: Сообщение с ошибкой исполенния запроса на витрине без header
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
        replyTo: agent-fias
        errorCode: DATAMART-001
        message: Непредвиденная ошибка
    - name: errorWithHeader
      summary: Сообщение с ошибкой исполенния запроса на витрине
      headers:
        MESSAGE_TYPE: DatamartExecuteQueryError:0.1
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
        replyTo: agent-fias
        errorCode: DATAMART-001
```

```
      message: Непредвиденная ошибка
```

**Avro-схема сообщения**

```
datamartExecuteQueryError:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: QueryError
  namespace: datamart.query
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      type:
        type: string
        logicalType: uuid
    - name: replyTo
      description: Служебная информация маршрутизации сообщения. Заполняется
соответсвующим значением из запроса
      type: string
    - name: errorCode
      description: Код возникшей ошибки
      type: string
    - name: message
      description: Сообщение с ошибкой исполнения
      type: string
  examples:
    - requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
      replyTo: agent-fias
      errorCode: DATAMART-001
      message: Непредвиденная ошибка
```

## 18 query.estimation.rs

Топик `QUERY.ESTIMATION.RS` предоставляет возможность произвести предварительную оценку объема получаемых данных при выполнении запроса к Витрине данных, а также, ограничить выгрузку данных в случае, если количество получаемых данных превысит заданное количество строк (параметр `rowCountThreshold`). В этом случае, ответом на запрос будет предварительная оценка объема.

Например, если вам нужна информация из какой-либо таблицы контактов, то, возможно, следует предварительно узнать, какой объем данных вы можете получить на такой запрос т.к ответ может содержать несколько гигабайт информации и выполнение запроса может занять много времени. Вы сможете установить ограничение на получение данных, например, не более 10 контактов из таблицы. В этом случае, если ответом на запрос будет 5 контактов, то Витрина предоставит ответ полностью. Если ответом будет 1000 контактов, то в качестве ответа будет сформирована предварительная оценка такого ответа, а именно, что данный ответ будет содержать 1000 строк и содержать информацию, например, на 15000 байт. Используя топик `query.estimation.rs` можно прогнозировать объем получаемых данных, в соответствии с которыми оптимизировать запросы к Витрине.

В случае использования топика `query.estimation.rs` запрашивается не конечный ответ на запрос, а приблизительная оценка объема (байт) и количество строк в ответе.

Примечание:

Данное требование не распространяется на механизм подписок Потребителей данных ПОДД.

**Алгоритм работы query.estimation.rs**

1. Витрина получает запрос query.rq с признаком `isForEstimation` оценивает объем результата по этому запросу (в байтах и количестве строк).
2. Витрина сравнивает результаты оценки объема запроса со значением предельного числа строк в параметре `rowCountThreshold` (топик query.rq).
3. Если значение в оценке меньше, чем предельное значение в `rowCountThreshold`, то Витрина возвращает результат запроса в качестве ответа в топик query.rs.
4. Если значение оценки превышает предельное значение, возвращает предварительную оценку объема в качестве ответа.

**Структура сообщения**

```yaml
datamartQueryEstimationMessage:
  description: Оценка по запросу
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  payload:
    $ref: '#/components/schemas/datamartQueryEstimation'
  examples:
    - name: estimation
      summary: Сообщение с оценкой по исполнению запроса
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
        estimatedRowCount: 100
        estimatedSize: 1000
        estimatedTime: 50
```

**Avro-схема сообщения**

```yaml
datamartQueryEstimation:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: Estimation
  namespace: datamart.query
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      type:
        type: string
        logicalType: uuid
    - name: estimatedRowCount
      description: Оценка количества строк результата выполнения запроса
      type: long
    - name: estimatedSize
      description: Оценка объема результата выполнения запроса, в байтах
      type: long
```

```
    - name: estimatedTime
      description: Оценка времени выполнения запроса в миллисекундах
      type: long
  examples:
    - requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
      estimatedRowCount: 100
      estimatedSize: 1000
      estimatedTime: 50
```

## 19 query.rq

`query.rq` - Топик sql запросов на исполнение

**Структура сообщения**

```
datamartExecuteQueryRequestMessage:
  description: Исполнение sql запроса на витрине
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      MESSAGE_TYPE:
        description: Тип сообщения
        type: string
        const: DatamartExecuteQueryRequest:0.1
      REQUEST_ID:
        description: Идентификатор запроса
        type: string
      QUERY_DEADLINE:
        description: Время в миллисекундах от эпохи, до которого запрос должен быть
выполнен
        type: string
        format: int64
      AGENT_CONSUMER_ID:
        description: Мнемоника потребителя (мнемоника агента)
        type: string
      QUERY_MNEMONIC:
        description: '<Полная мнемоника РЗ>.<версия РЗ>'
        type: string
  payload:
    $ref: '#/components/schemas/datamartExecuteQueryRequest'
  examples:
    - name: simple
      summary: Простой запрос на исполнение без параметров
      headers:
        MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
        replyTo: agent-fias
        datamartMnemonic: fias
        sql: select * from v1_addrobj
        parameters: [ ]
        namedParams: [ ]
        tableParams: [ ]
        isForEstimation: false
```

```
        rowCountThreshold: -1
        customerId: aaa
        customerOgrn: ""
        queryMnemonic: fias.selectAllAddrobj.1.0
    - name: estimation
      summary: Запрос на оценку
      headers:
        MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
      payload:
        requestId: 403eada5-05f6-480c-bca9-03328091efeb
        subRequestId: 451000b8-dff2-4a1b-ab1b-42500a70d232
        replyTo: agent-fias
        datamartMnemonic: fias
        sql: select * from v1_addrobj
        parameters: [ ]
        namedParams: [ ]
        tableParams: [ ]
        isForEstimation: true
        rowCountThreshold: 1000
        customerId:
          string: agent-fias
        customerOgrn:
          string: "1053600591197"
        queryMnemonic:
          string: fias.selectAllAddrobj.1.0
    - name: complex
      summary: Запрос с параметрами
      headers:
        MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
      payload:
        requestId: 68758a92-0027-4258-bf17-aa3d24f85094
        subRequestId: 96e6eb99-7ff1-4efa-abae-ef1c5744b723
        replyTo: agent-fias
        datamartMnemonic: fias
        sql: select * from v1_addrobj where oktmo = ? and name = @tbl.fullname
        parameters:
          - type: STRING
            value:
              string: asdasdasd
          - type: LONG
            value: null
        namedParams: [ ]
        tableParams: [ ]
        isForEstimation: false
        rowCountThreshold: -1
        customerId:
          string: agent-fias
        customerOgrn:
          string: "1053600591197"
        queryMnemonic:
          string: fias.selectAddrobjWithParams.1.0
    - name: complex_named
      summary: Запрос с именованными параметрами
      headers:
        MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
      payload:
        requestId: 12358a92-0027-4258-bf17-aa3d24f85094
        subRequestId: 56e6eb99-7ff1-4efa-abae-ef1c5744b723
        replyTo: agent-fias
        datamartMnemonic: fias
        sql: select * from @tbl.fullname el LEFT JOIN v1_addrobj where oktmo = @p1 and
kod = @p2
```

```
      parameters: [ ]
      namedParams:
        - name: p1
          type: STRING
          value:
            string: asdasdasd
        - name: p2
          type: LONG
          value: null
      tableParams: [ ]
      isForEstimation: false
      rowCountThreshold: -1
      customerId:
        string: agent-fias
      customerOgrn:
        string: "1053600591197"
      queryMnemonic:
        string: fias.selectAddrobjWithParams.1.0
  - name: deadline
    summary: Простой запрос на исполнение без параметров
    headers:
      MESSAGE_TYPE: DatamartExecuteQueryRequest:0.1
      QUERY_DEADLINE: 1629289006904
    payload:
      requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
      replyTo: agent-fias
      datamartMnemonic: fias
      sql: select * from v1_addrobj
      parameters: [ ]
      namedParams: [ ]
      tableParams: [ ]
      isForEstimation: false
      rowCountThreshold: -1
      customerId: agent-fias
      customerOgrn: "1053600591197"
      queryMnemonic: fias.selectAllWithDeadline.1.0
```

**Avro-схема сообщения**

```
datamartExecuteQueryRequest:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: QueryRequest
  namespace: datamart.query
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      type:
        type: string
        logicalType: uuid
    - name: replyTo
      description: Служебная информация маршрутизации сообщения. Ответ, формируемый
витриной, обязан содержать переданное значение без каких либо искажений
      type: string
    - name: datamartMnemonic
      description: Мнемоника витрины, к которой выполняется запрос
```

```
      type: string
    - name: sql
      description: SQL запрос на исполнение, либо имя хранимой процедуры для
регламентированных запросов
      type: string
    - name: parameters
      description: Параметры к SQL запросу
      default: [ ]
      type:
        type: array
        items:
          type: record
          name: QueryParameter
          description: Описание параметра
          fields:
            - name: type
              type: string
              description: Тип параметра
              enum:
                - BIG_DECIMAL
                - BINARY
                - BOOLEAN
                - DATE
                - DOUBLE
                - FLOAT
                - INTEGER
                - LONG
                - SHORT
                - STRING
                - TIME
                - TIMESTAMP
            - name: value
              description: Значение параметра
              type:
                - string
                - 'null'
    - name: namedParams
      description: Именованные параметры запроса
      default: [ ]
      type:
        type: array
        items:
          type: record
          name: NamedParam
          description: Описание именованного параметра
          fields:
            - name: name
              description: Имя (мнемоника) параметра
              type: string
            - name: type
              type: string
              description: Тип параметра
              enum:
                - BIG_DECIMAL
                - BINARY
                - BOOLEAN
                - DATE
                - DOUBLE
                - FLOAT
                - INTEGER
                - LONG
                - SHORT
```

```
              - STRING
              - TIME
              - TIMESTAMP
          - name: value
            description: Значение параметра
            type:
              - string
              - 'null'
    - name: tableParams
      description: Табличные параметры запроса
      default: [ ]
      type:
        type: array
        items:
          type: record
          name: TableParam
          fields:
            - name: id
              description: Уникальный идентификатор
              type:
                type: string
                logicalType: uuid
            - name: name
              description: Имя параметра
              type: string
            - name: columns
              description: Описание колонок таблицы
              type:
                type: array
                items:
                  type: record
                  description: Описание колонки
                  name: TableParamColumnInfo
                  fields:
                    - name: name
                      type: string
                      description: Имя колонки
                    - name: type
                      type: string
                      description: Тип атрибута
                      enum:
                        - BIG_DECIMAL
                        - BINARY
                        - BOOLEAN
                        - DATE
                        - DOUBLE
                        - FLOAT
                        - INTEGER
                        - LONG
                        - SHORT
                        - STRING
                        - TIME
                        - TIMESTAMP
    - name: isForEstimation
      description: Признак необходимости вернуть статистику по запросу в качестве
результата. В случае, если оценка по результату исполнения sql запроса не превышает
rowCountThreshold записей, должен сразу отдаваться результат без отправки оценки в ядро
      type: boolean
      default: false
    - name: rowCountThreshold
      description: Максимальное оценочное количество строк результата, при превышении
которого возвращается статистика по запросу. Если оценка по запросу не превышет данный
```

```
параметр, витрина сразу возвращает ответ с результатом. Заполняется в случае
isForEstimation = true
      type: long
      default: -1
    - name: customerId
      description: Мнемоника ИС Потребителя
      type:
        - 'null'
        - string
      default: null
    - name: customerOgrn
      description: ОГРН ИС Потребителя
      type:
        - 'null'
        - string
      default: null
    - name: queryMnemonic
      description: 'Мнемоника РЗ, сформированная по правилу: <мнемоника
витрины>.<мнемоника РЗ>.<версия РЗ> Если запрос распределенный, то формируется по
правилу: podd.<мнемоника РЗ>.<версия РЗ>'
      type:
        - 'null'
        - string
      default: null
```

## 20 query.rs

`query.rs` - Топик с чанками данных исполнения запросов

**Структура сообщения**

```
datamartExecuteQueryResultChunkMessage:
  description: Чанк с данными по исполнению запроса
  contentType: 'application/octet-stream'
  bindings:
    kafka:
      key:
        $ref: '#/components/schemas/datamartExecuteQueryResultChunk'
  headers:
    type: object
    properties:
      MESSAGE_TYPE:
        description: Тип сообщения
        type: string
        const: DatamartExecuteQueryResultChunk:0.1
  payload:
    description: Бинарные данные чанка
  examples:
    - name: base64
      headers:
        MESSAGE_TYPE: DatamartExecuteQueryResultChunk:0.1
      payload:
        value: JEEJNodyLO7p1pgsRHG9pEiXeYGvHW4YCl4FgrgBmu5C92iVX1PV2GZdcqsb66bx8sk=
```

**Avro-схема сообщения**

```
datamartExecuteQueryResultChunk:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: QueryResultChunk
  namespace: datamart.query
  fields:
    - name: requestId
```

```
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      type:
        type: string
        logicalType: uuid
    - name: replyTo
      description: Служебная информация маршрутизации сообщения. Заполняется
соответсвующим значением из запроса
      type: string
    - name: chunkNumber
      description: Номер порции по порядку
      type: int
      minimum: 1
    - name: isLastChunk
      description: Признак последнего сообщения
      type: boolean
    - name: streamNumber
      description: Номер стрима дланных
      minimum: 1
      type:
        - int
        - "null"
    - name: streamTotal
      description: Общее количество стримов
      minimum: 1
      type:
        - int
        - "null"
    - name: isFragmented
      description: Признак присутствия в чанке неполных строк (строк, которые были
разбиты на несколько чанков)
      type: boolean
    - name: uncompressedSize
      description: Признак присутствия в чанке неполных строк (строк, которые были
разбиты на несколько чанков)
      type: int
      minimum: 0
  examples:
    - requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId: 608c5a5a-01d4-4439-8220-dda41a8519fe
      replyTo: agent-fias
      chunkNumber: 1
      isLastChunk: true
      streamNumber:
        int: 1
      streamTotal:
        int: 1
      isFragmented: false
      uncompressedSize: 10
```

## 21 query.tp

`query.tp` - Топик чанков табличных параметров

**Структура сообщения**

```
datamartTableParamChunkMessage:
  description: Чанк с данными табличных параметров
  contentType: 'application/octet-stream'
```

```
  bindings:
    kafka:
      key:
        $ref: '#/components/schemas/datamartTableParamChunkKey'
  headers:
    type: object
    properties:
      MESSAGE_TYPE:
        description: Тип сообщения
        type: string
        const: DatamartTableParamChunkKey:0.1
      REQUEST_ID:
        description: Идентификатор запроса
        type: string
      QUERY_DEADLINE:
        description: Время в миллисекундах от эпохи, до которого запрос должен быть
выполнен
        type: string
        format: int64
      AGENT_CONSUMER_ID:
        description: Мнемоника потребителя (мнемоника агента)
        type: string
      QUERY_MNEMONIC:
        description: '<Полная мнемоника РЗ>.<версия РЗ>'
        type: string
  payload:
    description: Бинарные данные чанка
  examples:
    - name: base64
      headers:
        MESSAGE_TYPE: DatamartTableParamChunkKey:0.1
      payload:
        value: JEEJNodyLO7p1pgsRHG9pEiXeYGvHW4YCl4FgrgBmu5C92iVX1PV2GZdcqsb66bx8sk=
```

**Avro-схема Key сообщения**

```
datamartTableParamChunkKey:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: DatamartTableParamChunkKey
  namespace: datamart.query
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      type:
        type: string
        logicalType: uuid
    - name: tableParamId
      description: Идентификатор табличного параметра
      type:
        type: string
        logicalType: uuid
    - name: datamartMnemonic
      description: Мнемоника витрины, к которой выполняется запрос
      type: string
    - name: chunkNum
      description: Номер порции по порядку
```

```
      type: int
      minimum: 1
    - name: isLast
      description: Признак последнего сообщения
      type: boolean
    - name: streamNum
      description: Номер стрима данных
      type: int
      minimum: 1
    - name: streamTotal
      description: Общее количество стримов
      type: int
      minimum: 1
    - name: customerId
      description: Мнемоника ИС Потребителя
      type:
        - 'null'
        - string
      default: null
    - name: customerOgrn
      description: ОГРН ИС Потребителя
      type:
        - 'null'
        - string
      default: null
    - name: queryMnemonic
      description: 'Мнемоника РЗ, сформированная по правилу: <мнемоника
витрины>.<мнемоника РЗ>.<версия РЗ> Если запрос распределенный, то формируется по
правилу: podd.<мнемоника РЗ>.<версия РЗ>'
      type:
        - 'null'
        - string
      default: null
    - name: queryParams
      description: Информация о запросе
      type:
        type: record
        name: QueryParams
        namespace: datamart.query
        fields:
        - name: sql
          description: Текст SQL-запроса
          type: string
        - name: replyTo
          description: Служебная информация маршрутизации сообщения. Ответ,
формируемый витриной, обязан содержать переданное значение без каких либо искажений
          type: string
        - name: parameters
          default: [ ]
          description: Параметры к SQL запросу
          type:
            type: array
            items:
              type: record
              name: QueryParameter
              description: Описание параметра
              fields:
                - name: type
                  type: string
                  description: Тип параметра
                  enum:
                    - BIG_DECIMAL
```

```
                - BINARY
                - BOOLEAN
                - DATE
                - DOUBLE
                - FLOAT
                - INTEGER
                - LONG
                - SHORT
                - STRING
                - TIME
                - TIMESTAMP
            - name: value
              description: Значение параметра
              type:
                - string
                - 'null'
      - name: namedParams
        description: Именованные параметры запроса
        default: [ ]
        type:
          type: array
          items:
            type: record
            name: NamedParam
            description: Описание именованного параметра
            fields:
              - name: name
                description: Имя (мнемоника) параметра
                type: string
              - name: type
                type: string
                description: Тип параметра
                enum:
                  - BIG_DECIMAL
                  - BINARY
                  - BOOLEAN
                  - DATE
                  - DOUBLE
                  - FLOAT
                  - INTEGER
                  - LONG
                  - SHORT
                  - STRING
                  - TIME
                  - TIMESTAMP
              - name: value
                description: Значение параметра
                type:
                  - string
                  - 'null'
      - name: tableParams
        description: Табличные параметры запроса
        type:
          type: array
          items:
            type: record
            name: TableParam
            fields:
              - name: id
                description: Уникальный идентификатор
                type:
                  type: string
```

```
                logicalType: uuid
            - name: name
              description: Имя параметра
              type: string
            - name: columns
              description: Описание колонок таблицы
              type:
                type: array
                items:
                  type: record
                  description: Описание колонки
                  name: TableParamColumnInfo
                  fields:
                    - name: name
                      type: string
                      description: Имя колонки
                    - name: type
                      type: string
                      description: Тип атрибута
                      enum:
                        - BIG_DECIMAL
                        - BINARY
                        - BOOLEAN
                        - DATE
                        - DOUBLE
                        - FLOAT
                        - INTEGER
                        - LONG
                        - SHORT
                        - STRING
                        - TIME
                        - TIMESTAMP
examples:
  - queryParams:
      summary: Пример запроса с параметрами
      sql: select * from v1_addrobj where oktmo = ? and name = @tbl.fullname
      replyTo: agent-fias
      parameters:
        - type: STRING
          value:
            string: asdasdasd
      namedParams: []
      tableParams:
        - id: 6bba1b55-031c-4931-94ef-daccdc203f8d
          name: tbl
          columns:
            - name: fullname
              type: STRING
    requestId: 114a6659-4b7b-4a2a-8c22-d2177f91091a
    subRequestId: 8c1e7130-989a-4943-b23b-75e242e9d77e
    tableParamId: f3ff5c24-5039-4a00-8212-063114d6411b
    datamartMnemonic: fias
    chunkNum: 1
    isLast: true
    streamNum: 1
    streamTotal: 1
    customerId:
      string: agent-fias
    customerOgrn:
      string: "1053600591197"
    queryMnemonic:
      string: fias.selectAllAddrobj.1.0
```

```
      customerId:
        string: agent-fias
      customerOgrn:
        string: "1053600591197"
      queryMnemonic:
        string: fias.selectAllAddrobj.1.0
    - queryParams:
        summary: Пример запроса с именованными параметрами
        sql: select * from v1_addrobj where oktmo = p1 and name = @tbl.fullname
        replyTo: agent-fias
        parameters: []
        namedParams:
          - name: p1
            type: STRING
            value:
              string: asdasdasd
        tableParams:
          - id: 6bba1b55-031c-4931-94ef-daccdc203f8d
            name: tbl
            columns:
              - name: fullname
                type: STRING
      requestId: 114a6659-4b7b-4a2a-8c22-d2177f91091a
      subRequestId: 8c1e7130-989a-4943-b23b-75e242e9d77e
      tableParamId: f3ff5c24-5039-4a00-8212-063114d6411b
      datamartMnemonic: fias
      chunkNum: 1
      isLast: true
      streamNum: 1
      streamTotal: 1
      customerId:
        string: agent-fias
      customerOgrn:
        string: "1053600591197"
      queryMnemonic:
        string: fias.selectAllAddrobj.1.0
```

## 22 replication.cancel.rq

`replication.cancel.rq` - Топик запросов на отмену подписки у поставщика

**Структура сообщения**

```
subscriptionCancelRequestMessage:
  description: Запрос на отмену подписки у поставщика
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    $ref: '#/components/schemas/subsriptionCancelRequest'
  examples:
    - name: simple
      headers:
```

```
      AGENT_CONSUMER_ID: agent-fias
    payload:
      requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId:
        string: 00000000-0000-0000-0000-000000000000
      subscriptionId: dcf43fc7-e152-459b-8af5-48d91d4b6a21
```

**Avro-схема сообщения**

```
subsriptionCancelRequest:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: SubscriptionAgentCancelRequest
  namespace: ru.rtlabs.common.replication.subscription
  fields:
    - name: requestId
      description: Уникальный идентификатор операции
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Идентификатор подзапроса, идентифицирует одно взаимодействие типа запрос-ответ
      default: null
      type:
        - 'null'
        - type: string
          logicalType: uuid
    - name: subscriptionId
      description: Уникальный идентификатор подписки
      type:
        type: string
        logicalType: uuid
```

## 23 replication.cancel.rs

`replication.cancel.rs` Топик с ответами на регистрацию подписки у поставщика.

Успешный ответ с результатом регистрации подписки у поставщика. Содержит сведения о структуре таблиц, необходимой для хранения реплик.

**Структура сообщения**

```
subscriptionCancelResultMessage:
  description: Ответ с результатом отмены подписки у поставщика
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    $ref: '#/components/schemas/subscriptionCancelResult'
  examples:
    - name: success
      headers:
```

```
      AGENT_CONSUMER_ID: agent-fias
    payload:
      requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId:
        string: 00000000-0000-0000-0000-000000000000
      subscriptionId: dcf43fc7-e152-459b-8af5-48d91d4b6a21
      success: true
      message: Подписка успешно отменена
  - name: error
    headers:
      AGENT_CONSUMER_ID: agent-fias
    payload:
      requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      subRequestId:
        string: 00000000-0000-0000-0000-000000000000
      subscriptionId: dcf43fc7-e152-459b-8af5-48d91d4b6a21
      success: false
      message: Непредвиденная ошибка отмены подписки
```

**Avro-схема сообщения**

```
subscriptionCancelResult:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: SubscriptionAgentCancelResult
  namespace: ru.rtlabs.common.replication.subscription
  fields:
    - name: requestId
      description: Уникальный идентификатор операции
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Идентификатор подзапроса, идентифицирует одно взаимодействие типа
запрос-ответ
      default: null
      type:
        - 'null'
        - type: string
          logicalType: uuid
    - name: subscriptionId
      description: Уникальный идентификатор подписки
      type:
        type: string
        logicalType: uuid
    - name: success
      description: Успешность выполнения отмены подписки
      type: boolean
    - name: message
      description: Сообщения с результатом выполнения операции
      default: null
      type:
        - 'null'
        - string
```

## 24 replication.err

`replication.err` - Топик с ошибками регистрации подписки у поставщика

Получение Агентом ПОДД ошибки при обработке подписки на репликацию от Витрины.

Негативный ответ на запрос подписки на репликацию - описание причины ошибки, передается только в случае невозможности выполнения запроса. Один запрос - один ответ (об

ошибке). Один ответ - одно сообщение.

**Структура сообщения**

```yaml
subscriptionRegistrationErrorMessage:
  description: Неуспешный результат регистрации подписки у поставщика
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    $ref: '#/components/schemas/subscriptionRegistrationError'
  examples:
    - name: simple
      headers:
        AGENT_CONSUMER_ID: agent-fias
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId:
          string: 00000000-0000-0000-0000-000000000000
        subscriptionId: dcf43fc7-e152-459b-8af5-48d91d4b6a21
        errorCode: DATAMART-001
        message: Непредвиденная ошибка
```

**Avro-схема сообщения**

```yaml
subscriptionRegistrationError:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: SubscriptionRegistrationError
  namespace: ru.rtlabs.common.replication.subscription
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса
      default: null
      type:
        - 'null'
        - type: string
          logicalType: uuid
    - name: subscriptionId
      description: Уникальный идентификатор подписки
      type:
        type: string
        logicalType: uuid
    - name: errorCode
      description: Код ошибки выполнения
      type: string
    - name: message
      description: Сообщение об ошибке
```

```
      type: string
```

### 25 replication.in.err

`replication.in.err` - Топик с ошибками обработки запросов на формирование структуры хранения реплик.

**Структура сообщения**

```
createReplicationStorageErrorMessage:
  description: Ответ с ошибкой обработки запроса на формирование структуры хранения реплик
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    $ref: '#/components/schemas/createReplicationStorageError'
  examples:
    - name: simple
      headers:
        AGENT_CONSUMER_ID: agent-oktmo
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId:
          string: 00000000-0000-0000-0000-000000000000
        subscriptionId: dcf43fc7-e152-459b-8af5-48d91d4b6a21
        errorCode: DATAMART-001
        message: Непредвиденная ошибка создания структуры хранения реплик
```

**Avro-схема сообщения**

```
createReplicationStorageError:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: CreateReplicationStorageError
  namespace: ru.rtlabs.common.replication.storage
  fields:
    - name: requestId
      description: Уникальный идентификатор операции
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор идентифицирует одно взаимодействие типа запрос-ответ
      default: null
      type:
        - 'null'
        - type: string
          logicalType: uuid
    - name: subscriptionId
      description: Уникальный идентификатор подписки
      type:
```

```
        type: string
        logicalType: uuid
    - name: errorCode
      description: Код ошибки выполнения
      type: string
    - name: message
      description: Сообщение об ошибке
      type: string
```

## 26 replication.in.rq

`replication.in` - Топик запросов к витрине потребителя на формирование хранилищ реплик.

**Структура сообщения**

```
createReplicationStorageRequestMessage:
  description: Запрос к витрине потребителя на формирование хранилища реплик
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    $ref: '#/components/schemas/createReplicationStorageRequest'
  examples:
    - name: replication
      headers:
        AGENT_CONSUMER_ID: agent-oktmo
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId:
          string: 00000000-0000-0000-0000-000000000000
        subscriptionId: dcf43fc7-e152-459b-8af5-48d91d4b6a21
        datamartMnemonic: oktmo
        lastSynId: 10
        table:
          tableId:
            string: 00000000-0000-0000-0000-000000000000
          tableName:
            string: tab
          sql:
          string: select * from v1_addrobj
          fields:
            - name: oktmo
              type: VARCHAR
              length: null
              precision: null
              scale: null
              primaryKey: null
              shardingKey: null
```

**Avro-схема сообщения**

```
createReplicationStorageRequest:
```

```
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: CreateReplicationStorageRequest
  namespace: ru.rtlabs.common.replication.storage
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса регистрации
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Идентификатор подзапроса, идентифицирует одно взаимодействие типа
запрос-ответ
      default: null
      type:
        - 'null'
        - type: string
          logicalType: uuid
    - name: subscriptionId
      description: Уникальный идентификатор подписки
      type:
        type: string
        logicalType: uuid
    - name: datamartMnemonic
      description: Наименование целевого датамарта, приходит в ПОДД в момент
регистрации подписки через ЕИП
      type: string
    - name: lastSynId
      description: Последний id синхронизации в витрине
      type: int
      default: 0
    - tables:
      description: Структура таблиц хранения реплик
      default: []
      type:
        type: array
        items:
          type: record
          name: ReplicationTableInfo
          namespace: ru.rtlabs.common.replication.storage
          fields:
            - name: tableId
              description: Идентификатор таблицы
              default: null
              type:
                - 'null'
                - type: string
                  logicalType: uuid
            - name: tableName
              description: Наименование таблицы
              type:
                - 'null'
                - type: string
            - name: sql
              description: sql, для которого должна быть создана таблица
              default: null
              type:
                - 'null'
                - type: string
            - name: fields
              description: Поля таблицы
              type:
```

```
              type: array
              items:
                type: record
                name: ReplicationFieldInfo
                namespace: ru.rtlabs.common.replication.storage
                description: Описание поля таблицы
                fields:
                  - name: name
                    description: Имя поля
                    type: string
                  - name: type
                    description: Тип поля
                    type: string
                  - name: length
                    description: Максимальная длина строки, если пусто, то нет
ограничений
                    type:
                      - int
                      - 'null'
                  - name: precision
                    description: Количество значимых цифр у decimal и numeric. Т.е.
количество в целой части + количество в дробной части. Если пусто, то без размера
                    type:
                      - int
                      - 'null'
                  - name: scale
                    description: ¥use only PODD¥ Масштаб decimal и numeric. Т.е.
количество значимых цифр в дробной части. Если пусто, то нет дробной части
                    default: null
                    type:
                      - 'null'
                      - int
                  - name: primaryKey
                    description: Порядковый номер поля (начиная с 0) в составе
первичного ключа. null - если не входит в состав первичного ключа
                    type:
                      - int
                      - 'null'
                  - name: shardingKey
                    description: Порядковый номер поля (начиная с 0) в составе ключа
шардирования. null - если не входит в состав ключа шардирования
                    type:
                      - int
                      - 'null'
```

## 27 replication.in.rs

`replication.in.rs` - Топик ответов с успешными результатами обработки запроса на формирование структуры хранения реплик.

**Структура сообщения**

```
createReplicationStorageResultMessage:
  description: Ответ с успешным результатом обработки запроса на формирование структуры
хранения реплик
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
```

```
  headers:
    type: object
    properties:
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    $ref: '#/components/schemas/createReplicationStorageResult'
  examples:
    - name: replication
      headers:
        AGENT_CONSUMER_ID: agent-oktmo
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId:
          string: 00000000-0000-0000-0000-000000000000
        subscriptionId: dcf43fc7-e152-459b-8af5-48d91d4b6a21
```

**Avro-схема сообщения**

```
createReplicationStorageResult:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: CreateReplicationStorageResult
  namespace: ru.rtlabs.common.replication.storage
  fields:
    - name: requestId
      description: Уникальный идентификатор операции
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Идентификатор подзапроса, идентифицирует одно взаимодействие типа
запрос-ответ
      default: null
      type:
        - 'null'
        - type: string
          logicalType: uuid
    - name: subscriptionId
      description: Уникальный идентификатор подписки
      type:
        type: string
        logicalType: uuid
```

## 28 replication.rq

`replication.rq` - Топик запросов регистрации подписки репликации у поставщика

**Структура сообщения**

```
subscriptionRegistrationRequestMessage:
  description: Запрос регистрации подписки репликации у поставщика
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
```

```
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    $ref: '#/components/schemas/subscriptionRegistrationRequest'
  examples:
    - name: simple
      headers:
        AGENT_CONSUMER_ID: agent-fias
      payload:
        requestId: 00000000-0000-0000-0000-000000000000
        subRequestId:
          string: 00000000-0000-0000-0000-000000000000
        subscriptionId: 00000000-0000-0000-0000-000000000000
        datamartMnemonic: fias
        sql:
          string: select * from v1_addrobj
        queries:
          - id: 00000000-0000-0000-0000-000000000000
            sql: select 1
            type: DATA
        isReplication:
          boolean: true
    - name: multiple
      headers:
        AGENT_CONSUMER_ID: agent-fias
      payload:
        requestId: 00000000-0000-0000-0000-000000000000
        subRequestId:
          string: 00000000-0000-0000-0000-000000000000
        subscriptionId: 00000000-0000-0000-0000-000000000000
        datamartMnemonic: fias
        sql:
          string: select * from v1_addrobj; select * from v1_house
        isReplication:
          boolean: true
```

**Avro-схема сообщения**

```
subscriptionRegistrationRequest:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: SubscriptionRegistrationRequest
  namespace: ru.rtlabs.common.replication.subscription
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса регистрации
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса, идентифицирует одно
взаимодействие типа запрос-ответ
      default: null
      type:
        - 'null'
        - type: string
          logicalType: uuid
    - name: subscriptionId
      description: Уникальный идентификатор подписки
      type:
        type: string
```

```
        logicalType: uuid
    - name: datamartMnemonic
      description: Мнемоника витрины, к которой выполняется запрос
      type: string
    - name: sql
      description: sql запрос текущей подписки для старых витрин тут может приходить
массив.
      default: null
      type:
        - 'null'
        - string
    - name: isReplication
      description: ¥version 1.1¥ Признак типа подписки true - подписка на репликацию,
false - подписка на уведомления
      default: null
      type:
        - 'null'
        - boolean
```

## 29 replication.rs

`replication.rs` - Топик с ответами на регистрацию подписки у поставщика

Позитивный ответ на запрос подписки на репликацию передается только в случае, если Витрина успешного обработала все выражения из параметра sql запроса. Один запрос - один ответ. Один ответ - одно сообщение.

**Структура сообщения**

```
subscriptionRegistrationResultMessage:
  description: Успешный ответ с результатом регистрации подписки у поставщика. Содержит
сведения о структуре таблиц, необходимой для хранения реплик
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор подзапроса
  headers:
    type: object
    properties:
      AGENT_CONSUMER_ID:
        description: Идентификатор агента потребителя
        type: string
  payload:
    $ref: '#/components/schemas/subscriptionRegistrationResult'
  examples:
    - name: simple
      headers:
        AGENT_CONSUMER_ID: agent-fias
      payload:
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        subRequestId:
          string: 00000000-0000-0000-0000-000000000000
        subscriptionId: dcf43fc7-e152-459b-8af5-48d91d4b6a21
        datamartMnemonic: fias
        lastSynId: 10
        snapshotKeySize: 1024
        deltaUuid: 79a928f5-eab5-49ea-a947-2c99fbf39b6e
        tables:
          - tableId: 90722850-b1c7-42eb-836c-b2f88849e22c
            tableName:
```

```
            string: tab
          sql: select * from v1_addrobj
          fields:
            - name: oktmo
              type: VARCHAR
              length: null
              precision: null
              scale: null
              primaryKey: null
              shardingKey: null
      tables:
        ru.rtlabs.common.replication.storage.ReplicationTableInfo:
          tableId:
            string: 00000000-0000-0000-0000-000000000000
          tableName:
            string: tab
          sql:
            string: select * from v1_addrobj
          fields:
            - name: oktmo
              type: VARCHAR
              length: null
              precision: null
              primaryKey: null
              scale: null
              shardingKey: null
```

**Avro-схема сообщения**

```
subscriptionRegistrationResult:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: SubscriptionRegistrationResult
  namespace: ru.rtlabs.common.replication.subscription
  fields:
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: subRequestId
      description: Уникальный идентификатор подзапроса,
      default: null
      type:
        - 'null'
        - type: string
          logicalType: uuid
    - name: subscriptionId
      description: Уникальный идентификатор подписки
      type:
        type: string
        logicalType: uuid
    - name: datamartMnemonic
      description: Мнемоника витрины-источника
      type: string
    - name: lastSynId
      description: Последний id дельты примененной в витрине на момент подписки (если
дельт нет, то придет -1)
      type: int
    - name: tables
      description: Структура таблиц хранения реплик
      default: []
```

```
      type:
        type: array
        items:
          type: record
          name: ReplicationTableInfo
          namespace: ru.rtlabs.common.replication.storage
          fields:
            - name: tableId
              description: Идентификатор таблицы
              default: null
              type:
                - 'null'
                - type: string
                  logicalType: uuid
            - name: tableName
              description: Наименование таблицы
              type:
                - 'null'
                - type: string
            - name: sql
              description: sql, для которого должна быть создана таблица
              default: null
              type:
                - 'null'
                - type: string
            - name: fields
              description: Поля таблицы
              type:
                type: array
                items:
                  type: record
                  name: ReplicationFieldInfo
                  namespace: ru.rtlabs.common.replication.storage
                  description: Описание поля таблицы
                  fields:
                    - name: name
                      description: Имя поля
                      type: string
                    - name: type
                      description: Тип поля
                      type: string
                    - name: length
                      description: Максимальная длина строки, если пусто, то нет
ограничений
                      type:
                        - int
                        - 'null'
                    - name: precision
                      description: Количество значимых цифр у decimal и numeric. Т.е.
количество в целой части + количество в дробной части. Если пусто, то без размера
                      type:
                        - int
                        - 'null'
                    - name: scale
                      description: Масштаб decimal и numeric. Т.е. количество значимых
цифр в дробной части. Если пусто, то нет дробной части
                      default: null
                      type:
                        - 'null'
                        - int
                    - name: primaryKey
                      description: Порядковый номер поля (начиная с 0) в составе
```

```
первичного ключа. null - если не входит в состав первичного ключа
                    type:
                      - int
                      - 'null'
                  - name: shardingKey
                    description: Порядковый номер поля (начиная с 0) в составе ключа
шардирования. null - если не входит в состав ключа шардирования
                    type:
                      - int
                      - 'null'
```

## 30 statistics.err

`statistics.err` - Топик с ошибками получения статистики витрины

**Структура сообщения**

```yaml
datamartStatisticErrorMessage:
  description: Неуспешный результат обработки запроса на получение статистики
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор запроса
  headers:
    type: object
    properties:
      REQUEST_ID:
        description: Идентификатор запроса
        type: string
  payload:
    $ref: '#/components/schemas/datamartStatisticError'
  examples:
    - name: simple
      headers:
        REQUEST_ID: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      payload:
        protocol: read.statistic.protocol.v.1
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        errorCode: DATAMART-001
        message: Непредвиденная ошибка
```

**Avro-схема сообщения**

```yaml
datamartStatisticError:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: DatamartStatisticError
  namespace: ru.rtlabs.common.statistic
  fields:
    - name: protocol
      type: string
      description: Версия протокола. Указывается константа read.statistic.protocol.v.1
      conts: read.statistic.protocol.v.1
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: errorCode
      description: Код ошибки
```

```
      type: string
    - name: message
      description: Сообщение об ошибке
      type: string
```

## 31 statistics.rq

`statistics.rq` - Топик запросов статистики витрины

**Структура сообщения**

```
datamartStatisticRequestMessage:
  description: Запрос статистики витрины
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        $ref: '#/components/schemas/datamartStatisticRequestKey'
  headers:
    type: object
    properties:
      REQUEST_ID:
        description: Идентификатор запроса
        type: string
  payload:
    $ref: '#/components/schemas/datamartStatisticRequest'
  examples:
    - name: simple
      headers:
        REQUEST_ID: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      payload:
        protocol: read.statistic.protocol.v.1
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        datamart:
          mnemonic: fias
          version:
            major: 1
            minor: 0
```

**Avro-схема сообщения**

```
datamartStatisticRequest:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  type: record
  name: DatamartStatisticRequest
  namespace: ru.rtlabs.common.statistic
  fields:
    - name: protocol
      description: Версия протокола. Указывается константа read.statistic.protocol.v.1
      type: string
    - name: requestId
      description: Уникальный идентификатор запроса
      type:
        type: string
        logicalType: uuid
    - name: datamart
      description: Витрина
      type:
        type: record
        name: DatamartInfo
        fields:
          - name: mnemonic
            description: Мнемоника витрины
```

```
          type: string
        - name: version
          description: Версия
          type:
            type: record
            name: SemanticVersion
            namespace: ru.rtlabs.common.model.metadata
            fields:
              - name: major
                type: int
                minimum: 1
              - name: minor
                type: int
                minimum: 0
```

## 32 statistics.rs

`statistics.rs` - Топик со статистикой витрины

**Структура сообщения**

```
datamartStatisticResponseMessage:
  description: Статистика витрины
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
  bindings:
    kafka:
      key:
        type: string
        format: uuid
        description: Уникальный идентификатор запроса
  headers:
    type: object
    properties:
      REQUEST_ID:
        description: Идентификатор запроса
        type: string
  payload:
    $ref: '#/components/schemas/datamartStatisticResponse'
  examples:
    - name: simple
      headers:
        REQUEST_ID: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
      payload:
        protocol: read.statistic.protocol.v.1
        requestId: 2e8c8ab2-44db-4dcb-8ae5-2365121b4e14
        datamart:
          mnemonic: fias
          version:
            major: 1
            minor: 0
          tables:
            - mnemonic: addrobj
              columns:
                - mnemonic: oktmo
                  notGreater10: 10.0
                  inRange11And100: 50.0
                  inRange101And1000: 30.0
                  moreThan1000: 10.0
```

**Avro-схема сообщения**

```
datamartStatisticResponse:
  schemaFormat: 'application/vnd.apache.avro;version=1.9.0'
```

```yaml
type: record
name: DatamartStatisticResponse
namespace: ru.rtlabs.common.datamart.profile
fields:
  - name: protocol
    description: Версия протокола. Указывается константа read.statistic.protocol.v.1
    type: string
  - name: requestId
    description: Уникальный идентификатор запроса
    type:
      type: string
      logicalType: uuid
  - name: datamart
    description: Статистика по витрине
    type:
      type: record
      name: DatamartStatistic
      fields:
        - name: mnemonic
          description: Мнемоника витрины
          type: string
        - name: version
          description: Версия
          type:
            type: record
            name: SemanticVersion
            namespace: ru.rtlabs.common.model.metadata
            fields:
              - name: major
                type: int
                minimum: 1
              - name: minor
                type: int
                minimum: 0
        - name: tables
          type:
            type: array
            items:
              type: record
              name: TableStatistic
              fields:
                - name: mnemonic
                  description: Мнемоника витрины
                  type: string
                - name: columns
                  description: Колонки
                  type:
                    type: array
                    items:
                      type: record
                      name: ColumnStatistic
                      description: Статистика по колонке
                      fields:
                        - name: mnemonic
                          type: string
                        - name: notGreater10
                          type: double
                        - name: inRange11And100
                          type: double
                        - name: inRange101And1000
                          type: double
                        - name: moreThan1000
```

```
                        type: double
```

# 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 Avro**

Линейно-ориентированный (строчный) формат передачи наборов данных, используемый в качестве платформы сериализации, разрабатываемый в рамках фонда Apache.

**Apache Hadoop**

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

**Apache Kafka**

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

**Apache Spark**

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

**API**

Application programming interface (англ.) - Программный интерфейс приложения, описание сервисов взаимодействия компьютерной программы с другими программами.

**BLOB-адаптер**

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

**ClickHouse**

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

**Counter-Provider**

Сервис генерации уникального номера.

**CSV**

Comma-Separated Values (англ.) - 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.

**DAG**

Файл, содержащий блок данных.

**DATA-uploader**

Модуль исполнения асинхронных заданий.

**DBeaver**

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

**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.

**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**.

**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.

**Grafana**

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

**Greenplum**

Массово-параллельная СУБД для хранилищ данных на основе PostgreSQL.

**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**

Библиотека классов, реализующая стандарт JDBC и подключения к источнику данных с использованием специализированного протокола, поддерживаемого источником данных.

**JDBC-extractor**

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

**JSON**

JavaScript Object Notation - Общий формат для представления значений и объектов в соответствии со стандартом RFC 4627.

**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, сервис исполнения запросов.

**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.

**SQL-запрос**

Запрос к Витрине данных Поставщика. Произвольный или регламентированный запрос к данным, сформулированный на языке SQL.

**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 СМЭВ4 (Агент)**

Типовое программное обеспечение, устанавливаемое в контуре ИС УВ и обеспечивающее сопряжение Витрин данных и ИС УВ с Ядром СМЭВ4.

**База данных**

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

**(Большой) Двоичный объект (BLOB / БЛОБ)**

Тип данных, значение которого представляет собой массив байт, размер которого существенно превышает размер базовых скалярных типов (int, float, double, date)

**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 взаимодействия, обеспечивающий хранение и предоставление данных другим Участникам взаимодействия с использованием СМЭВ4.

**Вид сведения СМЭВ (ВС)**

Комплекс документальных и программных компонентов, зарегистрированный в СМЭВ 3.x, обеспечивающий взаимодействие ИС ведомств в определённом формате и по определённым правилам.

**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.

**ИС УВ**

Информационная система Участника взаимодействия.

**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.

**ЛК УВ**

Личный кабинет участника взаимодействия. Система, предназначенная для управления информационными системами и мониторинга информационных обменов в СМЭВ 3 и СМЭВ 4 участниками взаимодействия.

**Логическая модель данных**

Схема базы данных, выраженная в понятиях бизнес-требований.

**Мнемоника Витрины**

Уникальное строковое значение, определяющее модель данных Витрины.

**Модель данных Витрины**

Описание структуры Витрины (общая информация, перечень сущностей, атрибутный состав), загруженное в Ядро СМЭВ4.

**Набор данных**

Совокупность систематизированных данных (датасетов), представляющих собой базовый элемент для работы с данными.

**НСУД**

Национальная система управления данными.

**ОГРН**

Основной государственный регистрационный номер, присваивается юридическим лицам сразу же после регистрации в ФНС РФ.

**Параметр запроса**

Символическое имя, входящее в текст SQL-запроса и не содержащееся в Модели данных Витрины, в терминах которой сформулирован SQL-запрос.

**ПО**

Программное обеспечение.

**СМЭВ4-адаптер**

Программно-технический продукт, обеспечивающий взаимодействие витрины и СМЭВ4.

**СМЭВ4-адаптер - Модуль исполнения запросов**

Логический модуль СМЭВ4-адаптера, предназначен для исполнения запросов СМЭВ4 (через протокол коммуникации Агент СМЭВ4).

**СМЭВ4-адаптер - Модуль MPPR**

Логический модуль СМЭВ4-адаптера, предназначен для чтения данных в многопоточном режиме (massively parallel processing, **MPP**).

**СМЭВ4-адаптер - Модуль MPPW**

Логический модуль СМЭВ4-адаптера выполняет загрузку данных в многопоточном режиме.

**Подписка (потребителя)**

Предоставление права Потребителю данных СМЭВ4 на информационный обмен с использованием Регламентированного запроса типа «Рассылка».

**Поставщик данных**

Участник взаимодействия, являющийся источником данных для других участников и использующий СМЭВ4 для передачи данных.

**Потребитель данных**

Участник взаимодействия, получающий данные от Поставщиков данных для дальнейшей их обработки и использующий для передачи запросов и получения данных СМЭВ4.

**Распределенный запрос**

Регламентированный запрос, инициированный Потребителем, SQL-выражение которого содержит наборы данных из двух или более Витрин данных.

**Регламентированный SQL-запрос (РЗ)**

SQL-запрос, выраженный в терминах Модели данных, загруженной в СМЭВ4, и зарегистрированный в Ядре СМЭВ4 под символической мнемоникой, используемой ИС Потребителя СМЭВ4 для выполнения регламентированного запроса. Может иметь параметры, значения которых задаются Потребителем данных СМЭВ4 при выполнении регламентированного запроса.

**Реплика**

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

**Сервис Формирования документов**

Модуль витрины, предназначенный для работы с формируемыми документами.

**СМЭВ**

Система межведомственного электронного взаимодействия.

**СМЭВ 3**

Единая система межведомственного электронного взаимодействия, функционирующая в соответствии с Методическими рекомендациям по работе со СМЭВ версии `3.x`.

**СМЭВ3-адаптер**

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

**СМЭВ4**

единый сервис доступа к данным СМЭВ, предназначенный для автоматизации процесса передачи данных и уведомлений об изменении данных между организациями или органами власти, ответственными за формирование и ведение информационных ресурсов, зарегистрированных в НСУД.

**Сообщение**

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

**СУБД**

Система управления базами данных.

**Табличный параметр (запроса)**

Параметр, значение которого представляет собой двумерный массив с именованными колонками и неупорядоченными строками. Формальный табличный параметр может использоваться в инструкциях `FROM`, `JOIN` как источник данных.

**Токен**

Ключ безопасности (Цифровой сертификат).

**Участник взаимодействия**

Орган или организация, участвующий в информационном обмене через СМЭВ.

**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.

**Чанк**

Фрагмент результирующих данных оптимального для передачи по сети размера.