# Arenadata™ Database

*Версия - v6.11.2-arenadata9-dev*

## Интеграция кластера ADB и Kafka

*2021 Arenadata™*

# Оглавление

**1 ADB to Kafka (kafka-connector) 3**
1.1 Установка с помощью ADCM . . . 3
1.2 Установка из rpm-пакетов . . . 3
1.3 Пример использования . . . 3

**2 Kafka to ADB (kadb-fdw) 5**
2.1 Установка с помощью ADCM . . . 5
2.2 SQL-интерфейс . . . 5
2.3 Примеры использования . . . 11

В документе приведено краткое описание интеграции кластера Arenadata DB и кластера брокера сообщений Kafka.

**Important:** Контактная информация службы поддержки – e-mail: info@arenadata.io

# Глава 1

# ADB to Kafka (kafka-connector)

## 1.1 Установка с помощью ADCM

Для установки коннектора с помощью **ADCM** требуется инсталлировать сервис *PXF* на все сегментные ноды кластера, а также в списке сервисов выбрать сервис *ADB to Kafka*. При этом необходимые пакеты и файлы автоматически устанавливаются на машины кластера.

## 1.2 Установка из rpm-пакетов

Установка из rpm-пакетов предполагает, что в кластере *ADB* установлен сервис *PXF*.

Для установки коннектора из rpm-пакетов необходимо:

1. Установить пакет *kafka-connector* на всех хостах кластера *ADB*, где установлен сервис *PXF*;
2. Добавить в файл */etc/pxf/conf/pxf-profiles-default.xml* на каждом хосте кластера следующую секцию:

```
<profile>
    <name>kafka</name>
    <description>A profile for export data into Apache Kafka</description>
    <plugins>
        <accessor>org.greenplum.pxf.plugins.kafka.KafkaAccessor</accessor>
        <resolver>org.greenplum.pxf.plugins.kafka.KafkaResolver</resolver>
    </plugins>
    <optionMappings>
        <mapping option="BOOTSTRAP_SERVERS" property="kafka.bootstrap.servers"/>
        <mapping option="BATCH_SIZE" property="kafka.batch.size"/>
    </optionMappings>
</profile>
```

3. Перезапустить сервис *PXF* на всех хостах кластера.

## 1.3 Пример использования

```
CREATE WRITABLE EXTERNAL TABLE kafka_tbl (a TEXT, b TEXT, c TEXT)
  LOCATION ('pxf://<topic>?PROFILE=kafka&BOOTSTRAP_SERVERS=<server>')
  FORMAT 'CUSTOM' (FORMATTER='pxfwritable_export');

INSERT INTO kafka_tbl VALUES ('a', 'b,c', 'd'), ('x', 'y', 'z');
```

```
DROP EXTERNAL TABLE kafka_tbl;
```

# Глава 2

# Kafka to ADB (kadb-fdw)

**kadb-fdw** - это расширение для ADB/GPDB, позволяющее производить транзакционную загрузку данных из кластера брокера сообщений Kafka.

**Особенности**:

- AVRO десереализация;
- Хранение смещений Kafka вне кластера Kafka, на стороне потребителя;
- Поддержка транзакций ADB/GPDB;
- Поддержка Kerberos-аутентификации.

## 2.1 Установка с помощью ADCM

Для установки коннектора с помощью **ADCM** требуется инсталлировать сервис *PXF* на все сегментные ноды кластера, а также в списке сервисов выбрать сервис *Kafka to ADB*. При этом необходимые пакеты и файлы автоматически устанавливаются на машины кластера.

**Important:** Процесс установки включает в себя выполнение следующей команды: **RESET client_min_messages**

## 2.2 SQL-интерфейс

### 2.2.1 Пример

```
-- Create a SERVER
DROP SERVER IF EXISTS ka_server;
CREATE SERVER ka_server
FOREIGN DATA WRAPPER kadb_fdw
OPTIONS (
    k_brokers 'localhost:9092'
);

-- Create a FOREIGN TABLE
DROP FOREIGN TABLE IF EXISTS ka_table;
CREATE FOREIGN TABLE ka_table(field1 INT, field2 TEXT)
SERVER ka_server
```

```
OPTIONS (
    format 'avro',
    k_topic 'test',
    k_consumer_group 'my_consumer_group',
    k_seg_batch '100',
    k_timeout_ms '1000',
    k_initial_offset '42'
);

-- Issue a SELECT query as usual
SELECT * FROM ka_table;
```

### 2.2.2 Таблица смещений

В ходе установки расширения создается служебная схема `kadb`, содержащая таблицу `kadb.offsets`. В данную таблицу помещаются пары соответствий партиция-смещение для любой когда-либо создававшейся в текущей базе внешней таблицы (FOREIGN TABLE).

Таблицы идентифицируются по значению OID, который можно узнать, выполнив следующую команду:

```
SELECT '<table name>'::regclass::oid;
```

При выполнении SELECT-запроса к внешней таблице чтение сообщений из Kafka производится, начиная со смещения, указанного для данной внешней таблицы в таблице `kadb.offsets`.

Смещения можно изменять при помощи обычного SQL-запроса к таблице `kadb.offsets`.

Для новых партиций, записей о которых нет в таблице `kadb.offsets`, начальное смещение по умолчанию устанавливается равным 0 (значение может быть изменено в параметре *k_initial_offset*).

После успешного выполнения SELECT-запроса к внешней таблице смещение обновляется в соответствии со значениями, полученными от Kafka. Например, если последнее прочитанное сообщение из некоторой партиции имело смещение 84, значение смещения для данной партиции в таблице `kadb.offsets` будет равным 85.

### 2.2.3 Опции внешних таблиц

Для сервера (SERVER) и внешней таблицы (FOREIGN TABLE) возможно указание опций в виде пары ключ-значение.

Опции, определенные для сервера и внешней таблицы, не отличаются друг от друга. Иными словами, не имеет значения, для какого объекта они были указаны. Однако опции, определенные для внешней таблицы, более приоритетны, чем опции сервера (в случае, если для обоих объектов была указана одинаковая опция).

**Поддерживаемые опции:**

#### k_brokers

*Обязательна.*

Список брокеров Kafka, разделенный запятыми, где каждый элемент — это строка вида <host> или <host>:<port>.

#### k_topic

*Обязательна.*

Идентификатор топика Kafka.

### k_consumer_group

*Обязательна.*

Идентификатор группы потребителей Kafka.

### k_initial_offset

*Неотрицательное целое число. Значение по умолчанию: 0.*

Смещение, которое следует использовать для партиций, записей о которых нет в *таблице смещений*.

### k_allow_offset_increase

*Булево значение.*
*Значение по умолчанию: true.*

Позволять коннектору автоматически увеличивать смещение для любой партиции, если он обнаруживает, что наименьшее смещение среди существующих сообщений в Kafka больше, чем смещение, хранящееся в *таблице смещений*, или чем значение опции *k_initial_offset*. В случае, если опция имеет значение false, и наименьшее смещение существующего в Kafka сообщения больше, чем желаемое начальное смещение, генерируется ошибка.

### k_seg_batch

*Обязательна.*
*Положительное целое число.*

Максимальное количество сообщений Kafka, запрашиваемое каждым сегментом кластера ADB/GPDB.

При выполнении запроса с условием LIMIT сообщения в Kafka продолжают запрашиваться батчами. В результате смещения в *таблице смещений* устанавливается на основании *последнего* полученного сообщения для каждой партиции, даже если данные не содержались в результатах запроса.

При достижении конца партиции Kafka отправляет специальное сообщение. Таким образом, если значение `k_seg_batch` меньше, чем число партиций, определенное для одного сегмента ADB/GPDB и `k_seg_batch` этих партиций было полностью прочитано, может возникнуть ситуация, при которой не будет прочитано никаких полезных данных. По этой причине опция `k_seg_batch` *должна быть больше, чем максимальное число партиций определенных для одного сегмента.*

### k_timeout_ms

*Обязательна.*
*Неотрицательное целое число.*

Время ожидания выполнения запроса к Kafka в миллисекундах. За это время Kafka-клиентом будут извлечены и представлены в качестве результата SELECT-запроса только сообщения, доступные в Kafka в этот период времени.

SELECT-запрос может выполняться быстрее, если в топике Kafka достаточно сообщений.

Реальное время выполнения SELECT-запроса может быть значительно больше. Для оценки максимального значения используйте следующее выражение:

*[duration] = [k_timeout_ms] * (2 + ceil( [number of partitions] / [number of segments in cluster] ))*

Более всего на длительность запроса влияют партиции Kafka, в которых недостаточно сообщений: коннектор ожидает появления в этих партициях новых сообщений в течение *k_timeout_ms* миллесекунд.

На некоторых этапах выполнения SELECT-запроса его принудительная отмена может быть невозможна до окончания периода k_timeout_ms.

#### k_security_protocol

*Обязательна, если используется Kerberos-аутентификация.*

Протокол безопасности, который необходимо использовать для подключения к Kafka. В данный момент доступен только протокол sasl_plaintext.

#### kerberos_keytab

*Обязательна, если используется Kerberos-аутентификация.*

Путь до keytab-файла для Kerberos-аутентификации. Этот файл должен быть доступен для пользователя, от имени которого происходит запуск процесса ADB/GPDB, а также должен присутствовать на каждом сегменте кластера (по одному и тому же пути).

Установка данной опции включает Kerberos-аутентификацию.

#### kerberos_principal

*Значение по умолчанию: kafkaclient.*

Имя Kerberos-принципала клиента, который получает доступ к Kafka.

#### kerberos_service_name

*Значение по умолчанию: kafka.*

Имя Kerberos-принципала, который использует Kafka (не включая `/hostname@REALM`).

При указании данного параметра librdkafka передает его значение как первый аргумент (service) в вызов sasl_client_new()

#### kerberos_min_time_before_relogin

*Неотрицательное целое число.*
*Значение по умолчанию: 6000.*

Минимальное время в миллисекундах между попытками обновления ключа. Для отключения автоматического обновления ключа неоходимо выставить опцию в значение 0.

### format

*Обязательна.*
*Одно из предопределенных значений (без учета регистра).*

Формат сереализованных данных:

- avro
- csv

### avro_schema

AVRO-схема, которую необходимо использовать. Десериализованная AVRO схема представляет из себя JSON. Получаемые сообщения десериализуются двумя способами:

- Если присутствует опция `avro_schema`, используется указанная схема (сообщение так же должно быть представлено в OCF формате)
- В противном случае, схема извлекается из полученного сообщения в OCF формате.

**Important:** Пользовательская схема не может быть валидирована. Если реальная схема не соответствует указанной, десериализация завершается с ошибкой *ERROR: invalid memory alloc request size*. По этой причине опция `avro_schema` должна использоваться только в целях повышения производительности и только после тщательного изучения.

### csv_quote

*Одиночный символ, представимый одним байтом в текущей кодировке.*
*Значение по умолчанию: “”“.*

Символ, используемый в качестве кавычек при парсинге CSV.

### csv_delimeter

*Одиночный символ, представимый одним байтом в текущей кодировке.*
*Значение по умолчанию: “,“.*

Символ, используемый в качестве разделителя полей при парсинге CSV.

### csv_null

Строка, используемая в качестве значения `NULL` в CSV.

*По умолчанию пустая строка интерпретируется как NULL.*

### csv_ignore_header

*Булево значение.*
*Значение по умолчанию: false.*

Игнорировать первую строку каждого сообщения при парсинге.

#### csv_attribute_trim_whitespace

*Булево значение.*
*Значение по умолчанию: true.*

Удалять пробелы в начале и конце строки для каждого атрибута (поля) записи.

### 2.2.4 Десериализация

В настоящее время расширение *kafka-fdw* поддерживает сообщения Kafka, сериализованные при помощи одного из следующих форматов:

- AVRO OCF
- CSV

Метод десериалиализации должен быть указан в опции *format*.

Вне зависимости от используемого метода десериализуется только сообщение Kafka, ключ сообщения игнорируется.

#### AVRO

*kadb-fdw* предоставляет поддержку формата сериализации AVRO OCF с некоторыми ограничениями.

Схема должна содержать только примитивные типы данных или объединения одного примитивного типа данных с типом `NULL`.

В настоящее время логические типы данных не поддерживаются. Они обрабатываются как обычные типы.

**Пример схемы:**

```
{
   "name": "doc",
   "type": "record",
   "fields": [
      {
         "name": "id",
         "type": "int"
      },
      {
         "name": "text",
         "type": ["string", "null"]
      },
      {
         "name": "issued_on",
         "type": "int",
      },
   ]
}
```

#### CSV

Поддержка CSV реализована при помощи libcsv. Таким образом, учитываются все соглашения о формате CSV, устанавливаемые этой библиотекой. Перечень соглашений представлен в данном документе. Спецификация CSV определена в RFC 4180.

Принимая во внимание эти рекомендации, *kadb-fdw* использует следующие правила парсинга CSV:

- Поля (атрибуты) разделяются специальным символом (*разделителем*);

- Строки (записи) разделяются символом новой строки;
- Поле может быть заключено в специальные символы (*кавычки*);
- Поля, содержащие символ разделителя, кавычки или новой строки, должны быть заключены в кавычки;
- Каждый символ *кавычки* должен быть экранирован путем его дублирования (добавления идентичного символа перед ним);
- Пустое поле всегда интерпретируется как значение NULL;
- Пустые строки игнорируются;
- Пробелы в начале и конце поля, не заключенного в кавычки, удаляются, если соответствующая *опция* выставлена в значение true.

Значения могут быть конвертированы в любой тип данных PostgreSQL. Используемые правила конвертации аналогичны правилам, применяемым для входных данных *psql*.

## 2.3 Примеры использования

### 2.3.1 Таблица для данных в формате AVRO

```
DROP SERVER IF EXISTS ka_server CASCADE;
CREATE SERVER ka_server
FOREIGN DATA WRAPPER kadb_fdw
OPTIONS (
    k_brokers 'localhost:9092'
);

CREATE FOREIGN TABLE ka_avro_table(
    i INT,
    t TEXT
)
SERVER ka_server
OPTIONS (
    format 'avro',
    k_topic 'my_topic',
    k_consumer_group 'my_consumer_group',
    k_seg_batch '5',
    k_timeout_ms '1000'
);
```

### 2.3.2 Таблица для данных в формате CSV

```
DROP SERVER IF EXISTS ka_server CASCADE;
CREATE SERVER ka_server
FOREIGN DATA WRAPPER kadb_fdw
OPTIONS (
    k_brokers 'localhost:9092'
);

CREATE FOREIGN TABLE ka_csv_table(
    i INT,
    t TEXT
)
SERVER ka_server
OPTIONS (
    format 'csv',
```

```
    k_topic 'my_topic',
    k_consumer_group 'my_consumer_group',
    k_seg_batch '5',
    k_timeout_ms '1000'
);
```

### 2.3.3 Таблица с Kerberos-аутентификацией

```
DROP SERVER IF EXISTS ka_kerberized_server CASCADE;
CREATE SERVER ka_kerberized_server
FOREIGN DATA WRAPPER kadb_fdw
OPTIONS (
    k_brokers 'ke-kafka-sasl.ru-central1.internal:9092',
    k_security_protocol 'sasl_plaintext',
    kerberos_keytab '/root/adbkafka.service.keytab',
    kerberos_principal 'adbkafka'
);

CREATE FOREIGN TABLE ka_table(
    i INT,
    t TEXT
)
SERVER ka_kerberized_server
OPTIONS (
    format 'avro',
    k_topic 'my_topic',
    k_consumer_group 'my_consumer_group',
    k_seg_batch '5',
    k_timeout_ms '1000'
);
```