

Arenadata™ Database

Версия - latest_5.X

Интеграция кластера ADB и Kafka

Оглавление

1	Системные требования	3
2	Установка коннектора	4
3	Использование коннектора	5
4	Ограничения	7
5	Пример интеграции	8

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

Системные требования

Для работы коннектора **ADB Kafka** необходимо выполнение следующих условий:

1. Все сегмент-сервера кластера ADB имеют сетевой доступ до всех брокеров и Zookeeper-нод кластера Kafka.
2. Все сегмент-сервера кластера ADB имеют записи в */etc/hosts*, содержащие *hostname* всех брокеров и Zookeeper-нод кластера Kafka и их ip-адреса.

Глава 2

Установка коннектора

Для инсталляции коннектора необходимо установить пакет *adb_kafka_external_connector-<version>.rpm* на все сервера-сегменты кластера. Пакет добавляет на сервера следующие исполняемые файлы:

1. */usr/lib/adbkafka/adb-kafka-consumer*.
2. */usr/lib/adbkafka/adb-kafka-producer*.

Глава 3

Использование коннектора

Для чтения данных из кластера **Kafka** в кластер **ADB** необходимо воспользоваться внешней таблицей (*WEB EXTERNAL TABLE*), обращающейся к установленному consumer'у. Пример:

```
create external web table <table_name> (<column_name> <data_type> , ...)
EXECUTE '/usr/lib/adbkafka/adb-kafka-consumer -kafka-topic <topic_name> -zookeeper-hosts
→<zookeeper_hosts> 2>><log_file>_$GP_SEGMENT_ID.log' ON ALL
FORMAT 'CSV';
```

Директива *ON ALL* указывает, что из кластера **Kafka** данные запрашиваются всеми сегментами СУБД (максимальный уровень параллельности).

Important: Число партиций в топике Kafka должно быть равно или больше числа сегментов ADB, запрашивающих данные. В случае, если партиций в топике меньше, чем сегментов в кластере, необходимо скорректировать число сегментов при создании внешней таблицы. Например, *ON 2*

При необходимости указываются пользовательские значения разделителей и escape-символов. При этом доступны следующие опции consumer'a:

- *-consumer-group string* – название общей группы всех consumer'ов ADB. В рамках одной группы используется *read once* – каждая строка читается ровно один раз (по умолчанию *ADB_consumers*);
- *-kafka-topic string* – название топика Kafka (обязательно);
- *-key* – флаг выдачи ключа каждой записи (опционально);
- *-timeout int* – таймаут ожидания данных;
- **-timestamp* – флаг выдачи времени создания каждой записи (опционально);
- *-verbose* – флаг расширенного логирования (опционально);
- *-zookeeper-hosts string* – разделенные запятыми *хост:порт* узлов Zookeeper (по умолчанию *localhost:2181*).

Для записи используется producer и внешняя таблица на запись (*WRITABLE EXTERNAL WEB TABLE*), соответственно:

```
create writable external web table <table_name> (<column_name> <data_type> , ...)
EXECUTE '/usr/lib/adbkafka/adb-kafka-producer -kafka-brokers <brokers> -kafka-topic <topic> 2>>/
→tmp/<log_file>_$GP_SEGMENT_ID.log'
FORMAT 'CSV'
DISTRIBUTED BY (<distribution_key>);
```

Доступны следующие опции producer'a:

- `-batch int` – размер батча при записи данных (по умолчанию *5000*);
- `-kafka-brokers string` – разделенные запятыми *хост:порт* Kafka брокеров (по умолчанию *localhost:9092*);
- `-kafka-topic string` – имя топика Kafka;
- `-pause int` – длительность паузы между записью батчей данных (по умолчанию *0*).

Глава 4

Ограничения

При интеграции кластера **ADB** и **Kafka** имеются следующие ограничения:

- Создание внешних таблиц коннектора доступно только суперпользователям БД. Использование таблиц не ограничено;
- Необходимо, чтобы число партиций в топике Kafka было равно или больше числа сегментов ADB, запрашивающих данные. В случае, если партиций в топике меньше, чем сегментов в кластере, следует скорректировать число сегментов при создании внешней таблицы. Например, ON 2.

Глава 5

Пример интеграции

1. Создание таблицы для тестовых данных:

```
create table kafka_test_data (  
  id int,  
  text1 text,  
  text2 text,  
  some_number real,  
  dtm timestamp with time zone  
) with (appendonly=true, orientation=column, compresstype=zstd, compresslevel=1)  
distributed by (id);  
  
create table kafka_test_data_input (like kafka_test_data) with (appendonly=true,  
→orientation=column, compresstype=zstd, compresslevel=1)  
distributed by (id);
```

2. Генерация тестовых данных:

```
insert into kafka_test_data select gen, 'Some text #' || gen::text, 'Some another text #' ||  
→gen::text, gen/(gen+1)::real, timeofday()::timestamp from generate_series(1, 1000000) gen;
```

3. Создание топика в Kafka с необходимым числом партиций (выполняется на сервере Kafka):

```
bin/kafka-topics.sh --create --zookeeper 10.0.214.15:2181 --replication-factor 1 --partitions 30 -  
→-topic mytopic7
```

4. Создание WRITABLE-внешней партиции:

```
create writable external web table kafka_write_example (like kafka_test_data)  
EXECUTE '/usr/lib/adbkafka/adb-kafka-producer -kafka-brokers 10.0.214.20:9092,10.0.214.21:9092 -  
→kafka-topic mytopic7 -batch 100000 -pause 0 2>>/tmp/adb_kafka_producer$(date "+%Y-%m-%d")_$GP_  
→SEGMENT_ID.log'  
FORMAT 'CSV'  
DISTRIBUTED BY (id);
```

5. Перенос данных в Kafka:

```
insert into kafka_write_example select * from kafka_test_data;
```

6. Создание READABLE-внешней таблицы:

```
create external web table kafka_read_example (like kafka_test_data)
EXECUTE '/usr/lib/adbkafka/adb-kafka-consumer -kafka-topic mytopic7 -zookeeper-hosts 10.0.214.
→15:2181 -consumer-group adb7 -timeout 20000 2>>/tmp/adb_kafka_consumer$(date "+%Y-%m-%d")_$GP_
→SEGMENT_ID.log' ON ALL
FORMAT 'CSV';
```

7. Чтение данных:

```
select count(*) from kafka_read_example;
```