ETL КОНВЕЙЕР ДЛЯ ПОТОКОВОЙ ОБРАБОТКИ ТЕКСТОВЫХ ДАННЫХ ПОД УПРАВЛЕНИЕМ РАСПРЕДЕЛЕННОГО БРОКЕРА СООБЩЕНИЙ APACHE KAFKA

Кузьменко Д.Е.[1], Кайда А.Ю.[2]
[1] НИ ТПУ, ИШИТР, 8ПМ21, e-mail: dek29@tpu.ru
[2] НИ ТПУ, ИШИТР, ст. преп. ОИТ, e-mail: ayk13@tpu.ru

Введение

С развитием онлайн-сервисов объемы обрабатываемых данных многократно выросли. Они, как это было раньше, перестали быть статическими и храниться централизованно. В настоящее время для данных, собранных из разных источников нужны передовые аналитические инструменты для их передачи, обработки и хранения [1].

Одним из таких решений как раз выступает промежуточное программное обеспечение, ориентированное на обработку сообщений – брокеры сообщений. Они позволяют по определенному протоколу обмениваться информацией между приложениями или отдельными модулями в режиме реального времени. В сообщении может находиться абсолютно любая информация, будь то банковская транзакция или же целый словарь [2].

В конвейере существуют различные модули, которые обмениваются между собой данными. В каждый модуль изначально поступают данные, затем они обрабатываются и передаются в следующий модуль. Для этого реализуются ETL – процессы, которыми можно управлять с помощью брокера сообщений. Из ETL- процессов собирается ETL-конвейер.

Основная часть

Для реализации потоковой обработки текстовых данных необходимо проанализировать существующие программные решения брокеров сообщений.

Брокер сообщений – это архитектурный паттерн в распределенных системах. Брокер, преобразует сообщение по одному протоколу от приложения-источника в сообщение протокола приложения-приемника, а также выступает посредником между ними [3].

Были рассмотрены три различных брокера сообщений: RabbitMQ, ActiveMQ, Apache Kafka. Apache Kafka, в отличие от RabbitMQ и ActiveMQ не удаляет сообщения после прочтения. В Apache Kafka существует журнал логов, и каждый подписчик может получить полный набор этих логов. Данные факторы являются преимуществом Apache Kafka перед другими брокерами сообщений [4].

После рассмотрения вышеупомянутых брокеров сообщений был выбран брокер Apache Kafka. Данный брокер использует паттерн Producer/Consumer [4].

Определение терминов в Apache Kafka [4].:

Продюсер – поставщик данных, который генерирует сообщения;

Консьюмер – потребитель данных, который читает и использует события;

Сообщение – пакет данных, необходимый для совершения какой-либо операции;

Брокер – узел (сервер) передачи сообщения от процесса-продюсера приложению-потребителю;

Топик (тема) – виртуальное хранилище сообщений (журнал записей) одинакового или похожего содержания, из которого приложение-потребитель извлекает необходимую ему информацию.

Одной из важных особенностей Apache Kafka является ZooKeeper. ZooKeeper – это централизованный сервис, который работает как база для хранения метаданных о состоянии узлов кластера и расположении сообщений. Он обеспечивает гибкую и надежную синхронизацию в распределенной системе, позволяя нескольким клиентам выполнять одновременно чтение и запись.

Жизненный цикл сообщений в Apache Kafka (рисунок 1) [4]:

1. Продюсер отправляет сообщение на сервер.

2. Консьюмер извлекает сообщение и его уникальный идентификатор сервера.

3. Сервер помечает сообщение как in-flight (в полете). Сообщения в таком состоянии все еще хранятся на сервере, но временно не доставляются другим консьюмерам. Таймаут этого состояния контролируется специальной настройкой.

4. Консьюмер обрабатывает сообщение. Затем отправляют ack или nack-запросы обратно на сервер, как показано на рисунке 1.

5. В случае успеха сообщение удаляется с сервера навсегда. В случае ошибки или таймаута состояния in-flight сообщение доставляется консьюмеру для повторной обработки [4].

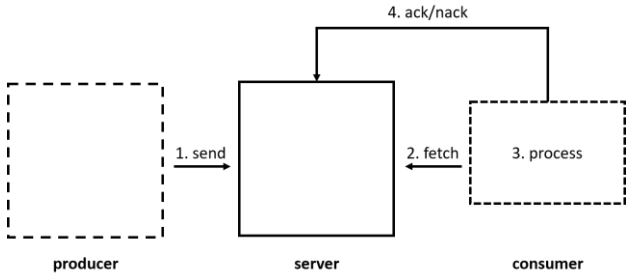

Рис. 1. Жизненный цикл сообщений в Apache Kafka

Данные обрабатываются в потоковом режиме. Поток данных – это упорядоченная последовательность данных, которой соответствует определенный источник или получатель. Поток данных проходит через разные модули и в каждом из таких модулей присутствует ETL-процесс – один из основных процессов в управлении хранилищами данных, который включает в себя: извлечение данных из внешних источников; их трансформацию и очистку, для того, чтобы они соответствовали заданным условиям; и загрузку их в последующий модуль или в хранилище данных. С точки зрения процесса ETL, архитектуру хранилища данных можно представить в виде трех компонентов [3]:

1) источник данных: содержит данные в виде таблиц, совокупности таблиц или просто файла;
2) промежуточная область: содержит вспомогательные таблицы, создаваемые временно и исключительно для организации процесса выгрузки;
3) получатель данных: хранилище данных или база данных, в которую должны быть помещены извлеченные данные.

Ряд ETL–процессов образует ETL – конвейер, который на каждом из шагов этого конвейера извлекает данные, обрабатывает, загружает в последующий модуль или помещает их в конечное хранилище [3].

Преимуществами ETL – конвейера являются управляемость: каждым из ETL – процессов можно управлять с помощью Apache Kafka, который будет загружать и извлекать данные для каждого процесса; а также возможность использовать распараллеливание процесса обработки данных, схематическое представление представлено на рисунке 2.

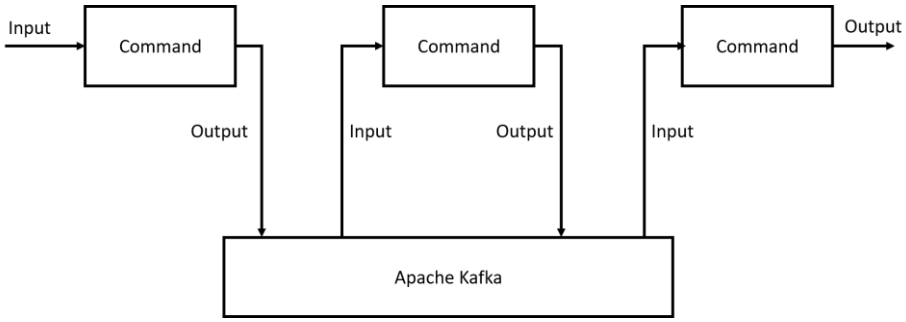

Рис. 2. Схематичное представление ETL – конвейера с использованием Apache Kafka

Для разработки модулей потоковой обработки тестовых данных был выбран язык программирования Python. Для него создано много различных библиотек по обработке текста. С его помощью возможно взаимодействие со следующими продуктами: Apache Kafka, MongoDB.

В конвейере используются три основных модуля для обработки текстовых данных. Загрузка в модуль и выгрузка из него осуществляется при помощи модуля os и методов stdin и stdout. Конвейер запускается при помощи скрипта bash. Схематичное представление ETL - конвейера для потоковой обработки текстовых данных под Apache Kafka представлено на рисунке 3.

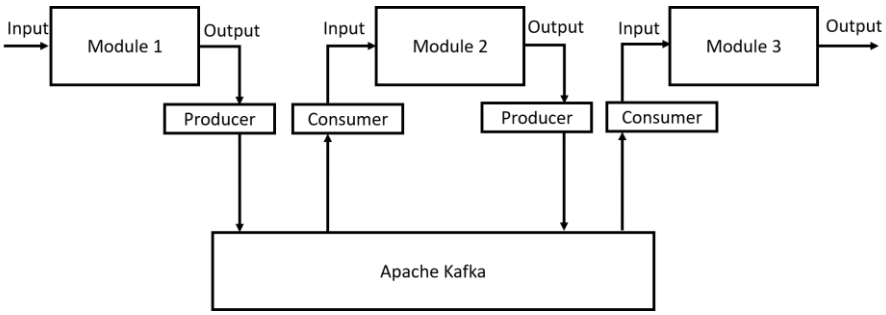

Рис. 3. Схематичное представление ETL – конвейера для потоковой обработки текстовых данных с использованием Apache Kafka

В первый модуль подается корпус из 100 текстов. Из текста при помощи регулярных выражений удаляются ссылки, знаки препинания, иные лишние знаки, пробелы и табуляция. Все слова в тексте преобразуются в нижний регистр. Далее эти данные поступают на вход продюсера, который отправляет их на сервер.

Во второй модуль данные поступают на вход при помощи консьюмера Apache Kafka. При помощи библиотеки nltk текст разбивается на лексемы, из которого удаляются стоп-слова (или шумовые слова) – например, такие слова, как: предлоги, союзы, частицы и т.д. Выходные данные из второго модуля – это список лексем, они поступают на вход продюсера, который отправляет их на сервер.

В третий модуль, аналогично со вторым модулем, данные поступают при помощи консьюмера Apache Kafka. При помощи библиотеки pymorphy2 выполняется нормализация – приведение всех лексем в словарную форму.

Для того, чтобы избежать повторяемости данных, поступающих на вход второго и третьего модулей, принято решение разнести выходные данные, отправляемые продюсером с первого и второго модулей по разным топикам.

Выходные данные из третьего модуля поступают на вход продюсера Apache Kafka, откуда публикуются в специальный топик и считываются консьюмером. Консьюмер записывает подготовленные данные в нереляционную базу данных MongoDB.

Заключение

В результате проделанной работы реализована потоковая обработка текстовых данных, при помощи ETL - конвейера под управлением распределенного брокера сообщений Apache Kafka. На вход ETL - конвейера подается корпуса документов. На выходе получаются нормализованные слова, которые загружаются в нереляционную базу данных MongoDB. В дальнейшей работе планируется продолжить разработку ETL-конвейера для потоковой обработки текстовых данных, под управлением Apache Kafka.

Список использованных источников

1. Knowledge Graphs and Big Data Processing// LNCS, volume 12072, July 2020. 209p. [Электронный ресурс]. – URL: https://link.springer.com/chapter/10.1007/978-3-030-53199-7_1 (дата обращения 20.01.2023).
2. Кайда А.Ю. Магистерская диссертация: Разработка протокола. передачи данных для системы управления потоками данных. – ТПУ, 2019. – 108с. [Электронный ресурс]. – URL: https://earchive.tpu.ru/handle/11683/53908 (дата обращения 21.01.2023).
3. Big Data Ecosystem. [Электронный ресурс]. – URL: https://www.sciencedirect.com/topics/computer-science/big-data ecosystem#:~:text=Big%20data%20ecosystem%20is%20the,potentials%20of%20big%20data%20analytics (дата обращения 20.01.2023).
4. Narkhede N. Kafka: The Definitive Guide/ N. Narkhede, G. Shapira, T. Palino. – USA: O'Reilly Media, Inc., 2017. – P. 12-16.