**1.** Добрый день, Марина Юрьевна, Азат Шавкатович, коллеги!

Представляю вашему вниманию свой итоговый проект по программе «Дата-инженер».

**2.** Целью проекта является создание минимально жизнеспособного продукта, который собирает, обрабатывает и анализирует данные по использованию услуги интерактивного телевидения.

**3, 4.** Непосредственно поставленные задачи взяты из предлагаемого задания № 1. Предлагаю отдельно на них не останавливаться, а каждую рассмотреть непосредственно с решением.

Рассмотрим их на широко известной в узких кругах схеме:

* сбор данных с использованием распределенного программного брокера сообщений Apache Kafka, являющегося компонентом продукта RT.Streaming;
* хранение собранных данных в распределенной файловой системе HDFS, являющейся элементом корпоративного хранилища данных RT.DataLake;
* обработка и агрегация данных с использованием СУБД Apache Hive, так же являющейся компонентом корпоративного хранилища данных RT.DataLake;
* перенос данных в массивно-параллельную СУБД GreenPlum (компонент RT.Warehouse);
* аналитика с помощью фреймворка Apache Spark, так же являющимся элементом корпоративного хранилища данных RT.DataLake;
* визуализация и построение дашборда при помощи платформы Apache Superset, являющейся компонентом продукта RT.Datavision.

**5.** Схема решения поставленных задач многим может показаться знакомой:

* так как датасет для проекта никто не предоставил, необходимо было его сгенерировать;
* сбор данных реализован на языке Python. Для этого был создан продюсер, который симулирует данные о поведении пользователей интерактивного телевидения и отправляет их в топик Kafka;
* хранение сырых данных осуществляется с помощью потребителя на Python, который читает данные с Kafka и сохраняет их в HDFS в формате CSV;
* обработка сырых данных произведена в СУБД Hive, так же в ней сделаны некоторые запросы для агрегации данных;
* перенос данных из Hive в Greenplum осуществлен с помощью ETL процесса – ориентированного ациклического графа Airflow DAG;
* данные для аналитики в Spark взяты из обработанных данных СУБД Hive;
* для построения Superset-дашборда, в Greenplum было создано материализованное представление.

**6.** Так как датасет предоставлен не был, пришлось его генерировать самостоятельно. Для этого я нагуглил рейтинг просмотров одного кабельного канала, произносить не буду, а так же немного почитал исследования от Тинькова. В конце презентации на них есть ссылки.

Рейтинг был копипастой преобразован в CSV файл, а на основании его с помощью Python сгенерирован датасет на все 148 каналов и 10000 пользователей на 7 дней. Датасет был разделен на три таблицы в третьей нормально форме. Чтобы генерация датасета не превратилась в еще один итоговый проект, сделано допущение, что один пользователь в течении дня смотрел только один канал. Основная таблица включает в себя следующие колонки:

* идентификатор сессии;
* идентификатор пользователя;
* идентификатор канала;
* время начала и окончания сессии;
* статус сессии.

**7.** На следующем слайде выведен код генерации датасета. Коллеги, на слайд простыню с кодом уместить сложно, если есть вопросы, могу открыть сам файл с большим масштабом.

**8.** Итак, первым пунктом нашего задания является сбор данных с использованием распределенного программного брокера сообщений Apache Kafka, являющегося компонентом продукта RT.Streaming. Сбор данных реализован на языке Python. Для этого был создан продюсер, который симулирует данные о поведении 10000 пользователей интерактивного телевидения и отправляет их в топик Kafka.

На картинке представлены сообщения топика, каждое из которых – это действия нашего пользователя.

В итоге мой датасет представляет собой изначально сгенерированный файл за 7 дней и перехваченные во время стимуляции данные за 8 день.

**9.** На следующем слайде представлен код продюсера.

**10.** Следующим пунктом задания является реализация хранения собранных данных в распределенной файловой системе HDFS, являющейся элементом корпоративного хранилища данных RT.DataLake. Хранение осуществляется с помощью потребителя на Python, который читает данные с Kafka и сохраняет их в HDFS в формате CSV.

Здесь я отошел от предлагаемого задания и не использовал формат хранения parquet или AVRO, в моем случае конкретно AVRO из-за необходимости в кратчайшие сроки реализовать решение, т.к. у меня сейчас происходит сезонный пик работы. Ну а далее из-за нестабильности работы кластера под возросшей нагрузкой я не стал ломать то, что и так работает.

На слайде видно, что 3 файла сгенерированного датасета я загрузил в Hadoop напрямую, а файл tv-stream – это результат работы стриминга, запущенного под пользователем olejnikov.

**11.** На следующем слайде представлен код консумера.

**12.** Следующим пунктом задания является реализация обработки и агрегации данных с использованием СУБД Apache Hive, являющейся компонентом корпоративного хранилища данных RT.DataLake.

Для этого расположенные в HDFS сырые данные, полученные в результате генерации и стриминга я загрузил и преобразовывал в таблицы Hive.

**13.** Для агрегации нескольких таблиц в одну я использовал виртуальную таблицу. В ней уже нет колонки со статусом начала и конца сессии, а непосредственно указаны ее время начала и конца. А так же вместо id канала стоит его название. Код и виртуальную таблицу вы можете видеть на слайде.

**14.** Так же в Hive можно сделать аналитические запросы. На слайде вы видите код и результаты запросов рейтинга каналов по количеству и общей длительности просмотра, а так же количество и длительность просмотров в разрезе по дням.

**15.** Следующим пунктом задания является реализация переноса обработанных данных из Hive в массивно-параллельную СУБД GreenPlum (компонент RT.Warehouse) при помощи ETL процесса – ориентированного ациклического графа Airflow DAG.

В графе я использовал три задачи – отдельно для каждой таблицы датасета.

В Greenplum SQL-диалект свежее и можно построить более сложный запрос – например распределение количества просмотров в разрезе каждого часа.

**16.** На следующем слайде представлен код графа.

**17.** Следующим пунктом задания является реализация аналитики с помощью фреймворка Apache Spark, так же являющимся элементом корпоративного хранилища данных RT.DataLake. Данные для аналитики в Spark взяты из обработанных данных СУБД Hive.

С помощью SQL-запросов и Python-библиотек в Spark были построены аналитические графики:

* распределение по количеству и времени просмотра телеканалов по дням;

**18.**

* диаграмма просматриваемых каналов. На ней мы можем видеть, что общее время просмотра 6 каналов из первого и второго мультиплекса составляет столько же, сколько и остальные 142 канала;
* рейтинг наименее просматриваемых каналов, полученный как отношение времени просмотра к количеству пользователей, которые этот канал смотрели. Такой подход считаю более объективный т.к. есть каналы с малым количеством зрителей, но которые смотрели данный канал длительное время.

**19, 20.** И последний пункт задания - визуализация и построение дашборда при помощи платформы Apache Superset, являющейся компонентом продукта RT.Datavision. Для построения Superset-дашборда, в Greenplum было создано материализованное представление. Код я приводить не стал, т.к. на 90% оно совпадает с виртуальной таблицей в Hive. Разве что было добавлено случайное распределение по сегментам.

**21.** Итоги и инсайты. Инсайты мы видели ранее на слайдах. Как сказал в свое время генералиссимус Суворов: «Теория без практики мертва, практика без теории слепа». Считаю, что данное выражение на все 200% реализовано во всех направлениях IT-отрасли. В итоговом проекте я пощупал не отдельные друг от друга домашние задания, а реализовал взаимосвязь доброй половины пройденных за курс технологий и продуктов. Многое стало понятно. Например – в озере данных не стоит использовать 3НФ, т.к. в дальнейшем это приведет к очень тяжелым и ресурсоемким запросам.

Питоньячий код у меня заточен на конечные и повторяемые действия, но изменения буквально в паре строк сделают их масштабируемым для работы с большими объемами. Используемые технологии позволяют сохранять данные не в CSV, а в AVRO формате, что тоже реализуется изменением буквально одной строки кода.

Считаю работу достойным завершением курса.

**22.** На следующем слайде указаны ссылки на репозиторий проекта, рейтинги и статистика, на основании которых генерировался датасет. Ссылки на визуализацию, но доступны они только внутри развернутого кластера. И то, ради чего все это затевалось – ссылка платформа управления данными ПАО «Ростелеком».

**23.** Уважаемые коллеги! Доклад закончил. Спасибо за внимание! Готов ответить на ваши вопросы.