Spark Developer

Spark Developer

Длительность курса: 114 академических часов

1 Scala

1 Архитектура приложения Spark

Цели занятия:

познакомиться с API для работы с приложениями Спарк: spark-shell, spark-submit, Idea, jupiter;
познакомиться с режимами работы Спарк-приложения: local, client, cluster;
познакомиться с менеджерами ресурсов для Спарка;
познакомиться с основными компонентами архитектура Спарка приложения;
познакомиться с возможностями управления.

Краткое содержание:

local;
cleint;
cluster modes;
Yarn, Kubernetus resource manager;
Spark-shell;
spark-submit.

---

2 Основы Scala

Цели занятия:

научиться читать код Scala;
научиться писать простые приложения на языке Scala.

Краткое содержание:

Scala.

---

3 **Сборка проектов на Scala**

Цели занятия:

научиться собирать готовые проекты для использования приложения на Scala;
научиться использовать библиотеки в вашем проекте.

Краткое содержание:

Scala;
SBT.

Домашние задания

1 Первое приложение на Scala для работы с файлами в формате JSON

Цель: Данное задание позволит вам создать свой первый Scala-проект и собрать его с помощью SBT, а также познакомиться с базовыми инструментами обработки данных нативными средствами языка Scala.

1. Загрузите файл с географическими данными различных стран (https://raw.githubusercontent.com/mledoze/countries/master/countries.json)

2. Среди стран выберите 10 стран Африки с наибольшей площадью.

3. Запишите данные о выбранных странах в виде JSON-массива объектов следующей структуры:
[{
“name”: <Официальное название страны на английском языке, строка>,
“capital”: <Название столицы, строка>(если столиц перечислено несколько, выберите первую),
“area”: <Площадь страны в квадратных километрах, число>,
}]

4. Обеспечьте проект инструкциями для сборки JAR-файла, принимающего на вход имя выходного файла и осуществляющего запись в него.

4 **Сборка проектов на Scala. Практика**

Цели занятия:

понять зачем и с помощью чего собирать проект;
научиться видеть проблемы при сборке в больших данных;
разобрать пример Scala-проекта на Sbt.

Краткое содержание:

сборка Scala-проекта;
Sbt.

2 Hadoop

1 Hadoop

Цели занятия:

понять назначение Hadoop и его роль в системах обработки данных;
рассмотреть основные дистрибутивы Hadoop.

Краткое содержание:

Hadoop;
дистрибутивы: CDH/HDP/CDP.

---

2 HDFS

Цели занятия:

научиться использовать Hadoop для хранения данных;
научиться описывать архитектуру и роль Zookeeper в распределенных системах.

Краткое содержание:

до занятия необходимо установить докер.

HDFS;
Zookeeper.

Домашние задания

1 Работа с файлами на HDFS: запись, считывание и управление файлами

Цель: В этом ДЗ вы сможете на практике поработать с файлами на HDFS, научитесь записывать и считывать файлы, а также управлять ими.

Задача описана в репозитории: https://github.com/Gorini4/hadoop_course_homework/tree/master/hw1

3	**YARN**	**Цели занятия:** понять назначение систем контейнеризации; научиться использовать парадигму MapReduce для решения задач обработки данных. **Краткое содержание:** YARN; MapReduce.
4	**Форматы данных**	**Цели занятия:** рассмотреть разницу между основными форматами хранения данных в Hadoop; научиться выбирать подходящие форматы для каждого из типов задач; научиться настраивать сжатие при записи данных. **Краткое содержание:** JSON, CSV, Avro, Parquet, Orc, SequenceFile; HadoopInputFormat/HadoopOutputFormat.
5	**Q&A**	**Цели занятия:** получить ответы на вопросы по ДЗ; получить ответы на вопросы по приложениям. **Краткое содержание:** типичные ошибки при выполнении ДЗ; наставники ответят на ваши вопросы.

3 Spark

1 RDD/Dataframe/Dataset

Цели занятия:

познакомиться с Data API Spark: RDD, Dataframe, Dataset для манипуляции с данными;
познакомиться со способами описания логики обработки: Scala Lambda, Spark DSL, Spark SQL;
разобрать плюсы и минусы Data API и способов описания логики.

Краткое содержание:

RDD;
Dataframe;
Dataset;
Scala Lambda;
Spark DSL;
Spark SQL;
Catalyst Optimiser.

Домашние задания

1. Аналитическая витрина на основе сырых данных, используя Spark

   Цель: Выполнив домашнее задание вы получите опыт работы с RDD API, DataFrame API, Dataset API. Научитесь строить аналитическую витрину на основе сырых данных, используя Spark и различные API.

   Предварительная инструкция для ДЗ:

   1. Скачать и установить Idea Community - https://www.jetbrains.com/idea/download/#section=windows
   2. Установить плагин Скала
   3. Скачать и установить Java JDK 11 - https://www.oracle.com/java/technologies/javase-jdk11-downloads.html
   4. Скачать и установить git - https://git-scm.com/downloads
   5. Скачать и установить локально дистрибутив Hadoop (инструкция для

Windows - https://www.datasciencecentral.com/profiles/blogs/how-to-install-and-run-hadoop-on-windows-for-beginners)
6. Скачать стартовый проект с Гитхаб с помощью команды git clone. Ссылка на проект: https://github.com/vadopolski/otus-hadoop-homework
7. Запустить Idea и открыть скаченный проект File -> Open -> project folder/build.sbt
8. Открыть в проекте файл src/main/scala/homework2/DataApiHomeWorkTaxi.scala и запустить его, Ctrl + Shift10
9. Скачать и установить docker-compose
10. Из корневой папки проекта запустить сделать запуск - docker-compose up

Основная инструкция по ДЗ и задания к занятию по Spark Data API:

Основная инструкция, задание 1:

Загрузить данные в первый DataFrame из файла с фактическими данными поездок в Parquet (src/main/resources/data/yellow_taxi_jan_25_2018). Загрузить данные во второй DataFrame из файла со справочными данными поездок в csv (src/main/resources/data/taxi_zones.csv) С помощью DSL построить таблицу, которая покажет какие районы самые популярные для заказов. Результат вывести на экран и записать в файл Паркет.

Результат: В консоли должны появиться данные с результирующей таблицей, в файловой системе должен появиться файл. Решение оформить в github gist.

Основная инструкция, задание 2:

Загрузить данные в RDD из файла с фактическими данными поездок в Parquet (src/main/resources/data/yellow_taxi_jan_25_2018). С помощью lambda построить таблицу, которая покажет В какое время происходит больше всего вызовов. Результат вывести на экран и в txt файл с пробелами.

Результат: В консоли должны появиться

данные с результирующей таблицей, в файловой системе должен появиться файл. Решение оформить в github gist.

Основная инструкция, задание 3:

Загрузить данные в DataSet из файла с фактическими данными поездок в Parquet (src/main/resources/data/yellow_taxi_jan_25_2018). С помощью DSL и lambda построить таблицу, которая покажет. Как происходит распределение поездок по дистанции? Результат вывести на экран и записать в бд Постгрес (докер в проекте). Для записи в базу данных необходимо продумать и также приложить инит sql файл со структурой.

(Пример: можно построить витрину со следующими колонками: общее количество поездок, среднее расстояние, среднеквадратическое отклонение, минимальное и максимальное расстояние)

Результат: В консоли должны появиться данные с результирующей таблицей, в бд должна появиться таблица. Решение оформить в github gist.

---

2 **Методы оптимизации приложений Spark**

Цели занятия:

рассмотреть проблемы с перфомансом в приложениям Spark;
познакомиться с моделью памяти приложений Spark;
познакомиться с оптимизацией медленных джойнов, Blume Filter;
познакомиться с Spark 3.0: Adaptive Query Execution, Dynamic Partition Pruning.

Краткое содержание:

Adaptive Query Execution;
Dynamic Partition Pruning;
Blume Filter;
Out of Memory Exception.

---

3 **Написание коннекторов для Spark**

Цели занятия:

познакомиться со стандартным Spark API для external storage: RDBMS, Kafka, MPP;
узнать как взаимодействовать с external storage в параллельном режиме;
рассмотреть, зачем и как писать свой собственный коннектор к нестандартному хранилищу для работы.

Краткое содержание:

RDBMS;
Kafka;
MPP;
Data Source v 2.0.

Домашние задания

1 Разработка собственного коннектора на Spark

Цель: В данном ДЗ вы поработаете с DataSource API V2, научитесь писать свои собственные коннекторы для Spark.
Задача - доработать data source для Postgres для партиционированного чтения.

1. Склонируйте репозиторий https://github.com/Gorini4/spark_datasource_example
2. Доработайте код в файле src/main/scala/org/example/datasource/postgres/PostgresDatasource.scala так, чтобы тест в файле src/test/scala/org/example/PostgresqlSpec.scala при выполнении читал таблицу users не в одну партицию, а в несколько (размер одной партиции должен задаваться через метод .option("partitionSize", "10")).

Рекомендация: Архитектуру решения можно обсудить в общей группе в Slack.

---

4 **Тестирование**

Цели занятия:

приложений Spark

узнать как организовать тестирование на проекте больших данных с Apache Spark;
познакомиться с библиотеками для юнит тестирования Apache Spark, интеграционных тестов и генерации тестовых данных.

Краткое содержание:

Apache Spark;
Test Containers;
Scalacheck.

Домашние задания

1 Тесты для приложения Spark

Цель: Выполнив это домашнее задание, вы научитесь писать авто-тесты для Spark jobs.

Предварительная инструкция для ДЗ:

1. Скачать и установить Idea Community - https://www.jetbrains.com/idea/download/#section=windows
2. Установить плагин Скала
3. Скачать и установить Java JDK 11 - https://www.oracle.com/java/technologies/javase-jdk11-downloads.html
4. Скачать и установить git - https://git-scm.com/downloads
5. Скачать и установить локально дистрибутив Hadoop (инструкция для Windows - https://www.datasciencecentral.com/profiles/blogs/how-to-install-and-run-hadoop-on-windows-for-beginners)
6. Скачать стартовый проект с Гитхаб с помощью команды git clone. Ссылка на проект: https://github.com/vadopolski/otus-hadoop-homework
7. Запустить Idea и открыть скаченный проект File -> Open -> project folder/build.sbt
8. Открыть в проекте файл src/main/scala/homework2/DataApiHomeWorkTaxi.scala запустить его, Ctrl + Shift10
9. Скачать и установить docker-compose
10. Из корневой папки проекта запустить сделать запуск - docker-compose up

Основная инструкция по выполнению ДЗи задания к занятию по Spark Testing:

!!!!! ВНИМАНИЕ, для корректной работы необходимо сделать reimport dependency

Основная инструкция, задание 1:

Логически разбить на методы, написанный в домашнем задании к занятию Spark Data API. Пример src/main/scala/lesson2/OtusFragmentedByMethod.scala

Результат: В репозитории должен появиться код с описанием методов. Решение оформить в github.

Основная инструкция, задание 2:

Сформировать ожидаемый результат и покрыть простым тестом (с библиотекой AnyFlatSpec) витрину из домашнего задания к занятию Spark Data API, построенную с помощью RDD. Пример src/test/scala/lesson2/SimpleUnitTest.scala

Результат: В репозитории должен появиться код с тестом. Тест должен успешно выполняться при запуске. Решение оформить в github.

Основная инструкция, задание 3:

Сформировать ожидаемый результат и покрыть Spark тестом (с библиотекой SharedSparkSession) витрину из домашнего задания к занятию Spark Data API, построенную с помощью DF и DS. Пример src/test/scala/lesson2/TestSharedSparkSession.scala

Результат: В репозитории должен появиться код с тестом. Тест должен успешно выполняться при запуске. Решение оформить в github.

5 **Spark ML**

Цели занятия:

научиться обучать и применять обученные модели на больших объемах данных с помощью Spark.

Краткое содержание:

Spark;
Scala;
XGBoost.

4 Streaming

1 Kafka

Цели занятия:

научиться описывать архитектуру Apache Kafka;
научиться использовать консольные утилиты и клиенты для работы с Kafka;
научиться оптимизировать запись и чтение топиков Kafka.

Краткое содержание:

Kafka.

Домашние задания

1 Приложение для чтения данных из Kafka

Цель: В этом ДЗ вы научитесь использовать Apache Kafka - управлять топиками и читать/писать данные с помощью Scala.

Перед началом выполнения требуется развернуть Kafka через docker-compose (https://github.com/Gorini4/kafka_scala_example) и создать топик books с 3 партициями.

Вам нужно написать приложение, которое будет выполнять следующее:

1. Вычитывать из CSV-файла, который можно скачать по ссылке - https://www.kaggle.com/sootersaalu/amazon-top-50-bestselling-books-2009-2019, данные, сериализовывать их в JSON, и записывать в топик books локально развернутого сервиса Apache Kafka.

2. Вычитать из топика books данные и распечатать в stdout последние 5 записей (с максимальным значением offset) из каждой партиции. При чтении топика одновременно можно хранить в памяти только 15 записей.

2 **Spark Streaming**

Цели занятия:

научиться писать приложения для потоковой обработки данных на Spark;
научиться использовать DStreams и Structured Streaming.

Краткое содержание:

Dataflow model;
Spark Streaming;
интеграция с Kafka;
интеграция со Spark ML.

Домашние задания

1 Настройка применения предобученной модели в Spark Streaming

Цель: Выполнив это ДЗ, вы научимся обучать и сохранять модели Spark ML. А также использовать предобученные модели Spark ML в Spark Streaming.

1. Построить модель классификации Ирисов Фишера и сохранить её.
- Описание набора данных: https://ru.wikipedia.org/wiki/Ирисы_Фишера
- Набор данных в формате CSV: https://www.kaggle.com/arshid/iris-flower-dataset
- Набор данных в формате LIBSVM: https://github.com/apache/spark/blob/master/data/mllib/iris_libsvm.txt
- Должен быть предоставлен код построения модели (ноутбук или программа)

2. Разработать приложение, которое читает из одной темы Kafka (например, "input") CSV-записи с четырьмя признаками ирисов, и возвращает в другую тему (например, "predictition") CSV-записи с теми же признаками и классом ириса.
- Должен быть предоставлен код программы.

3 **Structured Streaming**

Цели занятия:

научиться писать приложения на Structured Streaming, используя state и агрегаты.

Краткое содержание:

Dataflow model;
Structured Streaming;
интеграция с Kafka;
интеграция со Spark ML.

Домашние задания

1 Настройка применения предобученной модели в Spark Structured Streaming

Цель: Выполнив это ДЗ вы научитесь обучать и сохранять модели Spark ML. А также использовать предобученные модели Spark ML в Spark Structured Streaming

1) Построить модель классификации Ирисов Фишера и сохранить её.

Описание набора данных:
https://ru.wikipedia.org/wiki/Ирисы_Фишера
Набор данных в формате CSV:
https://www.kaggle.com/arshid/iris-flower-dataset
Набор данных в формате LIBSVM:
https://github.com/apache/spark/blob/master/data/mllib/iris_libsvm.txt

Должен быть предоставлен код построения модели (ноутбук или программа)

2) Разработать приложение, которое читает из одной темы Kafka (например, "input") CSV-записи с четырми признаками ирисов, и возвращает в другую тему (например, "predictition") CSV-записи с теми же признаками и классом ириса.

Должен быть предоставлен код программы.

4 **Flink - часть 1**

Цели занятия:

научиться выбирать подходящий фреймворк для задачи обработки потоковых данных;
научиться использовать базовые операции из API Flink.

Краткое содержание:

Scala;
Flink.

---

5 **Flink - часть 2**

Цели занятия:

научиться писать приложения потоковой обработки данных, используя Flink.

Краткое содержание:

Scala;
Flink.

---

6 **Q&A**

Цели занятия:

получить ответы на вопросы по ДЗ;
получить ответы на вопросы по приложениям.

Краткое содержание:

типичные ошибки при выполнении ДЗ;
наставники ответят на ваши вопросы.

5 Apache Hive

1 Обзор Hive

Цели занятия:

научиться описывать архитектуру Hive;
рассмотреть назначение Hive и аналогичных инструментов.

Краткое содержание:

Hive;
Impala;
Presto.

2 HiveQL

Цели занятия:

научиться писать запросы на HiveQL;
научиться создавать таблицы в Hive.

Краткое содержание:

форматы файлов;
типы данных;
DDL;
DML;
запросы к данным.

Домашние задания

1 Построение аналитической витрины в Hive

Цель: В это ДЗ вы напишите код для построения аналитической витрины в Hive.

1. Выбрать любой интересный открытый набор данных для работы (можно воспользоваться сайтом Kaggle, пример набора данных: https://www.kaggle.com/tylerx/flights-and-airports-data)
2. Создать database в Hive и загрузить туда эти таблицы
3. На этих данных построить витрины (5-6) с использованием конструкций: where, count, group by, having, order by, join, union, window.
4. К каждой созданной витрине добавить описание, что данная витрина демонстрирует (например: количество перелетов за год, топ самых загруженных аэропортов и т.п.).

6 Обслуживающие системы

1 **Оркестрация процессов обработки данных**

Цели занятия:

рассмотреть назначение оркестраторов в ETL-процессах;
научиться использовать Oozie для оркестрации.

Краткое содержание:

Oozie;
Airflow.

---

2 **Мониторинг и логирование для Spark-приложений**

Цели занятия:

научиться настраивать мониторинг и логирование для Spark-приложений.

Краткое содержание:

Spark;
Grafana.

Домашние задания

1 Настройка мониторинга для своего приложения

Цель: В этом ДЗ вы настроите мониторинг для своего приложения.

1. Настроить для приложения

---

3 **CI/CD для Spark и Hive**

Цели занятия:

научиться настраивать процессы сборки и деплоя для Spark и Hive.

Краткое содержание:

Spark;
SBT.

---

4 **Q&A**

Цели занятия:

получить ответы на вопросы по ДЗ;
получить ответы на вопросы по приложениям.

Краткое содержание:

типичные ошибки при выполнении ДЗ;
наставники ответят на ваши вопросы.

7 Проектная работа

1 Выбор темы и организация проектной работы

Цели занятия:

выбрать и обсудить тему проектной работы;
спланировать работу над проектом;
ознакомиться с регламентом работы над проектом.

Краткое содержание:

правила работы над проектом и специфика проведения итоговой защиты;
требования к результату проекта и итоговой документации.

Домашние задания

1 Проектная работа

Цель: В этом ДЗ необходимо выбрать тему проектной работы и закрепить её в чате по дз. Написать проект и защить.

1. Выбрать тему и утвердить
2. Сделать проект
3. Защита проект

---

2 Консультация по проектам и домашним заданиям

Цели занятия:

получить ответы на вопросы по проекту, ДЗ и по курсу.

Краткое содержание:

вопросы по улучшению и оптимизации работы над проектом;
затруднения при выполнении ДЗ;
вопросы по программе.

3	**Защита проектных работ**	**Цели занятия:** защитить проект и получить рекомендации экспертов. **Краткое содержание:** презентация проектов перед комиссией; вопросы и комментарии по проектам.