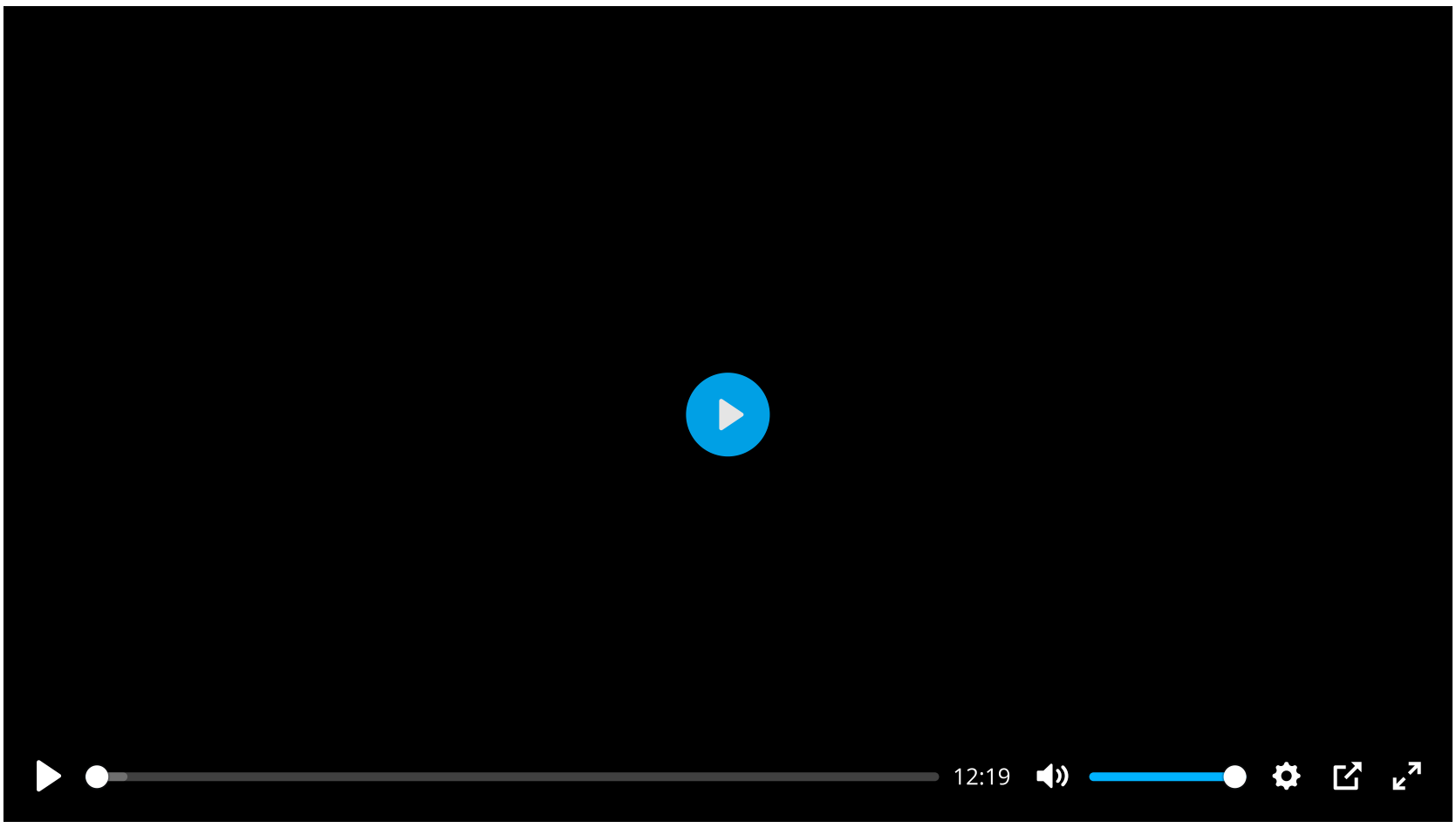
4.2 Что такое Spark и зачем он нужен DE. Введение в Resilient Distributed Datasets



Текстовая расшифровка видео:

НАЧАЛО РАБОТЫ

План:

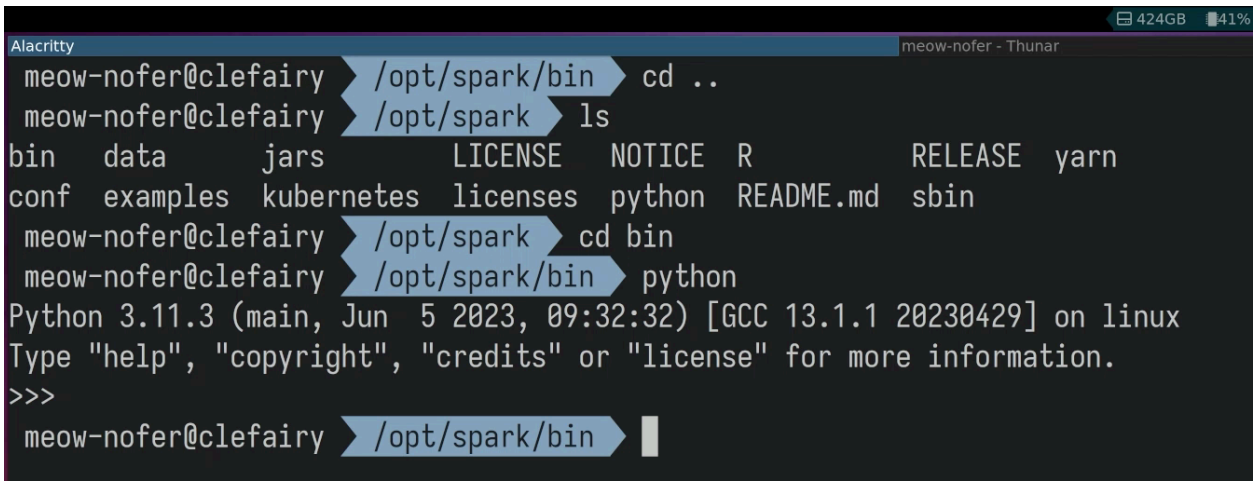
- Самодостаточная система;
- Запуск приложений;
- С точки зрения кода;
- Отличия SparkContext и SparkSession;
- Источники, приемники, форматы.

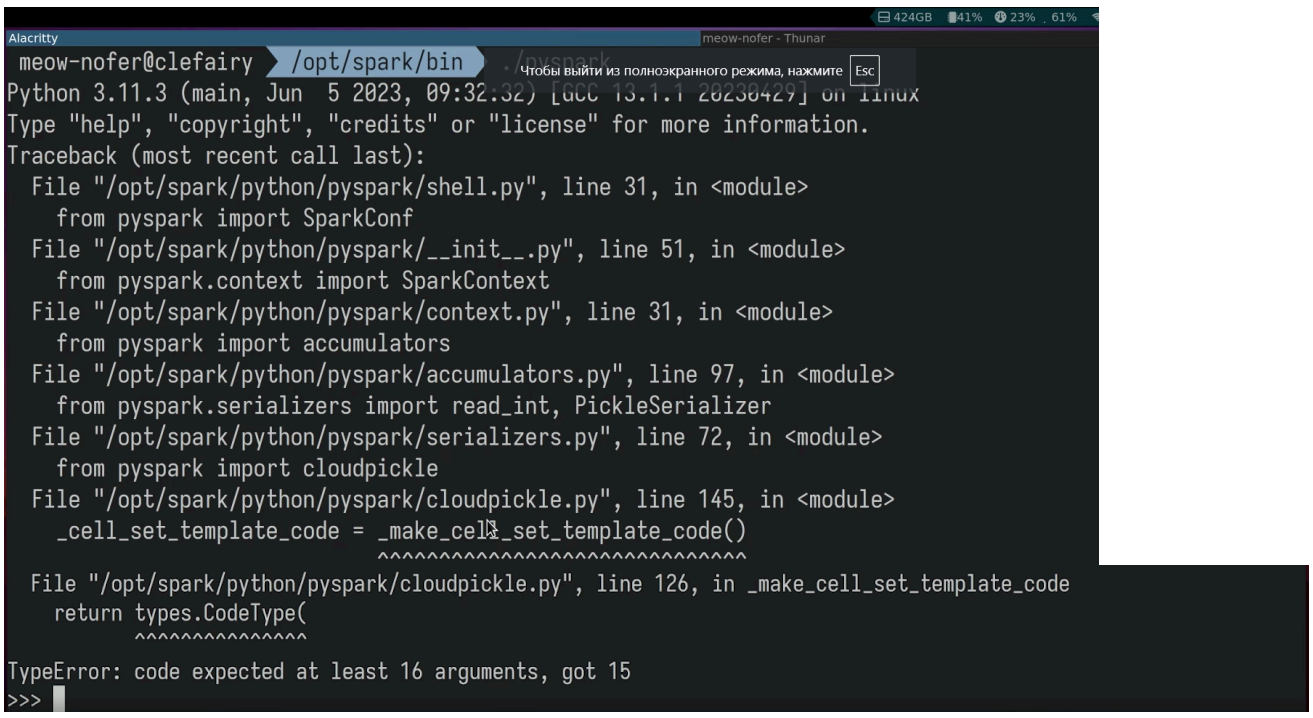
Самодостаточная система

Spark можно скачать с сайта (аналогично с Hadoop). Однако есть нюанс: если вы хотите писать под Spark на Python, то **скачанный с сайта Spark поддерживает только конкретные версии Python**.

Рассмотрим на примере.

Это скачанный Spark (версия 2.4.8), который мы распаковали. Python, который стоит – 3.11:





Подобный подход можно использовать, но важно подобрать подходящую версию Python.

Еще один вариант – **поставить клиент для Spark вместе с пакетом PySpark.** Он доступен из pip’a:

```
~$ pip install pyspark
```

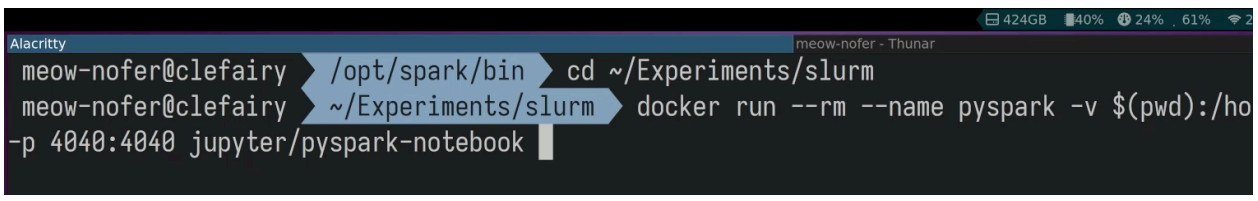
Для практики лучше воспользоваться самым простым вариантом, а именно взять подсобранный PySpark в контейнере с уже готовым Jupyter’ом:

```
~$ docker pull jupyter/pyspark-notebook
```

```
~$ docker run --name pyspark -p 8888:8888 -p 4040:4040 jupyter/pyspark-notebook
```

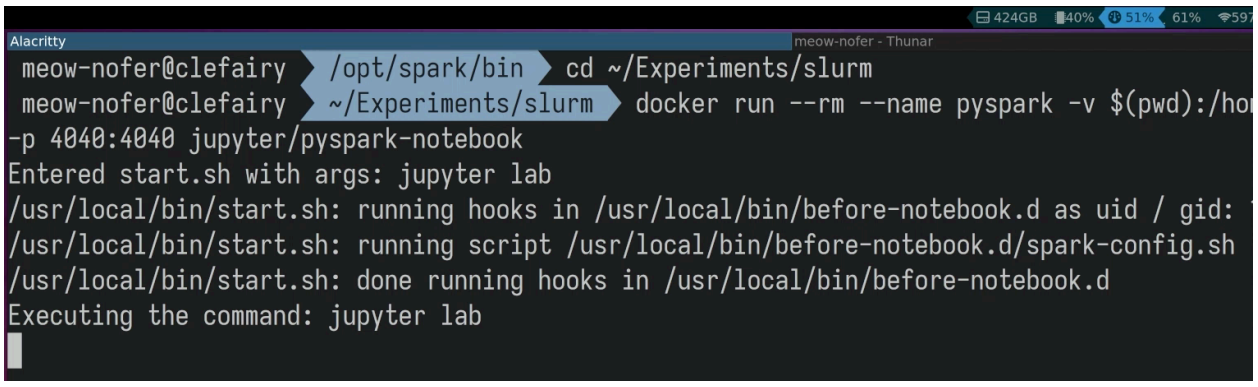
Нужно скачать соответствующий образ докера с Docker Hub и запустить его, прокинув порты. Так как у нас Jupyter – 8888. 4040 – отладочный UI Spark’a.

Перейдем в каталог и скажем «docker run»:



Добавим «-v» в качестве домашнего каталога пользователя внутри контейнера, так как в текущем каталоге есть дополнительные файлы.

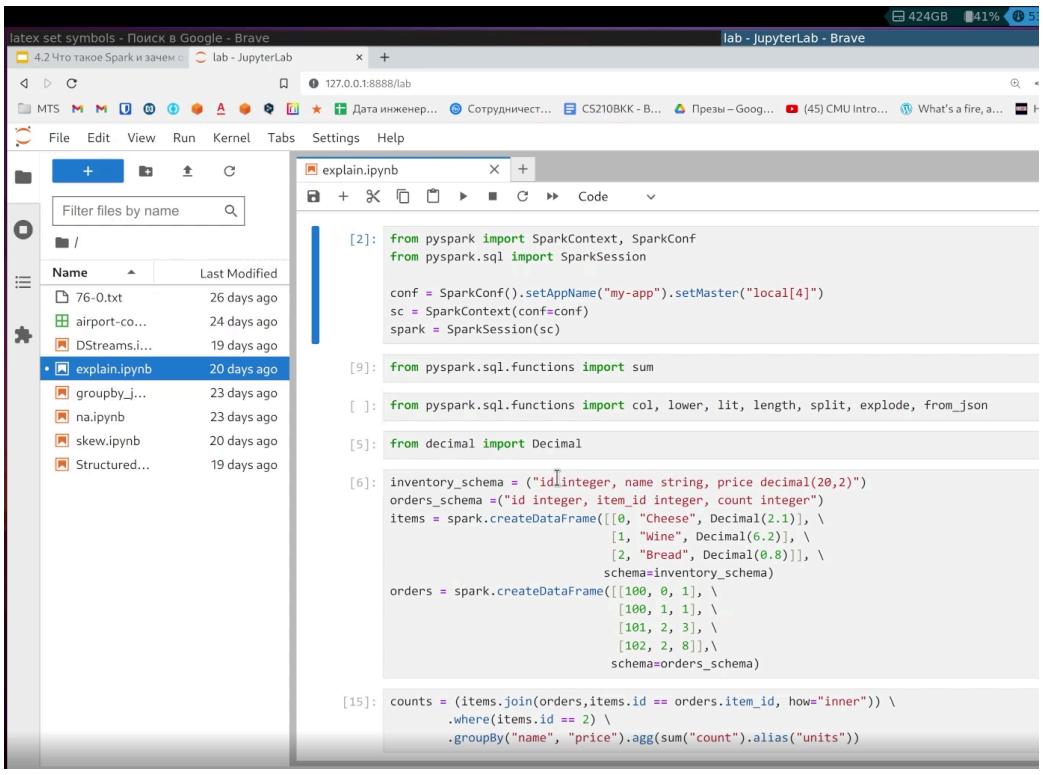
Запустили:



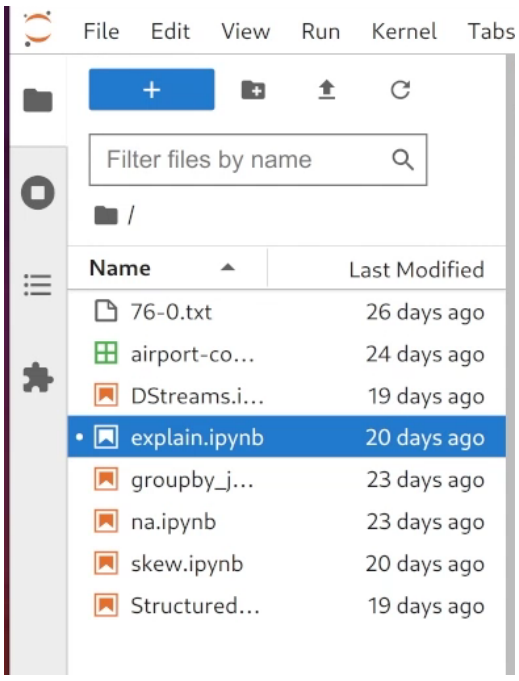
Если вы уже пробовали работать с Jupyter’ом, то знаете, что он «выплевывает» адрес, куда нужно подключаться с токеном:

Мы можем зайти в браузер, скопировать целиком URL; у нас откроется интерфейс Jupyter’a, где можно

Дедлайн 07 июля, 23:59 Мск



Обратите внимание: поскольку мы добавили локальный каталог, у нас появились несколько разных демонстрационных notebooks:



Запуск приложений

Запуск приложений на кластере Spark происходит несколькими способами:

- **Использовать shell**. Можно использовать Spark-shell, если используете язык программирования Scala, а PySpark, если пишете на Python.
- **Использовать отдельную утилиту – Spark-submit**. Мы пишем код, работающий с драйвером Spark'a. С помощью Spark-submit мы задаем дополнительный параметр того, как все сконфигурировать, и передаем код для запуска на кластере.

Дополнительно можно ознакомиться с UI Spark'a, перейдя по ссылке: http://127.0.0.1:4040/jobs/

Важно: чтобы ссылка открылась, и у Spark'a запустился UI нужно проинициализировать драйвер, создать контекст.

С точки зрения кода

Рассмотрим пример:

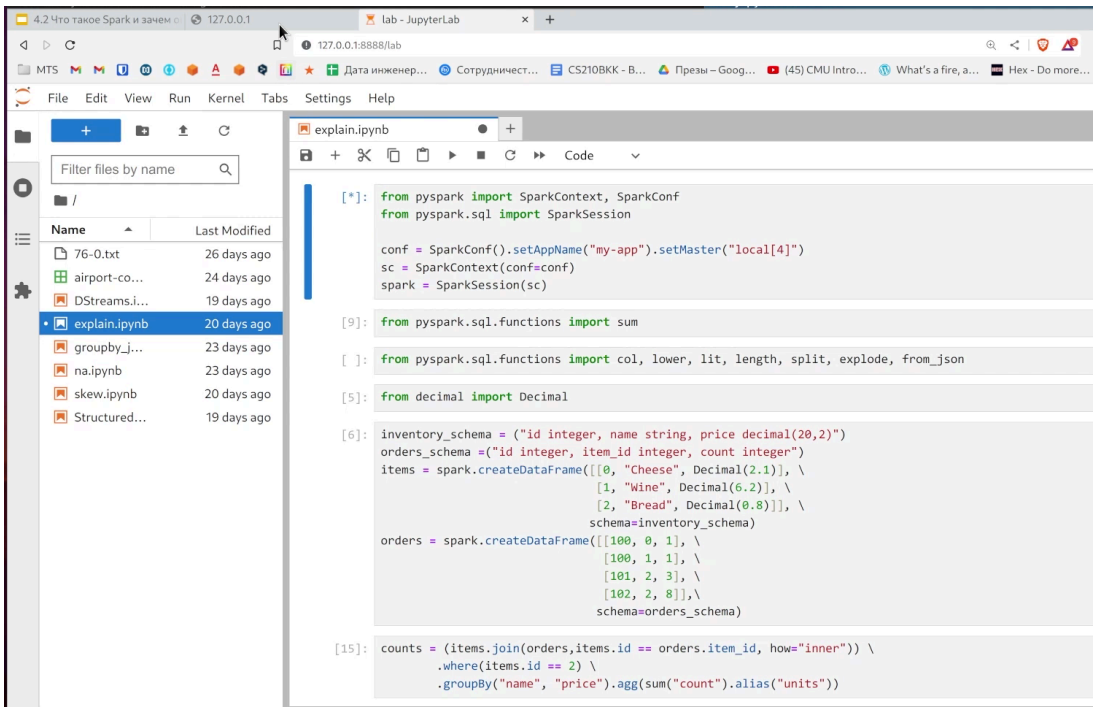
```
1. from pyspark import SparkContext, SparkConf
2.
3. conf = SparkConf().setAppName("my-app").setMaster("local[4]")
4. sc = SparkContext(conf=conf)
```

Мы импортируем SparkContext для получения доступа к кластеру (также можно взять SparkConf, чтобы сконфигурировать кластер так, как мы хотим).

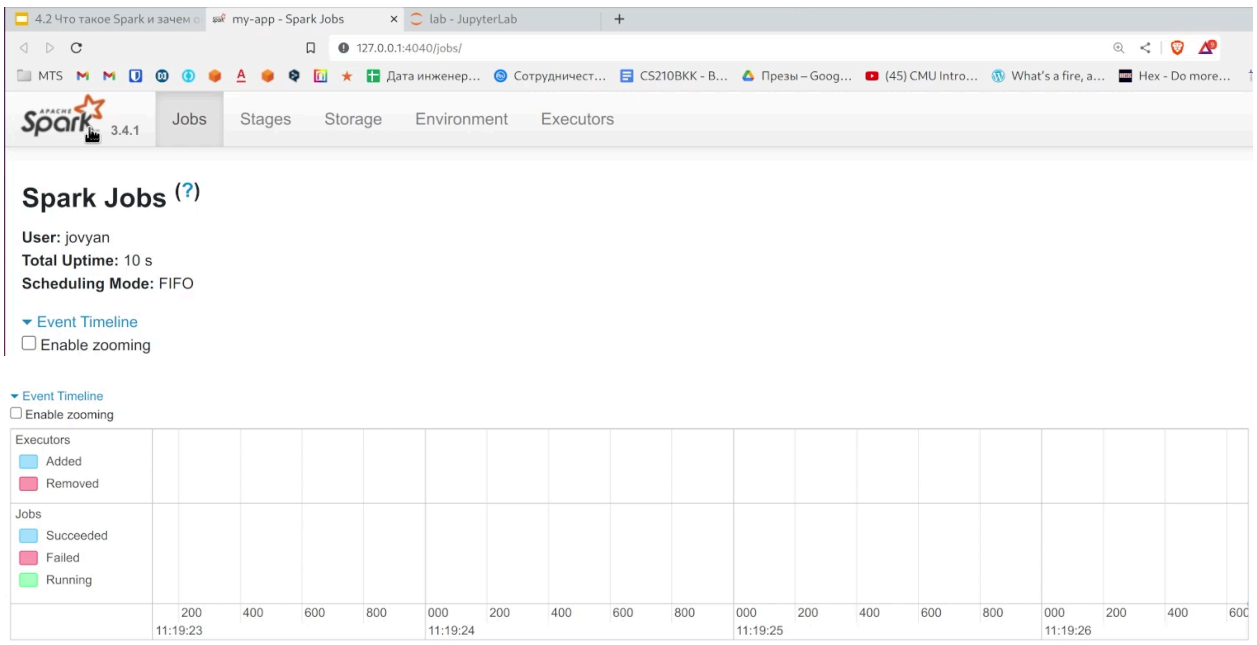
Дедлайн 07 июля, 23:59 Мск

В целом требования минималистичные: мы можем задать базовое имя для приложения, а также задать то, какую конфигурацию хотим использовать. Поскольку в данном примере мы запускаем Spark на локальной машине, то говорим «запусти конфигурацию с четырьмя воркерами», все будет считаться на них. Далее, мы создаем объект SparkContext'а, указывая соответствующий конфигурационный файл.

В примерах с notebooks код более развесистый:



Для начала запустим ячейку с инициализатором. По порту 4040 должен подняться UI самого Spark'а:



Здесь пока ничего особенного не будет, однако, если потребуется запускать сложные нагрузки и отлаживать их, то мы получим большую поддержку от данного UI.

Отличия SparkContext и SparkSession

SparkContext:

- Появился в Spark 1.x;
- Исторически использовался для создания RDD и «размазывания» данных по кластеру;
- Доступен в spark-shell, как sc.

SparkSession:

- Появился в Spark 2.x;
- Фактически заменил собой большую часть SparkContext;
- Включает в себя обертки для Hive, streaming'а и прочего;
- Позволяет работать как с RDD, так и с Dataframe/Dataset;
- Доступен в spark-shell, как spark.

Дедлайн 07 июля, 23:59 Мск

Источники, приемники, форматы

Так как мы запускаем все на своей локальной машине, мы можем работать с локальной файловой системой. Однако, если говорить о распределенном кластере, там необходимо хранилище.

Очевидно, поддерживаются HDFS, S3, а также коннекторы для других баз данных: Redis, Elasticsearch, MangoDB.

Для большинства реляционных баз и баз, которые поддерживают работу через SQL-запросы, есть реализация Java – Database connector (JDBC). Мы можем подключаться к Postgres SQL, My SQL и т.д.

Форматы, с которыми Spark может работать:

- CSV;
- JSON;
- XML;
- RCFile;
- Parquet;
- ORC;
- Avro.

Как вам урок?

Изучил, далее

Слёрм ©

+7 (495) 248-05-80

Лицензия №ДЛ-1368 от 22.08.2019

Политика конфиденциальности

Публичная оферта

Дедлайн 07 июля, 23:59 Мск