



Spark Developer

Spark UDAF: разрабатываем свой агрегатор



Проверить, идет ли запись

Напишите «+» в чат, если меня слышно и видно





Тема открытого урока

Spark UDAF: разрабатываем свой агрегатор



Заигрин Вадим

Ведущий эксперт по технологиям

Контакты:

vzaigrin@yandex.ru

<https://t.me/vzaigrin>



Маршрут вебинара

1. Знакомство

2. Об ОТУС

3. Команда курса

4. О курсе, программа
обучения

5. Обзор Spark

6. Spark UDAF

7. Бонус: карьерная информация

8. Рефлексия

Правила вебинара



Активно
участвуем



Задаем вопрос
в чат



Вопросы вижу в чате,
могу ответить не сразу

Условные обозначения



Индивидуально



Документ



Ответьте себе или
задайте вопрос

Расскажите о себе

- Как вас зовут? Откуда вы?
- Ваш опыт работы в IT?
- С какой основной целью вы записались на занятие?



06 OTUS

О компании



Сфера

ОТУС специализируется на обучении в IT.

Наша фишка — продвинутые программы для специалистов с опытом и быстрый запуск курсов по новым набирающим популярность технологиям.



Клиенты

Наши партнеры современные технологичные компании.

А обучение и открытые материалы привлекают специалистов разных грейдов: junior, middle, senior, lead.



Образование в OTUS



Программы курсов

OTUS имеет образовательную лицензию, поэтому наши курсы являются программами повышения квалификации и профессиональной переподготовки.

Направления курсов

Обучение специалистов разных грейдов: junior, middle, senior, lead



- Программирование
- Инфраструктура
- Тестирование
- Аналитика



- Data Science
- Управление
- GameDev
- Информационная безопасность

Мы в цифрах

130+

курсов для junior, middle, senior специалистов и менеджеров

600+

преподавателей делятся актуальными знаниями и реальными кейсами, востребованными в IT-индустрии



6

лет со дня основания компании

20 000+

выпускников уже прошли обучение по программам, адаптированным под запросы ведущих работодателей

430 000+

ИТ-специалистов в нашем сообществе, читают наши материалы, учатся и общаются на наших площадках

Напишите, пожалуйста, в чат подходящую цифру

- 1 - если уже учились у нас в компании
- 2 - если НЕ учились, но слышали о нас
- 3 - если впервые знакомитесь с OTUS



Знакомство с командой и программой курса

Процесс обучения



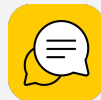
Обучение выстроено в формате вебинаров (онлайн). Онлайн-вебинары проводятся по вечерам или в выходные дни



Все записи занятий и материалы, предоставляемые преподавателями, сохраняются в личном кабинете и остаются доступны даже после окончания обучения



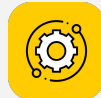
Домашние задания позволят Вам применить на практике полученные во время вебинаров знания. По каждому домашнему заданию преподаватель дает развернутый фидбек



В процессе обучения Вы можете задавать преподавателю вопросы по материалам лекций и домашних заданий, уточнять моменты, которые были непонятны на уроке



Время на обучение: от 4 ак. часов на занятия и 4-8 часов на домашнюю работу в неделю




Программа обучения на курсах обновляется каждый запуск в зависимости от актуальных запросов в сфере IT-технологий

Spark Developer

● КУРС ПЕРЕРАБОТАН

● РАССРОЧКА ?

ПРИ ПОДДЕРЖКЕ

Yandex  Cloud

Spark Developer

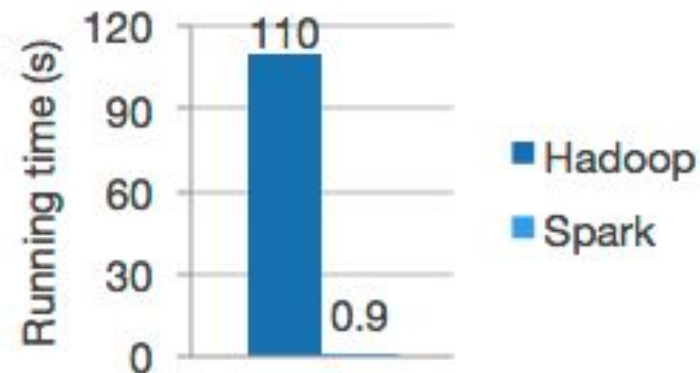
Углубленный курс по самым мощным инструментам обработки больших данных.

Вступительное тестирование

Apache Spark

Apache Spark

Apache Spark is a unified analytics engine for large-scale data processing



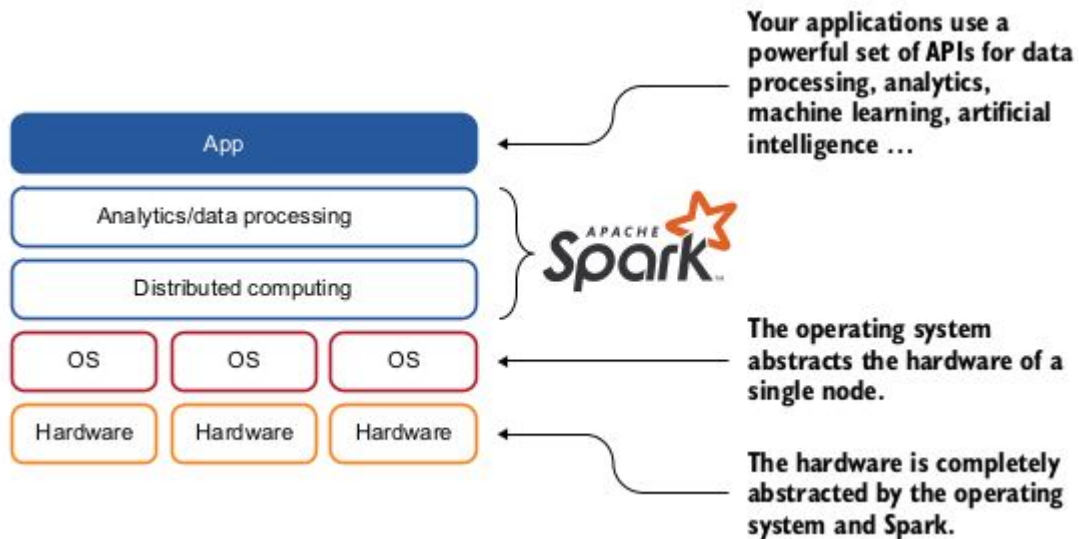
Logistic regression in Hadoop and Spark

```
df = spark.read.json("logs.json")
df.where("age > 21").select("name.first").show()
```

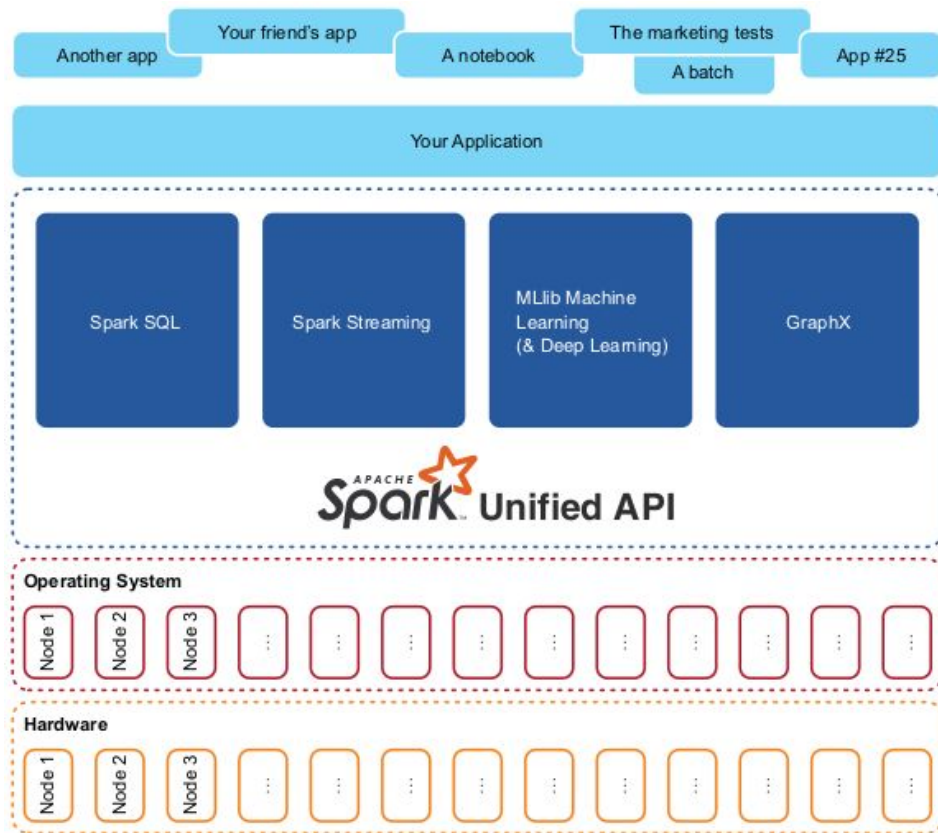
Apache Spark

- Фреймворк распределенной обработки данных
- Стандарт де-факто
- Работает со структурированными и слабо структурированными данными
- Запускается на разных платформах
- Удобный API

Что такое Spark



Что такое Spark



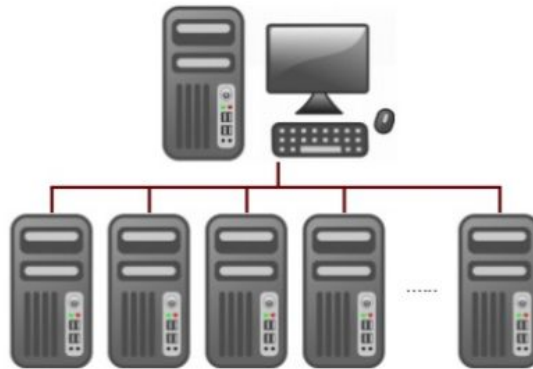
Развёртывание Spark

- Local

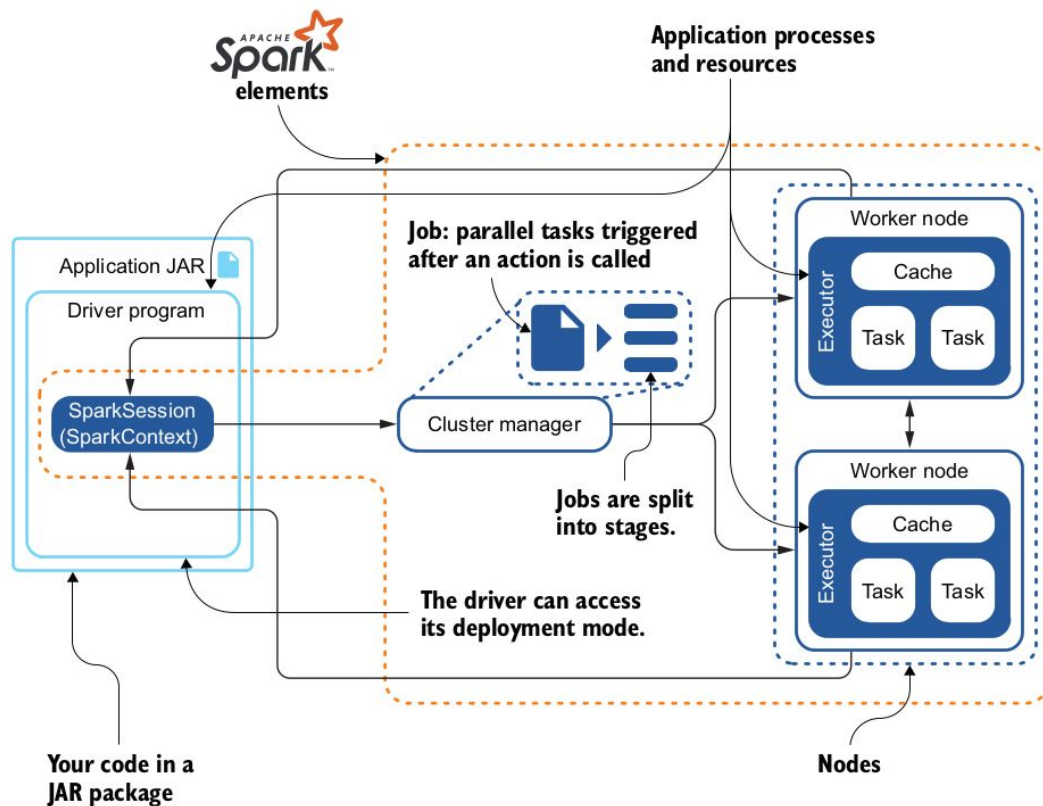


- Cluster

- Standalone
- YARN
- Mesos
- Kubernetes



Приложение Spark



Компоненты приложения

- **Application** — программа на Spark. Состоит из драйвера и исполнителей
- **Driver program** — Процесс, запускающий функцию `main()` приложения и создающий `SparkContext`. Все начинается здесь.
- **Worker node** — Любой узел, который может запускать код приложения в кластере
- **Executor** — Процесс, запущенный для приложения на рабочем узле. Исполнитель выполняет задачи и хранит данные в памяти или на диске

Запускаем Spark

- Устанавливаем JDK
- Устанавливаем `$JAVA_HOME`
- Развёртываем SPARK
- Устанавливаем `$SPARK_HOME`
- Меняем настройки `$SPARK_HOME/conf` (опционально)
- Запускаем пример:
 - `$SPARK_HOME/bin/run-example SparkPi 10`

Spark на Windows

- Создаём папку `C:\HADOOP\BIN`
- Копируем в неё `WINUTILS.EXE`
(<https://cwiki.apache.org/confluence/display/HADOOP2/WindowsProblems>)
- Может понадобиться *Microsoft Visual C++2010 Redistributable Package*
- Создаём переменную `%HADOOP_HOME%=C:\HADOOP`
- Добавляем в переменную `Path` `%HADOOP_HOME%\BIN`
- Создаём папку `C:\Temp`
- В файл `%SPARK_HOME%\conf\spark-defaults.conf` добавляем
`spark.local.dir=C:\\Temp`
- Запускаем `winutils chmod -R 777 C:\Temp`
- В файл `%SPARK_HOME%\conf\log4j.properties` добавляем
`log4j.logger.org.apache.spark.util.ShutdownHookManager=OFF`

Spark в интерактивном режиме

- `spark-shell` - Scala
- `pyspark` - Python
- Apache Zeppelin <http://zeppelin.apache.org>
- JetBrains Big Data Tools
<https://plugins.jetbrains.com/plugin/12494-big-data-tools>
- JupyterLab / Jupyter <https://jupyter.org>
- JetBrains Data Spell <https://www.jetbrains.com/ru-ru/dataspell>
- Almond <https://almond.sh>
- Databricks <https://community.cloud.databricks.com>



Запуск приложения Spark

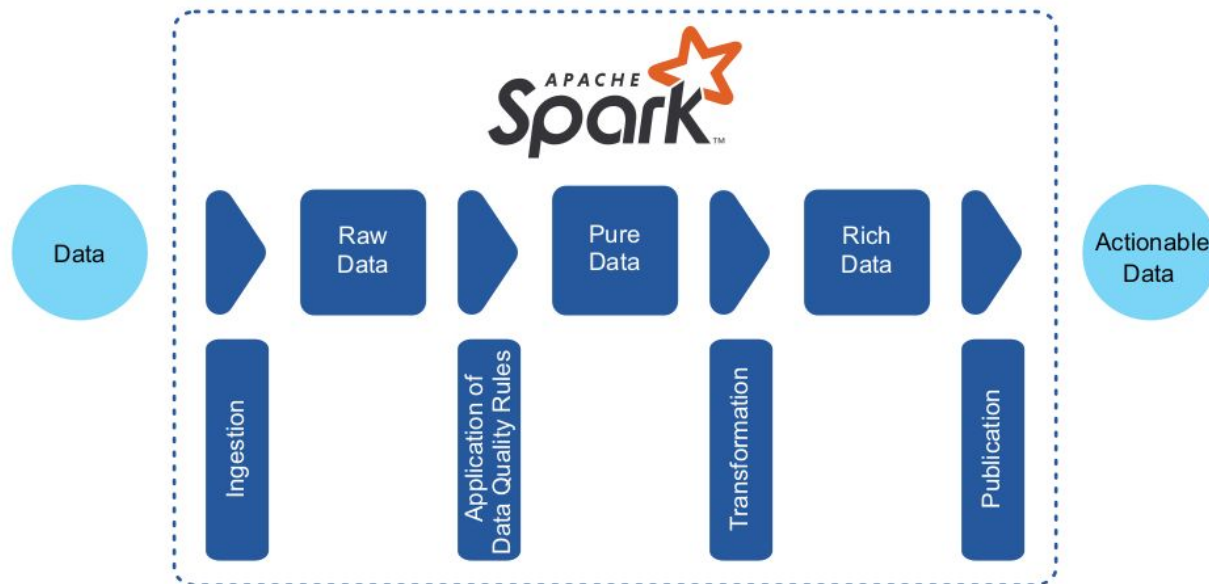
```
spark-submit [options] <app jar | python file | R file> [app arguments]
```

- `--master MASTER_URL` spark://host:port, mesos://host:port, yarn, k8s://https://host:port, or local (Default: local[*])
- `--deploy-mode DEPLOY_MODE` "client" or "cluster" (Default: client)
- `--class CLASS_NAME` Your application's main class (Java/Scala)
- `--jars JARS` Comma-separated list of jars to include on the driver and executor classpaths
- `--packages` Comma-separated list of maven coordinates of jars to include on the driver and executor classpaths
- `--py-files PY_FILES` Comma-separated list of .zip, .egg, or .py files] to place on the PYTHONPATH for Python apps
- `--driver-memory MEM` Memory for driver (e.g. 1000M, 2G) (Default: 1024M)
- `--executor-memory MEM` Memory per executor (e.g. 1000M, 2G) (Default: 1G)

Spark API

Структура программы на Spark

- Ввод данных
- Операции над данными
- Вывод результата



Spark API

- **RDD** — Resilient Distributed Datasets
- **DataFrame** — колоночная структура, представляющая собой подобие SQL-таблицы
- **Dataset** — классы, строгая типизация (Scala/Java)
 - `DataFrame = Dataset[Row]`
- **Pandas API** — новый (с 3.2.0) API для PySpark

RDD

Resilient Distributed Datasets — это отказоустойчивая коллекция элементов, которая может обрабатываться параллельно

- Коллекция разбита на партиции — “куски” данных
- Внутри коллекции элементы представлены сериализуемыми классами
- Автоматически перестраивается в случае сбоя

Недостатки RDD

- Не похож на SQL
- Нет возможности использования оптимизаций хранения данных как в колоночных форматах

DataFrame

DataFrame — колоночная структура, представляющая собой подобие SQL-таблицы

- Физические партиции
- Элементы теперь имеют специальную колоночную структуру, позволяющую выполнять SQL-запросы

RDD

```
rdd.map(x => (x.dept, (x.age, 1)))  
  .reduceByKey { case (x,y) => (x._1+y._1, x._2+y._2) }  
  .mapValues { case (x,y) => x / y }
```

DataFrame

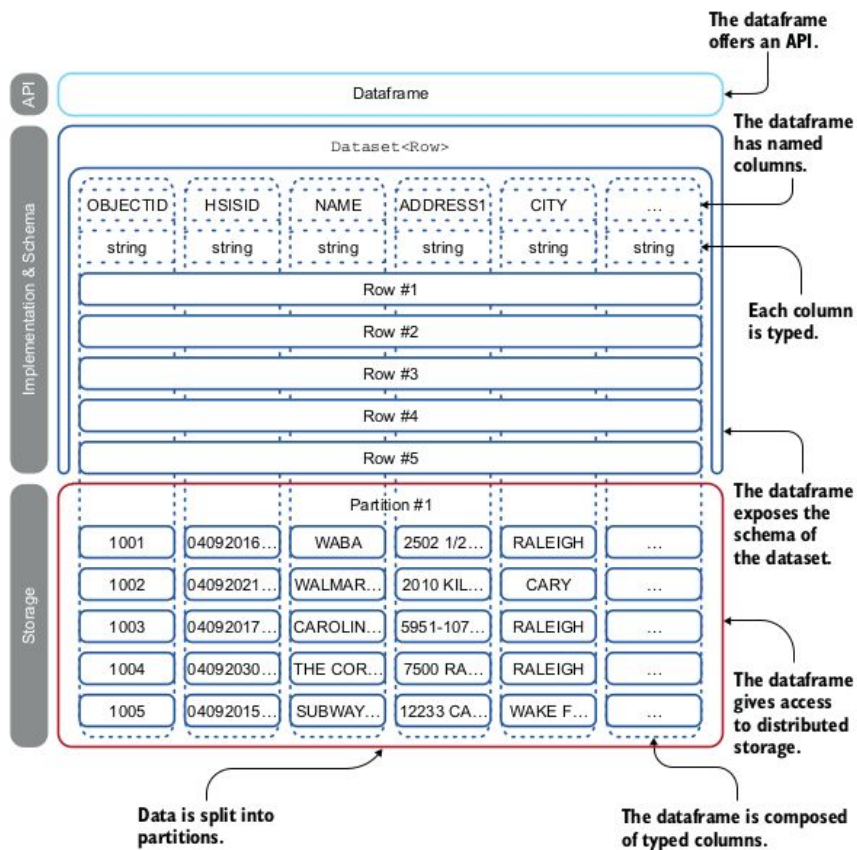
```
df.groupBy("dept")  
  .avg("age")
```

SQL

```
sqlContext.sql("SELECT avg(age) " +  
  "FROM people GROUP BY dept")
```



DataFrame



Инициализация

SparkSession — точка входа приложения

```
import org.apache.spark.sql.SparkSession

val spark = SparkSession
    .builder()
    .appName("Spark SQL basic example")
    .config("spark.some.config.option", "some-value")
    .getOrCreate()

// For implicit conversions like converting RDDs to DataFrames
import spark.implicits._
```



DataFrame DSL

- Высокоуровневые операции
 - Выбор колонок
 - Фильтры
 - Соединение
 - Агрегация
- Библиотека с большим количеством функций
 - 100+ оптимизированных функций
 - String, DateTime, Math,...
- UDF
- UDAF

DataFrame DSL Примеры

```
// Select everybody, but increment the age by 1
df.select($"name", $"age" + 1).show()
// +-----+-----+
// |  name|(age + 1)|
// +-----+-----+
// |Michael|    null|
// |   Andy|     31|
// |  Justin|     20|
// +-----+-----+

// Select people older than 21
df.filter($"age" > 21).show()
// +---+---+
// |age|name|
// +---+---+
// | 30|Andy|
// +---+---+

// Count people by age
df.groupBy("age").count().show()
// +-----+-----+
// |  age|count|
// +-----+-----+
// |   19|     1|
// | null|     1|
// |   30|     1|
// +-----+-----+
```

UDF - User Defined Function

```
// Define and register a zero-argument non-deterministic UDF
// UDF is deterministic by default, i.e. produces the same result for the same input.
val random = udf(() => Math.random())
spark.udf.register("random", random.asNondeterministic())
spark.sql("SELECT random()").show()

// +-----+
// |UDF() |
// +-----+
// |xxxxxxx|
// +-----+

// Define and register a one-argument UDF
val plusOne = udf((x: Int) => x + 1)
spark.udf.register("plusOne", plusOne)
spark.sql("SELECT plusOne(5)").show()

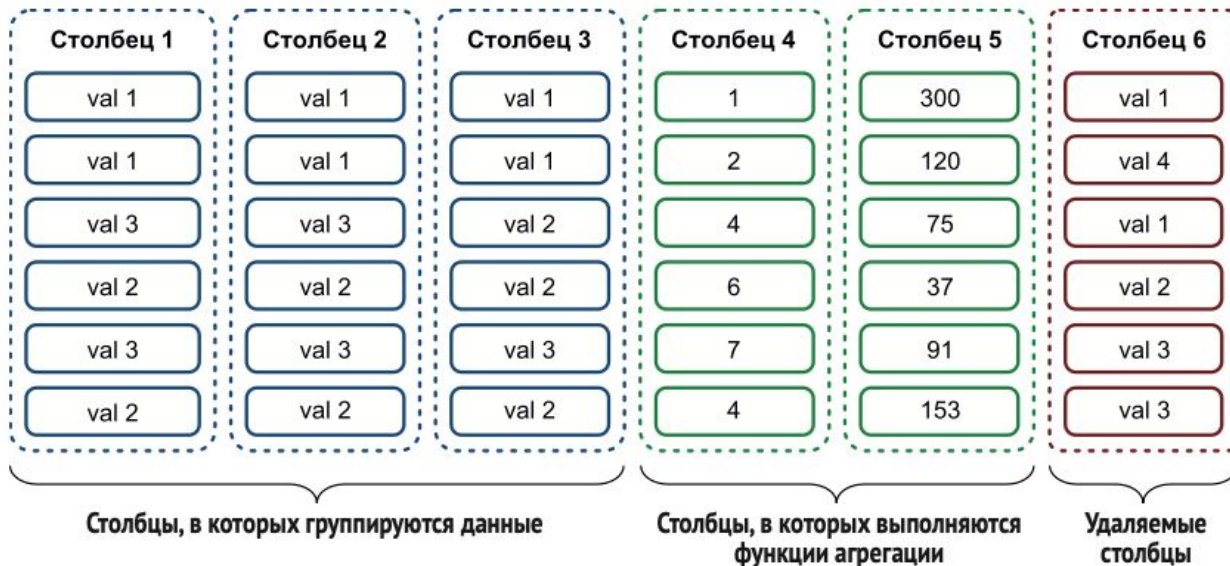
// +-----+
// |UDF(5)|
// +-----+
// |      6|
// +-----+
```

LIVE

Агрегирование данных

Агрегирование данных

Агрегирование данных — это способ группировки данных для последующего анализа обобщенных характеристик объектов



Пример

firstName	lastName	state	quantity	revenue	timestamp
Jean-Georges	Perrin	NC	1	300	1551903533
Jean-Georges	Perrin	NC	2	120	1551903567
Jean-Georges	Perrin	CA	4	75	1551903599
Holden	Karau	CA	6	37	1551904299
Ginni	Rometty	NY	7	91	1551916792
Holden	Karau	CA	4	153	1552876129

Summing the quantity

Summing the revenue and calculating an average

Пример (продолжение)

firstName	lastName	state	sum(quantity)	sum(revenue)	avg(revenue)
Jean-Georges	Perrin	NC	1	300	
			2	120	
			SUM 3	SUM 420	AVG 210
Jean-Georges	Perrin	CA	4	75	
			SUM 4	SUM 75	AVG 75
Holden	Karau	CA	6	37	
			4	153	
			SUM 10	SUM 180	AVG 95
Ginni	Rometty	NY	7	91	
			SUM 7	SUM 91	AVG 91

Пример (продолжение)

firstName	lastName	state	sum(quantity)	sum(revenue)	avg(revenue)
Jean-Georges	Perrin	NC	3	420	210
Jean-Georges	Perrin	CA	4	75	75
Holden	Karau	CA	10	180	95
Ginni	Rometty	NY	7	91	91

Агрегирование на SQL

```
SELECT
    "firstName",
    "lastName",
    "state",
    SUM(quantity) AS sum_qty,
    SUM(revenue) AS sum_rev,
    AVG(revenue::numeric::float8) AS avg_rev
FROM public.ch13
GROUP BY ("firstName", "lastName", "state");
```

Агрегирование в Spark

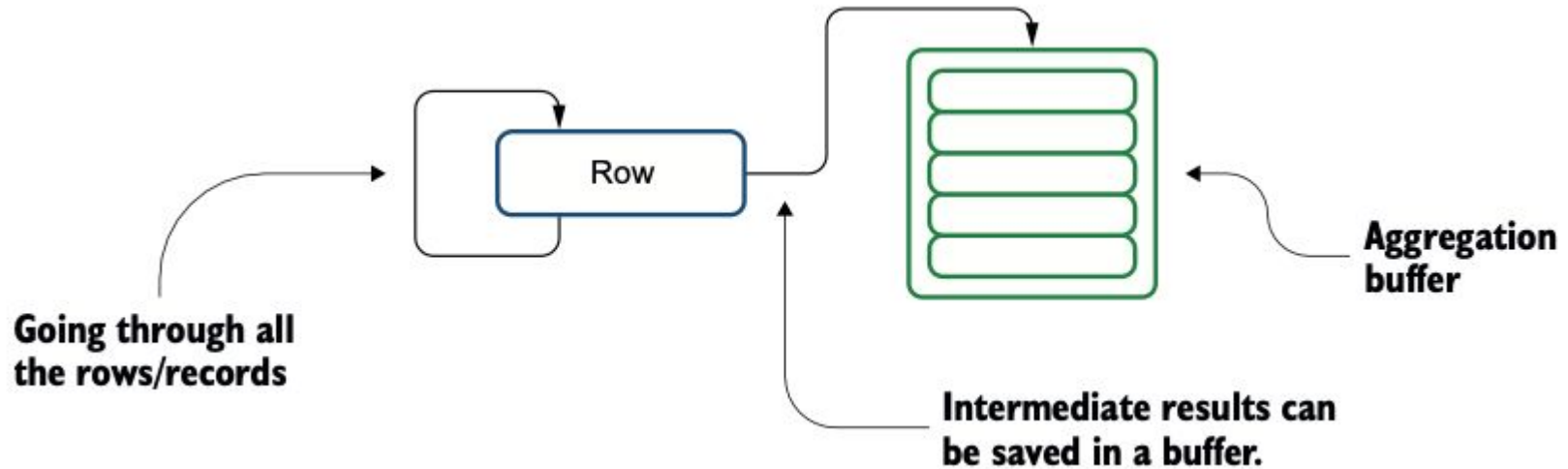
```
val aggDf = data
  .groupBy($"firstName", $"lastName", $"state")
  .agg(
    sum("quantity").as("sum_qty"),
    sum("revenue").as("sum_rev"),
    avg("revenue").as("avg_rev"))

aggDf.show()
```

LIVE

UDAF

User Defined Aggregate Functions (UDAFs)



Aggregator

Aggregator[-IN, BUF, OUT]

Параметры:

IN - Тип входных данных

BUF - Тип промежуточного значения

OUT - Тип выходного результата

Методы:

bufferEncoder: Encoder[BUF] - Кодировщик для типа промежуточного значения

finish(reduction: BUF): OUT - Преобразует выходные данные

merge(b1: BUF, b2: BUF): BUF - Объединяет два промежуточных значения

outputEncoder: Encoder[OUT] - Кодировщик для типа выходного значения

reduce(b: BUF, a: IN): BUF - Преобразует входное значение *a* в текущее промежуточное значение. Для повышения производительности функция может изменять *b* и возвращать его вместо создания нового объекта для *b*

zero: BUF - Начальное значение промежуточного результата

Простой пример: Среднее

```
case class Average(var sum: Long, var count: Long)

class MyAverage extends Aggregator[Long, Average, Double] {
  // Начальное значение. Должно соответствовать свойству: любое b + zero = b
  def zero: Average = Average(0L, 0L)
  // Объединение двух значений в новое значение.
  // Для повышения производительности функция может изменять `buffer` и
  // возвращать его вместо создания нового объекта.
  def reduce(buffer: Average, data: Long): Average = {
    buffer.sum += data
    buffer.count += 1
    buffer
  }
  // Объединение двух промежуточных значения
  def merge(b1: Average, b2: Average): Average = {
    b1.sum += b2.sum
    b1.count += b2.count
    b1
  }
  // Преобразование выходных данных
  def finish(reduction: Average): Double = reduction.sum.toDouble / reduction.count
  // Кодировщик для типа промежуточного значения
  def bufferEncoder: Encoder[Average] = Encoders.product
  // Кодировщик для типа выходного значения
  def outputEncoder: Encoder[Double] = Encoders.scalaDouble
}
```

LIVE

Вопросы?



Задаем
вопросы в чат



Ставим “-”,
если вопросов нет



Карьерная информация



Анализ позиции Data инженера

1613

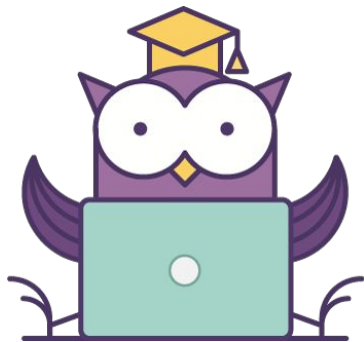
Вакансий Data инженера в июне 2023
г. для соискателей в России.

Источник — hh

Аналитика зарплатных предложений

Медиана

187 000+



Источник — hh

Вакансии работодателей

Администратор Hadoop

Что Вам предстоит делать?

- Развертывание, настройка, консультирования в продуктах экосистемы Hadoop: Impala, Hive, Spark, HDFS, MapReduce, Kylin, YARN, HBase
- Обеспечение бесперебойной работы и развития производственных систем;
- Организация работ по исправлению выявленных недостатков;
- Создание технической документации для внедряемых систем;
- Автоматизация управления конфигурациями;
- Обеспечение резервирования, восстановления баз.

Вакансии работодателей

Разработчик Spark\Scala

Твои задачи:

- Создание, оптимизация и доработка скриптов на Spark/PySpark
- Постановка витрин на регламент в AirFlow
- Доработка существующих фреймворков (Scala/Python) и создание новых

Мы ждем от тебя:

- Знание Spark/PySpark
- Знание Hive/Impala
- Умение работать с AirFlow
- Понимание принципов работы экосистемы Hadoop (Spark, Yarn) и знание базовых команд
- Знание Scala и Python

Вакансии работодателей

ML engineer / Data engineer



Сбер для экспертов ☺

Требования

не менее 3 лет работы в качестве Data Engineer/Data Analyst/Data Scientist

продвинутые знания SQL и Python: опыт разработки и, желательно, внедрения в пром. от 1 года

уверенные знания основных компонентов Apache Hadoop

опыт работы с фреймворком Apache Spark

понимание основ Machine Learning

высшее техническое образование

Будет плюсом:

знание Scala

опыт работы с Python ML-frameworks (Scikit-Learn, LightGBM, PyTorch и другие)

опыт работы с Spark streaming, Kafka, Hbase, GreenPlum

опыт работы с DevOps (Jenkins), Git (BitBucket)

опыт работы по методологии Agile (SCRUM, Kanban)

Условия



Вакансии работодателей

Senior разработчик Scala/Spark (Крипто-Анклав)

Какие знаний и навыки для нас важны:

- Опыт работы на позиции Разработчика не менее 3-х лет;
- Экспертные знания в Scala, Spark;
- Опыт проектирования перераспределенных систем;
- Умение разбираться в чужом коде, в том числе проводить review;
- Знание подхода DevOps в разработке;

Преимуществом будет:

- Имеете опыт работы со Java;
- Высокие коммуникативные навыки.

Рефлексия

Список материалов для изучения

1. Spark в действии <https://dmkpress.com/catalog/computer/data/978-5-97060-879-1>
2. Изучаем Spark <https://dmkpress.com/catalog/computer/data/978-5-97060-323-9>
3. Spark для профессионалов
<https://www.piter.com/collection/all/product/spark-dlya-professionalov-sovremennye-patterny-obrabotki-bolshih-dannyh>
4. Эффективный Spark. Масштабирование и оптимизация
https://www.piter.com/product_by_id/106943448
5. SCALA для нетерпеливых. Второе издание
<https://dmkpress.com/catalog/computer/programming/functional/978-5-97060-536-3>
6. Scala. Профессиональное программирование
<https://www.piter.com/collection/all/product/scala-professionalnoe-programmirovani-e-5-e-izd>

0 курсе

Spark Developer



старт обучения: 27.07.2023



Следующий открытый урок: 11.07.2023



Заполните, пожалуйста,
опрос о занятии

Спасибо за внимание!