

Оглавление

Часть I	■ Теория, разбавленная превосходными примерами	35
1	■ Так что же такое Spark?	36
2	■ Архитектура и рабочий процесс.....	56
3	■ Важнейшая роль фрейма данных.....	72
4	■ Природная лень	112
5	■ Создание простого приложения для развертывания	138
6	■ Развертывание простого приложения.....	165
Часть II	■ Потребление данных.....	190
7	■ Потребление данных из файлов.....	192
8	■ Потребление из баз данных.....	226
9	■ Более сложный процесс потребления: поиск источников данных и создание собственных.....	255
10	■ Потребление через структурированные потоки	288
Часть III	■ Преобразование данных	313
11	■ Работа с языком SQL	314
12	■ Преобразование данных	329
13	■ Преобразование документов в целом.....	364
14	■ Расширенные преобразования с помощью функций, определенных пользователем.....	378
15	■ Агрегирование данных	396
Часть IV	■ Продолжаем изучение Spark.....	424
16	■ Кеширование и копирование данных в контрольных точках: улучшение производительности Spark.....	426
17	■ Экспорт данных и создание полноценных конвейеров обработки данных	455
18	■ Описание ограничений процесса развертывания: объяснение экосистемы	478

Содержание

Оглавление.....	5
Словарь терминов	15
Вступительное слово	17
Предисловие	19
Благодарности.....	21
О чем эта книга.....	24
Об авторе.....	32
Иллюстрация на обложке	33

Часть I	Теория, разбавленная превосходными примерами.....	35
1	Так что же такое Spark?.....	36
1.1	Общая картина: что такое Spark и что он делает.....	37
1.1.1	Что такое Spark	37
1.1.2	Четыре столпа маны.....	40
1.2	Как можно использовать Spark	41
1.2.1	Spark в процессе обработки данных / инженерии данных	41
1.2.2	Spark в научных исследованиях в области обработки данных	44
1.3	Что можно делать с помощью Spark	45
1.3.1	Spark прогнозирует качество пунктов питания Северной Каролины	46
1.3.2	Spark обеспечивает быструю передачу данных для Lumeris	47
1.3.3	Spark анализирует журналы наблюдения за оборудованием CERN	48
1.3.4	Другие варианты использования	48

1.4	Почему вам очень понравится фрейм данных.....	48
1.4.1	Фрейм данных с точки зрения Java	49
1.4.2	Фрейм данных с точки зрения СУРБД	49
1.4.3	Графическое представление фрейма данных.....	50
1.5	Первый пример.....	51
1.5.1	Рекомендуемое программное обеспечение	51
1.5.2	Скачивание исходного кода	52
1.5.3	Запуск первого приложения	52
1.5.4	Первый исходный код для вас	53
	Резюме	54
2	Архитектура и рабочий процесс	56
2.1	Создание собственной мысленной (когнитивной) модели	57
2.2	Использование кода Java для создания мысленной (когнитивной) модели.....	58
2.3	Подробный разбор приложения	61
2.3.1	Установка соединения с ведущим узлом.....	62
2.3.2	Загрузка или потребление содержимого CSV-файла	63
2.3.3	Преобразование данных	66
2.3.4	Сохранение работы, сделанной в фрейме данных, в базе данных.....	68
	Резюме	71
3	Важнейшая роль фрейма данных	72
3.1	Чрезвычайно важная роль фрейма данных в Spark	73
3.1.1	Внутренняя организация фрейма данных	74
3.1.2	Неизменяемость – это не клятва	75
3.2	Использование фреймов данных на примерах.....	77
3.2.1	Фрейм данных после простой операции потребления CSV-файла.....	79
3.2.2	Данные хранятся в разделах.....	84
3.2.3	Подробнее о схеме.....	86
3.2.4	Фрейм данных после потребления формата JSON	87
3.2.5	Объединение двух фреймов данных	94
3.3	Фрейм данных как структура Dataset<Row>.....	99
3.3.1	Повторное использование простых старых объектов Java (POJO).....	100
3.3.2	Создание набора данных из строк	101
3.3.3	Преобразование фрейма данных в набор данных и обратно	103
3.4	Предшественник фрейма данных: RDD.....	109
	Резюме	110

4	Природная лень	112
4.1	Пример рациональной лени из реальной жизни.....	113
4.2	Пример рациональной лени в Spark.....	114
4.2.1	Рассмотрение результатов преобразований и действий.....	115
4.2.2	Процесс преобразования шаг за шагом.....	116
4.2.3	Код реализации процесса преобразования/действия.....	119
4.2.4	Загадка создания 7 миллионов точек данных за 182 мс.....	123
4.2.5	Загадка, связанная с измерением времени для действий.....	125
4.3	Сравнение с СУРБД и обычными приложениями.....	130
4.3.1	Обработка набора данных с коэффициентами рождаемости для подростков.....	130
4.3.2	Анализ различий между обычным приложением и приложением Spark.....	131
4.4	Spark великолепно подходит для приложений, ориентированных на обработку данных.....	133
4.5	Catalyst – катализатор приложения.....	133
	Резюме.....	137
5	Создание простого приложения для развертывания	138
5.1	Пример без операции потребления данных.....	139
5.1.1	Вычисление π	139
5.1.2	Исходный код для вычисления приближенного значения π	142
5.1.3	Что такое лямбда-функции в Java.....	148
5.1.4	Приближенное вычисление π с использованием лямбда-функций.....	150
5.2	Взаимодействие со Spark.....	152
5.2.1	Локальный режим.....	153
5.2.2	Режим кластера.....	154
5.2.3	Интерактивный режим Scala и Python.....	158
	Резюме.....	163
6	Развертывание простого приложения	165
6.1	Подготовка к изучению примера: роль компонент.....	168
6.1.1	Краткий обзор компонент и взаимодействий между ними.....	168
6.1.2	Рекомендации по устранению проблем в архитектуре Spark.....	172
6.1.3	Дополнительная информация для изучения.....	173
6.2	Создание кластера.....	174
6.2.1	Создание собственного кластера.....	174
6.2.2	Настройка среды кластера.....	176

6.3	Создание приложения для работы в кластере.....	179
6.3.1	Создание файла <i>uberJAR</i> для приложения.....	180
6.3.2	Создание приложения с использованием <i>Git</i> и <i>Maven</i>	182
6.4	Выполнение приложения в кластере.....	185
6.4.1	Передача файла <i>uberJAR</i>	185
6.4.2	Выполнение приложения	186
6.4.3	Анализ пользовательского интерфейса <i>Spark</i>	187
	Резюме	188

Часть II Потребление данных 190

7	Потребление данных из файлов.....	192
7.1	Общее поведение парсеров	194
7.2	Сложная процедура потребления данных из CSV-файла.....	194
7.2.1	Требуемый вывод результата	196
7.2.2	Код	197
7.3	Потребление CSV-данных с известной схемой	198
7.3.1	Требуемый вывод результата	199
7.3.2	Код	200
7.4	Потребление данных из JSON-файла.....	201
7.4.1	Требуемый вывод результата	203
7.4.2	Код	204
7.5	Потребление данных из многострочного JSON-файла.....	205
7.5.1	Требуемый вывод результата	207
7.5.2	Код	207
7.6	Потребление данных из файла XML	208
7.6.1	Требуемый вывод результата	210
7.6.2	Код	211
7.7	Потребление данных из текстового файла.....	213
7.7.1	Требуемый вывод результата	214
7.7.2	Код	214
7.8	Форматы файлов для больших данных.....	215
7.8.1	Проблема с обычными форматами файлов.....	215
7.8.2	<i>Avro</i> – формат сериализации на основе схемы	217
7.8.3	<i>ORC</i> – формат хранения данных в столбцах.....	217
7.8.4	<i>Parquet</i> – еще один формат хранения данных в столбцах	218
7.8.5	Сравнение форматов <i>Avro</i> , <i>ORC</i> и <i>Parquet</i>	218
7.9	Потребление данных из файлов <i>Avro</i> , <i>ORC</i> и <i>Parquet</i>	218
7.9.1	Потребление данных в формате <i>Avro</i>	219
7.9.2	Потребление данных в формате <i>ORC</i>	221
7.9.3	Потребление данных в формате <i>Parquet</i>	222
7.9.4	Справочная информация по организации потребления данных в форматах <i>Avro</i> , <i>ORC</i> , <i>Parquet</i>	224
	Резюме	224

8	Потребление из баз данных	226
8.1	Потребление из реляционных баз данных.....	228
8.1.1	Контрольный перечень операций при установлении соединения с базой данных	228
8.1.2	Объяснение происхождения данных, используемых в следующих примерах.....	229
8.1.3	Требуемый вывод результата	231
8.1.4	Код	232
8.1.5	Другая версия кода	234
8.2	Роль диалекта	236
8.2.1	Что такое диалект	236
8.2.2	Диалекты JDBC, предоставляемые в Spark.....	237
8.2.3	Создание собственного диалекта.....	237
8.3	Расширенные запросы и процесс потребления	240
8.3.1	Фильтрация с использованием ключевого слова <i>WHERE</i>	240
8.3.2	Соединение данных в базе данных.....	243
8.3.3	Выполнение потребления и распределение данных	246
8.3.4	Итоги изучения расширенных функциональных возможностей	249
8.4	Потребление данных из Elasticsearch.....	249
8.4.1	Поток данных.....	249
8.4.2	Набор данных о ресторанах Нью-Йорка, извлекаемый Spark.....	250
8.4.3	Исходный код для потребления набора данных о ресторанах из Elasticsearch.....	252
	Резюме	253
9	Более сложный процесс потребления: поиск источников данных и создание собственных	255
9.1	Что такое источник данных	257
9.2	Преимущества прямого соединения с источником данных.....	259
9.2.1	Временные файлы.....	260
9.2.2	Скрипты для улучшения качества данных.....	260
9.2.3	Данные по запросу.....	261
9.3	Поиск источников данных на сайте Spark Packages	261
9.4	Создание собственного источника данных.....	261
9.4.1	Обзор примера проекта.....	262
9.4.2	Интерфейс API специализированного источника данных и его параметры.....	264
9.5	Что происходит внутри: создание самого источника данных.....	267
9.6	Использование файла регистрации и заявочного класса.....	268

9.7	Объяснение взаимоотношения между данными и схемой.....	270
9.7.1	Источник данных создает отношение.....	271
9.7.2	Внутри отношения.....	274
9.8	Создание схемы из JavaBean.....	277
9.9	Создание фрейма данных – манипуляции с утилитами....	280
9.10	Другие классы	286
	Резюме	286

10	Потребление через структурированные потоки	288
10.1	Что такое потоковая обработка.....	290
10.2	Создание первого потока данных	292
10.2.1	Генерация потока данных.....	293
10.2.2	Потребление записей.....	296
10.2.3	Считывание записей, а не строк	302
10.3	Потребление данных из сетевых потоков	303
10.4	Работа с несколькими потоками	306
10.5	Различия между дискретизированными и структурированными потоками.....	311
	Резюме	312

Часть III	Преобразование данных.....	313
------------------	-----------------------------------	------------

11	Работа с языком SQL.....	314
11.1	Работа со Spark SQL.....	315
11.2	Различия между локальными и глобальными представлениями	319
11.3	Совместное использование API фрейма данных и Spark SQL.....	321
11.4	Не удаляйте (DELETE) данные	324
11.5	Рекомендации для дальнейшего изучения SQL.....	327
	Резюме	327

12	Преобразование данных	329
12.1	Что такое преобразование данных	330
12.2	Процесс и пример преобразования данных на уровне записи	331
12.2.1	Обследование данных для оценки их сложности	333
12.2.2	Отображение данных для создания схемы процесса	335
12.2.3	Написание исходного кода преобразования	338
12.2.4	Итоговый обзор результата преобразования данных для обеспечения качества обработки	345
12.2.5	Несколько слов о сортировке	347

12.2.6	<i>Завершение первого процесса преобразования с использованием Spark</i>	347
12.3	Соединение наборов данных	348
12.3.1	<i>Более подробно о соединяемых наборах данных</i>	348
12.3.2	<i>Создание списка вузов по округам</i>	350
12.3.3	<i>Выполнение соединений</i>	356
12.4	Выполнение других преобразований	362
	Резюме	362

13	Преобразование документов в целом	364
13.1	Преобразование документов в целом и их структура	365
13.1.1	<i>Упрощение структуры документа в формате JSON</i>	365
13.1.2	<i>Создание документов с вложенной структурой для передачи и сохранения</i>	371
13.2	Секреты статических функций	376
13.3	Выполнение других преобразований	377
	Резюме	377

14	Расширенные преобразования с помощью функций, определенных пользователем	378
14.1	Расширение функциональности Apache Spark	379
14.2	Регистрация и вызов UDF	381
14.2.1	<i>Регистрация UDF в Spark</i>	384
14.2.2	<i>Использование UDF совместно с API фрейма данных</i>	385
14.2.3	<i>Использование UDF совместно с SQL</i>	387
14.2.4	<i>Реализация UDF</i>	388
14.2.5	<i>Написание кода сервиса</i>	390
14.3	Использование UDF для обеспечения высокого уровня качества данных	392
14.4	Ограничения использования UDF	394
	Резюме	395

15	Агрегирование данных	396
15.1	Агрегирование данных в Spark	397
15.1.1	<i>Краткое описание агрегаций</i>	397
15.1.2	<i>Выполнение простых агрегаций с использованием Spark</i>	400
15.2	Выполнение агрегаций с оперативными данными	403
15.2.1	<i>Подготовка набора данных</i>	403
15.2.2	<i>Агрегация данных для получения более точной информации о школах</i>	408
15.3	Создание специализированных агрегаций с использованием UDAF	415
	Резюме	422

Часть IV Продолжаем изучение Spark 424

16 Кеширование и копирование данных в контрольных точках: улучшение производительности Spark 426

- 16.1 Кеширование и копирование данных в контрольных точках могут повысить производительность 427
 - 16.1.1 Полезность кеширования в Spark 429
 - 16.1.2 Изысканная эффективность механизма копирования данных в контрольных точках в Spark 431
 - 16.1.3 Использование кеширования и копирования данных в контрольных точках 431
- 16.2 Кеширование на практике 442
- 16.3 Дополнительные материалы по оптимизации производительности 452
- Резюме 453

17 Экспорт данных и создание полноценных конвейеров обработки данных 455

- 17.1 Экспорт данных 456
 - 17.1.1 Создание конвейера с наборами данных NASA 456
 - 17.1.2 Преобразование столбцов в метки времени *datetime* 459
 - 17.1.3 Преобразование процентов степени достоверности в уровень достоверности 461
 - 17.1.4 Экспорт данных 462
 - 17.1.5 Экспорт данных: что происходит в действительности 465
- 17.2 Delta Lake: удобная база данных прямо в системе 466
 - 17.2.1 Объяснение, почему необходима база данных 467
 - 17.2.2 Использование Delta Lake в конвейере обработки данных 468
 - 17.2.3 Потребление данных из Delta Lake 473
- 17.3 Доступ к сервисам облачного хранилища из Spark 475
- Резюме 477

18 Описание ограничений процесса развертывания: объяснение экосистемы 478

- 18.1 Управление ресурсами с использованием YARN, Mesos и Kubernetes 479
 - 18.1.1 Встроенный автономный режим управления ресурсами 480
 - 18.1.2 YARN управляет ресурсами в среде Hadoop 481
 - 18.1.3 Mesos – автономный диспетчер ресурсов 483
 - 18.1.4 Kubernetes управляет оркестровкой контейнеров 484
 - 18.1.5 Правильный выбор диспетчера ресурсов 486

18.2	Совместное использование файлов с помощью Spark	486
18.2.1	Доступ к данным, содержащимся в файлах.....	487
18.2.2	Совместное использование файлов с помощью распределенных файловых систем.....	488
18.2.3	Доступ к файлам на совместно используемых накопителях или на файловом сервере	490
18.2.4	Работа с сервисами совместного использования файлов для распределения файлов	491
18.2.5	Другие варианты обеспечения доступа к файлам в Spark	492
18.2.6	Гибридное решение совместного использования файлов в Spark	492
18.3	Уверенность в безопасности приложения Spark	492
18.3.1	Безопасность сетевых компонентов инфраструктуры.....	493
18.3.2	Безопасность при использовании диска Spark.....	494
	Резюме	495
Приложение А	Установка Eclipse	496
Приложение В	Установка Maven.....	502
Приложение С	Установка Git	506
Приложение D	Загрузка исходного кода и начало работы в Eclipse.....	508
Приложение E	Хронология корпоративных данных.....	514
Приложение F	Справочная информация по реляционным базам данных	519
Приложение G	Статические функции упрощают преобразования	524
Приложение H	Краткий справочник по Maven	533
Приложение I	Справочник по преобразованиям и действиям.....	538
Приложение J	Немного Scala.....	548
Приложение K	Установка Spark в реальной эксплуатационной среде и несколько рекомендаций	550
Приложение L	Справочник по операциям потребления.....	563
Приложение M	Справочник по соединениям	574
Приложение N	Установка Elasticsearch и пример набора данных	586
Приложение O	Генерация потоковых данных.....	592
Приложение P	Справочник по обработке потоковых данных	597
Приложение Q	Справочник по экспорту данных.....	608
Приложение R	Где искать помощь при затруднениях	616
	Предметный указатель	621

Словарь терминов

Краткое описание терминов Spark, используемых в процессе развертывания.

Термин на английском языке	Термин на русском языке	Определение
Application	Приложение	Программа, которая создается в рабочей среде Spark и для функционирования в Spark. Состоит из программы-драйвера и исполнителей в кластере
Application JAR	JAR-приложение	Архивный файл Java (JAR), содержащий приложение Spark. Это может быть файл uber-JAR, включающий все зависимости
Cluster manager	Диспетчер кластера	Внешний сервис для распределения ресурсов в кластере. Возможно использование встроенного в Spark диспетчера кластера
Deploy mode	Режим развертывания	Определяет, где запускается и работает процесс драйвера. В режиме кластера (cluster mode) фреймворк запускает драйвер внутри кластера. В режиме клиента (client mode) ведомый (подчиненный) запускает драйвер за пределами кластера. Можно определить, в каком режиме вы находитесь, выполнив метод <code>deployMode()</code> . Метод возвращает свойство, защищенное от изменения (только для чтения)
Driver program	Программа-драйвер	Процесс, выполняющий основную функцию <code>main()</code> приложения и создающий контекст <code>SparkContext</code> . Все начинается именно здесь
Executor	Исполнитель	Процесс, запускаемый для приложения на рабочем узле. Исполнитель выполняет задачи и сохраняет промежуточные данные в памяти или на диске. У каждого приложения имеются собственные исполнители
Job	Задание	Параллельное вычисление (выполнение), состоящее из нескольких задач, которые порождаются в ответ на некоторое действие Spark (например, <code>save()</code> или <code>collect()</code> , подробнее см. приложение I)
Stage	Этап	Каждое задание разделяется на более мелкие наборы задач, называемые этапами, зависящими друг от друга (подобно этапам отображения и сокращения в MapReduce)
Task	Задача	Минимальная единица работы, которая передается одному исполнителю
Worker node	Рабочий узел	Любой узел, который может выполнять код приложения в кластере



Вступительное слово

Аналитическая операционная система

В XX веке масштабы деятельности в деловой сфере значительно укрупнились благодаря глобальному расширению и распространению. Любая компания, выполняющая бизнес-операции по всему миру, получила вытекающее из этого преимущество в стоимости и степени распространения, что привело к появлению более конкурентоспособной продукции. Предприятия розничной торговли с глобальной сетью магазинов получили преимущество в распространении товаров, несравнимое с деятельностью более мелких компаний. Результаты этого глобального укрупнения определили преимущества в конкуренции на несколько десятилетий вперед.

Всю ситуацию в целом изменил интернет. В настоящее время существует три главных результата глобального укрупнения:

- сеть – замкнутость, управляемая надежной сетью (Facebook, Twitter, Etsy и т. д.);
- экономика глобального укрупнения – более низкая цена единицы товара, управляемая объемом (Apple, TSMC и т. д.);
- данные – всеобъемлющее машинное обучение и интуиция на основе динамически изменяющихся совокупностей данных.

В книге *Big Data Revolution* (Wiley, 2015 г.) я исследовал несколько компаний, которые извлекали реальную выгоду из данных в результате глобального укрупнения бизнес-деятельности. Но сейчас, в 2019 году, большие данные все еще остаются в основном неисследованной областью в организациях по всему миру. Spark – аналитическая операционная система – представляет собой средство, способное изменить существующее положение.

Spark стал инструментом, изменившим инновационную политику компании IBM. Spark – это аналитическая операционная система, обеспечивающая стандартизацию и унификацию источников данных и средства доступа к данным. Стандартизированная программная мо-

дель Spark делает эту систему самым лучшим вариантом выбора для разработчиков, создающих приложения анализа огромных объемов данных. Spark сокращает время и снижает сложность создания аналитических рабочих потоков, позволяя разработчикам сосредоточиться на задачах машинного обучения и создания экосистемы на основе Spark. Мы наблюдали ранее и видим сейчас, как проект с открытым исходным кодом быстро становится источником инноваций в широком масштабе.

Эта книга погружает вас в мир Spark. В ней рассматривается мощь этой технологии и гибкость ее экосистемы, а также практические приложения Spark для работы в конкретной современной компании. Вне зависимости от того, являетесь ли вы инженером по обработке данных, научным исследователем в области обработки данных, разработчиком прикладных программ или инженером поддержки ИТ-операций, эта книга расскажет об инструментальных средствах и откроет секреты, которые вы должны знать, чтобы управлять инновациями в своей компании или сообществе.

Наша стратегия в IBM – создание собственных проектов на основе и с использованием успешных открытых платформ, но эти проекты должны быть значительными и должны выделяться на общем фоне. Spark является именно такой платформой. В IBM можно найти множество примеров применения этой стратегии, и вы получите такой же результат в своей компании, если выберете этот путь.

Spark можно считать инновацией – это аналитическая операционная система, в которой будут успешно развиваться новые решения, устраняющие сдерживающий фактор обработки больших данных. Кроме того, Spark – это сообщество высококвалифицированных ученых и аналитиков в области обработки данных, хорошо знающих Spark, которые могут быстро превратить любые сегодняшние задачи в завтрашние решения. Spark представляет собой один из самых быстро развивающихся проектов с открытым исходным кодом в истории ИТ. Присоединяйтесь к этому движению.

– Роб Томас (Rob Thomas),
первый вице-президент,
подразделение Cloud and Data Platform, IBM

Предисловие

Не думаю, чтобы для Apache Spark требовалось какое-то предварительное представление. Если вы читаете эти строки, то, вероятнее всего, уже кое-что знаете о теме этой книги: инженерии данных и науки о данных, применяемой в крупных масштабах с использованием распределенной обработки. Но Spark – это нечто большее, и об этом вы скоро узнаете, прочитав вступительное слово Роба Томаса и главу 1.

Как Обеликс упал в котел с магическим зельем¹, я «упал в котел» Spark в 2015 году. В то время я работал во французской компании, производящей аппаратные компоненты для компьютеров, где участвовал в проектировании высокопроизводительных систем для анализа данных. Вполне естественно, первое впечатление от Spark было скептическим. Затем я начал постоянно работать со Spark и понял, что результат зависит от самого пользователя. Первоначальный скептицизм превратился в настоящее глубокое увлечение превосходным инструментом, который позволяет обрабатывать данные чрезвычайно простым способом, – это мое искреннее убеждение.

Я начал работу над несколькими проектами с использованием Spark, и это позволило мне выступить с сообщениями на Spark Summit, IBM Think, а также сблизиться с инициативами All Things Open, Open Source 101. Кроме того, через местную пользовательскую группу Spark я принял участие в анимации региона Роли-Дурэм в Северной Каролине. Это позволило мне встретиться с замечательными людьми и наблюдать за многочисленными проектами, связанными со Spark. В результате я увлекся Spark еще больше.

В этой книге я попытался поделиться с вами своим увлечением.

¹ Обеликс (Obelix) – известный персонаж комиксов и мультфильмов. Обеликс – неизменный спутник Астерикса (Asterix). Когда галл Астерикс выпивает магическое зелье, он получает суперсилу, которая помогает ему постоянно побеждать римлян (и пиратов). В раннем детстве Обеликс упал в котел, в котором варилось магическое зелье, воздействие которого на Обеликса стало постоянным. В Европе комиксы (и мультфильмы) про Астерикса (и Обеликса) весьма популярны. Больше информации см. на www.asterix.com/en/.

Примеры (или домашние задания) в этой книге основаны на использовании языка Java, но в репозитории книги содержится также код на языках Scala и Python. Сразу после выпуска версии Spark 3.0 сотрудники издательства Manning и я решили убедиться в том, что книга соответствует самым свежим версиям, а не отстает от действительности.

Как вы могли догадаться, мне очень нравятся комиксы. Я вырос на комиксах. Мне нравится такой способ передачи информации, в чем вы можете убедиться сами, читая эту книгу. Это не книжка комиксов, но около 200 иллюстраций в ней должны помочь вам понять великолепный инструмент, который называется Apache Spark.

Спутником Астерикса является Обеликс, точно так же и у книги «Spark в действии» (второе издание) есть спутник – постоянные средства поддержки, которые можно загрузить бесплатно из раздела ресурсов сайта издательства Manning по быстрой ссылке: <http://jgp.net/sia>. Эти средства поддержки содержат справочную информацию о статических функциях Spark и в итоге должны превратиться в более полезные справочные ресурсы.

Неважно, понравится вам книга или нет, в любом случае отправьте мне сообщение в твиттер на @jgperrin. Если книга понравилась, напишите отзыв на Amazon. Если книга не понравилась, то, как говорят при бракосочетании, (скажите сейчас) или храните молчание вечно. И все же я надеюсь, что книга вам понравится.

*Alea iacta est – Жребий брошен*¹.

¹ По-английски это высказывание звучит так: «The die is cast». Эту фразу приписывают Юлию Цезарю (заклятому врагу Астерикса), так как именно Цезарь перевел свою армию через Рубикон: событие произошло, и отменить его невозможно – как в случае, когда эта книга вышла из печати и стала доступна читателям.

Благодарности

В этом разделе я выражаю благодарность всем, кто помог мне в работе над этой книгой. Возможно, в этом разделе я забыл упомянуть некоторых людей, и если кто-то не обнаружил здесь своего имени, то я очень сожалею об этом. Искренне сожалею. Эта книга потребовала огромных усилий, и работа над ней в одиночку, вероятнее всего, была бы оценена в две или три звезды на Amazon вместо пятизвездочного рейтинга, который мы с вами получим очень скоро (это награда и благодарность за труд!).

Я хотел бы начать с благодарности тем рабочим группам, которые доверили мне этот проект, начиная с Zalonі (Анупам Ракшит (Anupam Rakshit) и Туфейл Хан (Tufail Khan)), Lumeris (Йон Фарн (Jon Farn), Сурья Кодуру (Surya Koduru), Ноэль Фостер (Noel Foster), Дивья Пенметса (Divya Penmetsa), Срини Гаддам (Srini Gaddam) и Брайс Татт (Bruce Tutt); всем, кто почти вслепую следовал за мной в массовом движении Spark, всем людям из Veracity Solutions и моей новой команде в Advance Auto Parts.

Спасибо Мэри Паркер (Mary Parker) из отдела статистики (Department of Statistics) Техасского университета (University of Texas) в Остине (Austin) и Кристiane Страччиалана Парада (Cristiana Straccialana Parada). Их вклад помог уточнить и сделать более ясным содержимое некоторых разделов.

Я хочу поблагодарить все сообщество в целом, в том числе Джима Хьюджеса (Jim Hughes), Майкла Бен-Давида (Michael Ben-David), Марселя-Яна Крейгсмана (Marcel-Jan Krijgsman), Жана-Франсуа Морена (Jean-Francois Morin) и всех неизвестных участников, присылавших запросы на включение изменений в репозиторий GitHub. Также хочу выразить искреннюю благодарность сотрудникам Databricks, IBM, Netflix, Uber, Intel, Apple, Alluxio, Oracle, Microsoft, Cloudera, NVIDIA, Facebook, Google, Alibaba, многочисленных университетов и всем, кто сделал Spark таким, какой он есть. В частности, спасибо за работу, вдохновение и поддержку Холдену Карау (Holden Karau), Яцеку Ласковски (Jacek Laskowski), Шону Оуэну (Sean Owen), Матею Захариа (Matei Zaharia) и Жюлю Дамжи (Jules Damji).

Во время работы над этим проектом я участвовал в нескольких подкастах. Благодарю Тобиаса Мейси (Tobias Macey) за подкаст «Data Engineering Podcast» (<http://mng.bz/WPjX>), Эла Мартина (Al Martin) из IBM за подкаст «Making Data Simple» (<http://mng.bz/8p7g>) и за подкаст «Roaring Elephant» Джона Масскелейна (Jhon Masschelein) и Дэйва Рассела (Dave Russell) (<http://mng.bz/EdRr>).

В качестве члена команды IBM я был счастлив работать со многими сотрудниками IBM на протяжении всего этого проекта. Они помогали прямо, косвенно или оказывали вдохновляющее воздействие: Роб Томас (Rob Thomas) (мы должны работать вместе и в дальнейшем), Мариус Чиортеа (Marius Ciortea), Алберт Мартин (Albert Martin) (он, помимо всего прочего, выпускает великолепный подкаст «Make Data Simple»), Стив Мур (Steve Moore), Сурав Мазумдер (Sourav Mazumder), Стейси Ронэхэн (Stacey Ronaghan), Мей-Мей Фу (Mei-Mei Fu), Виджай Боммиреддипалли (Vijay Bommireddipalli) (сохраняй присущее тебе равновесие на всех этих холмах Сан-Франциско!), Сунита Камбампати (Sunitha Kambhampati), Сахдев Зала (Sahdev Zala) и мой брат Стюарт Лайтел (Stuart Litel).

Хочу поблагодарить сотрудников издательства Manning, которые приняли в работу этот сумасшедший проект. Как в хороших кинофильмах, перечисляю «в порядке их появления»: рецензент издательства Майкл Стивенс (Michael Stephens), издатель Марьян Бэйс (Marjan Base), редакторы-консультанты Марина Майклз (Marina Michaels) и Тони Арритола (Toni Arritola); производственный персонал: Ирин Туюхи (Erin Twohey), Ребекка Райнхарт (Rebecca Rinehart), Берт Бейтс (Bert Bates), Кэндэс Джиллхулли (Candace Gillhoolley), Радмила Эрцеговац (Radmila Ercegovac), Алекс Драгосавлевич (Aleks Dragosavljevic), Матко Хрватин (Matko Hrvatin), Кристофер Кауфманн (Christopher Kaufmann), Ана Ромак (Ana Romac), Шерил Вайсман (Cheryl Weisman), Лори Уэйдерт (Lori Weidert), Шэрон Уилки (Sharon Wilkey) и Мелоди Долаб (Melody Dolab).

Также хочу выразить признательность и поблагодарить всех рецензентов издательства Manning. Это Анупам Сенгупта (Anupam Sengupta), Арун Лаккакулам (Arun Lakkakulam), Кристиан Кройтцер-Бек (Christian Kreutzer-Beck), Кристофер Карделл (Christopher Kardell), Конон Редмонд (Conor Redmond), Эзра Шрёдер (Ezra Schroeder), Габор Ласло Хайба (Gábor László Hajba), Гэри А. Стаффорд (Gary A. Stafford), Джордж Томас (George Thomas), Джулиано Араухо Бертоти (Giuliano Araujo Bertoti), Игор Франка (Igor Franca), Игор Карп (Igor Karp), Йерун Бенкхейсен (Jeroen Benckhuijsen), Хуан Руфес (Juan Rufes), Келвин Джонсон (Kelvin Johnson), Келвин Роулз (Kelvin Rawls), Марио-Леандер Раймер (Mario-Leander Reimer), Маркус Бройер (Markus Breuer), Массимо Далла Ровере (Massimo Dalla Rovere), Паван Мадхира (Pavan Madhira), Самбаран Хазра (Sambaran Hazra), Шоба Айвер (Shobha Iyer), Убальдо Пескаторе (Ubaldo Pescatore), Виктор Дюран (Victor Durán) и Уильям И. Уилер (William E. Wheeler). С помощью всей этой компании удалось написать (надеюсь) хорошую книгу. Также хочу поблагодарить Петара Зечевица (Petar Zečević) и Марко Банаси (Marco Banaci), которые написали первую редакцию этой книги. Спасибо Томасу Локни (Thomas Lockney) за подробную техническую

рецензию, а также Рамбабу Посу (Rambabu Posa) за включение исходного кода в книгу. Спасибо Йону Риу (Jon Rioux) (merci, Йонатан!) за первоначальный вариант проекта PySpark in Action. Йон подал идею «создания группы Spark в издательстве Manning».

И еще раз спасибо Марине. Марина (Marina Michaels) была моим редактором-консультантом в течение большей части времени работы над книгой. Она всегда была рядом, когда я выдавал предварительные версии, когда нужны были ее советы, она была строга со мной (да, твой контроль никогда не ослабевал!), но ее участие было весьма важным в этом проекте. Я всегда буду помнить наши долгие споры об этой книге (вне зависимости от того, существовали или нет предлоги для разговоров о чем-то еще). Я буду скучать по тебе, старшая сестра (до того момента, когда начнется работа над другой книгой).

Наконец, хочу сказать спасибо своим родителям, которые поддерживали меня больше, чем я того заслуживал, а также тем, кому я посвятил эту книгу: моей жене Лиз, которая помогала всем, чем могла, на любом уровне, в том числе и в достижении взаимопонимания с редакторами, и нашим детям Пьеру-Николя, Джеку, Натаниелю и Руби, у которых я украл так много времени, работая над этой книгой.

О чем эта книга

Когда я начинал работу над этим проектом, который стал книгой «Spark в действии», второе издание, моими целями были:

- помощь сообществу Java в использовании Apache Spark с наглядной демонстрацией того, что нет необходимости дополнительно изучать языки Scala и/или Python;
- описание главных концепций, лежащих в основе Apache Spark, инженерии (больших) данных и науке о данных, требующих только знаний о реляционных базах данных и основ языка SQL, и ничего больше;
- тщательное разъяснение того, что Spark – это операционная система, специально предназначенная для распределенных вычислений и анализа.

Я верю в метод обучения по любой теме из области информационных технологий с использованием большого количества примеров. Примеры в этой книге являются чрезвычайно важной частью процесса обучения. Я создавал примеры так, чтобы они были как можно ближе к ситуациям, возникающим в настоящей профессиональной деятельности. Предлагаемые здесь наборы данных взяты из реальных ситуаций, со всеми присущими им качественными недостатками. Это не идеализированные учебные наборы данных, которые «работают всегда». Именно поэтому, объединяя примеры и такие наборы данных, вы будете работать и учиться более практическим способом по сравнению со «стерилизованным» подходом к обучению. Я называю эти примеры лабораторными работами (labs) с надеждой, что они окажутся интересными для вас и вы захотите поэкспериментировать с ними.

В книге очень много иллюстраций. Основываясь на широко известном высказывании «Рисунок заменяет тысячу слов», я избавил вас от чтения 183 000 дополнительных слов.

Для кого предназначена эта книга

Трудно связать название профессии с названием книги, поэтому если ваша профессия называется инженер по обработке данных, ученый-ис-

Конец ознакомительного фрагмента.

Приобрести книгу можно

в интернет-магазине

«Электронный универс»

e-Univers.ru