计算机科学与技术学院大数据分析与实践课程实验报告

实验题目：SPARK 实践		学号：202322130197
日期：	班级：数据	姓名：崔嘉铭
Email：cjm13969665900@gmail.com		

实验目的：

本实验旨在介绍学习者如何配置和运行 Apache Spark，以及如何使用 Spark 进行简单的数 据处理和分析。实验将涵盖以下内容： 1. 安装和配置 Apache Spark。 2. 运行一个简单的 Spark 应用程序，以理解 Spark 的基本概念。 3. 使用 Spark 进行数据处理和分析。

实验软件和硬件环境：

Apache Spark

实验原理和方法：

实验步骤：（不要求罗列完整源代码）

1、安装和配置 Apache Spark

2、运行一个简单的 Spark 应用程序 从销售数据中统计衣物商品每日的总营收

代码：

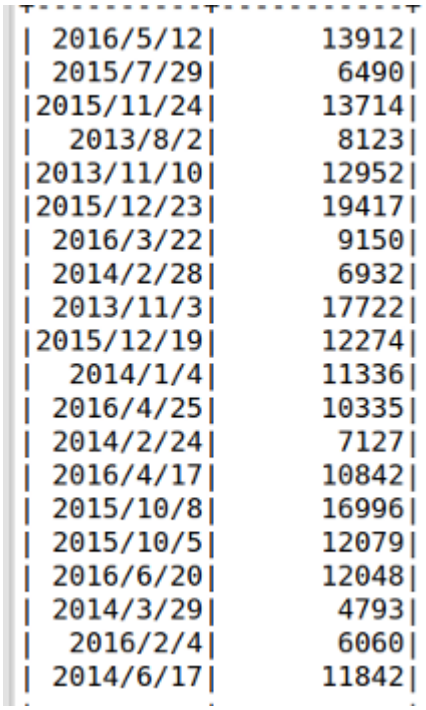
```
1. public static void main(String[] args) {
2.
3.     // 1. 初始化 SparkSession
4.     SparkSession spark = SparkSession.builder()
5.             .appName("SimpleSparkApp")
6.             .master("local[*]")
7.             .getOrCreate();
8.
9.     // 2. 加载 CSV 数据（开启 header + 自动推断 Schema）
10.    Dataset<Row> data = spark.read()
11.            .option("header", "true")
12.            .option("inferSchema", "true")
13.            .csv("/home/hadoop/Desktop/sales_data.csv"); // 数据文件路径
14.
15.    // 3. 数据处理
16.    Dataset<Row> result = data
17.            // 过滤
18.            .filter(col("product_category").equalTo("Clothing"))
19.            // 按 date 分组
20.            .groupBy("date")
21.            // 求和 revenue
```

```
            .agg(sum("revenue").alias("sum_revenue"));

    // 4. 显示结果
    result.show();

    // 5. 关闭 SparkSession
    spark.stop();
}

```



```
| 2016/5/12|      13912|
| 2015/7/29|       6490|
|2015/11/24|      13714|
|  2013/8/2|       8123|
|2013/11/10|      12952|
|2015/12/23|      19417|
| 2016/3/22|       9150|
| 2014/2/28|       6932|
| 2013/11/3|      17722|
|2015/12/19|      12274|
|  2014/1/4|      11336|
| 2016/4/25|      10335|
| 2014/2/24|       7127|
| 2016/4/17|      10842|
| 2015/10/8|      16996|
| 2015/10/5|      12079|
| 2016/6/20|      12048|
| 2014/3/29|       4793|
|  2016/2/4|       6060|
| 2014/6/17|      11842|
```

```
public static void main(String[] args) {

    // 1. 初始化 SparkSession
    SparkSession spark = SparkSession.builder()
            .appName("JavaSparkWordCountAdvanced")
            .master("local[*]")
            .getOrCreate();

    // 2. 读取文本文件
    String inputPath = "/home/hadoop/Desktop/input.txt";
    JavaRDD<String> lines = spark.read()
            .textFile(inputPath)
            .javaRDD();

    // 3. 数据预处理（清洗 + 分词 + 过滤停用词）
    JavaRDD<String> words = lines
```

```
            // 去除首尾空格
            .map(String::trim)
            // 过滤空行
            .filter(line -> !line.isEmpty())
            // 转小写
            .map(String::toLowerCase)
            // 分词
            .flatMap(line ->
                    Arrays.asList(WORD_PATTERN.split(line)).iterator()
            )
            // 过滤空字符串 + 过滤停用词 + 过滤单字符无意义词
            .filter(word -> !word.isEmpty()
                    && !STOP_WORDS.contains(word)
                    && word.length() > 1
            );

    // 4. 词频统计 + 降序排序
    JavaPairRDD<String, Integer> sortedWordCounts = words
            .mapToPair(word -> new Tuple2<>(word, 1))
            .reduceByKey(Integer::sum)
            .mapToPair(Tuple2::swap)
            .sortByKey(false)
            .mapToPair(Tuple2::swap);

    // 5. 输出结果
    System.out.println("过滤停用词后的词频统计（前 10）：");
    sortedWordCounts.take(10).forEach(tuple ->
            System.out.println(tuple._1() + ": " + tuple._2())
    );

    // 6. 关闭 SparkSession
    spark.stop();
}

```

```
spark: 2
hello: 2
java: 2
learn: 1
big: 1
data: 1
framework: 1
easy: 1
```

```
public static void main(String[] args) {
```

```
2.
3.     // 1. 初始化 SparkSession
4.     SparkSession spark = SparkSession.builder()
5.             .appName("MaxWeekdayRevenueCategory")
6.             .master("local[*]")
7.             .getOrCreate();
8.
9.     // 2. 读取销售数据，解析日期
10.    Dataset<Row> df = spark.read()
11.            .option("header", "true")
12.            .option("inferSchema", "true")
13.            .csv("/home/hadoop/Desktop/sales_data.csv")
14.            // 统一日期格式
15.            .withColumn("date_parsed",
16.                    to_date(col("Date"), "yyyy/M/d"));
17.
18.    // 3. 筛选工作日数据（周一=2 ~ 周五=6）
19.    Dataset<Row> weekdayDF =
20.            df.filter(dayofweek(col("date_parsed")).between(2, 6));
21.
22.    // 4. 按产品类别聚合工作日总销售额
23.    Dataset<Row> categoryRevenueDF = weekdayDF
24.            .groupBy("Product_Category")
25.            .agg(sum("Revenue").alias("total_weekday_revenue"));
26.
27.    // 按销售额降序输出
28.    categoryRevenueDF
29.            .orderBy(desc("total_weekday_revenue"))
30.            .show();
31.
32.    // 5. 关闭 SparkSession
33.    spark.stop();
34. }
35.
```

```
|Product_Category|total_weekday_revenue|
+----------------+---------------------+
|           Bikes|             43698486|
|     Accessories|             10663199|
|        Clothing|              6015432|
+----------------+---------------------+
```

```
1. public static void main(String[] args) {
2.
3.     // 1. 初始化 SparkSession
4.     SparkSession spark = SparkSession.builder()
```

```
5.              .appName("DailyAvgRevenueBySubCategory")
6.              .master("local[*]")
7.              .getOrCreate();
8.
9.      // 2. 读取数据并解析日期
10.     Dataset<Row> df = spark.read()
11.             .option("header", "true")
12.             .option("inferSchema", "true")
13.             .csv("/home/hadoop/Desktop/sales_data.csv")
14.             .withColumn(
15.                     "date_parsed",
16.                     to_date(col("Date"), "yyyy/M/d")
17.             );
18.
19.     // 3. 按产品子类分组，统计总销售额、总营业天数
20.     Dataset<Row> subcategoryStatsDF = df
21.             .groupBy("Sub_Category")
22.             .agg(
23.                     sum("Revenue").alias("total_revenue"),
24.                     countDistinct("date_parsed").alias("total_days")
25.             );
26.
27.     // 4. 计算日均销售额（保留 2 位小数）
28.     Dataset<Row> dailyAvgRevenueDF = subcategoryStatsDF.withColumn(
29.             "daily_avg_revenue",
30.             round(
31.                     col("total_revenue").divide(col("total_days")),
32.                     2
33.             )
34.     );
35.
36.     // 5. 展示结果（降序）
37.     dailyAvgRevenueDF
38.             .orderBy(desc("daily_avg_revenue"))
39.             .show();
40.
41.     // 6. 关闭 SparkSession
42.     spark.stop();
43. }
44.
```

Sub_Category	total_revenue	total_days	daily_avg_revenue
Road Bikes	33363061	1814	18391.99
Mountain Bikes	21123526	1644	12848.86
Touring Bikes	7295547	726	10048.96
Helmets	5741081	792	7248.84
Tires and Tubes	4670902	792	5897.6
Jerseys	4113742	786	5233.77
Shorts	1740710	676	2575.01
Bottles and Cages	1409174	790	1783.76
Vests	949063	534	1777.27
Fenders	1245733	774	1609.47
Hydration Packs	990406	624	1587.19
Bike Racks	517800	410	1262.93
Gloves	871419	740	1177.59
Bike Stands	344075	350	983.07
Caps	548777	782	701.76
Cleaners	198821	692	287.31
Socks	147171	566	260.02

本次实验围绕 Apache Spark 的环境配置与数据处理分析展开，成功完成了所有预设任务， 验证了 Spark 在结构化数据和非结构化数据处理中的高效性，认识到 Spark 的核心优势：一是 数据处理效率，通过 RDD 的惰性求值与分布式计算快速响应；二是 API 的灵活性，无论是结构 化数据的 Dataset 操作，还是非结构化数据的 RDD 处理，均能通过简洁 API 实现复杂逻辑。