实验报告模板

山东大学计算机科学与技术学院

大数据分析实践课程实验报告

实验题目：SPARK实践	
实验学时：2	实验日期：2025.11.28

实验目标：

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。

实验步骤：

1. 运行一个简单的Spark应用程序

- 创建 SparkSession
- 加载 CSV 文件
- 数据处理：筛选 + 分组求和
- 显示结果

完整代码如下：

```python
from pyspark.sql import SparkSession

# 初始化 SparkSession
spark = SparkSession.builder \
    .appName("SimpleSparkApp") \
    .config("spark.hadoop.fs.defaultFS", "file:///") \
    .getOrCreate()

try:
    # 加载数据
    data = spark.read.csv('sales_data.csv', header=True, inferSchema=True)

    # 执行一些数据处理操作
    result = data.filter(data["Product_Category"] == "Accessories").groupBy("Date")

    # 显示结果
    result.show()
finally:
    # 关闭 SparkSession
    spark.stop()
```

运行结果：

```
hadoop@ubuntu:~/Desktop/exp_spark$ python3 1.py
2025-11-28 01:51:31,118 WARN util.Utils: Your hostname, ubuntu resolves to a loopback address: 127.0.1.1; using 192.168.126.129 instead (on interface ens33)
2025-11-28 01:51:31,121 WARN util.Utils: Set SPARK_LOCAL_IP if you need to bind to another address
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
2025-11-28 01:51:32,619 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
+---------+-----------+
|     Date|sum(Revenue)|
+---------+-----------+
| 2016/5/12|       21505|
| 2015/7/29|        5994|
|  2013/8/2|       17700|
|2013/11/10|       20464|
|2015/11/24|       28005|
|2015/12/23|       19262|
| 2014/2/28|       28865|
| 2016/3/22|       16684|
|2015/12/19|       30493|
| 2013/11/3|       26902|
| 2016/4/17|       22595|
| 2016/4/25|       19838|
|  2014/1/4|       14332|
| 2014/2/24|       19069|
|  2016/2/4|       17017|
| 2015/10/5|       25070|
| 2014/3/29|       16641|
| 2015/10/8|       22254|
| 2016/6/20|       27200|
| 2013/8/28|       25919|
+---------+-----------+
only showing top 20 rows
```

2. 根据文档运行word count：

- 创建 SparkSession
- 读取文本文件
- WordCount 的核心逻辑
- 收集 & 打印结果

完整代码如下：

```python
from pyspark.sql import SparkSession

# 创建 SparkSession
spark = SparkSession.builder \
    .appName("WordCount") \
    .config("spark.hadoop.fs.defaultFS", "file:///") \
    .getOrCreate()

try:
    # 读取文本文件
    text_file = spark.sparkContext.textFile("text.txt")

    # 执行 WordCount 操作
    word_counts = (text_file
                   .flatMap(lambda line: line.split())  # 将每一行按空格拆分成单词
                   .map(lambda word: (word, 1))  # 每个单词映射为 (word, 1)
                   .reduceByKey(lambda a, b: a + b)  # 对相同的单词进行累加
                   )

    # 打印结果
    print("词频统计结果：")
    for word, count in word_counts.collect():
        print(f"{word}: {count}")
finally:
    # 关闭 SparkSession
    spark.stop()
```

运行结果：

```
hadoop@ubuntu:~/Desktop/exp_spark$ python3 2.py
2025-11-28 01:52:16,161 WARN util.Utils: Your hostname, ubuntu resolves to a loopback address: 127.0.1.1; using 192.168.126.129 instead (on interface ens33)
2025-11-28 01:52:16,163 WARN util.Utils: Set SPARK_LOCAL_IP if you need to bind to another address
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
2025-11-28 01:52:17,804 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
词频统计结果：
Saying: 1
Good-bye: 1
Cambridge: 3
Again: 1
Very: 2
quietly: 4
take: 2
leave: 2
as: 2
came: 2
In: 4
The: 2
willows: 1
riverside: 1
young: 1
sun;: 1
reflections: 1
shimmering: 1
waves: 2
depth: 1
of: 5
sludge: 1
Sways: 1
leisurely: 1
water;: 1
gentle: 1
would: 1
water: 2
That: 1
shade: 1
elm: 1
Holds: 1
but: 1
rainbow: 1
sky;: 1
```

3. 使用Spark求工作日流量最大的门店ID：

- 创建 SparkSession
- 加载 CSV 数据
- 数据清洗与统计

> step 1：筛选工作日
>
> step 2：过滤掉订单量 <= 0 的无效记录

> step 3：按 State 分组求总订单数
>
> step 4：重命名列名
>
> step 5：按订单量从高到低排序

- 打印结果

完整代码：

```python
from pyspark.sql import SparkSession
from pyspark.sql.functions import to_date, dayofweek

# 初始化SparkSession
spark = SparkSession.builder \
    .appName("MaxWeekdayStoreTraffic") \
    .config("spark.hadoop.fs.defaultFS", "file:///") \
    .config("spark.sql.legacy.timeParserPolicy", "LEGACY") \
    .getOrCreate()

try:
    # 加载数据
    sales_data = spark.read.csv(
        "sales_data.csv",
        header=True,
        inferSchema=True
    )

    # 筛选工作日（周一至周五：dayofweek返回2-6）并计算各地区的总订单量
    max_traffic_store = sales_data \
        .filter(dayofweek(to_date("Date", "yyyy/MM/dd")).between(2, 6)) \
        .filter("Order_Quantity > 0") \
        .groupBy("State") \
        .sum("Order_Quantity") \
        .withColumnRenamed("sum(Order_Quantity)", "total_order_quantity") \
        .orderBy("total_order_quantity", ascending=False)

    # 显示结果
    print("工作日各地区总订单量：")
    max_traffic_store.show(20)
finally:
    # 停止会话
    spark.stop()
```

运行结果：

```
hadoop@ubuntu:~/Desktop/exp_spark$ python3 3.py
2025-11-28 02:20:01,130 WARN util.Utils: Your hostname, ubuntu resolves to a loopback address: 127.0.1.1; using 192.168.126.129 instead (on interface ens33)
2025-11-28 02:20:01,132 WARN util.Utils: Set SPARK_LOCAL_IP if you need to bind to another address
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
2025-11-28 02:20:02,629 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
工作日各地区总订单量：
+------------------+--------------------+
|             State|total_order_quantity|
+------------------+--------------------+
|        California|              191174|
|  British Columbia|              137485|
|           England|              112637|
|        Washington|              100059|
|   New South Wales|               81750|
|            Oregon|               47347|
|          Victoria|               46523|
|        Queensland|               41560|
|          Saarland|               22387|
|Nordrhein-Westfalen|               20190|
|     Seine (Paris)|               19656|
|            Hessen|               18877|
|           Hamburg|               14403|
| Seine Saint Denis|               14265|
|              Nord|               13728|
|   South Australia|               11211|
|            Bayern|               10811|
|    Hauts de Seine|                9165|
|           Essonne|                8751|
|            Yveline|                7746|
+------------------+--------------------+
only showing top 20 rows
```

4. 求每个门店的日均客流量

- 创建 SparkSession
- 加载 CSV 数据

- 核心逻辑：计算每个州工作日日均订单量

> Step 1：筛选工作日
>
> Step 2：过滤掉无效数据
>
> Step 3：按州（State）分组
>
> Step 4：计算平均订单量
>
> Step 5：重命名结果列
>
> Step 6：按平均订单量排序

- 显示最终结果

完整代码：

```python
from pyspark.sql import SparkSession
from pyspark.sql.functions import to_date, dayofweek

# 初始化SparkSession
spark = SparkSession.builder \
    .appName("StoreDailyAvgTraffic") \
    .config("spark.hadoop.fs.defaultFS", "file:///") \
    .config("spark.sql.legacy.timeParserPolicy", "LEGACY") \
    .getOrCreate()

try:
    # 加载数据
    sales_data = spark.read.csv(
        "sales_data.csv",
        header=True,
        inferSchema=True
    )

    # 计算每个地区的工作日日均订单量
    store_daily_avg = sales_data \
        .filter(dayofweek(to_date("Date", "yyyy/MM/dd")).between(2, 6)) \
        .filter("Order_Quantity > 0") \
        .groupBy("State") \
        .avg("Order_Quantity") \
        .withColumnRenamed("avg(Order_Quantity)", "avg_order_quantity") \
        .orderBy("avg_order_quantity", ascending=False)

    # 显示结果
    print("每个地区的工作日日均订单量：")
    store_daily_avg.show(20)
finally:
    # 停止会话
    spark.stop()
```

运行结果：

```
hadoop@ubuntu:~/Desktop/exp_spark$ python3 4.py
2025-11-28 02:21:15,759 WARN util.Utils: Your hostname, ubuntu resolves to a loopback address: 127.0.1.1; using 192.168.126.129 instead (on interface ens33)
2025-11-28 02:21:15,763 WARN util.Utils: Set SPARK_LOCAL_IP if you need to bind to another address
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
2025-11-28 02:21:17,327 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
每个地区的工作日日均订单量：
+----------------+------------------+
|           State|avg_order_quantity|
+----------------+------------------+
|         Alabama|              25.5|
|     Mississippi|             22.75|
|       Minnesota|22.333333333333332|
|            Ohio|20.833333333333332|
|         Ontario|20.333333333333332|
|         Montana|20.333333333333332|
|        Virginia|              16.5|
|  North Carolina|              16.5|
|   Pas de Calais|14.061538461538461|
|        New York|              14.0|
|British Columbia|13.666500994035784|
|         Georgia|13.571428571428571|
|          Oregon|12.734534696073158|
|    Val de Marne| 12.63063063063063|
|      Val d'Oise|12.570621468926554|
|      Washington|12.304353172651254|
|          Somme|12.146067415730338|
|         Essonne|12.070344827586206|
|      California|11.973819366153075|
|Seine Saint Denis|11.907345575959933|
+----------------+------------------+
only showing top 20 rows
```