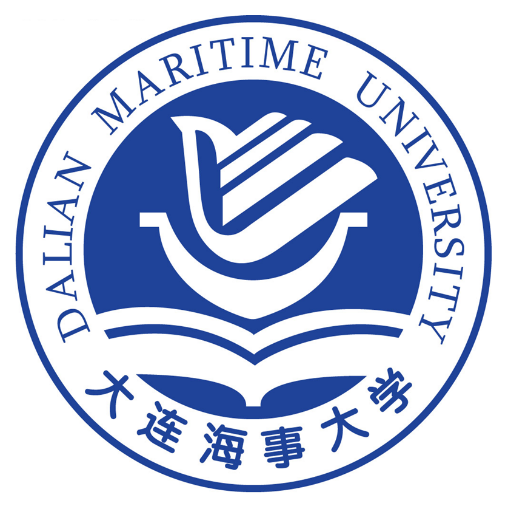
大连海事大学

**实 验 报 告**

《 大数据系统架构 》

2024～2025学年第 一 学期



**学 号：**

**姓 名：**

**指导教师：**

**报告时间：**

# 实验名称：\*\*\*\*\*\*\*\*

1. **实验目的**

本实验旨在使用 Apache Spark 对应聘者数据进行分析，实现以下两个主要目标：

1. 统计不同年龄的应聘者人数。
2. 计算各应聘职位中应聘者的平均海龄、最大海龄和最小海龄。  
   通过实验，熟悉大数据处理工具 Spark 的基本操作和功能，实现分布式数据处理与分析。
3. **实验基本原理**
4. **数据预处理**：

* 使用 Spark 提供的 API 对原始数据进行清洗与格式转换，确保数据的结构化和可用性。
* 转换数据类型，例如将“海龄”列中的字符串数据（包含“年”字符）转换为数值类型以便后续统计分析。

1. **分组与聚合**：

* 利用 Spark SQL 的 groupBy 和 agg 函数对数据进行分组和统计。
* 针对年龄分组统计总人数；针对应聘职位分组统计海龄的平均值、最大值和最小值。

1. **分布式存储与输出**：

* 通过 HDFS（Hadoop 分布式文件系统）将分析结果保存为 CSV 文件，展示 Spark 在分布式数据处理和存储上的强大能力。

1. **实验步骤及实验结果**

**1. 实验步骤：**

1.1 准备实验环境：

* 启动 Hadoop 和 Spark 环境，确保 HDFS 和 Spark 正常运行。
* 将原始数据文件 homework.csv 上传至 HDFS 指定目录。

1.2 数据加载与预处理：

* 使用 Spark 读取 HDFS 中的 CSV 文件，加载数据集。
* 对 "海龄" 列进行预处理，移除“年”后缀并转换为数值类型。

1.3 年龄人数统计：

* 按照 "年龄" 列分组，统计每个年龄的应聘者人数。
* 按统计结果的总人数降序排列。
* 将结果保存至 HDFS 指定目录。

1.4 职位海龄统计：

* 按照 "应聘" 列分组，计算每个职位的平均海龄、最大海龄和最小海龄。
* 按照平均海龄降序排列。
* 将结果保存至 HDFS 指定目录。

1.5 查看与验证结果：

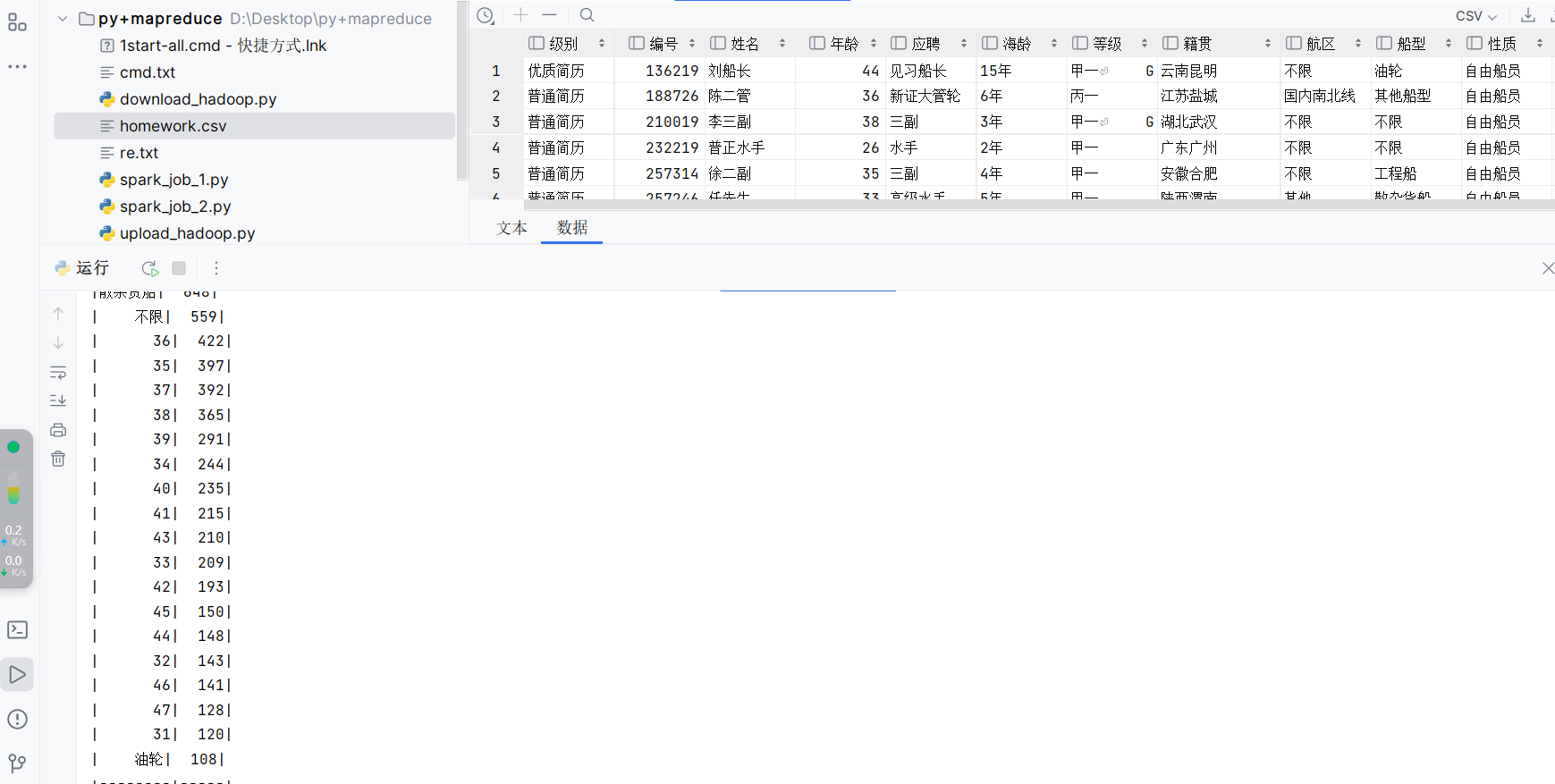
* 使用 Spark 提供的 show() 方法打印统计结果。
* 通过 HDFS 命令检查输出文件是否正确生成

1. **实验总结**

实验结果：

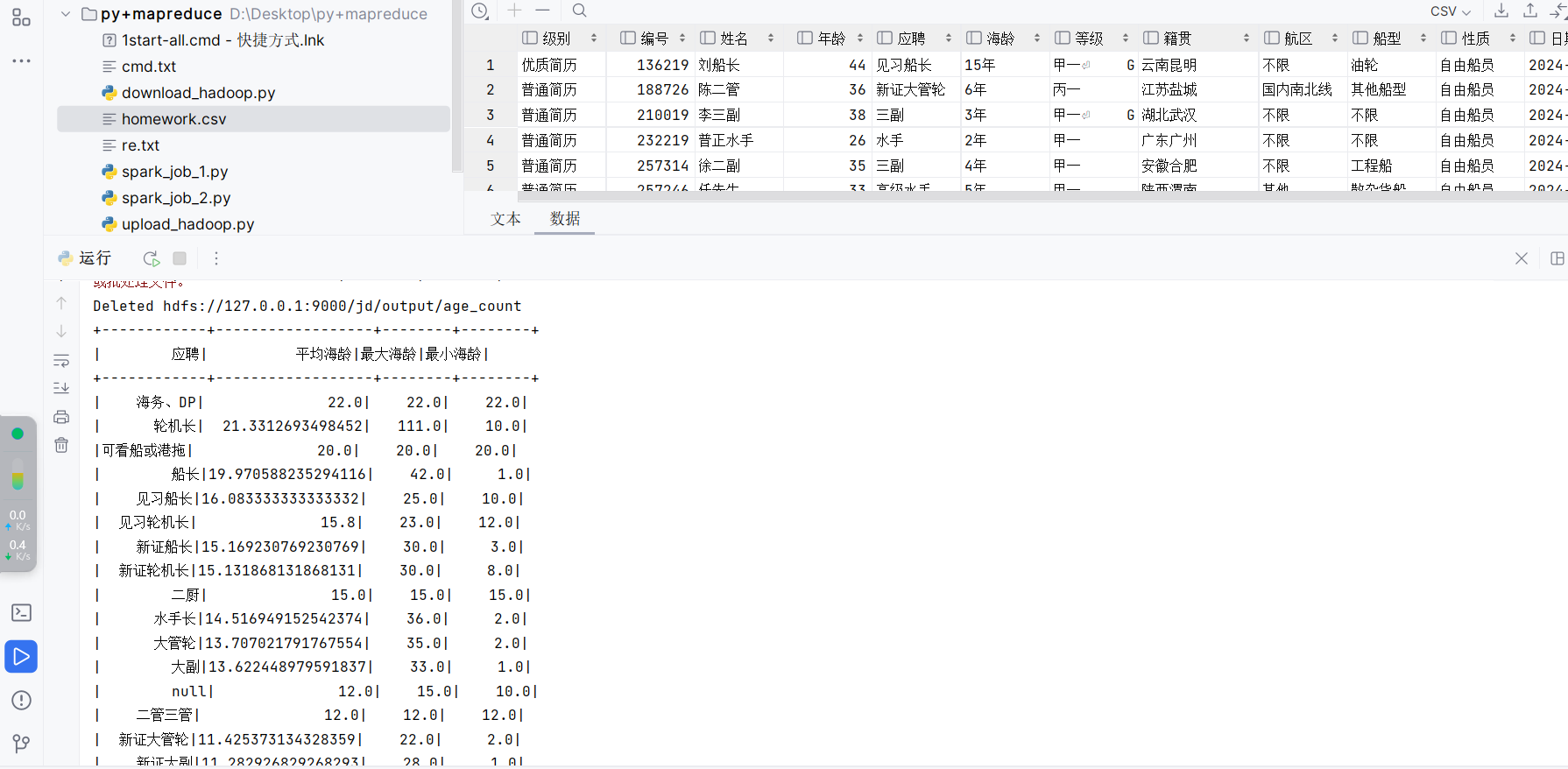
年龄人数统计结果：

按年龄分组，得到了每个年龄对应的应聘者人数。例如：



职位海龄统计结果：

按职位分组，得到了各职位的平均海龄、最大海龄和最小海龄。例如：



**附录：程序代码**

****

import os  
  
*# 执行命令hadoop fs -ls / 查看是否创建成功*hadoop\_url = 'hdfs://127.0.0.1:9000/{dir}/{file}'  
hadoop\_upload\_command = 'hadoop fs -put {local\_path} {hadoop\_path}'  
  
  
def upload\_file\_to\_hadoop(local\_path, hadoop\_path):  
 os.system(hadoop\_upload\_command.format(local\_path=local\_path, hadoop\_path=hadoop\_path))  
  
  
def loadFileList(dir):  
 return os.listdir(dir)  
  
  
commands = [  
 'hadoop fs -rm -r hdfs://127.0.0.1:9000/jd',  
 'hadoop fs -mkdir hdfs://127.0.0.1:9000/jd',  
 'hadoop fs -mkdir hdfs://127.0.0.1:9000/jd/dataset',  
 'hadoop fs -mkdir hdfs://127.0.0.1:9000/jd/output',  
]  
  
*# 创建文件夹*for command in commands:  
 print(f'执行命令：{command}')  
 os.system(command)  
  
  
  
print('正在上传文件：homework.csv')  
upload\_file\_to\_hadoop('homework.csv', hadoop\_url.format(dir='jd/dataset', file='homework.csv'))



import os

import findspark

from pyspark.sql import SparkSession

from pyspark.sql.functions import col, avg, max, min, count, regexp\_replace

findspark.init()

hadoop\_url = 'hdfs://127.0.0.1:9000/{dir}/{file}'

spark = SparkSession.builder.appName("resume\_analysis").getOrCreate()

# 加载应聘者数据

resume\_data = spark.read.csv(hadoop\_url.format(dir='jd/dataset', file='homework.csv'), header=True)

resume\_data.show()

# 预处理海龄列 - 移除“年”后转换为浮点数

resume\_data = resume\_data.withColumn('海龄', regexp\_replace(col('海龄'), '年', '').cast('float'))

# 1. 统计不同年龄的应聘者人数

age\_count = resume\_data.groupBy('年龄').count().sort('count', ascending=False)

age\_count.show()

# 修改列名

age\_count = age\_count.withColumnRenamed('年龄', 'name').withColumnRenamed('count', 'value1')

# 删除已存在的输出路径

output\_path = hadoop\_url.format(dir='jd/output', file='age\_count')

if os.system(f"hadoop fs -test -e {output\_path}") == 0:

os.system(f"hadoop fs -rm -r {output\_path}")

# 保存到 HDFS

age\_count.write.csv(output\_path, header=True)

# 2. 统计各应聘职位的平均海龄、最大海龄和最小海龄

position\_stats = resume\_data.groupBy('应聘').agg(

avg('海龄').alias('平均海龄'),

max('海龄').alias('最大海龄'),

min('海龄').alias('最小海龄')

).sort('平均海龄', ascending=False)

position\_stats.show()

# 删除已存在的输出路径

output\_path = hadoop\_url.format(dir='jd/output', file='position\_stats')

if os.system(f"hadoop fs -test -e {output\_path}") == 0:

os.system(f"hadoop fs -rm -r {output\_path}")

# 保存到 HDFS

position\_stats.write.csv(output\_path, header=True)

spark.stop()