第 6 章 Spark 集群环境搭建

★本章导读★

本章将介绍 Spark 集群的安装与相关配置，涉及 Spark 集群的搭建以及 Pyspark Notebook 的各种模式的启动方法。通过本章节的学习，读者能够更深一步认识 Spark 集群，从零搭建 Spark 集群，并使用 Pyspark Notebook 进行 WordCount 的练习。

★知识要点★

通过本章内容的学习，读者将掌握以下知识：

- Spark 集群搭建与配置方法
- Pyspark Notebook 各个模式的启动方法
- 能够使用 Pyspark Notebook 实现 WordCount 任务

6.1 Spark 集群环境介绍

本节将详细介绍 Spark 集群的搭建方法。

6.1.1 Spark 组件选择

本书使用的 Spark 版本为 2.4.3，读者也可以选择 2.4.x 版本的 Spark。2.4.x 版本的 Spark 各个功能较为完善，特别是 Structred Streaming 的融入，而且，在工业界生产中，Spark 2.4.x 也是使用非常广泛的一个版本。

Spark 可以从官网选择相应的版本进行下载，下载地址如下所示：

https://archive.apache.org/dist/spark/

Hadoop 集群 Web 端完整页面如图 6-1 所示。

图 6-1 Spark 安装包下载页面

首先我们选择下载的 Spark 版本为 2.4.3；其次我们选择的安装包为编译后的文件，即名字中带有“bin”的文件；最后我们需要下载的 Spark 安装包其对应的 Hadoop 版本应与本书前面章节安装的 Hadoop 版本相匹配，即为 2.6.x 系列。在悉知上述内容之后，点击图 6-1 中蓝色字样自动弹出下载的链接下载即可。

在这里，如果不清楚如何选择的读者也可以在终端通过 wget 命令直接下载，下载命令如下：

```
wget  https://archive.apache.org/dist/spark/spark-2.4.3/spark-2.4.3-bin-hadoop2.6.tgz
```

下载完成之后，将安装包移动到我们软件安装目录，并解压，命令如下：

```
mv spark-2.4.3-bin-hadoop2.6.tgz  /usr/local/src
cd /usr/local/src
tar zxvf spark-2.4.3-bin-hadoop2.6.tgz
```

完成之后软件安装目录如图 6-2 所示。

```
[root@master src]# ls
anaconda3                       conf          jdk1.8.0_172  spark-2.4.3-bin-hadoop2.6
Anaconda3-5.2.0-Linux-x86_64.sh hadoop-2.6.5  scala-2.11.8  tars
[root@master src]#
```

图 6-2 Spark 软件包

6.1.2 Spark 集群节点配置

接下来，进行 Spark 安装包的配置工作，步骤如下所示。

步骤 1：配置环境变量

在终端中输入如下命令并保存。

```
gedit  ~/.bashrc
```

输入完成后，按【Enter】键，打开环境变量配置文件，输入配置，如下图 6-3 所示。

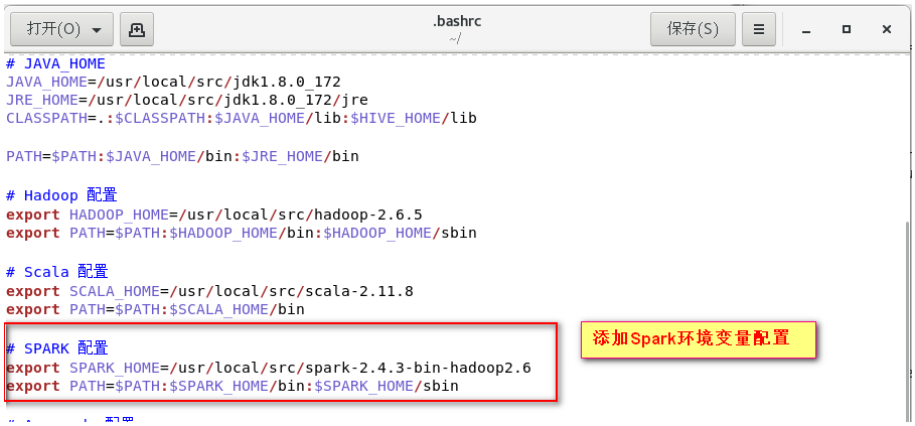


图 6-3 Spark 环境变量配置

步骤 2：生效环境变量，输入如下命令。

```
source  ~/.bashrc
```

同样，如 Hadoop 环境变量配置，需要通过命令使环境变量配置生效。

步骤 3：配置 Spark

接下来，需要对 Spark 集群进行配置，在终端中输入命令进入到 Spark 软件的配置文件目录，如图 6-4 所示。

```
cd  /usr/local/src/spark-2.4.3-bin-hadoop2.6/conf
```

```
[root@master conf]# cd /usr/local/src/spark-2.4.3-bin-hadoop2.6/conf
[root@master conf]# ls
docker.properties.template  metrics.properties.template  spark-defaults.conf.template
fairscheduler.xml.template  slaves                       spark-env.sh
log4j.properties.template   slaves.template              spark-env.sh.template
[root@master conf]#
```

图 6-4 Spark 配置文件目录

步骤 4：拷贝配置文件

按如下命令，将配置文件模板复制并修改名字。

```
cp  spark-env.sh.template  spark-env.sh
cp  slaves.template  slaves
```

接下来分别对上述配置文件进行配置，首先是 spark-env.sh 文件，同样使用 gedit 打开文件进行编辑，配置内容如下图 6-6 所示。

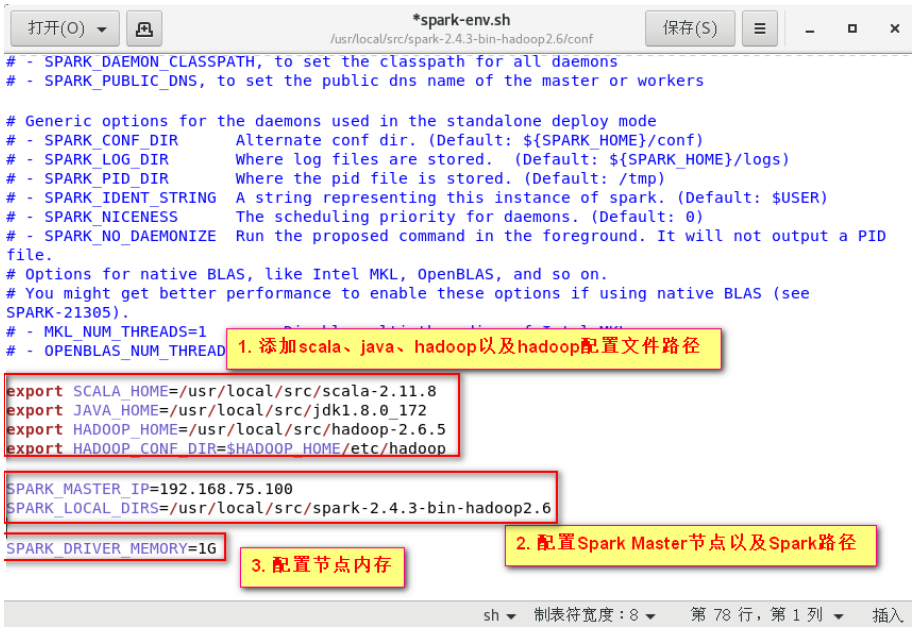


图 6-6 spark-env.sh 文件配置

同样，打开 slaves 文件进行配置，配置内容如下图 6-7 所示。

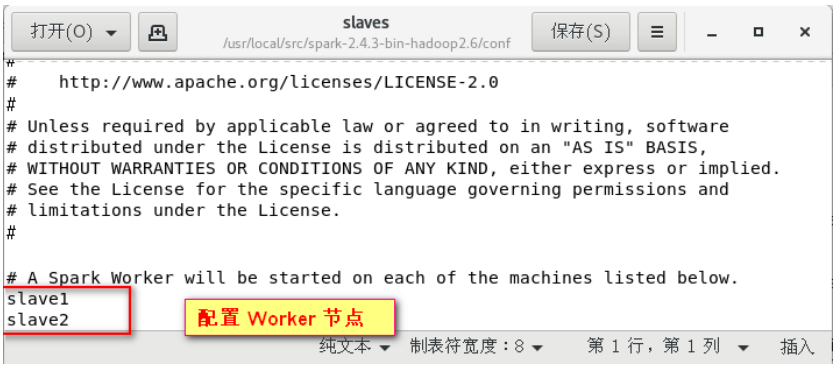


图 6-7 slaves 文件配置

由于本书大数据集群环境使用了三台虚拟机，所以 worker 节点配置两台机器。

步骤 5：同步 Spark 安装软件

至此，master 节点的配置工作已经完成，接下来我们需要将 Spark 软件包同步到 slave1 和 slave2 节点，命令如下。

```
scp -r /usr/local/src/spark-2.4.3-bin-hadoop2.6 root@slave1: /usr/local/src/
scp -r /usr/local/src/spark-2.4.3-bin-hadoop2.6 root@slave2: /usr/local/src/
```

步骤 6：slave 节点配置

同样的，slave 节点也需要进行环境变量的配置工作，而 Spark 的配置在 master 节点已经完成。环境变量的配置方法同步骤 1 和步骤 2。

6.2 Spark 集群监控

本小节中，我们将介绍如何启动 Spark 集群，以及如何查看 Spark 集群的健康状态与节点信息。通过对 Spark 集群的监控方法的学习，可以验证是否正确搭建完成 Spark 集群。

6.2.1 启动 Spark 集群

步骤 1：启动 Spark 集群

在 master 节点，通过终端输入命令进入到 Spark 执行脚本目录，执行 Spark 启动命令，启动命令如下所示。

```
cd /usr/local/src/spark-2.4.3-bin-hadoop2.6/sbin
./start-all.sh /local/src/
```

启动后终端日志如图 6-8 所示。

```
[root@master sbin]# cd /usr/local/src/spark-2.4.3-bin-hadoop2.6/sbin
[root@master sbin]# ./start-all.sh
starting org.apache.spark.deploy.master.Master, logging to /usr/local/src/spark-2.4.3-bin-hadoop2.6/logs/spark-root-org.apache.spark.d
eploy.master.Master-1-master.out
slave1: starting org.apache.spark.deploy.worker.Worker, logging to /usr/local/src/spark-2.4.3-bin-hadoop2.6/logs/spark-root-org.apache
.spark.deploy.worker.Worker-1-slave1.out
slave2: starting org.apache.spark.deploy.worker.Worker, logging to /usr/local/src/spark-2.4.3-bin-hadoop2.6/logs/spark-root-org.apache
.spark.deploy.worker.Worker-1-slave2.out
[root@master sbin]#
```

图 6-8 Spark 集群启动

步骤 2：查看各节点进程

分别在 master、slave1、slave2 节点输入命令“jps”可以查看当前机器的进程，从而判断是否正常启动 Spark，如图 6-9 所示。

图 6-9 Spark Master 节点进程

批注 [雷1]: 未添加图片，需添加正确图片

如上图所示，master 节点进程中有 Master，即 Spark 主节点服务进程，说明 master 节点已经启动。在 master 节点可以通过 ssh 命令登录 slave1 和 slave2 节点，同样的方法查看节点进程，如图 6-10 所示。

```
[root@master src]# ssh slave1
Last login: Sat Feb  5 12:06:28 2022 from master
[root@slave1 ~]# jps
5602 Worker
5701 Jps
5223 NodeManager
5135 DataNode
[root@slave1 ~]# ssh slave2
Last login: Sat Feb  5 15:10:53 2022 from slave1
[root@slave2 ~]# jps
5219 NodeManager
5131 DataNode
5660 Jps
5567 Worker
```

图 6-10 Spark Slave 节点进程

由上图可知，我们已经正确启动 Spark 集群，各节点均存在应有进程。但是，Spark 集群状态是否健康，集群是否真的正常呢？我们需要通过 Spark 集群监控页面进行进一步确认。

6.2.2 Spark 集群监控页面

Spark 集群监控网页默认端口是 8080，通过 master 节点 IP 或者 hostname 指定端口可以打开 Spark 监控页面，如图 6-11 所示。

图 6-11 Spark 监控页面

在监控页面中，我们可以看出，Spark 集群存活节点数为 2，Workers 数量为 2，在页面 Workers 中可以看到两个 Worker 节点的 Id、Address、State、Cores、Memory 信息。这说明我们正常启动 Spark 集群，并且集群各个节点处于健康状态。

但是如果由于我们配置错误或者其他原因导致集群各节点进程虽然存在，在 Spark 监控页面中却不存在 Worker 节点，则说明集群没有正常启动，如图 6-12 所示。

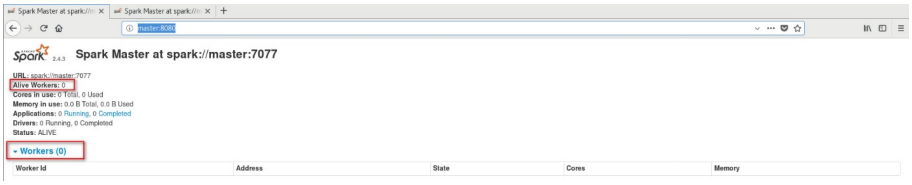


图 6-12 Spark 监控页面

由上图可以看到，我们可以正常访问 Spark 监控页面，但是在 Workers 下却看不到 Worker 节点，这说明 Worker 节点启动不正常，我们需要通过 6.2.1 节步骤 1 启动时打印出的各个节点的日志进一步排查确定原因并解决。

6.3 Spark 交互式界面

在本小节中，我们将介绍如何进入 spark 交互界面以及在不同模式下 pyspark 的交互界面。在实际开发中，通过交互界面可以方便的帮助我们进行调试、开发。

6.3.1 Spark Shell 交互界面

在终端中，输入如下命令，然后按【Enter】键，就可以进入到 Spark 交互界面。

```
spark-shell
```

值得注意的是，该命令进入的 Spark 交互界面使用的是 Scala 语言，启动 Spark 交互界面如图 6-13 所示。

图 6-13 启动 spark-shell

从启动日志中可以看到 Spark 的版本为 2.4.3，以及 Java 和 Scala 的版本。

输入如下命令可以查看当前 spark-shell 应用的模式为本地 local 模式，如图 6-14 所示。

```
scala> sc.master
res0: String = local[*]
```

图 6-14 spark-shell 模式

该命令的具体含义在后续章节会详细介绍，再对输出内容进行详解叙述。输出内容 local[N]代表在本地运行，使用 N 个线程，而 local[*]表示会尽可能使用机器上的所有 CPU 核心。

6.3.2 Local 模式 Pyspark 交互界面

由于本书是基于 Python 语言实现大数据分析处理，所以着重介绍 Pyspark 在不同模式下如何启动交互式界面。首先，本小节介绍使用最多、最方便的本地模式。

在终端输入如下命令启动 Pyspark 本地模式。

```
pyspark
```

启动界面如图 6-15 所示。

图 6-15 pyspark 本地模式界面

从上图可以看到 Spark 和 Python 的版本信息，同样可以通过命令 sc.master 查看当前的运行模式，看到 local[*]，代表当前为本地模式，以及使用 CPU 资源的情况。

当然，我们也可以通过参数指定，并且设置该进程的 CPU 使用情况，在终端输入如下命令启动 Pyspark 本地模式。

```
pyspark  --master  local[2]
```

上述命令中 local[2]的含义在上一小节中已经进行了介绍。下面说明一下—master 的含义，--master 即代表启动模式，通过该参数指定本地模式还是集群模式等。

启动界面如图 6-16 所示。

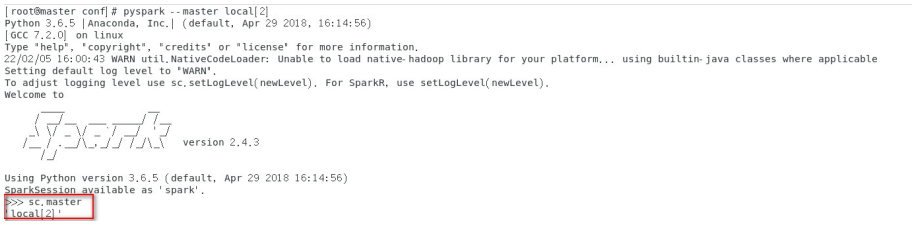


图 6-16 pyspark 本地模式界面

6.3.3 Yarn 模式 Pyspark 交互界面

启动 Yarn 模式的 Pyspark 交互界面命令如下，请注意下面代码是一行代码，由于书面打印限制分列在两行显示。

```
HADOOP_CONF_DIR=/usr/local/src/hadoop-2.6.5/etc/hadoop pyspark --master yarn --deploy-mode client
```

启动界面如图 6-16 所示。

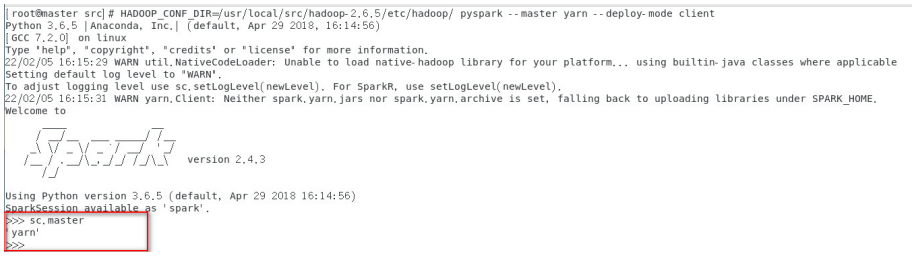


图 6-16 pyspark 本地模式界面

上述命令参数含义如下表 6-1 所示。

表 6-1 命令参数含义

命令代码	含义
HADOOP_CONF_DIR=/usr/local/src/hadoop-2.6.5/etc/hadoop	设置 Hadoop 配置文件路径
pyspark	启动 pyspark 应用程序
--master yarn -deploy-mode client	指定 yarn-client 模式启动

同样，通过命令可以查看当前运行模式为 yarn。接下来我们可以打开 yarn 的页面查看该任务是否存在，在浏览器中输入如下地址并按【Enter】键进入。

```
master:8088        # yarn 默认端口为 8088
```

打开页面如下图 6-17 所示。

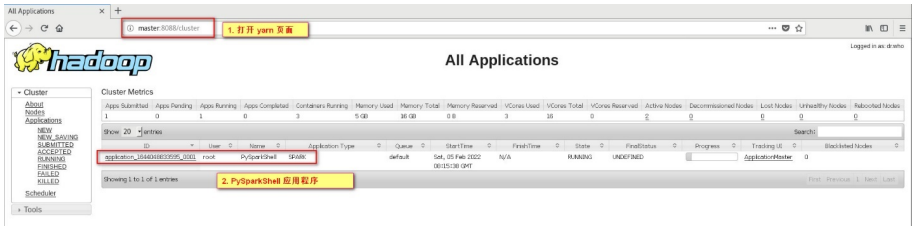


图 6-17 pyspark yarn 模式界面

6.3.4 Standalone 模式 Pyspark 交互界面

步骤 1：启动 Spark 集群

首先需要启动三台虚拟机，即 master、slave1 和 slave2，然后在 master 节点终端输入如下命令。

```
cd /usr/local/src/spark-2.4.3-bin-hadoop2.6/sbin
sh start-all.sh
```

集群的启动验证请参考 6.2 节。

步骤 2：启动 Pyspark Standalone 模式

启动 Pyspark Standalone 模式，启动命令如下。

```
pyspark  --master  spark://master:7077
```

启动界面如图 6-18 所示。

图 6-18 pyspark standalone 模式界面

我们可以通过 Spark 集群查看任务的信息，在浏览器中输入如下地址并按【Enter】键进入，如图 6-19 所示。

```
master:8080        # spark 集群默认端口为 8080
```

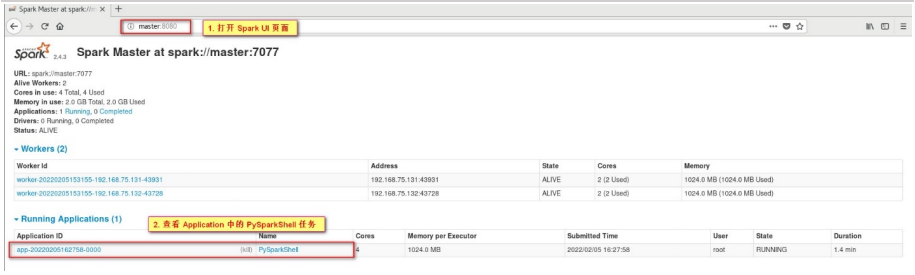


图 6-19 pyspark standalone web 界面

6.4 Jupyter Notebook PySpark 搭建

通过上一节的学习我们掌握了 Pyspark 各种模式的启动方法，在本节中我们将介绍 Pyspark Notebook 在不同模式下的启动方法，为后续开发工作奠定基础。

6.4.1 Local 模式运行 PySpark Notebook

本小节介绍 Local 模式的 PySpark Notebook 启动方法。

步骤 1：启动 Pyspark Notebook

在 master 节点终端输入如下命令启动 Local 模式的 Pyspark Notebook。

```
PYSPARK_DRIVER_PYTHON=ipython PYSPARK_DRIVER_PYTHON_OPTS="notebook" pyspark
```

在终端中可以看到启动过程中的打印日志，启动成功后，终端会打印出 Pyspark Notebook 的网址链接，复制链接到浏览器中就可以打开 Pyspark Notebook，如图 6-20 所示。

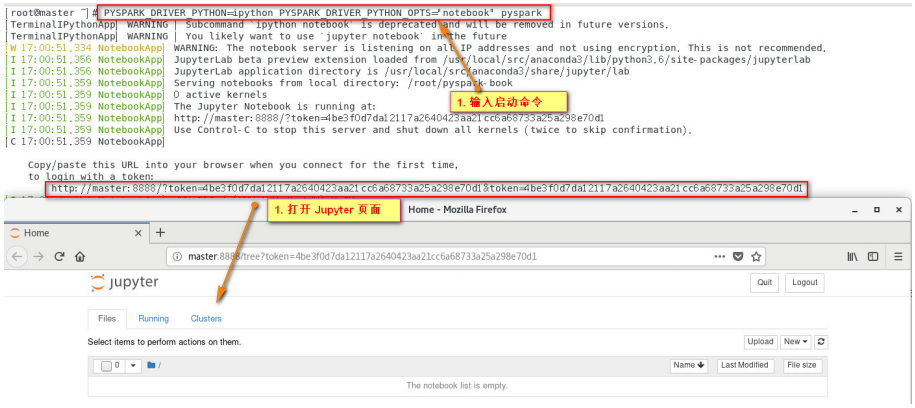


图 6-20 Loca 模式 Pyspark Notebook 页面

步骤 2：新建 Notebook 文件

点击右上角的【New】下拉菜单，为本章节新建一个 Notebook 文件，如图 6-21 所示。

图 6-21 新建 Notebook 文件

步骤 3：重命名 Notebook 文件

首先单击页面左上方的【Untitled】，在弹出的【Rename Notebook】对话框中输入“Demo”，最后单击【Rename】按钮，为 Notebook 重命名，如下图 6-22 所示。

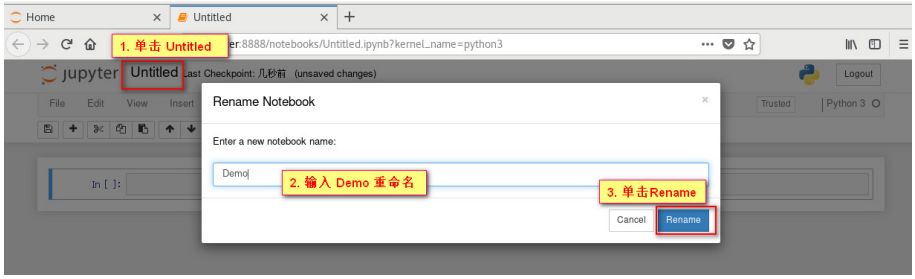


图 6-22 重命名 Notebook 文件

步骤 4：查看新建 Notebook 文件

在刚才打开的 Jupyter 首页可以看到我们新建的 Notebook 文件，如图 6-23 所示。

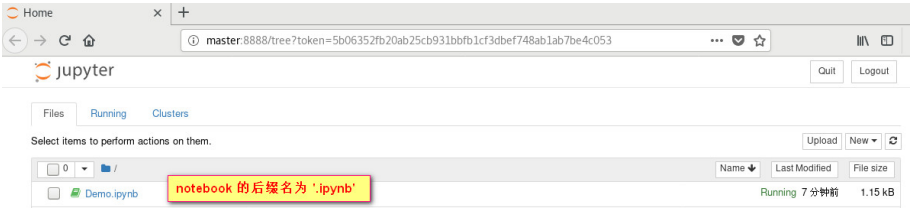


图 6-23 查看 Notebook 文件

步骤 5：查看 Pyspark 运行模式

同样，类似上一节中的命令，我们可以在 Notebook 中通过命令查看当前程序的运行模式，如下图 6-24 所示。

图 6-24 查看 Pyspark 运行模式

6.4.2 Yarn 模式运行 PySpark Notebook

接下来本小节介绍 Yarn 模式的 PySpark Notebook 启动方法。

步骤 1：启动 Pyspark Notebook

在 master 节点终端输入如下命令启动 Local 模式的 Pyspark Notebook，请注意这是一行代码，由于书面打印限制分列在了两行。

```
PYSPARK_DRIVER_PYTHON=ipython  PYSPARK_DRIVER_PYTHON_OPTS="notebook"
HADOOP_CONF_DIR=/usr/local/src/hadoop-2.6.5/etc/hadoop  MASTER=yarn-client  pyspark
```

在终端中可以看到启动过程中的打印日志中的网址链接，复制链接到浏览器中打开 Pyspark Notebook，并同样新建 Notebook 文件，如图 6-25 所示。

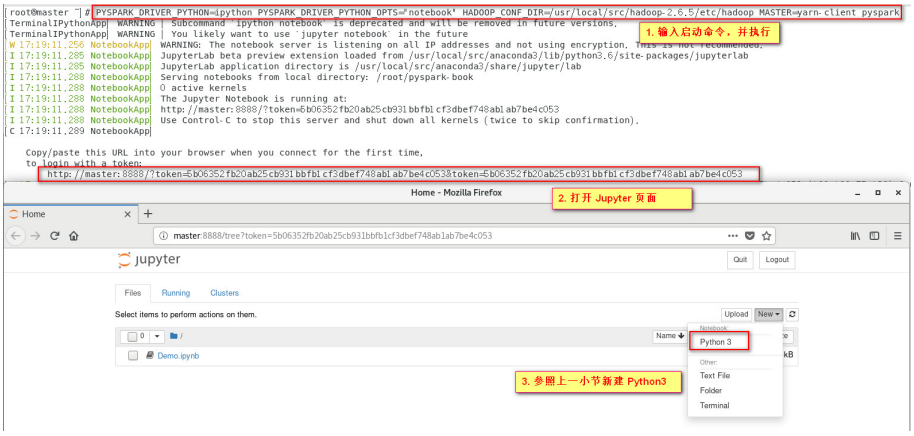


图 6-25 Yarn 模式 Pyspark Notebook 页面

步骤 2：查看 Pyspark 运行模式

同样，为 Notebook 文件进行重命名，并通过命令查看当前程序的运行模式，如下图 6-26 所示。

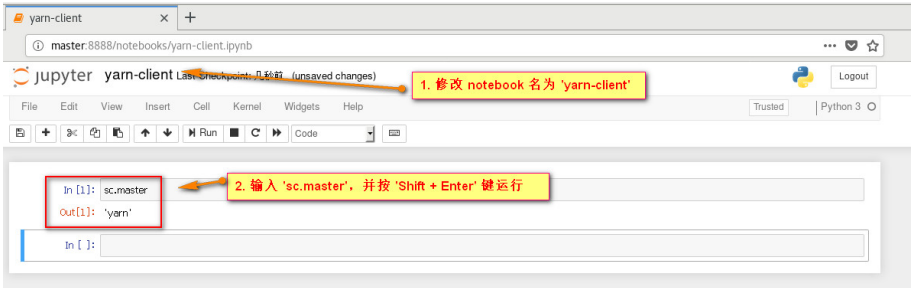


图 6-26 查看 Pyspark 运行模式

步骤 3：Yarn Web 查看 Pyspark 应用程序

通过 Yarn Web 页面同样可以查看该任务是否存在，如图 6-27 所示。

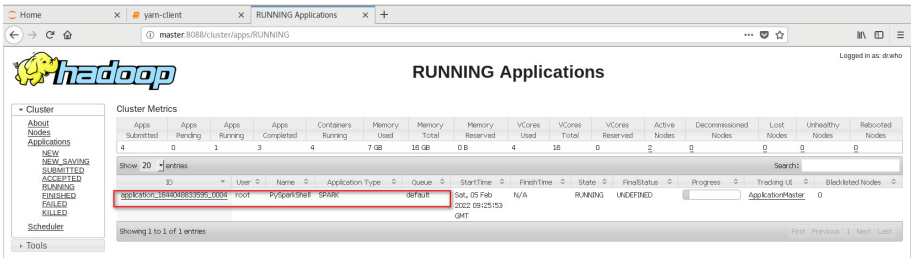


图 6-27 Yarn Web 查看 Pyspark 应用程序

最后，当我们需要停止 Pyspark Notebook 程序时，在启动 Pyspark Notebook 的终端中按【Ctrl+C】组合键可以关闭应用程序，如图 6-28 所示。

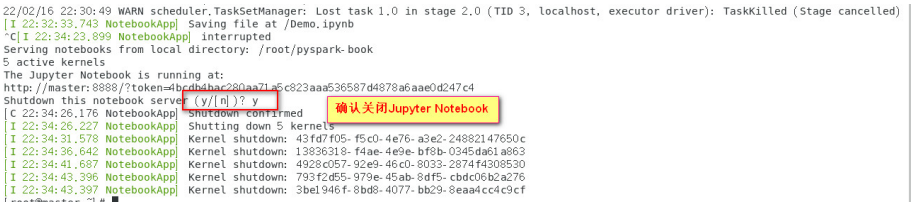


图 6-28 关闭 Pyspark Notebook 应用程序

6.5 PySpark Notebook 初体验

本节中，我们将介绍在 Pyspark Notebook 中如何实现与 Spark 的交互，并通过简单的 WordCount 代码程序完成体验 Pyspark 的简单、高效的大数据分析。

6.5.1 PySpark 在 Jupyter Notebook 中的交互

步骤 1：启动 Local 模式 Pyspark Notebook，代码如下所示。

```
PYSPARK_DRIVER_PYTHON=ipython PYSPARK_DRIVER_PYTHON_OPTS="notebook" pyspark
```

启动具体方法如 6.4.1 小节所介绍。

步骤 2：打开 Demo.ipynb

打开在上一节中我们新建的 Demo.ipynb 文件，打开之后可以简单验证一下 Jupyter 内核是否加载完成，步骤如下图 6-29 所示。

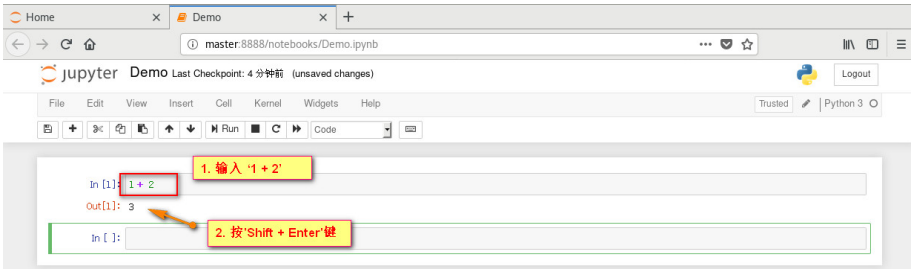


图 6-29 计算加法求和

步骤 3：读取 HDFS 文件

输入图 6-30 中的代码，并逐行运行，如图 6-30 所示。

图 6-30 读取 HDFS 文件

其中，第一行代码使用 Pyspark 函数实现了读取 HDFS 文件，结果返回给变量 book，第二行代码为使用 Pyspark 算子计算该文件的数据量。

6.5.2 Pyspark WordCount 实战

本小节将向读者介绍完整的 Pyspark WordCount 实战代码，代码及对应作用如图 6-31 所示。Pyspark 函数和算子的具体使用方法和含义在后续章节中会详解介绍。

这里主要包括 4 个步骤：

步骤 1：读取 HDFS 数据文件

```
book = sc.textFile("/data/The_DaVinci_Code.txt")
```

步骤 2：flatMap 打平数据

```
flatMap(lambda line : line.split(" "))      # 注意这里" "为一个空格字符串
```

步骤 3：map 映射

```
map(lambda word : (word, 1))
```

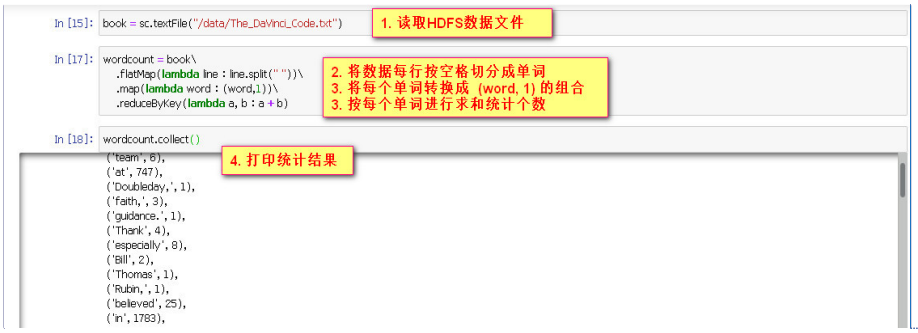
步骤 4：reduce 统计单词个数

```
reduceByKey(lambda a,b : a+b)
```

步骤 5：打印输出结果

```
wordcount.collect()
```

整体代码和运行结果如图 6-31 所示。



批注 [雷2]: 这个步骤图印到书上能看清吗？是否需要重新写在正文中？

批注 [HZ3R2]: 已经把代码核心内容在正文里也加上了，图片主要是展示输出结果。

图 6-31 Pyspark WordCount 代码

上述完整代码也可以参考本书配套本章节 Notebook 代码文件。

★回顾思考★-

01 Pyspark 交互界面的各个模式启动命令？

答：

（1）Local 模式

```
pyspark --master local[2]
```

（2）Yarn 模式

```
HADOOP_CONF_DIR=/usr/local/src/hadoop-2.6.5/etc/hadoop pyspark --master yarn --deploy-mode client
```

（3）Standalone 模式

```
pyspark --master spark://master:7077
```

02 Pyspark Notebook 各个模式启动命令？

（1）Local 模式

```
PYSPARK_DRIVER_PYTHON=ipython PYSPARK_DRIVER_PYTHON_OPTS="notebook" pyspark
```

（2）Yarn 模式

```
PYSPARK_DRIVER_PYTHON=ipython PYSPARK_DRIVER_PYTHON_OPTS="notebook"
HADOOP_CONF_DIR=/usr/local/src/hadoop-2.6.5/etc/hadoop MASTER=yarn-client pyspark
```

（3）Standalone 模式

```
PYSPARK_DRIVER_PYTHON=ipython PYSPARK_DRIVER_PYTHON_OPTS="notebook"
MASTER=spark://master:7077 pyspark
```

★练习★

一、选择题

1. 以下哪个参数可以指定 Pyspark 交互界面运行模式（ ）
   A. --master
   B. -master
   C. master
   D. --mode
2. 以下不是 Pyspark 交互界面本地模式命令的是（ ）
   A. pyspark --master local[6]
   B. pyspark
   C. pyspark --mode local[*]
   D. pyspark --master local[*]
3. 关闭 Jupyter Notebook 正确的方式是（ ）
   A. 按 Ctrl+Q
   B. 按 Ctrl+C
   C. 按 Ctrl+B
   D. 按 Ctrl+D

二、填空题

1. 启动 Pyspark 交互界面本地模式可以不指定参数。（√）
2. 启动 Pyspark 交互界面 Standalone 模式需要启动集群。（√）

三、实战练习

批注 [雷4]: 实战练习答案没有写

1. 使用命令启动不同模式的 Pyspark 交互界面。

（1）Local 模式

```
pyspark --master local[2]
```

（2）Yarn 模式

```
HADOOP_CONF_DIR=/usr/local/src/hadoop-2.6.5/etc/hadoop pyspark --master yarn --deploy-mode client
```

（3）Standalone 模式

```
pyspark --master spark://master:7077
```

2. 使用命令启动不同模式的 Pyspark Notebook 程序。

（1）Local 模式

```
PYSPARK_DRIVER_PYTHON=ipython PYSPARK_DRIVER_PYTHON_OPTS="notebook" pyspark
```

（2）Yarn 模式

```
PYSPARK_DRIVER_PYTHON=ipython PYSPARK_DRIVER_PYTHON_OPTS="notebook"
HADOOP_CONF_DIR=/usr/local/src/hadoop-2.6.5/etc/hadoop MASTER=yarn-client pyspark
```

（3）Standalone 模式

```
PYSPARK_DRIVER_PYTHON=ipython PYSPARK_DRIVER_PYTHON_OPTS="notebook"
MASTER=spark://master:7077 pyspark
```

3. 在 Jupyter Notebook 中独立实现 Pyspark WordCount 代码。
   参考本书配套本章节 Notebook 代码文件。

本章小结

本章详细介绍了 Spark 的下载与安装配置、如何搭建 Spark 集群，以及对 Spark 集群的监控方式进行了补充，启动 Spark 集群后，对集群健康状态的监控也是非常重要的环节，通过对 Spark 集群的搭建，能够更加熟悉 Spark 的运作方式。

同时介绍了 Pyspark 不同模式交互界面的启动以及 Pyspark Notebook 在不同模式下的启动方法。最后，通过 Pyspark WordCount 的代码实战，使读者更加深入的体验了 Pyspark 的简洁与强大以及了解了 Spark 在大数据分析处理中的重要地位。