|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 《大数据技术原理与应用》实验报告 | | | | |
| 题目： | 熟悉常用的HDFS操作 | 姓名 |  | 日期: |
| 实验环境：  操作系统：  PRETTY\_NAME="Ubuntu 22.04.1 LTS"  NAME="Ubuntu"  VERSION\_ID="22.04"  VERSION="22.04.1 LTS (Jammy Jellyfish)"  VERSION\_CODENAME=jammy  ID=ubuntu  ID\_LIKE=debian  HOME\_URL="https://www.ubuntu.com/"  SUPPORT\_URL="https://help.ubuntu.com/"  BUG\_REPORT\_URL="https://bugs.launchpad.net/ubuntu/"  PRIVACY\_POLICY\_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"  UBUNTU\_CODENAME=jammy  Hadoop 版本：3.3.6。  JDK 版本：1.8。  Java IDE：IntelliJ Idea Umltimate 远程开发） | | | | |
| 实验内容与完成情况：  一、基本命令练习  在HDFS中创建目录，  可以在Browse Directory页面浏览到，    级联创建，要加参数-p。    可以利用通配符进行模糊上传，    查看命令，dhfs dfs -h /    hdfs dfs -get命令进行下载，    hdfs dfs -touchz创建文件，    基本操作命令有：  以下是使用Hadoop提供的Shell命令完成你所描述的任务：  \*\*1. 向HDFS中上传文件：\*\*  ```shell  # 上传本地文件到HDFS中，如果文件已存在，则覆盖  hadoop fs -copyFromLocal /home/galaxy-chaos/Music/your\_local\_file.txt /data/Music/your\_hdfs\_file.txt  ```  \*\*2. 从HDFS中下载文件：\*\*  ```shell  # 从HDFS中下载文件到本地，如果本地文件名相同，将自动重命名  hadoop fs -copyToLocal /data/Music/your\_hdfs\_file.txt /home/galaxy-chaos/Music/  ```  \*\*3. 输出HDFS文件内容到终端：\*\*  ```shell  # 输出HDFS文件内容到终端  hadoop fs -cat /data/Music/your\_hdfs\_file.txt  ```  \*\*4. 显示HDFS文件信息：\*\*  ```shell  # 显示HDFS文件的权限、大小、创建时间、路径等信息  hadoop fs -ls /data/Music/your\_hdfs\_file.txt  ```  \*\*5. 显示HDFS目录下文件信息：\*\*  ```shell  # 显示HDFS目录下所有文件的权限、大小、创建时间、路径等信息  hadoop fs -ls -R /data/Music/  ```  \*\*6. 创建和删除文件：\*\*  ```shell  # 创建HDFS内的文件  hadoop fs -touchz /data/Music/your\_hdfs\_file.txt  # 删除HDFS内的文件  hadoop fs -rm /data/Music/your\_hdfs\_file.txt  ```  \*\*7. 创建和删除目录：\*\*  ```shell  # 创建HDFS目录，如果父目录不存在则自动创建  hadoop fs -mkdir -p /data/Music/your\_hdfs\_directory  # 删除HDFS目录，如果目录不为空，需要加上 -r 参数  hadoop fs -rm -r /data/Music/your\_hdfs\_directory  ```  \*\*8. 向HDFS中文件追加内容：\*\*  ```shell  # 向HDFS文件追加内容，追加到文件末尾  hadoop fs -appendToFile /local/path/to/append\_file /data/Music/your\_hdfs\_file.txt  ```  \*\*9. 删除HDFS中的文件：\*\*  ```shell  # 删除HDFS中指定文件  hadoop fs -rm /data/Music/your\_hdfs\_file.txt  ```  \*\*10. 删除HDFS中的目录：\*\*  ```shell  # 删除HDFS中指定目录，如果目录不为空，需要加上 -r 参数  hadoop fs -rm -r /data/Music/your\_hdfs\_directory  ```  \*\*11. 移动文件在HDFS中：\*\*  ```shell  # 在HDFS中将文件从源路径移动到目的路径  hadoop fs -mv /data/Music/source\_file.txt /data/Music/destination\_file.txt  ```  上述命令可以在Hadoop的Shell中使用，按照你的需求进行相应的操作。确保替换示例中的路径和文件名为实际的路径和文件名。  二、编程实现一个类“MyFSDataInputStream”  在用alias自定义好idea启动命令后，    可以通过支持x11协议的ssh工具来在其他主设备上直接调用服务器上的IntelliJ idea工具。    项目创建，Intellij中 File->New->Project 弹出的对话框中选择Maven ,Project SDK 选择1.8, Archetype中选择如图所示。    新建项目结构，    File->Settings 弹出对话框中将 将Target bytecode version 改为1.8，    添加依赖包，    编程实现一个类“MyFSDataInputStream”，该类继承“org.apache.hadoop.fs.FSDataInputStream”，要求如下：  （1）实现按行读取HDFS中指定文件的方法“readLine()”，如果读到文件末尾，则返回空，否则返回文件一行的文本。  先上传一个如下图所示的test.txt文本进行测试，  代码如下：  import org.apache.hadoop.conf.Configuration;  import org.apache.hadoop.fs.FSDataInputStream;  import org.apache.hadoop.fs.FileSystem;  import org.apache.hadoop.fs.Path;  import java.io.\*;  public class MyFSDataInputStream extends FSDataInputStream {  public MyFSDataInputStream(InputStream in) {  super(in);  }  /\*\*  \*实现按行读取  \*每次读入一个字符，遇到"n"结束，返回一行内容  \*/  public static String readline(BufferedReader br) throws IOException {  char[] data = new char[1024];  int read = -1;  int off = 0; //循环执行时，br每次会从上- -次读取结束的位置继续读取，因此该函数里，off每次都从0开始  while ( (read = br.read(data, off, 1)) != -1 ) {  if (String.valueOf(data[off]).equals("\n") ) {  off += 1;  break;  }  off += 1;  }  if (off> 0) {  return String.valueOf( data);  } else {  return null;  }  }  /\*\*  \*读取文件内容  \*/  public static void cat(Configuration conf, String remoteFilePath) throws IOException {  FileSystem fs = FileSystem.get(conf);  Path remotePath = new Path(remoteFilePath);  FSDataInputStream in = fs.open(remotePath);  BufferedReader br = new BufferedReader(new InputStreamReader(in));  String line = null;  while ( (line = MyFSDataInputStream.readline(br)) != null ) {  System.out.println(line);  }  br.close();  in.close();  fs.close();  }  /\*\*  \*主函数  \*/  public static void main(String[] args) {  Configuration conf = new Configuration();  conf.set(" fs.defaultFS", "hdfs://192.168.14.100:9000");  String remoteFilePath = "/data/text.txt"; // HDFS路径  try {  MyFSDataInputStream.cat(conf,remoteFilePath);  } catch (Exception e) {  e.printStackTrace();  }  }  }    运行结果如下：    （2）实现缓存功能，即利用“MyFSDataInputStream”读取若干字节数据时，首先查找缓存，如果缓存中有所需数据，则直接由缓存提供，否则向HDFS读取数据。  查看Java帮助手册或其它资料，用“java.net.URL”和“org.apache.hadoop.fs.FsURLStreamHandlerFactory”编程完成输出HDFS中指定文件的文本到终端中。  代码：  import org.apache.hadoop.fs.\*;  import org.apache.hadoop.io.IOUtils;  import java.io.\*;  import java.net.URL;  public class HDFS\_API {  static {  URL.setURLStreamHandlerFactory(new FsUrlStreamHandlerFactory());  }  /\*\*  \* 主函数  \*/  public static void main(String[] args) throws Exception {  String remoteFilePath = "hdfs://192.168.14.100:9000/data/text.txt"; // HDFS 文件  InputStream in = null;  try {  /\* 通过 URL 对象打开数据流，从中读取数据 \*/  in = new URL(remoteFilePath).openStream();  IOUtils.copyBytes(in, System.out, 4096, false);  } finally {  IOUtils.closeStream(in);  }  }  } | | | | |
| 出现的问题：  1.远程调试被显示出识别成本地路径。  2.尝试覆盖readLine()方法时出现错误，这是因为readLine()方法在java.io.DataInputStream中是final方法，不能被覆盖。为了实现按行读取的功能， | | | | |
| 解决方案（列出遇到的问题和解决办法，列出没有解决的问题）：  1.必须写上hdfs://localhost:9000部分，localhost配置为具体IP地址为宜。  2.可以考虑继承java.io.BufferedReader而不是java.io.DataInputStream，因为BufferedReader提供了方便的readLine()方法。 | | | | |