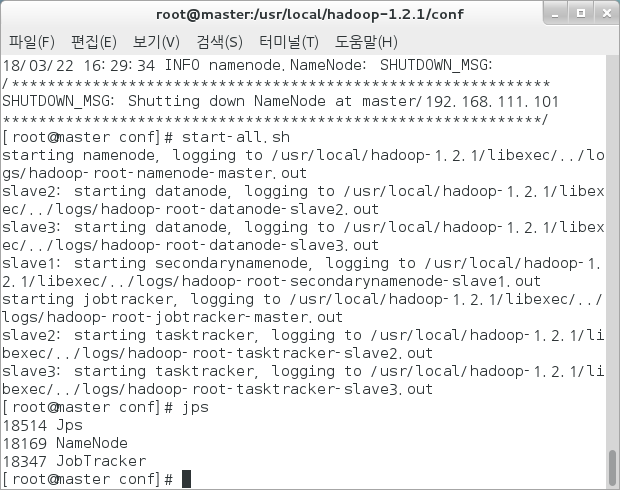
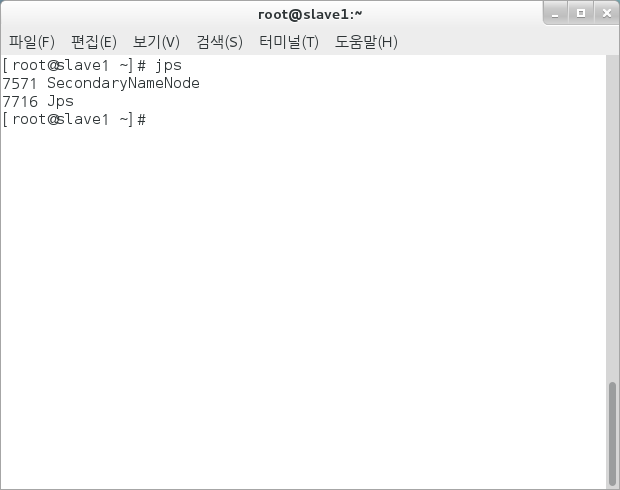
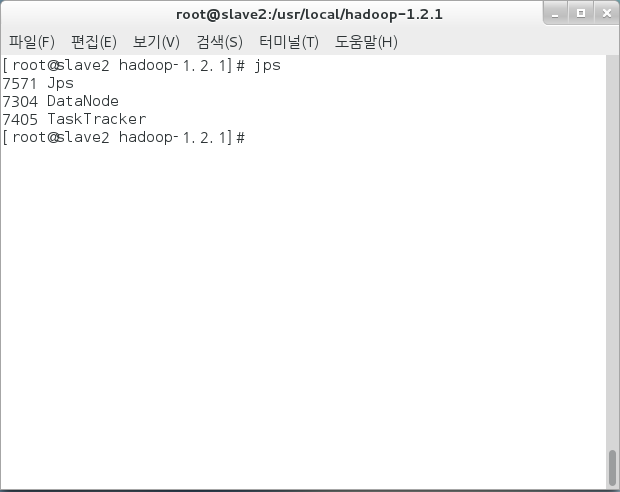
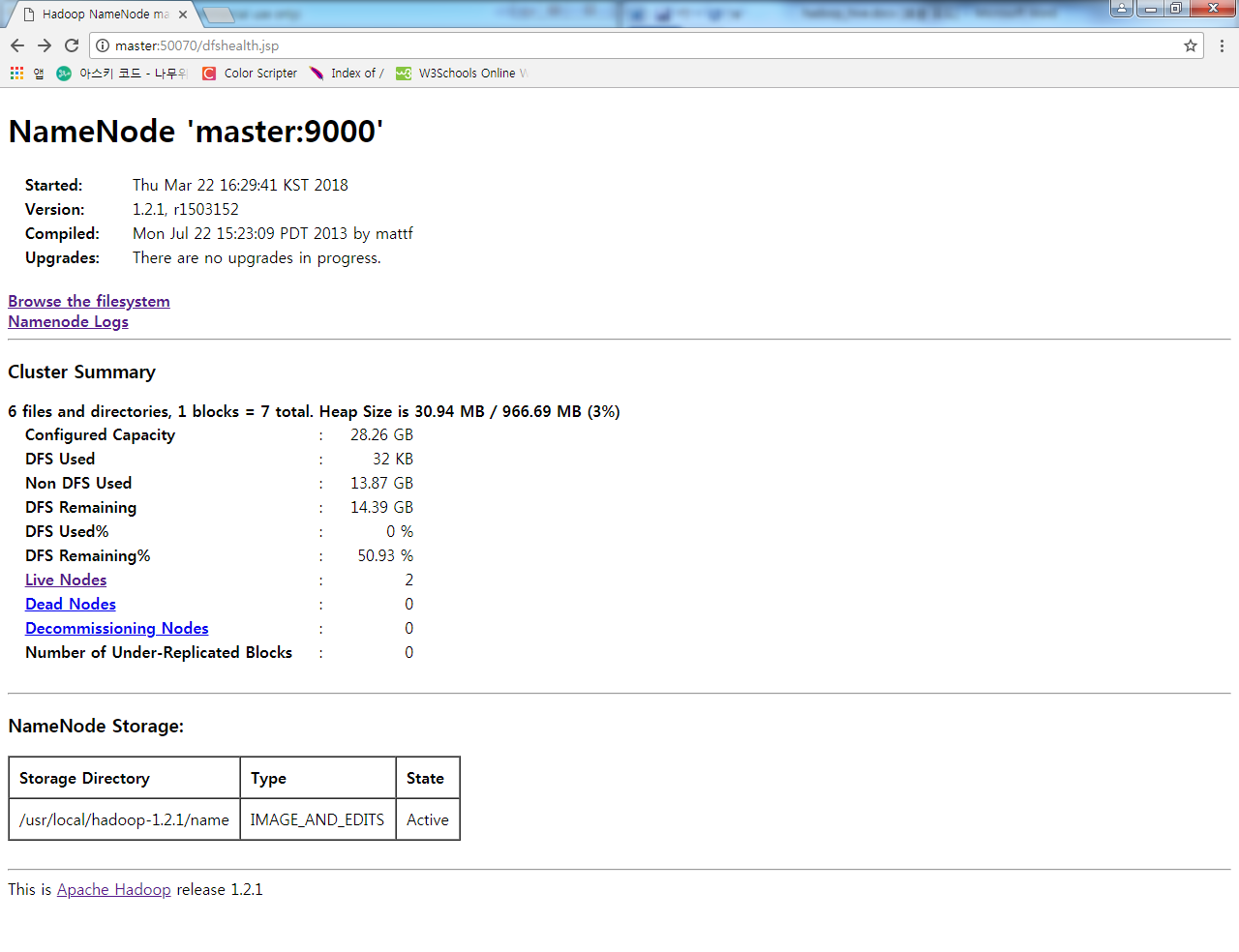
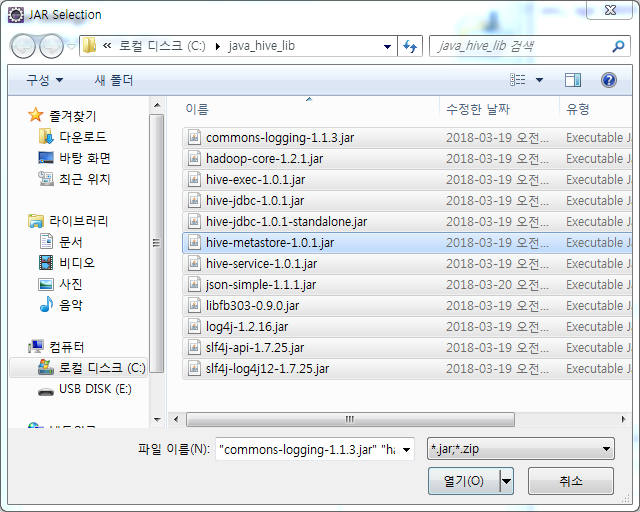
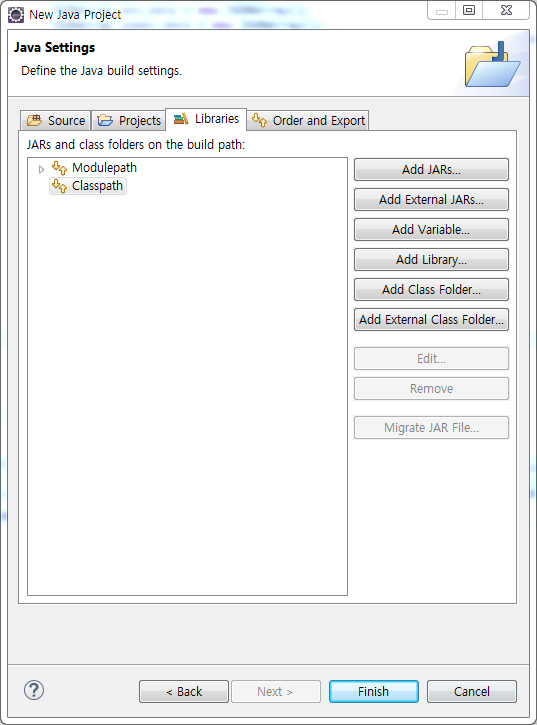
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **□ 수행평가 - 빅데이터 수집** | | | | | | |
|  |  |  | |  | |  |
| **과정명** | | 빅데이터 활용 Java 어플리케이션 개발자 양성 | | | | |
| **능력단위명** | | 빅데이터 수집 2001010502\_15v1 | | **훈련교사** | | 이진만 |
| **능력단위요소명** | | 데이터 수집 계획 수립하기,빅데이터 수집 시스템 구성하기 내부 데이터 수집하기,외부 데이터 수집하기 데이터 변환하기,수집 데이터 검증하기 | | | | |
| **과제개요** | | | | | | |
| 1. Hadoop을 설치 한다. 2. 설정 파일을 설정 한다. 3. Hadoop을 가동 한다. 4. Hadoop 관리화면 가동  5. Hive 설치  6. Java Application 연동 테스트 | | | | | | |
| **작성자** | | 백상우 | **작성일** | | 2018.03.23 | |

1. **Hadoop과 Hive를 설치하기 위한 준비 단계**
   1. **리눅스의 설치 및 네트워크 설정은 되어있다고 가정한다(CentOS-7.0.1406-x86)**
   2. **Hadoop과 Hive의 다운로드**<http://mirror.apache-kr.org/> 로 접속하여 Hadoop과 hive를 받는다.  
      문서에서 사용하는 버전은 hadoop-1.2.1과 hive-1.0.1이나 지금은 받을 수 없다.
   3. **JDK를 다운 받는다.**<http://www.oracle.com/technetwork/java/javase/downloads> 로 접속하여 jdk를 받는다.  
      문서에서 사용하는 버전은 8u161이다.
   4. **MariaDB를 다운 받는다.**<https://downloads.mariadb.org/mariadb/+releases/> 로 접속하여 MariaDB를 받는다.  
      필요한 파일은 client.rpm, common.rpm, server.rpm 세가지 이며 문서에서 사용하는 버전은 10.0.15 이다. 다음으로 <https://downloads.mariadb.org/connector-java>에 접속하여 jar 파일을 받는다. 문서에서 사용한 버전은 1.3.5 이다.
   5. **Hive jar 파일 받기**Java Application에서 Hive를 이용하기 위해 jar 파일을 받는다.  
      <https://docs.aws.amazon.com/ko_kr/emr/latest/ReleaseGuide/HiveJDBCDriver.html>
   6. **JDK 설치**터미널을 실행한 후 다운받은 파일이 있는 경로로 이동하여 JDK를 설치한다.  
      [root@master ~]# cd ~   
      [root@master ~]# tar xvfz jdk-8u161-linux-x64.tar.gz  
      [root@master ~]# mv jdk1.8.0\_161 /usr/local  
      [root@master ~]# cd /usr/bin[root@master bin]# ln -s /usr/local/jdk1.8.0\_161/bin/java java  
      [root@master bin]# java –version  
      java version "1.8.0\_161"  
      Java(TM) SE Runtime Environment (build 1.8.0\_161-b12)  
      Java HotSpot(TM) 64-Bit Server VM (build 25.161-b12, mixed mode)  
      [root@master bin]# vi /etc/profile  
      export JAVA\_HOME=/usr/local/jdk1.8.0\_161  
      export CLASS\_PATH=/usr/local/jdk1.8.0\_161/lib  
      export PATH=$JAVA\_HOME/bin:$PATH  
        
      수정한 profile을 실행한다.  
        
      [root@master bin]# source /etc/profile
   7. **MariaDB 설치**다운 받은 파일이 있는 경로로 이동하여 MariaDB를 설치한다.  
      [root@master bin]# cd ~  
      [root@master ~]# yum -y remove mariadb-libs  
      [rooot@master ~]# yum -y localinstall MariaDB-10.0.15-centos7\_0-x86\_64-\*  
      [rooot@master ~]# systemctl restart mysql  
      [rooot@master ~]# chkconfig mysql on
   8. **Firewall 끄기**사용할 포트를 방화벽에 등록하는 과정이 필요하나 지금은 방화벽을 끄고 진행한다.  
      [root@master ~]# systemctl stop firewalld  
      [root@master ~]# systemctl disable firewalld
   9. **HOST 파일 수정**하둡은 노드간에 SSH프로토콜을 이용해 통신한다. 호스트명을 정의하여 호스트 명으로 노드간에 통신을 할 수 있도록 설정한다.  
      [root@master ~]# vi /etc/hostname  
      master  
      [root@master ~]# vi /etc/hosts  
      192.168.111.101 master  
      192.168.111.102 slave1  
      192.168.111.103 slave2  
      192.168.111.104 slave3  
      [root@master ~]# systemctl restart network  
        
      세컨네임, 데이터노드의 설정에 맞춰 각각의 hostname과 hosts 파일 설정을 진행한다.
   10. **SSH 설정**이제 네임노드에서 데이터노드로 암호입력 없이 접근할 수 있도록 공개키를 만든다.  
       [root@master ~]# cd ~  
       [root@master ~]# ssh-keygen -t dsa -P '' -f ~/.ssh/id\_dsa  
       [root@master ~]# cd .ssh/  
       [root@master .ssh]# cat id\_dsa.pub >> authorized\_keys  
       [root@master .ssh]# ll  
       합계 12  
       -rw-r--r--. 1 root root 601 3월 22 14:34 authorized\_keys  
       -rw-------. 1 root root 668 3월 22 14:31 id\_dsa  
       -rw-r--r--. 1 root root 601 3월 22 14:31 id\_dsa.pub  
         
       생성된 공개키를 ssh copy를 통해 배포한다.   
         
       [root@master .ssh]# ssh-copy-id -i ~/.ssh/id\_dsa.pub root@slave1  
       The authenticity of host 'slave1 (192.168.111.102)' can't be established.  
       ECDSA key fingerprint is 59:4b:02:f8:5c:f6:aa:1e:7e:53:17:d2:e2:70:8e:b9.  
       Are you sure you want to continue connecting (yes/no)? yes  
       /usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed  
       /usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys  
       root@slave1's password: 111111  
         
       Number of key(s) added: 1  
         
       Now try logging into the machine, with: "ssh 'root@slave1'"  
       and check to make sure that only the key(s) you wanted were added.  
         
       공개키를 모든 데이터노드에 전송한다. 다음에 # ssh 데이터노드명 을 입력하여 비밀번호 입력없이 접속 되는지 확인하자.  
         
       [root@master ~]# ssh slave3  
       Last login: Thu Mar 22 15:00:59 2018  
       [root@slave3 ~]# exit  
       logout  
       Connection to slave3 closed.
2. **Hadoop 설치**
   1. **Hadoop 압축 파일 풀기**파일을 다운로드 한 경로로 이동하여 압축을 푼다.  
      [root@master ~]# cd ~  
      [root@master ~]# tar xvfz hadoop-1.2.1.tar.gz  
      [root@master ~]# mv hadoop-1.2.1 /usr/local
   2. **hadoop-env.sh 파일 수정**[root@master ~]# cd /usr/local/hadoop-1.2.1/conf  
      [root@master conf]# vi hadoop-env.sh  
        
      export JAVA\_HOME=/usr/local/jdk1.8.0\_161  
      export HADOOP\_HOME\_WARN\_SUPPRESS="TRUE"  
        
      export HADOOP\_PID\_DIR=/usr/local/hadoop-1.2.1/pids
   3. **masters, slaves 수정**[root@master conf]# vi masters  
      slave1  
        
      [root@master conf]# vi slaves  
      slave2  
      slave3
   4. **core-site.xml 수정**[root@master conf]# vi core-site.xml  
        
      <?xml version="1.0"?>  
      <?xml-stylesheet type="text/xsl" href="configuration.xsl"?>  
        
      <!-- Put site-specific property overrides in this file. -->  
        
      <configuration>  
       <property>  
       <name>fs.default.name</name>  
       <value>hdfs://master:9000</value>  
       </property>  
       <property>  
       <name>dfs.tmp.dir</name>  
       <value>/usr/local/hadoop-1.2.1/tmp</value>  
       </property>  
      </configuration>
   5. **hdfs-site.xml 수정**[root@master conf]# vi hdfs-site.xml  
        
      <?xml version="1.0"?>  
      <?xml-stylesheet type="text/xsl" href="configuration.xsl"?>  
        
      <!-- Put site-specific property overrides in this file. -->  
        
      <configuration>  
       <property>  
       <name>dfs.replication</name>  
       <value>2</value>  
       </property>  
      <property>  
       <name>dfs.http.address</name>  
       <value>master:50070</value>  
       </property>  
      <property>  
       <name>dfs.secondary.http.address</name>  
       <value>slave1:50090</value>  
       </property>   
      <property>  
       <name>dfs.name.dir</name>  
       <value>/usr/local/hadoop-1.2.1/name</value>  
       </property>  
       <property>  
       <name>dfs.data.dir</name>  
       <value>/usr/local/hadoop-1.2.1/data</value>  
       </property>  
       <property>  
       <name>dfs.webhdfs.enabled</name>  
       <value>true</value>  
       </property>  
      </configuration>
   6. **mapred-site.xml 수정**[root@master conf]# vi mapred-site.xml  
        
      <?xml version="1.0"?>  
      <?xml-stylesheet type="text/xsl" href="configuration.xsl"?>  
        
      <!-- Put site-specific property overrides in this file. -->  
        
      <configuration>  
       <property>  
       <name>mapred.job.tracker</name>  
       <value>master:9001</value>  
       </property>  
      </configuration>
   7. **/etc/procile 수정**[root@master ~]# vi /etc/profile  
        
      export JAVA\_HOME=/usr/local/jdk1.8.0\_161  
      export HADOOP\_HOME=/usr/local/hadoop-1.2.1  
      export CLASS\_PATH=/usr/local/jdk1.8.0\_161/lib  
      export PATH=$JAVA\_HOME/bin:$HADOOP\_HOME/bin:$PATH  
        
      [root@master ~]# source /etc/profile
   8. **Hadoop 환경설정 파일, /etc/profile 데이터 노드 전송**[root@master ~]# scp –r /usr/local/hadoop-1.2.1 root@slave1:/usr/local  
      [root@master ~]# scp –r /usr/local/hadoop-1.2.1 root@slave2:/usr/local  
      [root@master ~]# scp –r /usr/local/hadoop-1.2.1 root@slave3:/usr/local  
      [root@master ~]# scp /etc/profile root@slave1:/etc/profile  
      [root@master ~]# scp /etc/profile root@slave2:/etc/profile  
      [root@master ~]# scp /etc/profile root@slave3:/etc/profile  
      각 데이터 노드에서 source /etc/profile 을 실행해준다.
   9. **Hadoop-env.sh 및 namenode –format 실행**[root@master ~]# cd /usr/local/hadoop-1.2.1/conf  
      [root@master conf]# chmod ug+x hadoop-env.sh   
      [root@master conf]# ll hadoop-env.sh   
      -rwxrwxr--. 1 root root 2485 3월 22 15:26 hadoop-env.sh  
      [root@master conf]# . hadoop-env.sh  
      [root@master ~]# hadoop namenode –format
   10. **start-all.sh 로 Hadoop 실행(종료는 stop-all.sh)**[root@master ~]# start-all.sh  
       starting namenode, logging to /usr/local/hadoop-1.2.1/libexec/../logs/hadoop-root-namenode-master.out  
       slave3: starting datanode, logging to /usr/local/hadoop-1.2.1/libexec/../logs/hadoop-root-datanode-slave3.out  
       slave2: starting datanode, logging to /usr/local/hadoop-1.2.1/libexec/../logs/hadoop-root-datanode-slave2.out  
       The authenticity of host 'localhost (::1)' can't be established.  
       ECDSA key fingerprint is 59:4b:02:f8:5c:f6:aa:1e:7e:53:17:d2:e2:70:8e:b9.  
       Are you sure you want to continue connecting (yes/no)? yes
   11. **jps로 실행 되었는지 확인**각 노드에서 jps를 실행하면 아래와 같이 보여야 정상적으로 실행 된 것이다. ****  
         
       ****
3. **Hadoop 관리자 페이지 접속**
   1. **firefox에서 master:50070 을 입력하면 Hadoop화면이 보인다.**또는 namenode의 IP:50070을 입력하여서 접속할 수 있다.  
      만약 master:50070으로 접속할 수 없다면 hosts 파일을 확인해보자.  
      \* 윈도우OS는 C:\Windows\System32\drivers\etc\hosts 에서 확인 가능  
      
4. **Hive 설치를 위한 MariaDB 설정**
   1. **관리자 계정 설정 및 접속**[root @master conf]# mysqladmin -u root password '111111'  
      [root@master conf]# mysql -u root -p mysql  
      Enter password:111111
   2. **Hive 계정 생성 및 권한 할당**MariaDB [mysql]> grant all privileges on \*.\* to 'hive'@'localhost' identified by '111111';  
      MariaDB [mysql]> flush privileges;
   3. **hive\_db 생성 및 권한 할당**MariaDB [mysql]> create database hive\_db;  
      MariaDB [mysql]> grant all privileges on hive\_db.\* to 'hive'@'%' identified by '111111' with grant option;  
      MariaDB [mysql]> grant all privileges on hive\_db.\* to 'hive'@' localhost ' identified by '111111' with grant option;  
      MariaDB [mysql]> flush privileges;  
      MariaDB [mysql]> commit;
   4. **hive 계정으로 접속**[root@master conf]# mysql -u hive -p hive\_db  
      Enter password:111111  
      MariaDB [hive\_db]> show databases;
5. **Hive 설치**
   1. **Hive 압출 풀기**[root@master conf]# cd  
      [root@master ~]# tar xvfz apache-hive-1.0.1-bin.tar.gz  
      [root@master ~]# mv apache-hive-1.0.1-bin /usr/local/hive-1.0.1
   2. **MariaDB jar 파일 넣기**[root@master ~]# mv mariadb-java-client-1.3.5.jar /usr/local/hive-1.0.1/lib
   3. **/etc/profile 수정**[root@master ~]# vi /etc/profile  
      export JAVA\_HOME=/usr/local/jdk1.8.0\_161  
      export HADOOP\_HOME=/usr/local/hadoop-1.2.1  
      export HIVE\_HOME=/usr/local/hive-1.0.1  
      export CLASS\_PATH=/usr/local/jdk1.8.0\_161/lib  
      export PATH=$JAVA\_HOME/bin:$HADOOP\_HOME/bin:$HIVE\_HOME/bin:$PATH  
        
      [root@master ~]# source /etc/profile
   4. **hive-site.xml 파일 만들기**[root@master ~]# vi /usr/local/hive-1.0.1/conf/hive-site.xml  
        
      <?xml version="1.0"?>  
      <?xml-stylesheet type="text/xsl" href="configuration.xsl"?>  
      <configuration>  
       <property>  
       <name>hive.metastore.local</name>  
       <value>true</value>  
       <description>controls whether to connect to remove metastore server or open a new metastore server in Hive Client JVM</description>  
       </property>  
       <property>  
       <name>javax.jdo.option.ConnectionURL</name>  
       <value>jdbc:mariadb://localhost:3306/hive\_db?createDatabaseIfNotExist=true</value>  
       <description>JDBC connect string for a JDBC metastore</description>  
       </property>  
       <property>  
       <name>javax.jdo.option.ConnectionDriverName</name>  
       <value>org.mariadb.jdbc.Driver</value>  
       <description>Driver class name for a JDBC metastore</description>  
       </property>  
       <property>  
       <name>javax.jdo.option.ConnectionUserName</name>  
       <value>hive</value>  
       <description>username to use against metastore database</description>  
       </property>  
       <property>  
       <name>javax.jdo.option.ConnectionPassword</name>  
       <value>111111</value>  
       <description>password to use against metastore database</description>  
       </property>  
      </configuration>
   5. **Hive에 필요한 Hadoop 디렉토리 만들기**[root@master ~]# hadoop fs -mkdir /tmp/hive  
      [root@master ~]# hadoop fs -mkdir /user/hive/warehouse  
      [root@master ~]# hadoop fs -chmod 777 /tmp  
      [root@master ~]# hadoop fs -chmod 777 /tmp/hive  
      [root@master ~]# hadoop fs -chmod 777 /user  
      [root@master ~]# hadoop fs -chmod 777 /user/hive  
      [root@master ~]# hadoop fs -chmod 777 /user/hive/warehouse
   6. **Hive 실행**[root@master ~]# hive  
      18/03/22 17:32:55 WARN conf.HiveConf: DEPRECATED: Configuration property hive.metastore.local no longer has any effect. Make sure to provide a valid value for hive.metastore.uris if you are connecting to a remote metastore.  
      18/03/22 17:32:55 WARN conf.HiveConf: HiveConf of name hive.metastore.local does not exist  
        
      Logging initialized using configuration in jar:file:/usr/local/hive-1.0.1/lib/hive-common-1.0.1.jar!/hive-log4j.properties  
      hive>
6. **Hive에 데이터 올리기**
   1. **테스트를 위한 데이터 받기**<http://stat-computing.org/datasxpo/2009/>에 접속하여 2006~2008년 데이터를 받는다.  
      [root@master ~]# mkdir airline  
      [root@master ~]# mv 200\* airline  
      [root@master ~]# cd airline/  
      [root@master airline]# ll  
      합계 359500  
      -rwxrw-rw-. 1 root root 115019195 3월 15 16:43 2006.csv.bz2  
      -rwxrw-rw-. 1 root root 121249243 3월 15 16:42 2007.csv.bz2  
      -rwxrw-rw-. 1 root root 113753229 3월 15 16:33 2008.csv.bz2  
      [root@master airline]# bunzip2 200\*
   2. **데이터를 Hadoop에 올리기 위한 Hive에서 테이블 생성**[root@master ~]# hive  
        
      hive> CREATE TABLE airline\_delay(  
      Year INT,  
      MONTH INT,  
      DayofMonth INT,  
      DayofWeek INT,  
      DepTime INT,  
      CRSDepTime INT,  
      ArrTime INT,  
      CRSArrTime INT,  
      UniqueCarrier STRING,  
      FlightNum INT,  
      TailNum STRING,  
      ActualElapsedTime INT,  
      CRSElapsedTime INT,  
      AirTime INT,  
      ArrDelay INT,  
      DepDelay INT,  
      Origin STRING,  
      Dest STRING,  
      Distance INT,  
      TaxiIn INT,  
      TaxiOut INT,  
      Cancelled INT,  
      CancellationCode STRING  
      COMMENT 'A = carrier, B = weather, C = NAS, D = security',  
      Diverted INT COMMENT '1 = yes, 0 = no',  
      CarrierDelay STRING,  
      WeatherDelay STRING,  
      NASDelay STRING,  
      SecurityDelay STRING,  
      LateAircraftDelay STRING)  
      COMMENT 'TEST DATA'  
      PARTITIONED BY (delayYear INT)  
      ROW FORMAT DELIMITED  
      FIELDS TERMINATED BY ','  
      LINES TERMINATED BY '\n'  
      STORED AS TEXTFILE;
   3. **데이터 업로드**hive> LOAD DATA LOCAL INPATH '/root/airline/2006.csv’  
      OVERWRITE INTO TABLE airline\_delay  
      PARTITION (delayYear='2006');
   4. **관리자 페이지에서 확인**관리자 페이지로 접속하여 정상적으로 파일이 올라갔는지 확인한다.
   5. **Hive 서버 가동**[root@master ~]# hive --service hiveserver2
7. **Java Application 연동하기**
   1. **Java Project 만들기**새 Java Project를 만들기 -> Project 명 입력 -> next -> Libraries -> Classpath -> Add External JARs -> 받아 놓은 hive jar 파일 전부 추가  
      
   2. **Class 생성**

**package** hive\_test;

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.ResultSet;

**import** java.sql.Statement;

**import** org.json.simple.JSONArray;

**public** **class** hive\_test {

**public** **static** **void** main(String[] args) **throws** Exception {

Class.*forName*("org.apache.hive.jdbc.HiveDriver");

String sql = "SELECT year, month, COUNT(\*) FROM airline\_delay WHERE delayYear=2006 AND month IN (1, 2, 3, 4) AND ArrDelay > 0 GROUP BY year, month";

Connection conn = DriverManager.*getConnection*("jdbc:hive2://master:10000/default","root","111111");

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery(sql);

JSONArray ja = **new** JSONArray();

JSONArray data = **null**;

**while**(rs.next()) {

data = **new** JSONArray();

data.add(rs.getInt(2)+"월");

data.add(rs.getInt(3));

ja.add(data);

}

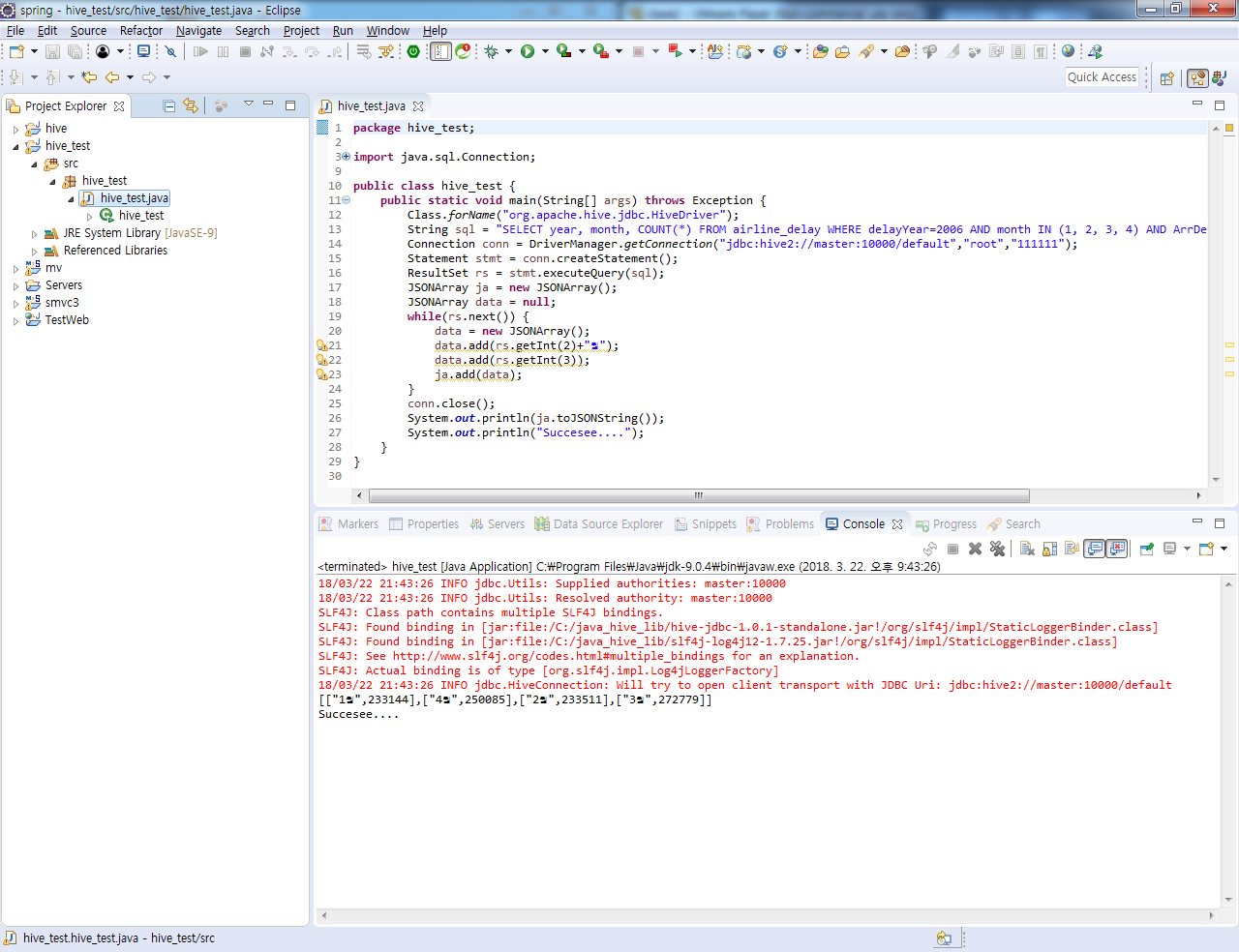
conn.close();

System.***out***.println(ja.toJSONString());

System.***out***.println("Succesee....");

}

}

****