|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **□ 수행평가 - 빅데이터를 활용한 IoT 시스템 개발(feat.커넥티드카)** | | | | | | |
|  |  |  | |  | |  |
| **과정명** | | 빅데이터를 활용한 IoT 시스템 개발(feat.커넥티드카) | | | | |
| **교과목명** | | 빅데이터저장및분석시스템구축기술 | | **훈련교사** | | 이진만 |
| **과정명** | | Linux OS 이해  빅데이터 수집시스템 개발  빅데이터 저장시스템 개발  빅데이터 분석시스템 개발 | | | | |
| **수행날짜** | | 2019.08.09 | 훈련생명 | | 김소희 | |
| **과제개요** | | | | | | |
| 1. Linux OS 를 설치 하고 빅데이터 시스템 환경을 구축 하시오 2. 빅데이터 수집 시스템을 구축 하시오 (log4j) 3. 빅데이터 저장 시스템을 구축 하시오 (HADOOP 설치) 4. 빅데이터 분석 시스템을 구축 하시오 (HIVE 설치) 5. 기존 Spring MVC와 빅데이터 시스템을 통합 하시오 | | | | | | |
| **최종 구조도**     1. **CentOS 설치 및 설정** 2. CMD에서   rundll32 "C:\Program Files (x86)\VMware\VMware Player\vmnetui.dll" VMNetUI\_ShowStandalone   1. cd /etc/yum. repos. d/ 2. gedit CentOS-Base. repo 3. gedit CentOS-Sources. repo 에서 release update를 지워준다. 4. mv CentOS-Base.repo CentOS-Base.repo.bak 5. wget <http://download.hanbit.co.kr/centos/7/CentOS-Base.repo> 6. rm \*.repo~ 7. yum clean all 8. vi /etc/sysconfig/network-scripts/ ifcfg-ens33 (수정)   BOOTPROTO=none  IPADDR=192.168.111.101 (컴퓨터 할당 ip로 수정)  NETMASK=255.255.255.0  GATEWAY=192.168.111.2  DNS1=192.168.111.2   1. systemctl restart network 2. hostnamectl set-hostname SERVER1 3. vi /etc/hosts (추가)   192.168.111.101 server1   1. vi /etc/sysconfig/selinux (수정)   SELINUX = disabled   1. **Log4j 사용** 2. web.xml에 추가   <listener>  <listenerclass>org.springframework.web.util.Log4jConfigListener</listener-class>  </listener>  <context-param>  <param-name>log4jConfigLocation</param-name>  <param-value>/WEB-INF/config/log4j.properties</param-value>  </context-param>     1. WEB-INF/config에 log4j.properties 파일 추가   log4j.logger.user = DEBUG, console, user  log4j.logger.work = DEBUG, console, work  log4j.logger.data = DEBUG, console, data  log4j.logger.product = DEBUG, console, product  # Product  log4j.appender.product.Threadhold=DEBUG  log4j.appender.product = org.apache.log4j.DailyRollingFileAppender  log4j.appender.product.DatePattern = '.'yyyy-MM-dd  log4j.appender.product.layout = org.apache.log4j.PatternLayout  log4j.appender.product.layout.ConversionPattern = %d{yyyy},%d{MM},%d{dd},%d{HH},%d{mm},%d{ss},%X{customer\_id},%X{product\_no} %n  log4j.appender.product.File = /root/glogs/product.log   1. src/mini.log에 Loggers.java 파일 추가   package mini.log;  import javax.servlet.http.HttpSession;  import org.apache.log4j.Logger;  import org.aspectj.lang.JoinPoint;  import org.aspectj.lang.annotation.After;  import org.aspectj.lang.annotation.Aspect;  import org.springframework.stereotype.Service;  import org.springframework.web.context.request.RequestContextHolder;  import org.springframework.web.context.request.ServletRequestAttributes;  @Service  @Aspect  public class Loggers {  private Logger work\_log = Logger.getLogger("work");  private Logger user\_log = Logger.getLogger("user");  private Logger data\_log = Logger.getLogger("data");  private Logger product\_log = Logger.getLogger("product");  // before  @After("execution(\* mini.controller..ProductController.\*(..))")  public void logging(JoinPoint jp) {  product\_log.debug(jp.getSignature().getName());  Object[] args = jp.getArgs();  }  }  → product log 파일 생성됨   1. **HADOOP 설치** 2. mariadb 다운 (책 참고) 3. hadoop 다운 ([https://archive.apache.org/dist/hadoop/common/hadoop-1.2.1/[hadoop-1.2.1.tar.gz](http://apache.mirror.cdnetworks.com/hadoop/core/hadoop-1.2.1/hadoop-1.2.1.tar.gz)](https://archive.apache.org/dist/hadoop/common/hadoop-1.2.1/%5Bhadoop-1.2.1.tar.gz%5D(http:/apache.mirror.cdnetworks.com/hadoop/core/hadoop-1.2.1/hadoop-1.2.1.tar.gz)))   cp -r hadoop-1.2.1 /etc/   1. vi /etc/profile   TOMCAT\_HOME=/etc/tomcat JAVA\_HOME=/etc/jdk1.8 HADOOP\_HOME=/etc/hadoop-1.2.1 CLASSPATH=/etc/jdk1.8/lib export CLASSPATH JAVA\_HOME TOMCAT\_HOME HADOOP\_HOME PATH=.:$JAVA\_HOME/bin:$TOMCAT\_HOME/bin:$HADOOP\_HOME/bin:$PATH   1. ssh   자기 자신 서버에 들어갈 때도 밖으로 나갔다가 다시 들어오게끔 하는 방법  -> 서버와 서버끼리 자유롭게 왕래하는 데에 있어서 걸림돌이 되므로 환경설정을 바꿔줘야함  ssh-keygen -t dsa -P '' -f ~/.ssh/id\_dsa  cd .ssh/ (home에서)  cat id\_dsa.pub >> authorized\_keys 배포키를 만들어줌   1. systemctl disable firewalld   systemctl stop firewalld   1. vi /etc/hadoop-1.2.1/conf/core-site.xml (추가)   <configuration>  <property>  <name>fs.default.name</name>  <value>hdfs://localhost:9000</value>  </property>  <property>  <name>dfs.tmp.dir</name>  <value>/etc/hadoop-1.2.1/tmp</value>  </property>  </configuration>   1. vi /etc/hadoop-1.2.1/conf/hdfs-site.xml (추가)   <configuration>  <property>  <name>dfs.replication</name>  <value>1</value>  </property>  <property>  <name>dfs.name.dir</name>  <value>/etc/hadoop-1.2.1/name</value>  </property>  <property>  <name>dfs.data.dir</name>  <value>/etc/hadoop-1.2.1/data</value> </property>  <property>  <name>dfs.webhdfs.enabled</name>  <value>true</value>  </property>  </configuration>   1. vi /etc/hadoop-1.2.1/conf/mapred-site.xml (추가)   <configuration>  <property>  <name>mapred.job.tracker</name>  <value>localhost:9001</value>  </property>  </configuration>   1. vi /etc/hadoop-1.2.1/conf/hadoop-env.sh (수정, 추가)   export JAVA\_HOME=/etc/jdk1.8  export HADOOP\_HOME\_WARN\_SUPPRESS="TRUE"   1. vi /etc/bashrc (추가)   . /etc/hadoop-1.2.1/conf/hadoop-env.sh  reboot   1. hadoop namenode -format   name이라는 디렉토리 생김   1. start-all.sh   data라는 디렉토리 생김  . ./hadoop-env.sh  끝낼 때에는 stop-all.sh 꼭 해야함   * **여러 대를 묶기**  1. Hadoop23에 ssh 부여하기 2. scp authorized\_keys [root@hadoop4:~/.ssh/authorized\_keys](mailto:root@hadoop4:~/.ssh/authorized_keys) 3. vi masters (수정)   hadoop2   1. vi slaves (수정)   hadoop2  hadoop3   1. systemctl disable firewalld   systemctl stop firewalld   1. vi /etc/hadoop-1.2.1/conf/mapred-site.xml (추가)   <configuration>  <property>  <name>mapred.job.tracker</name>  <value>70.12.114.206:9001</value>  </property> </configuration>   1. vi /etc/hadoop-1.2.1/conf/hdfs-site.xml (추가)   <configuration>  <property>  <name>dfs.permissions</name>  <value>false</value>  </property>  <property>  <name>dfs.replication</name>  <value>2</value>  </property>  <property>  <name>dfs.http.address</name>  <value>70.12.114.206:50070</value>  </property>  <property>  <name>dfs.secondary.http.address</name> <value>70.12.114.205:50090</value>  </property>  <property>  <name>dfs.name.dir</name>  <value>/etc/hadoop-1.2.1/name</value> </property>  8. vi /etc/hadoop-1.2.1/conf/core-site.xml (추가)  <configuration>  <property>  <name>fs.default.name</name>  <value>hdfs://70.12.114.206:9000</value>  </property>  <property>  <name>hadoop.tmp.dir</name>  <value>/etc/hadoop-1.2.1/tmp</value>  </property> </configuration>   1. 수정된 내용들 tar로 묶은 후 각 서버로 보내주기   [root@hadoop1 etc]# tar cvfz hadoop.tar.gz hadoop-1.2.1/  [root@hadoop1 etc]# scp hadoop.tar.gz root@hadoop4:/etc   1. 각 서버에서 tar파일 풀어줌   ssh root@hadoop4 "cd /etc;tar xvfz hadoop.tar.gz;rm -rf hadoop.tar.gz"   1. vi /etc/hadoop-1.2.1/conf/hadoop-env.sh (수정, 추가)   export JAVA\_HOME=/etc/jdk1.8  export HADOOP\_HOME\_WARN\_SUPPRESS="TRUE"   1. vi /etc/bashrc (추가)   . /etc/hadoop-1.2.1/conf/hadoop-env.sh   1. 각 서버로 bashrc와 profile 수정사항 보내줌   scp /etc/bashrc root@hadoop2:/etc  scp /etc/profile root@hadoop2:/etc   1. 각 서버 reboot 2. namenode에서 hadoop namenode –format 3. start-all.sh 4. **HIVE 설치** 5. use mysql 6. grant all privileges on *.* to 'hive'@'localhost' identified by '111111';   flush privileges;   1. create database hive\_db; grant all privileges on hive\_db.\* to 'hive'@'%' identified by '111111' with grant option; grant all privileges on hive\_db.\* to 'hive'@'localhost' identified by '111111' with grant option; flush privileges; commit; 2. exit 후mysql -u hive –p 비밀번호 입력, use hive\_유 3. 새로운 터미널에서tar xvf apache-hive-1.0.1-bin.tar.gz 4. mv apache-hive-1.0.1-bin hive 5. cp -r hive /etc 6. vi /etc/profile   export CLASSPATH JAVA\_HOME TOMCAT\_HOME HADOOP\_HOME HIVE\_HOME  PATH=.:$JAVA\_HOME/bin:$TOMCAT\_HOME/bin:$HADOOP\_HOME/bin:$HIVE\_HOME/bin:$PATH   1. reboot 후 start-all.sh 2. cp mariadb-java-client-2.4.2.jar /etc/hive/lib   -> error : maria jar 바꿈 (cp -r mariadb-java-client-1.3.5.jar /etc/hive/lib)  11. cd /etc/hive/conf   1. vi hive-site.xml (수정)   <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</namhae>  <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>  13. hadoop dfs -mkdir /tmp  hadoop dfs -mkdir /user/hive/warehouse  hadoop dfs -chmod g+w /tmp  hadoop dfs -chmod g+w /user/hive/warehouse  hadoop dfs -mkdir /tmp  hadoop dfs -chmod 777 /tmp  hadoop dfs -mkdir /tmp/hive  hadoop dfs -chmod 777 /tmp/hive  -> error : maria jar 바꿈 (cp -r mariadb-java-client-1.3.5.jar /etc/hive/lib)  14. hive -> 에러 (hadoop dfs -chmod 777 /tmp/hive)  hive -> 성공  15. hive DB에 HDI 테이블 만들기  hive> CREATE TABLE HDI(id INT, country STRING, hdi FLOAT, lifeex INT, mysch INT, eysch INT, gni INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE;  16. 내폴더에 hdi.txt 넣어준 후  hive>load data local inpath '/root/hdi.txt' into table HDI;  exit;  stop-all.sh   1. **통합** 2. hive --service hiveserver2 (linux에 치기, hive 밖에) 3. Java 파일에 연동하기 위해 코드 짜기   package hive;  import java.sql.Statement; import java.sql.Connection; import java.sql.DriverManager; import java.sql.ResultSet;  public class Hive {  public static void main(String[] args) throws Exception {  Class.forName("org.apache.hive.jdbc.HiveDriver");  Connection conn =  DriverManager.getConnection("jdbc:hive2://70.12.114.204:10000/default", "", "");  Statement stmt = conn.createStatement();  ResultSet rs = stmt.executeQuery("SELECT \* FROM airline\_delay where delayYear='2008' LIMIT 10");  while (rs.next()) {  System.out.println(rs.getString(5)); // n번재 자료 가져와라  }  conn.close();  System.out.println("Success....");  }   1. crongyul.sh shell script 작성   JAVA\_HOME=/etc/jdk1.8  export JAVA\_HOME  CLASSPATH=$JAVA\_HOME/lib  export CLASSPATH  TOMCAT\_HOME=/etc/tomcat  export TOMCAT\_HOME  HADOOP\_HOME=/etc/hadoop-1.2.1  export HADOOP\_HOME  HIVE\_HOME=/etc/hive  export HIVE\_HOME  PATH=.:$JAVA\_HOME/bin:$TOMCAT\_HOME/bin:$HADOOP\_HOME/bin:$HIVE\_HOME/bin:$PATH  fileName="product.log"  echo $fileName  if [ -f /root/glogs/$fileName ]  then  hive<<EOF  LOAD DATA LOCAL INPATH '/root/glogs/$fileName' OVERWRITE INTO TABLE PRODUCT\_VISIT;  EOF  echo "OK"  echo "OK"  else  echo "File Not Found"  fi  4. vi /etc/crontab (수정) -> 지속적으로 로그를 찍어 저장하기 위해  SHELL=/bin/bash  PATH=/sbin:/bin:/usr/sbin:/usr/bin  MAILTO=root  # For details see man 4 crontabs  # Example of job definition:  # .---------------- minute (0 - 59)  # | .------------- hour (0 - 23)  # | | .---------- day of month (1 - 31)  # | | | .------- month (1 - 12) OR jan,feb,mar,apr ...  # | | | | .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat  # | | | | |  \* \* \* \* \* root . /root/glogs/crongyul.sh  **최종 결과**    **UI 흐름도**    **REGISTER**     * **관리자로 로그인 했을 경우**   **PRODUCT REGISTER**    **PRODUCT LIST (UPDATE 및 DELETE 가능)**     * **CUSTOMER로 로그인 했을 경우**   **PRODUCT DETAIL**    **CART에 담을 경우**    **CART에서 ORDER 할 경우**    **PRODUCT DETAIL에서 ORDER할 경우**    **로그를 이용하여 상품 별 클릭 횟수 그래프로 나타낸 결과** | | | | | | |