**TDW安装**

TDW是腾讯基于hive和hadoop之上构建的数据仓库系统，在使用中，腾讯对hive做了大量定制，作为TDW的查询引擎（TDW QE）。这个项目将腾讯的hive分支以apache licence 2.0协议开源。

**环境依赖**

* CentOS-x86-64 6.x
* Oracle JDK 1.6.x
* Apache-ant 1.7.x及以上版本
* PostgreSQL 9.2.x及以上版本
* Hadoop 0.20.x版本
* Protobuf 2.3.0及以上版本
* hadoop-gpl-compression

**说明：**

**Centos**

所有软件以及环境依据此文档为准，并且需要用root超级用户来完成。不保证其他环境能成功

将所有防火墙关闭掉（service iptables stop）

vi 详解

http://wenku.baidu.com/link?url=lwNWeoLEG0UnzoMhdw7X2Vek0sNkIqAaRVk0Uh7CkB51yAbXXD8ZarQuIZjuKfGRtEoRXBwn5Yq5zF74EcA7SHIVs84ocI57Vpnid7ruFQ7

**编译**

**安装c/c++开发环境**

yum groupinstall "Development Tools"

yum install glibc-static wget unzip

**配置环境变量：**

vi ~/.bashrc

export JAVA\_HOME=/usr/local/java

export JRE\_HOME=/usr/local/java/jre

export ANT\_HOME=/usr/local/ant

export CLASSPATH=.:$JAVA\_HOME/lib:$JRE\_HOME/lib:$CLASSPATH

export PATH=$JAVA\_HOME/bin:$JRE\_HOME/bin:$ANT\_HOME/bin:$PATH

source ~/.bashrc

**安装java开发环境**

下载JDK 1.6.x和Apache-ant并安装，设置JAVA\_HOME和PATH环境变量，如添加以下到~/.bashrc中（这里将jdk和ant都安装在/usr/local中，请根据自己的情况替换相应的路径）

**JDK安装（根据环境不同有可能需要安装**yum install java-1.7.0-openjdk-devel

**）**

chmod +x dk-6u45-linux-x64.bin

./jdk-6u45-linux-x64.bin

mv jdk1.6.0\_45 /usr/local/java

**Apache-an安装**

ant官网：<http://ant.apache.org/>

tar -zxvf apache-ant-1.9.4-bin.tar.gz

mv apache-ant-1.9.4 /usr/local/ant

cd /usr/local/ant

ln -s /usr/local/ant/ bin/ant /usr/sbin/ant

输入 ant

出现

Buildfile: build.xml does not exist!

Build failed

这里面缺少build.xml文件，我们手动vi一个：

vi /usr/local/ant/bin/build.xml

vi /usr/local/ant/build.xml

|  |
| --- |
| <!--  Licensed to the Apache Software Foundation (ASF) under one  or more contributor license agreements. See the NOTICE file  distributed with this work for additional information  regarding copyright ownership. The ASF licenses this file  to you under the Apache License, Version 2.0 (the  "License"); you may not use this file except in compliance  with the License. You may obtain a copy of the License at  http://www.apache.org/licenses/LICENSE-2.0  Unless required by applicable law or agreed to in writing,  software distributed under the License is distributed on an  "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY  KIND, either express or implied. See the License for the  specific language governing permissions and limitations  under the License.  -->  <project xmlns:ivy="antlib:org.apache.ivy.ant" name="go-ivy" default="go">  <!--    this build file is a self contained project: it doesn't require anything else  that ant 1.6.2 or greater and java 1.4 or greater properly installed.    It is used to showcase how easy and straightforward it can be to use Ivy.    This is not an example of the best pratice to use in a project, especially  for the java source code "generation" :-) (see generate-src target)    To run copy this file in an empty directory, open a shell or a command window  in this directory and run "ant". It will download ivy and then use it to resolve  the dependency of the class which is itself "contained" in this build script.    After a successful build run "ant" again and you will see the build will be  much faster.    More information can be found at http://ant.apache.org/ivy/    -->  <!--  here is the version of ivy we will use. change this property to try a newer  version if you want  -->  <property name="ivy.install.version" value="2.0.0-beta1"/>  <property name="ivy.jar.dir" value="${basedir}/ivy"/>  <property name="ivy.jar.file" value="${ivy.jar.dir}/ivy.jar"/>  <property name="build.dir" value="build"/>  <property name="src.dir" value="src"/>  <target name="download-ivy" unless="skip.download">  <mkdir dir="${ivy.jar.dir}"/>  <!--  download Ivy from web site so that it can be used even without any special installation  -->  <echo message="installing ivy..."/>  <get src="http://repo1.maven.org/maven2/org/apache/ivy/ivy/${ivy.install.version}/ivy-${ivy.install.version}.jar" dest="${ivy.jar.file}" usetimestamp="true"/>  </target>  <!--  =================================  target: install-ivy  this target is not necessary if you put ivy.jar in your ant lib directory  if you already have ivy in your ant lib, you can simply remove this  target and the dependency the 'go' target has on it  =================================  -->  <target name="install-ivy" depends="download-ivy" description="--> install ivy">  <!--  try to load ivy here from local ivy dir, in case the user has not already dropped  it into ant's lib dir (note that the latter copy will always take precedence).  We will not fail as long as local lib dir exists (it may be empty) and  ivy is in at least one of ant's lib dir or the local lib dir.  -->  <path id="ivy.lib.path">  <fileset dir="${ivy.jar.dir}" includes="\*.jar"/>  </path>  <taskdef resource="org/apache/ivy/ant/antlib.xml" uri="antlib:org.apache.ivy.ant" classpathref="ivy.lib.path"/>  </target>  <!--  =================================  target: go  Go ivy, go!  =================================  -->  <target name="go" depends="install-ivy, generate-src" description="--> resolve dependencies, compile and run the project">  <echo message="using ivy to resolve commons-lang 2.1..."/>  <!--  here comes the magic line: asks ivy to resolve a dependency on  commons-lang 2.1 and to build an ant path with it from its cache  -->  <ivy:cachepath organisation="commons-lang" module="commons-lang" revision="2.1" pathid="lib.path.id" inline="true"/>  <echo message="compiling..."/>  <mkdir dir="${build.dir}"/>  <javac srcdir="${src.dir}" destdir="${build.dir}" classpathref="lib.path.id"/>  <echo>  We are now ready to execute our simple program with its dependency on commons-lang. Let's go!  </echo>  <java classname="example.Hello">  <classpath>  <path refid="lib.path.id"/>  <path location="${build.dir}"/>  </classpath>  </java>  </target>  <!--  =================================  target: generate-src  'Generates' the class source. It actually just echo a simple java  source code to a file. In real life this file would already be  present on your file system, and this target wouldn't be necessary.  =================================  -->  <target name="generate-src">  <mkdir dir="${src.dir}/example"/>  <echo file="${src.dir}/example/Hello.java">  package example; import org.apache.commons.lang.WordUtils; public class Hello { public static void main(String[] args) { String message = "hello ivy !"; System.out.println("standard message : " + message); System.out.println("capitalized by " + WordUtils.class.getName() + " : " + WordUtils.capitalizeFully(message)); } }  </echo>  </target>  <!--  =================================  target: clean  =================================  -->  <target name="clean" description="--> clean the project">  <delete includeemptydirs="true" quiet="true">  <fileset dir="${src.dir}"/>  <fileset dir="${build.dir}"/>  </delete>  </target>  <!--  =================================  target: clean-ivy  =================================  -->  <target name="clean-ivy" description="--> clean the ivy installation">  <delete dir="${ivy.jar.dir}"/>  </target>  <!--  =================================  target: clean-cache  =================================  -->  <target name="clean-cache" depends="install-ivy" description="--> clean the ivy cache">  <ivy:cleancache/>  </target>  </project> |

cd vi /usr/local/ant/bin

再次输入 ant



说明成功

检查java与ant环境及版本是否正确安装，运行如下命令检查：

[root@localhost ~]# ant -version

Apache Ant(TM) version 1.9.4 compiled on April 29 2014

[root@localhost ~]# javac -version

javac 1.6.0\_45

如果返回"-bash: xxx: command not found"，或者版本号低于TDW要求，请确认依赖软件是否安装正确，相应的环境变量是否设置生效。

**安装protobuf**

|  |
| --- |
| wget https://protobuf.googlecode.com/files/protobuf-2.3.0.zip  unzip protobuf-2.3.0.zip  cd protobuf-2.3.0  ./configure  make  make check  make install  protoc --version  可以看到版本号，说明安装成功。 |

**编译TDW QE二进制**

|  |  |
| --- | --- |
| 下载编译依赖的LZO压缩库：  wget http://hadoop-gpl-compression.googlecode.com/files/hadoop-gpl-compression-0.1.0-rc0.tar.gz  tar xf hadoop-gpl-compression-0.1.0-rc0.tar.gz  下载qe.zip到当前目录  https://github.com/amutu/tdw  unzip ./ tdw-master.zip  cd tdw-master  cp -a qe /usr/local/qe  cp /root/hadoop-gpl-compression-0.1.0/hadoop-gpl-compression-0.1.0.jar /usr/local/qe/lib/  进行编译：  注意：需要添加includeantruntime="on"，根据实际情况而定，修改位置有可能改动（需要cd /usr/local/qe && ant package时会提示）   |  | | --- | | [javac] /usr/local/qe/ant/build.xml:40: warning: 'includeantruntime' was not set, defaulting to build.sysclasspath=last; set to false for repeatable builds  [javac] /usr/local/qe/build-common.xml:154: warning: 'includeantruntime' was not set, defaulting to build.sysclasspath=last; set to false for repeatable builds  [javac] /usr/local/qe/jdbc/build.xml:51: warning: 'includeantruntime' was not set, defaulting to build.sysclasspath=last; set to false for repeatable builds  [javac] /usr/local/qe/metastore/build.xml:41: warning: 'includeantruntime' was not set, defaulting to build.sysclasspath=last; set to false for repeatable builds  [javac] /usr/local/qe/metastore/build.xml:50: warning: 'includeantruntime' was not set, defaulting to build.sysclasspath=last; set to false for repeatable builds  [javac] /usr/local/qe/ql/build.xml:122: warning: 'includeantruntime' was not set, defaulting to build.sysclasspath=last; set to false for repeatable builds  [javac] /usr/local/qe/recordio/build.xml:31: warning: 'includeantruntime' was not set, defaulting to build.sysclasspath=last; set to false for repeatable builds  [javac] /usr/local/qe/serde/build.xml:53: warning: 'includeantruntime' was not set, defaulting to build.sysclasspath=last; set to false for repeatable builds  [javac] /usr/local/qe/service/build.xml:51: warning: 'includeantruntime' was not set, defaulting to build.sysclasspath=last; set to false for repeatable builds  [javac] /usr/local/qe/shims/build.xml:42: warning: 'includeantruntime' was not set, defaulting to build.sysclasspath=last; set to false for repeatable builds  [javac] /usr/local/qe/contrib/src/java/org/apache/hadoop/hive/contrib/fileformat/build.xml:35: warning: 'includeantruntime' was not set, defaulting to build.sysclasspath=last; set to false for repeatable builds  [javac] /usr/local/qe/contrib/src/java/org/apache/hadoop/hive/contrib/fileformat/build.xml:36: warning: 'includeantruntime' was not set, defaulting to build.sysclasspath=last; set to false for repeatable builds |     cd /usr/local/qe && ant package |

看到



表示编译成功，生成的二进制包在qe/build/dist目录中，有以下几个目录：

* auxlib:运行时辅助lib
* bin：启动、重启脚本
* conf：配置文件所在的目录
* examples：SQL示例
* lib：库目录
* pl：过程语言库及脚本
* PLClient：交互式客户端

qe/build/dist目录用来打包部署生产环境。

编译后dist的代码命名为QE\_HOME环境变量~/.bashrc，后本QE\_HOME都指向这个目录：

|  |
| --- |
| export QE\_HOME=/usr/local/qe/build/dist |

**单机环境搭建**

我们快速搭建一个单机版的TDW QE环境，并运行简单的SQL进行体验。这个环境中，Hive，元数据，Hadoop都在一台机器上，并且HDFS和MapReduce都是本地运行模式，因此不宜处理较大数据量。

TDW QE安装包中自带了一个hadoop-0.20.0的包在qe/hadoopcore文件夹中，在单机环境中，我们使用这个hadoop在单机环境中运行SQL和MR。这时需要设置HADOOP\_HOME指向这个目录。

|  |
| --- |
| vi ~/.bashrc  export HADOOP\_HOME=/usr/local/qe/hadoopcore/hadoop-0.20.0  source ~/.bashrc |

**默认元数据配置**

这里补充一下，PostgresSQL安装（由于我的系统是最小化安装所以没有安装PostgreSQL）

|  |
| --- |
| yum install postgresql\*  yum localinstall http://yum.postgresql.org/9.3/redhat/rhel-6-x86\_64/pgdg-centos93-9.3-1.noarch.rpm  yum install postgresql93-server  初始化数据  service postgresql-9.3 initdb  service postgresql-9.3 start  chkconfig postgresql-9.3 on  进入postgres用户  su postgres  验证启动成功  psql -U postgres  退出  \q  回到root帐户  service postgresql-9.3 stop  注意需要修改配置文件：  vi /var/lib/pgsql/9.3/data/postgresql.conf  listen\_addresses = '\*'  service postgresql-9.3 start  修改postgres密码：  su postgres  psql -U postgres  ALTER USER postgres PASSWORD 'bjtest1234';  \q退出数据库  service postgresql-9.3 stop  vi /var/lib/pgsql/9.3/data/pg\_hba.conf  local all all md5  # IPv4 local connections:  host all all 127.0.0.1/32 md5  # IPv6 local connections:  host all all ::1/128 md5  service postgresql-9.3 start  重启数据库  进入root用户  psql -U postgres  是否提示输入密码。 |

首先在本机上要有PostgreSQL服务，使它监听127.0.0.1的5432端口（默认安装和初始化的PostgreSQL即监听127.0.0.1的5432端口），然后使用PG的管理员身份(一般是初始化PG数据库的linux账号,这里是postgres账户)，运行qe/script/tdw\_meta\_init.sql脚本，初始化元数据：

|  |
| --- |
| 在root用户提示下（注意：这里只需要执行一次psql -U postgres postgres -f /usr/local/qe/script/tdw\_meta\_init.sql）  cd /usr/local  chmod 777 qe/script/tdw\_meta\_init.sql  psql -h 127.0.0.1 -p 5432 -U postgres postgres -f script/tdw\_meta\_init.sql  提示输入密码 |

**启动Hive命令行（CLI模式）**

CLI模式区别于hvie server模式，在客户端进行SQL的解析和MR任务的提交。在开发和调试时，使用CLI模式比较方便。

启动hive的命令行： cd $QE\_HOME/bin && ./hive -u root -p tdwroot 进入如下命令行界面(TDW QE默认初始化账号root以及它的密码为tdwroot)：

|  |
| --- |
| [root@localhost data]# cd $QE\_HOME/bin && ./hive -u root -p tdwroot  session : root\_201411062206\_0.9663091272973413 start!  Connect to TDW successfully!  hive> |

然后就可以运行TDW SQL：

|  |
| --- |
| hive> create table tdw\_table(key int,value string);  OK  Time taken: 0.499 seconds  hive> show create table tdw\_table;  OK  CREATE TABLE tdw\_table(  key INT,  value STRING  );  Time taken: 0.149 seconds  hive> insert into tdw\_table values(1,'11'),(2,'22'),(3,'haha');  --...省略日志信息  hive> select \* from tdw\_table;  OK  1 11  2 22  3 haha  Time taken: 0.032 seconds  hive>quit; |

**启动Hive Server模式**

Hive Server模式是生产环境推荐的使用模式。它的启动方式是：

|  |
| --- |
| [root@localhost bin]# cd $QE\_HOME/bin && ./start-server.sh  [root@localhost bin]# nohup: 把输出追加到"nohup.out  回车 |

这个命令将hive server默认绑定在50000端口。如果希望指定端口，可以将端口号作为第一个参数传入，如：

|  |
| --- |
| [root@localhost bin]# cd $QE\_HOME/bin && ./start-server.sh 50001  [root@localhost bin]# nohup: 把输出追加到"nohup.out"  回车 |

然后可以通过PLClient进行连接。PLClient的使用方式，请参考下文。

**PLClient使用**

PLClient提供交互式的查询接口，一下为登陆和运行SQL的演示：

|  |  |
| --- | --- |
| [root@localhost bin]$ $QE\_HOME/bin/plclient/tdwpl  Python version is OK.  No arguments, we fall back to shell mode now ...  Entering shell mode now ...  Got config from cmd line: mode m connect to None:None@None:None  Username: root  Password: tdwroot  Use 'connect/c' to connect the server!  TDW/PL Client Version 0.2.3  Auto save config: False  Auto save passwd: False  User home library: '~/.tdwpl'  User: 'root'  Database name 'default\_db'  Server 'None:None' not connected  Welcome to TDW PL Shell, for help please input ? or help  root@TDW-PL$ server 127.0.0.1  You have changed server to '127.0.0.1'  root@TDW-PL$ port 50000  You have changed port to '50000'  root@TDW-PL$ c  connect to hive ip:127.0.0.1  Connect to server '127.0.0.1:50000' success.  root@TDW-PL$ new  New Session 6957574802551219 -1485586152  root@TDW-PL$ create table allison\_test(key int);  TDW-00000 SUCCEED ( session: 6957574802551219 query: create table allison\_test(key int) )  root@TDW-PL$ desc allison\_test  key int  TDW-00000 SUCCEED ( session: 6957574802551219 query: desc allison\_test )  root@TDW-PL$ help   |  | | --- | | Documented commands (type help <topic>):  ========================================  EOF color ed kill put sleep  a compile edit ls q start  attach config editor lsl quit status  autoed connect exec lsm reset test  autosaveconf d exit lsu run upload  c dbname get makejar save uploadm  cat detach gethistory n savepasswd user  ci discard getqp new server  cl disconnect getqueryplan passwd serverstatus  clusterstatus doc getschema plc sethome  co downloadm host port shows  Undocumented commands:  ======================  help | |

注：红色为重要命令