## Database Module:

Introduction:

In this part, we want to create the database, schemas, tables and some user defined functions. Please follow the steps below exactly. All the scripts mentioned in this module are saved in the folder “sql”.

1. Install db2 and db2 spatial extender, install universal fix packs.
2. Customize the createdb.sql for your case. Alter the name and path of database accordingly, our database named pais and we put our database in “/home/db2inst1/DB2/data/pais”, you can change them. Or just keep them for convenient and mkdirs /home/db2inst1/DB2/data/pais and /home/db2inst1/DB2/log/pais. A log directory also needs to be created. You need also modify the size of table spaces. In this script we set the size of MASTER table space to 5GB and SPATIALTBS32K table space to 15GB for test. You can change it as large as possible, like 20 and 100.
3. Create database using command (db2se enable included)

**$db2 -tf createdb.sql**

1. Restart database

**$db2stop force**

**$db2start**

1. connect to database using command

**$db2 connect to pais**

1. create tables of PAIS and PI using

**$db2 -tf table\_pais.sql**

**$db2 -tf table\_pidb.sql**

1. create stored procedures and user defined functions using

**$ db2 -td@ -f sp\_histogram.sql**

In some script error will occur such as no specified database, table or stored procedure is found. That’s normal because we want to drop the database, table or stored procedure before create them which means there always be a drop command in front of each create command. Just disregard this kind of error report.

## Data generating, preprocessing and uploading Module:

We assume that all data already be generated and preprocessed in our case, we need only upload them into database. All resources we need are saved in the folder “uploadtool/resource”. There are three types of resources for this module, the whole slide image which saved in folder wsi, the zipped boundary files saved in folder boundary and the files generated for image viewer which are saved in folder “DZImage”. I only prepared one image for your test. More images will be ready when this system works without errors on your system. Please follow the steps below.

1. Install openslide.

Openslide is necessary for our system. We use openslide to read the content of whole slide images. Therefore openslide should be installed before using any C or java programs. You can download it online or use the one I saved in folder. The newest version is 3.4.0, but I recommend 3.3.3 since some functions we used is deprecated in the new version. Firstly decompress the tar file, and enter the directory. Then run command with root authority: (some dependencies are necessary. Such as imagemagick, openjpeg, glib2, cairo and libxml etc. you can install them with yum command)

**$yum install openjpeg-devel.x86\_64**

**$yum install ImageMagick-devel.x86\_64**

**$yum install glib2-devel.x86\_64**

**$yum install cairo-devel.x86\_64**

**$yum install libxml2-devel.x86\_64**

**$./configure**

**$make**

**$make install**

Openslide installed to /usr/local/lib by default, please include this folder in LD\_LIBRARY\_PATH, add one line in /etc/profile

**export LD\_LIBRARY\_PATH=${LD\_LIBRARY\_PATH}:/usr/local/lib**

and then “source /etc/profile”

1. Compile and deploy openslidetools

Openslidetools is a C program we developed to manage whole slide images. It can read dimension of whole slide image, generate thumbnails for the whole image or one specified region, get the full resolution tile for any region, and generate tiles for the consumption of image viewer. This tool is necessary for both data uploading and web API. Therefore we need to compile it and put it into some where our system can reach. Please make sure that openslide already been installed before compiling.

You can find a source C file in folder “lib”, and compile it to the openslidetools and then move the compiled binary file into some where operating system can get from PATH. Otherwise our other program cannot run it. I also recommend /bin in which you need not to config the /etc/profile to add any export.

**$gcc -o openslidetools openslidetoolsbin.c -I/usr/local/include/openslide -L/usr/local/lib –lopenslide**

(the first –I is uppercase i, and the last –I is lowercase L )

**$mv openslidetools /bin**

run command “openslidetools” to make sure it already been compiled and moved correctly. One sentence “I’m ready” will appear if it is really ready. If you see:

***openslidetools: error while loading shared libraries: libopenslide.so.0: cannot open shared object file: No such file or directory***

you should define LD\_LIBRARY\_PATH in profile. Add this in file /etc/profile

**export LD\_LIBRARY\_PATH=${LD\_LIBRARY\_PATH}:/usr/lib:/lib:/usr/local/lib**

and then run command “source /etc/profile”.

1. Install JAVA

I’m sure JAVA already been installed in your system. Disregard this step if so.

1. download the newest jdk7, extract to /usr/local/java/jdk-version
2. in folder /usr/local/java run command to create a soft link

**$ln -s /usr/local/java/jdk-version /usr/local/java/jdk7**

1. edit file /etc/profile, add these two lines at the end of this file

**export JAVA\_HOME=/usr/local/java/jdk7**

**export PATH=${JAVA\_HOME}/bin: ${PATH}**

1. run command

**$source /etc/profile**

1. upload data

check list before upload data:

\*db2 is installed

\*db2 spatial extender is installed

\*universal fix pack is applied

\*db is created

\*pais and pi tables are created

\*stored procedures and user defined functions are created

\*openslidetools is compiled and can be executed using command “openslidetools” (double check before run any programs below, reboot or re login to make exports in /etc/profile count)

\*Java is installed

1. Modify the configuration files

All of the configuration files are saved in folder “uploadtool/config”, currently you need not to modify **paisidconfig.xml** and **loadconfig\_cell.xml**, you need only modify the database configuration files which are **paisdbc.xml** and **imagedbc.xml**. We use two different configuration files because the image uploader is implemented using hibernate, it need a hibernate style configuration file. It’s very simple to modify these two files. You need only change the hostname, port, database, username and password to your local or remote server.

1. Import the metadata of images into pi schema. Go into the folder “uploadtool”, and run command below.

**$java -jar imageloader.jar -dbc config/imagedbc.xml -pc config/paisidconfig.xml -i resource/wsi**

1. upload the boundary documents into database. run command

**$java -jar paistools.jar uploader -i resource/boundary -dbc config/paisdbc.xml**

1. load all uploaded boundaries into database. run command, I set argument t to 2 which means using 2 threads. This step is time consuming.

**$java -jar paistools.jar loadmanager -dbc config/paisdbc.xml -lc config/loadconfig\_cell.xml -t 2**

1. after uploaded all the images and boundary data, go to sql folder and generate histogram information and upload patient information. Switch to user db2inst1 and Run command:

**$db2 connect to pais**

**$db2 -tf gen\_histogram.sql**

**$db2 -tf load\_patient.sql**

## Web portal Module:

Make sure that all data have been uploaded into database correctly. Java and openslidetools are installed. Here we need to install the web server and deploy our web portal project and its API project. Please follow the steps below:

1. Install tomcat(root user)
2. download the newest version of tomcat and extract it to /usr/local/tomcat-version
3. in folder /usr/local create a softlink

**$ln -s /usr/local/tomcat-version /usr/local/tomcat7**

1. add the lines below at the end of /etc/profile, and “source /etc/profile” after adding

**export TOMCAT\_HOME=/usr/local/tomcat7**

**export PATH=${TOMCAT\_HOME}/bin: ${PATH}**

1. modify /usr/local/tomcat7/conf/server.xml, change the value of “appBase” in element “host” from “webapps” to “/var/www” or any other place for root project
2. try to start tomcat with command “startup.sh”
3. stop tomcat use “shutdown.sh”
4. install apache2(root user)
5. Some dependencies are needed before install httpd, please install these dependencies in order below. The source of apr, apr-util and pcre can be found in folder “dependency”. Decompress these compressed files, go into each of them and run command accordingly.

apr:

**$./configure --prefix=/usr/local/apr**

**$make**

**$make install**

apr-util:

**$./configure --prefix=/usr/local/apr-util --with-apr=/usr/local/apr**

**$make**

**$make install**

pcre:

**$./configure --prefix=/usr/local/pcre**

**$make**

**$make install**

1. download the newest version of apache2(httpd) or use the httpd-2.4.6.tar.gz in folder “dependency” and decompress it.
2. go to the extracted folder httpd-2.4.6 and install httpd with dependencies

**$./configure --with-apr=/usr/local/apr/ --with-apr-util=/usr/local/apr-util/ --with-pcre=/usr/local/pcre --prefix=/usr/local/apache2**

**$make**

**$make install**

1. same as above, add two lines in /etc/profile, and source /etc/profile

**export APACHE2\_HOME=/usr/local/apache2**

**export PATH=${APACHE2\_HOME}/bin:${PATH}**

1. run command using root user

**$apachectl start**

to run apache2. You’ll be warned could not reliable…… don’t worry, it will disappear after some configuration latter. You can see “it works” when you visit http://localhost

**$apachectl stop**

1. connect tomcat and apache2(root user)

Generally speaking, tomcat using 8080 port and this port cannot be accessed directly and it is hide by firewall. Apache2 uses port 80 which is opened to outer. Apache2 is in charge of static html pages and tomcat in charge of jsp pages and REST API. Therefore a connector is needed for apache2 to throw jsp pages and REST resource request to tomcat. please make sure tomcat and apache2 are installed.

1. download the newest tomcat apache2 connector, or use the tomcat-connectors-1.2.37-src.tar.gz in folder “dependency”, decompress
2. go into its subfolder “native” and install connector using command

**$./configure --with-apxs=/usr/local/apache2/bin/apxs**

**$make**

**$make install**

**You can see file “mod\_jk.so” in folder /usr/local/apache2/modules/**

1. Modify configuration files in folder /usr/local/apache2/conf
   1. add new file mod\_jk.conf with content:

**# where is the workers property file**

**JkWorkersFile /usr/local/apache2/conf/workers.properties**

**# Where to put jk logs**

**JkLogFile /usr/local/apache2/logs/mod\_jk.log**

**# Set the jk log level [debug/error/info]**

**JkLogLevel info**

**# Select the log format**

**JkLogStampFormat "[%a %b %d %H:%M:%S %Y]"**

**# JkOptions indicate to send SSL KEY SIZE,**

**JkOptions +ForwardKeySize +ForwardURICompat -ForwardDirectories**

**# JkRequestLogFormat set the request format**

**JkRequestLogFormat "%w %V %T"**

**# send all jsp and servlet request to tomcat via ajp13 protocol**

**JkMount /tcga/\* worker1**

**JkMount /\*.jsp worker1**

* 1. add new file workers.properties with content:

**worker.list=worker1**

**worker.worker1.type=ajp13**

**worker.worker1.host=localhost**

**worker.worker1.port=8009**

**worker.worker1.lbfactor=50**

**worker.worker1.cachesize=10**

**worker.worker1.cache\_timeout=600**

**worker.worker1.socket\_keepalive=1**

**worker.worker1.socket\_timeout=300**

* 1. modify httpd.conf: from up to down

change: **Listen 127.0.0.1:80**

add new lines to load mod\_jk.so：

**LoadModule jk\_module modules/mod\_jk.so**

**Include /usr/local/apache2/conf/mod\_jk.conf**

change: **ServerName LocalHost:80**

change: **DocumentRoot “/var/www”**

**<Directory “var/www”>**

add: **DirectoryIndex index.jsp** in element <IfModule dir\_module>(index.html already there)

1. deploy portal and API:
2. in folder war, there are two folder, portal is for web portal and tcga for its api. Go into tcga/WEB-INF/classes and modify the database configuration file: *connection.properties*, change the connect information inside.
3. copy portal and tcga folder in folder /var/www/
4. better to reboot system here.
5. start tomcat and apache2 using command

**$startup.sh**

**$apachectl start**

1. using web browser to visit localhost/portal to visit the portal
2. you can find a sample DZImage folder in *uploadtool/resource/*, create a link in */var/www* named *DZImage* pointing to it:

**$ln -s /path/to/DZImage/folder /var/www/DZImage**

you can also compile the c file in lib folder and use command “tilegen /path/to/input/folder /path/to/out/put/folder threadnumber level” to generate your own DZI file. And put the output folders into DZImage folder.

\*Notice:

\*when you switch to root user to start tomcat and apache2, please use command “su -” which will invoke the initial of /etc/profile instead of “su ”

\*when you add export in /etc/profile, make sure ${PATH} is always at the end of the export line. Therefore our new installed version of binary program will be called if more than one version is installed.

done!