Antelope 5.6 Tutorial for creating, managing, and utilizing databases

Antelope products have to be located in /opt/antelope

Help/man pages: All Antelope programs have man pages. $ANTELOPE/man/man3 contains the gory details of the programs and $ANTELOPE/man/man5 includes general documentation include dbdescriptor and dbexpressions, two useful man pages.

Table of Contents

For USArray Short-course: Many steps have been done in advance. As we work through the example red text indicates that you should type the command and blue text indicates a command done in advance. All example commands assume you have stayed in bash, the default shell in OSX, and I have therefore used a $ to indicate the command line. The $ is not part of the command you should type.

To set up access to Antelope

$source /opt/antelope/5.6/setup.sh

A. Creating a database

Example database is the New Madrid database combining seismic data from the Cooperative New Madrid Seismic Network and the Northern Embabyment Lithosphere Experiment

0. Change directory to our working directory for this example:
/Users/usarray/Modules/Wednesday/Database_Approach/NELE

```
$cd /Users/usarray/Modules/Wednesday/Database_Approach/NELE
```

1. Setup directory structure [Short-course: This has been done for you]

```
$mkdir data response pf dataless
```

#optional, not used here: db: where the database files that frequently change are located
#optional, not used here: dbmaster: where the metadata tables are located
#data: daily mseed data and log files
#pf: revised parameter files for antelope and passcal software
#response: response information for antelope
#dataless: dataless seed files

2. Download dataless seed volume from the DMC [Short-course: This has been done for you]
For this exercise, we are using data from the unresticted Cooperative New Madrid Seismic Network (FDSN code NM) and from the USArray Flexible Array NELE experiment (FDSN code ZL), which is resticted access through June 2017. Metadata, however, is not restricted, so we can still use breqfast or other IRIS tools (FetchMetaData, your favorite python script) to get the dataless seed volume. Additionally, I have already picked a day (2/7/2014) with a number of small New Madrid earthquakes and a few M6+ teleseisms.

Go to the IRIS webpage: http://ds.iris.edu/SeismiQuery/breq_fast.phtml
For network: NM,ZL
Lat/Lon: 38,34; -92,-88
Sample Rate: >=80, <=100
Dates: Feb 7 2014 00:00:00.0000 through Feb 7 2014 23:59:59.9999
Start Query –DO NOT REQUEST

Download the file via ftp: Feb7_NewMadrid.735326.dataless

```
$mv ~/Downloads/Feb7_NewMadrid.735326.dataless ./dataless/.
```

3. Setting up a database from dataless seed [Short-course: This has been done for you]
#seed2db creates the unchanged tables that the database will use including the site, calibration, network, sensor, instrument, etc. tables. Also creates a directory called response *with various sensor response files in it or uses the existing* response *directory to store response files*

```
$seed2db ./dataless/Feb7_NewMadrid.735326.dataless NewMadrid
```

Next, we need to create a file linking the schema to the table locations

```
$vim NewMadrid
        insert the following lines
        css3.0
        ./{NewMadrid}
```

If you have database tables in multiple locations, you can use this file to link those tables. Example:
./db/{NewMadrid}:./dbmaster/{NewMadrid}

4. Viewing the database tables:

This will bring up all the tables attached to the database and allow you to edit, join, view subsets, etc.
$dbe NewMadrid &

With the release of Antelope 5.6, BRTT changed the GUI and many capabilities of dbe. I personally still prefer the old dbe, which can be accessed as so:

$dbe_dep NewMadrid &

dbe tutorial: Follow along with the lecture and/or play with joining, separating, severing, and subset tables on your own.

5. Getting data from the DMC [Short-course: This has been done for you]

I used IRIS webservices scripts from the command line to get the unrestricted NM data (and didn't do anything fancy so that you can see the simplest example):

```
$FetchData -N NM -C EHE -s 2014-02-07T00:00:00 -e 2014-02-07T23:59:59 \
-o data/NM_EHE_0207.mseed

$FetchData -N NM -C EHN -s 2014-02-07T00:00:00 -e 2014-02-07T23:59:59 \
-o data/NM_EHN_0207.mseed

$FetchData -N NM -C EHZ -s 2014-02-07T00:00:00 -e 2014-02-07T23:59:59 \
-o data/NM_EHZ_0207.mseed
```

I used a bash script (scripts/GetData.sh) to show how to use webservices within a script and how to access restricted data. Note that the script I provide will not work because I have replaced by email and password with "fakeemail:fakepassword" and it writes to a non-existed Antelope database.

$vim GetData.sh

```
#!/bin/bash

days=(07)
months=(02)

stas=(A01 A01X A02 A03 A04 A04X A05 A05X A06 A07 A08 A08X A09 A10 A11 B01X B02 B03
B03X B04 B05 B06 B06X B07 B08 B09 B10 B11 B11X B12 B13 B14 B14X B15 C01 C01X
C01Y C02 C03 C04 C05 C06 C07 C08 C08X C09 C10 C11 C12 C13)

#NELE Stations
for s in ${stas[@]}; do
      for m in ${months[@]}; do
            for d in ${days[@]}; do
                  FetchData -v -S ${s} -N "ZL" -C "H*" -a fakeemail:fakepassword \
                    -lon -93:-86 -lat 33:40 -s 2014-${m}-${d},00:00:00.000000 \
                    -e 2014-${m}-${d},23:59:59.999999 -o data/ZL_${s}_${m}${d}.mseed \
            done
      done
done
```

6. Adding miniseed data to an existing database [Short-course: This has been done for you]

$miniseed2db data/*.mseed NewMadrid

7. Verifying the database [Short-course: We are going to skip this step]

$dbverify NewMadrid >& dbverify.out

$more dbverify.out

8. Completing the links between the wfdisc and the metadata

$dbfixchanids NewMadrid

You should see 3 stations that are in the site/sitechan tables but that have no data in the wfdisc.

$dbfix_calib NewMadrid

An unnecessary but interesting example is provide that also sets the calper column in the wfdisc by joining and modifying columns from the command line.

Join wfdisc and calibration tables and then set the wfdisc calper records to the same values as the calibration calper records. The –s flag with dbselect allows expressions to write to the database using the assignment operator ":=". Without the –s flag, dbselect would be used to print selected fields or expressions to a screen.

$dbjoin NVT.wfdisc calibration | dbselect -s - wfdisc.calper:=calibration.calper

8. Using dbe_dep to look at some basic data

The new dbe removes many of the useful graphics elements of the old dbe, so we will use the old dbe (dbe_dep) to look at a few things. Follow along with the lecture to look at instrument response and to look at filters. But first…..

From NELE directory
$cp .dbpickrc ~/.

.dbpickrc contains the filters and phases that show up when you use dbpick and dbloc2. Once you get used to the syntax, you have some control over the filters. Is it interactive and incredibly flexible, no.

$dbe_dep NewMadrid &

$dberesp –f ./response/Guralp_CMG3T:Reftek_130_Datalo.1

9. Uploading the wrong mseed volume and then removing those lines from the wfdisc
I purposefully uploaded a .mseed file that I did not want to use so that we could practice removing those lines from the wfdisc.

It is INCREDIBLY important that you check that the subset is only giving you want to you want to change/delete before you do anything else. It is also good to cp all the files to a backup directory when you know you are doing maintainence on the db. You really should avoid modifying tables by hand (vim, etc) or line by line within dbe, though that can be done by turning on the 'edit tables' option.

$dbsubset NewMadrid.wfdisc "dfile=='NMdata_EHN.mseed'" | dbselect – sta chan dir dfile

$dbsubset NewMadrid.wfdisc "dfile=='NMdata_EHN.mseed'" | dbdelete –

B. Event detection, association and location

1. Set up the parameter files [Short-course: This has been done for you]
The original files for these programs need to be copied initially but this stepped can then be skipped.

```
$cp $ANTELOPE/data/pf/dbdetect.pf pf
$cp $ANTELOPE/data/pf/dbgrassoc.pf pf
$cp $ANTELOPE/data/pf/ttgrid.pf pf
```

2. Create the travel time tables [Short-course: This has been done for you]
Change the ttgrid.pf file to reflect station spacing, network location, velocity model etc.

```
$ttgrid –pf pf/ttgrid.pf –time all NewMadrid > pf/newmadrid.ttgrid
```

If Antelope python is installed, you can use the following to view the ttgrid
```
$displayttgrid pf/newmadrid.ttgrid local
```

3. Run detection programs [Short-course: This has been done for you]
STA/LTA detecter with parameters to control filter

```
$dbdetect –pf pf/dbdetect.pf NewMadrid NewMadrid
```

4. Run event association and initial location program

Revise dbgrassoc.pf. Note that the velocity models named in the dbgrassoc file have to agree exactly with the names of the travel time models in the ttgrid file

Initial arrival associator and event locator
```
$dbgrassoc –pf pf/dbgrassoc.pf NewMadrid NewMadrid pf/newmadrid.ttgrid
```

Is this a good result with only good earthquakes and few false-positives?

You can clear the origin and associations made while practicing or before the database is complete. Delete the origin, assoc, origerror, and arrival files. The will be recreated when you rerun the associator programs. DO NOT REMOVE THE detection FILE!

Modify the dbgrassoc.pf file so that only the 4 known New Madrid local earthquakes end up "detected." Name your new file ./pf/dbgrassoc_yourname.pf

```
$dbgrassoc –pf pf/dbgrassoc_yourname.pf NewMadrid NewMadrid pf/newmadrid.ttgrid
```

Quick view of locations
```
$dbmapevents NewMadrid RDGT 5
```

5. Analyze local earthquakes, add S onsets, and calculate magnitudes in dbloc2

To view located events and relocate events; detections are located in the origin table. See notes below on how this program works.

```
$dbloc2 –p pf/dbloc2.pf NewMadrid
```

I have provided two additional 1D velocity models for New Madrid (dbgenloc->ttlvz/embyament, dbgenloc->ttlvz/enola).

After you decided which earthquake location is preferred, click the Magnitude button to have a true Richter ml calculated. You will have to modify the file pf/dbevproc.pf file and add your name to the allowed magnitude calculator list. This is the name that appears after the event location in the dbloc2 main window. Mine shows up as SMU:heather for example.

Information for dbloc2 and dbpick

- If two events show up on the same screen make sure when locating you are only using the arrivals for one event. Also make sure that each event is the right origin id-you may have to change the origin id for the second event by clicking on the origin id number and using the new event option.
- If you get an iterations error, change the max iterations bar. If this doesn't work, check that the assigned error bars are reasonable for each P and S pick.
- You want to click on the preferred location button for the best event. Save this event, chop the automatic, delete the trial events (unless you want to save them), and save the associated event if there is one.
- After picking arrivals for an event and relocating the event, click the next button to move on. This may go back to the new location you just picked so make sure you are picking arrivals for an event that has not been relocated and associated. Generally all this means is clicking on the next button twice rather than just once.
- Filters can be created or changed in the .dbpickrc (antelope home directory) or you can create your own filter file in the local directory.

MOUSE BUTTON	ACTION
Left-Click	Time Scroll to the left
Right-Click	Time scroll to the right
Middle-Drag	Time scroll left or right
Shift-Left-Click	Time zoom in
Shift-Right-Click	Time zoom out
Left-Click-on-Station-Label	Select/Deselect station
Left-Drag-across-Station-Label	Select/Deselect station
Left-Click-on-Arrival-Flag	Adjust arrival time
Shift-Left-Click-on-Arrival-Flag	Adjust arrival time uncertainty
Control-Shift-Left-Click-on-Arrival-Flag	Adjust arrival amplitude-period
Control-Left-Click-on-Arrival-Flag	Show arrival measurements
Middle-Click-on-Arrival-Flag	Phase code menu
Right-Click-on-Arrival-Flag	Arrival menu

C: Antelope and Matlab

Read Kent Linquist's pdf on Matlab and Antelope.

I have provided a couple of .m example scripts written by Blaine Buckholt that opens a database and plots waveforms at matlab/get_data.m. It has to be modified to work with the New Madrid database, however, as it was written to deal with time issues and is overly complicated for this tutorial.

1. To set up the Antelope/Matlab interface
matlab>run('/opt/antelope/5.6/data/matlab/2016a/antelope/scripts/setup.m')
matlab>help antelope %this is just to show you that help calls up the manual pages

2. Then it is simply about learning a slightly different syntax

```
>cd /Users/usarray/Modules/Wednesday/Database_Approach
>db=dbopen('./NewMadrid','r');
>db=dblookup_table(db,'sitechan');
>db=dbsubset(db, ['sta == " RDGT " && chan== " EHZ_00 " ']);
>dbw=dblookup_table(db,'wfdisc');
>db=dbjoin(db,dbw);
>tr=trload_css(db,starttime,endtime);   %Choose something that makes sense
        >tr=trload_css(db,'2014038:15:42:54','2014038:15:43:00');
>data=trextract_data(tr);
>plot(data,'r');
```

A Note On Time

Time is stored as seconds since January 1, 1970. This is known as epoch time. dbe translates epochtime to readable times, but for seismology data, epoch time is fantastic. The Antelope interfaces with Perl and Matlab (and others) contain a few useful commands to convert back and forth from epoch to readable (string) time.

Perl:
```
$string="$month/$day/$year $hr:$min:$sec";
$origintime=str2epoch($string);
$stringtime=epoch2str($otime,"%m/%d/%Y %H:%M:%S.%s")
$juliandaytime=epoch2str($otime,"%Y%j:%H:%M:%S.%s");
```

Matlab
```
string='month/day/year hr:min:sec';
origintime=str2epoch(string);
stringtime=epoch2str(otime,'%m/%d/%Y %H:%M:%S.%s')
juliandaytime=epoch2str(otime,'%Y%j:%H:%M:%S.%s')
```

Epoch has many options and time can be input in many formats. I just stick to a few so as to not get confused.

Month/Day/Year Hr:Min:Sec
YEARJULIANDAY:Hr:Min:Sec

Year -> %4d in printf format
JulianDay -> [001-0365] %3d
Month -> [01-12] %2d
Day -> [01-31] %2d

Hr -> [00-24] %2d
Min -> [00-59] %2d
Sec -> [00-59.99999] %6.3f

Appendices
[Not modified for the short-course. User beware]

A. Creating a batchfile

Short-course users – note that I have provided example batchfiles in a subdirectory so that you can see the range of formatting choices that are possible with batchfiles. I use batchfiles to create databases for new field projects for which I am responsible for providing full seed data to the DMC. For USArray experiments, IRIS created the seed files for PIs and handled all data archiving. For most projects, the PI is responsible for this activity and Antelope has been the standard software to use.

#Use dbbuild_batch to build the initial database tables describing metadata.

#das.latlons is a file storing DAS, station name, start time. Locations using the average GPS location found using logpeek on .log files and field notes from the installation and first set of recovered data.
vim das.latlons #or an other file/paperwork containing metadata
vim batchfile.NVT

#A note on location code .... Passcal documentation discourages the use of using a location code (00, 01, 02) for temporary data unless there will be a conflict with the channel names. Since the NVT experiment will have HH and LH channels (100 HZ and 1 HZ sampling rates), I chose not to use a loc code.

#What does a batchfile look like? It will be a series of station block information describing various features of the experiment

time 2009316 15:23:00.0 #Start time of the station
net Y8 New Madrid Nonvolcanic Tremor Array Experiment #FDSN Network Code[1]
sta S01 36.31898 -89.53264 0.08858 North One #Station name (short and long) and location. Elev in km
datalogger rt130 92C1 #Datalogger type and serial number[2]
sensor cmg40t 0 4779 #Sensor type, emplacement depth, and serial number[3]
samplerate 100sps
samplerate 1sps
add #THIS IS TERRIBLY IMPORTANT….DO NOT FORGET 'add' OR IT WON'T WORK

#Changed the datalogger due to maintenance issue
time 2010202:00:00:00.0 #Start time for the new datalogger
rt130 947A #Only need to specify the new datalogger serial number
add

close S01 12/31/2011 23:59:59.9 #End time for the NVT experiment

[1]: The FDSN Network Code is assigned by PASSCAL. The experiment name is chosen by the PI when an Instrument request is made (http://www.passcal.nmt.edu/inst_request). All archived data must have a network code. FDSN Network codes can be requested by filling in the form at http://www.iris.edu/scripts/getcode_permanent.html for a permanent network or http://www.iris.edu/scripts/getcode_temporary.html for a temporary network or experiment

[2]: Antelope comes with files describing the default settings (filters, etc.) for dataloggers, which may be found in $ANTELOPE/data/instruments/dataloggers. The datalogger type specified in the batchfile should

follow the Antelope naming convention. For example, by specifying 'datalogger rt130', dbbuild will automatically load parameters found in $ANTELOPE/data/instruments/dataloggers/rt130.pf.

[3]: Antelope comes with files describing the default settings (filters, etc.) for sensors, which may be found in $ANTELOPE/data/instruments/sensors. The sensor type specified in the batchfile should follow the Antelope naming convention. For example, by specifying 'sensor cmg40t', dbbuild will automatically load parameters found in $ANTELOPE/data/instruments/sensors/cmg40t.pf. As an aside, if you are using insturments that contain the datalogger and sensors (k2s, cmg6ts, etc), you will still need to specify a separate datalogger and sensor. For an example of a CERI cmt6t, see the Mt. Carmel batchfile.

3. Use dbbuild_batch to build the initial database tables describing metadata.
dbbuild -b -v NVT batchfile.NVT

You should now have files: NVT.calibration NVT.dlsensor NVT.instrument NVT.network NVT.schanloc NVT.sensor NVT.site NVT.sitechan NVT.snetsta NVT.stage

5. Create a file linking the schema to the table locations
vim NVT
insert the following lines
css3.0
./db/{NVT}:./dbmaster/{NVT}

6. Viewing the database tables:
dbe NVT

B: From Field to Database to DMC for RT-130s

Always do what IRIS prefers:
https://www.passcal.nmt.edu/webfm_send/2140

But there are a number of tutorials floating around, some of which are updates and reformats of mine. For example: **http://geophysics.eas.gatech.edu/people/jwalter/antelopetutorial.html**

Example database is the New Madrid Tremor Experiment (NVT)
/auto/home/hdeshon/SantaCruz2013/NVT

0. Follow PASSCAL directions for downloading data from field disks to a field computer. This tutorial starts out assuming you have reftek ZIP files such as 2011244.9889.ZIP (LASTDAYONDISK.DAS.ZIP)

1. Organize data directories
cd Database

mkdir raw_data data_files

foreach dir (raw_data data_files)
>cd $dir
>mkdir S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 *etc*
For above: modify with stations appropriate for you experiment
>cd ..
>end

The idea here is that you will have subdirectories for mseed hour files (raw_data) and mseed day

volumes (data_files) that are organized by station name or das number. It doesn't matter which from antelope's point of view. Antelope's default is to organize by DAS number because it assumes DASs don't move. However, DASs do move, so I prefer arranging by station name.

2. Create mseed files from reftek ZIP files

mkdir DUMP01

New way is to use **rt2ms**, which serves as both unchunky and fixhdr.

rt2ms should create a set of log files, error files, run files, and data directories. The log/run/err files are named by extraction time and the log files additionally include a DAS name. The data directories are named by julian day (R202.01, R202.02, R203.01, etc). The R stands for Reftek and the [.01, .02, 0.03] reflect how many times unchunky was run and may be ignored for now.

Within the data directories, unchunky or rt2ms extract mseed hour volumes for each DAS and each channel. Naming is for the begin time of the mseed file. 11.202.17.47.02.978C.1.m, for example, is data for Julian day 202, 2011 beginning at hour 17, minute 47, second 02. Data is for DAS 978C, channel 1 (vertical channel for this experiment).

2. Check log files to check for timing errors, double-check lat/lon/elev, power supply, etc.

cd Database/DUMP01
logpeek *logpeek opens a GUI which accesses all files ending in .log in the current directory.*

Time corrections larger than the sample interval should be referred to PASSCAL.
Useful to note the GPS location, start time and end time of the das/sensor setup, the sensor serial number, channel numbers (1-3;4-6) and vertical and horizontal channel orientation for use in next step.

Pencil and paper time: You are assessing the errors reported in the log file at this point. Look for clocks that never lock, DSP clock jumps, Q-packet errors, etc. If after reading the log file, you decide that the timing in the file is not accurate, be sure to write yourself a note so that this info can be recorded in the mseed headers in the next step. If you don't know what to do with an error, check the PASSCAL documents and/or send PASSCAL an email and ask. Don't assume errors can be ignored! Timing accuracy is essential in seismology.

3. Mark questionable timing using fixhdr in GUI form.

cd Database/DUMP01
fixhdr *#Open the GUI.*
Load each DAS/station separately and for problem times, mark the box 'timing questionable'

5. Organize data:

```
cd Database
cp DUMP01/*log DUMP01/*run DUMP01/*err   ./run_logs/.
foreach dir ( DUMP01 )
>cp $dir/*/*947A*.m raw_data/S01/.
>cp $dir/*/*9475*.m raw_data/S02/.
                    The rest of the stations would follow
                    .
                    .
>end
```

The above foreach loop actually resides in a script, which I happen to call copy_mseed.csh, because who wants to retype all of that every time you come back from the field. Also, I tend to

pack-rat data. The 'cp' calls above could be 'mv' calls to save harddrive space.

6. Create day volumes of the log files

Create a revised log2miniseed parameter file before first use:
cp $ANTELOPE/data/pf/log2miniseed.pf Database/pf/.
vim pf/log2miniseed.pf
change wfname line to
wfname data_files/%{sta}/%{sta}.%{net}.%{loc}.%{chan}.%Y.%j

Create a shell script to run log2miniseed that contains blocks of code, as follows, for each station. This would require modification for each trip to the field. My script is called run_log2miniseed.csh.

```
foreach file ( ./DUMP01/*947A*log )
>log2miniseed -n Y8 -s S01 $file
>end

foreach file ( ./DUMP01/*9475*log )
>log2miniseed -n Y8 -s S02 $file
>end
```

This will create day volumes (sta.net..LOG.year.jday) under directories in data_files filed under the station name. If files instead go into a subdirectory named after the year and julian day (ie, 2011), double check that your PFPATH environment variable contains the local Database/pf directory.

7. Create day volumes of the mseed files

```
foreach dir ( S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S16 S17 S18 S19 )
>miniseed2days –d TMPNVT \
>-w "./data_files/$dir/%{sta}.%{net}.%{loc}.%{chan}.%Y.%j" \
>./raw_data/$dir/*.m >&! miniseed2days.out1
>end
```

8. Make the wfdisc table

```
foreach dir ( data_files/S* )
>miniseed2db -v $dir/*.??? NVT
>end
```

The key here is that all files in data_files/S* will be added to the wfdisc. You need remove the .wfdisc file completely if you run the loop as written above because miniseed2db will append to an existing wfdisc table. A safer alternative is to write a script that allows you to set a start and endtime of the data you wanted added to wfdisc. You do not want duplicates line in your wfdisc! They are a pain to remove.

dbfix_calib NVT *Corrects the calib values in the wfdisc to be consistent with sensor table*
dbfixchanids NVT *Corrects the chanid in the wfdisc to link correctly with the sitechan table*

9. Verify the database
dbversdwf NVT >&! dbversdwf.out *Verifies the waveforms for archiving with the DMC*
dbverify NVT >&! dbverify.out *Verifies the database*

The chances that there are no mistakes is slim and fixing mistakes will be addressed in a later section.

10. Create dataless SEED file for the archive run and verify it
mk_dataless_seed -v -o Y8.09.20100742152.dataless NVT

creates a dataless file following NETCODE.YY.YEARJDAYHHMM.dataless where YEARJDAYHHMM are approximate time file is created and YY is the year of most of the data

seed2db -v Y8.09.20100742152.dataless *Allows verification of the wfdisc*

11. Sending to PASSCAL
Gui_DoFTP

ftp the day volumes (log and waveforms) from data_files and the current dataless seed volume

---

C. Adding data to a pre-existing wfdisc or creating a wfdisc from SAC data

Example database is a subset of the Nicoya CRSEIZE experiment from April 2001
/auto/home/hdeshon/SantaCruz2013/CostaRica

For this example, SAC headers already contain the station names and locations. Without this info in the headers, sac2db will not make site and sitechan tables. You can also stuff the SAC headers with event and phase information if available, then sac2db will even create event and origin tables for you.

Note: This method should work equally well if you wanted to merge two databases with different station information (moved stations, etc). For example, this is how I merge temporary CERI and Indiana networks for the Mt. Carmel earthquake with the permanent CNMSN network.

1. Make a temporay directory to hold a new database
cd Database
mkdir tmpdb; cd tmpdb

2. Create the tables for the OBS data, including wfdisc
sac2db /auto/hdeshon/SantaCruz2013/Nicoya_obsdata/2000.150/*.sac NicoyaOBS
dbe NicoyaOBS &

Note that in addition to the lastid, site, sitechan, and wfdisc sac2db creates event and origin tables. If the sac headers contained event information, this would be used in these two tables. However, the Nicoya data is continuous rather than event archived so these tables contain useless information and may be deleted.

rm NicoyaOBS.event NicoyaOBS.origin NicoyaOBS.lastid

3. Merge the NicoyaOBS database and AprilNicoya together:
dbmerge NicoyaOBS ../AprilNicoya

This will correctly set the chanid to be continuous with the AprilNicoya database. If you reopen the AprilNicoya.site table you can map all the stations.

Alternative approach: Reset the wfids in the temporary database to start from the last wfid plus 1 from the target database. (use dbset to accomplish this) Then you can just concate the temporary database wfdisc to the target database wfdis. You really should know what you are doing and how to fix errors (see Trouble-shooting below) if you take this approach.

4. Verify the database
dbverify AprilNicoya >&! dbverify.out *Verifies the database*

D. Trouble-shooting and Tricks of the Trade

Duplicate waveforms

Duplicate waveforms will appear orange when using dbpick or dbloc2. Duplicate waveforms can occur in two ways.

a) First, the mseed volume may have two copies of the waveform.

Fix using –DU flag with miniseed2days
**miniseed2days -DU -w "tmp/data_files/S*/%{sta}.%{net}.%{loc}.%{chan}.%Y.%j" **
tmp/data_files_tofix/*[ENZ]* >&! miniseed2days.out1

You should then recreate the wfdisc, or at minimum recreate the wfdisc lines pointing to the fixed mseed volume.

b) Second, the wfdisc may have multiple lines pointing to the same data or have lines pointing to the duplicate waveforms in the mseed volume. For example, if you fixed a mseed volume and added new lines to the wfdisc, but failed to remove the old wfdisc lines pointing to the original mseed volume, you will have introduced duplicate lines. The problem here is that all fields will be the same except the wfid and lddate fields, so a standard call to the unix 'sort' command will not work.

There is no quick way to fix this problem, to my knowledge. You can write some script that allows you to identify bad 'wfid' and use dbsubset and dbdelete to remove the lines.

Fix using dbsubset and dbdelete:

When deleting lines from the command line, you should always make sure your subset works like you think. Use dbselect to view the subseted lines
dbsubset AprilNicoya.wfdisc "wfid==’10’" | dbselect – sta chan wfid dir dfile
dbsubset AprilNicoya.wfdisc "wfid==’10’" | dbdelete –

Changing fields:

a) Example showing how to changed the the directory (dir) and channel (chan) fields in the wfdisc table using dbset/dbsubset [Short-course: I didn't introduce this problem either]

Set all dir fields from having a relative path to a full path name using pattern substitution. Antelope recognizes standard regular expressions (think sed, vim, etc.)
dbset AprilNicoya.wfdisc dir "*" 'patsub(dir,"../..","/auto/proj/[yourproject]")'

*Fix channel names by deleting the 01 location code (BHZ_01) that we are not interested in using. Use of the location code with this example database causes problems when using dbloc2 because that program defines vertical using *Z and horizontals using *[E,N] and therefore fails to read channels with the 01 tag. (Alternately, we could fix the dbloc2 parameter file to correctly identify *Z_01 as the vertical channel. But for now we will do it this way.)*
dbsubset AprilNicoya.wfdisc "time>=’2001120 00:00:00.000’" |
dbset AprilNicoya chan "*" 'patsub(chan,"_01","")'

b) Setting the channel id (chanid) field in the wfdisc table (example using dbtheta)

First some set up. The Costa Rica Ocean Bottom Seismometer data had channels named by numbers (1, 2, and 3). 1 was the vertical component, 2 & 3 were horizontals. Dbfixchanids fails to work with the OBS data because it depends on the sensor table being correct and the OBS don't have a sensor table. So…rather than create a sensor table for the OBS, let's fix this

another way. Without the proper chanids, the sitechan and wfdisc can not be joined using dbjoin and this a very useful join to link arrivals to their waveforms. Instead, we will use dbtheta to specify non-primary join keys and then set the wfdisc chanid to the sitechan.chanid.

First, test that the join will work like you think
dbsubset AprilNicoya.sitechan "chan=='1' || chan=='2' || chan=='3' " |
dbtheta –c "sta==sitechan.sta && chan=sitechan.chan" AprilNicoya.wfdisc sitechan |
dbselect – sta chan wfdisc.chanid sitechan.chanid

then fix it
dbsubset AprilNicoya.sitechan "chan=='1' || chan=='2' || chan=='3' " |
dbtheta –c "sta==sitechan.sta && chan=sitechan.chan" AprilNicoya.wfdisc sitechan |
dbselect –s – wfdisc.chanid:=sitechan.chanid

c) Resetting the offdate of stations (example of dbsubset/dbset)

Fix the fact that offdate is not set
dbsubset AprilNicoya.site "chan=='1' || chan=='2' || chan=='3' " |
dbset - offdate "*" "2001181"

dbsubset AprilNicoya.sitechan "chan=='1' || chan=='2' || chan=='3' " |
dbset - offdate "*" "2001181"

E. Antelope and Perl

Scripts should start with the following in order to access the Antelope Perl module

```
: # use perl
eval 'exec perl -S $0 "$@"'
if 0;

use lib "$ENV{ANTELOPE}/data/perl" ;
use Datascope ;
```

Perl scripts discussed below can be found in /NELE/scripts

The basic format to open a database in Perl

Example for reading in database tables in perl

```
$cr_hypos   = $ARGV[0] ;   #$cr_hypos=shift; would also work here
$db         = $ARGV[1];    #$db=shift; would also work here
#$ondate     = $ARGV[2];   #  The # represents a comment line in perl

@db     = dbopen  ( $db, "r+");   #This will open a database for read and write
@dborigin = dblookup( @db,"","origin","","");
@dbarrival = dblookup(@db,"","arrival","","");
#the above dblookup commands loads in the origin and arrival tables into separate arrays
```

Another example for reading in databases in perl

```
@db = dbopen($db, "r+");
@db = dblookup(@db,"","arrival","","");
@db = dbjoin(@db, dblookup(@db, "", "assoc", "", "") );
@db = dbjoin(@db, dblookup(@db, "", "origin", "","") );
@db = dbjoin(@db, dblookup(@db, "", "origerr", "","") );
```

```
@db = dbsubset(@db, "origin.auth!='orbassoc' && origin.auth!='UNA'" );
@db = dbsubset(@db, "lat < 11.0 && lat > 8.0");
@db = dbsubset(@db, "lon < -83.0 && lon > -85.00");
@db = dbsubset(@db, "nass >= 8 ");
@db = dbsort(@db, "origin.time", "sta", "iphase");

@db = dbsubset(@db, "origin.time >= '$stdate' && origin.time <= '$enddate'" );

$nrecords = dbquery(@db, "dbRECORD_COUNT" );

for( $db[3] = 0; $db[3] < $nrecords ; $db[3]++ ) {
   ($evid, $sta, $phase, $time, $auth) =
        dbgetv (@db, qw(evid sta iphase time arrival.auth ) );
}
```

Example A: Create an external database containing hypocenter and phase information from the CERI local earthquake catalog

1. Create CERI Permnet metadata tables for stations archived as part of NVT project using the CSS3.0 database for the CERI network created by W.Y. Kim quite a while ago.

cd /auto/home/hdeshon/SantaCruz2013/WYKIM

#revised dbsplit parameter file to work with site and sitechan

dbsplit -s "sta=='GLST' || sta=='HICK' || sta=='LEPT' || sta=='LNXT' || sta=='MORT' ||
sta=='PEBM' || sta=='PENM' || sta=='RDGT' || sta=='WYBT'" -p ./dbsplit.pf nmsz
nvtperm

cp nvtperm.site /Volumes/gaia/data/nvt/Database/external_databases/permnet.site

cp nvtperm.sitechan /Volumes/gaia/data/nvt/Database/ external_databases/permnet.sitechan

2. Add a wfdisc

cd /Volumes/gaia/data/nvt/Database/external_databases

miniseed2db ../PermNet/NM/*/* permnet

3. Fix some things

dbtheta -c "sta==sitechan.sta && chan==sitechan.chan" permnet.wfdisc sitechan |
dbselect -s - wfdisc.chanid:=sitechan.chanid

4. Add in hypocenter and pick information from CERI catalog

cd /Volumes/gaia/data/seisnet/NM/

foreach $file (2009/Loc/*/*arc 201[0,1]/Loc/*/*arc)
cat $file >>! /Volumes/gaia/data/nvt/Database/external_databases/20092011cericatalog.arc
end

*For years 19??, you would have to search as '**19??/Reg/*/*p**' to get the HypoInverse archive files

ceri_hypoe2phaID 20092011cericatalog.arc 20092011.cericatalog.pha NM 0 360

*ceri_hypoe2phaID is a fortran code modified to work with CERI data. Ask me for a copy.

First, make the origin and arrival tables

PERL SCRIPT: pha2db.pl 20092011.cericatalog.pha permnet

Second, make the assoc table

PERL SCRIPT: pha2db_step2.pl permnet

5. Fix some things

We are only interested in a subset of the network; remove all arrivals for stations NOT in the site table

dbjoin permnet.arrival assoc | dbnojoin - site | dbdelete –

Remove any earthquakes with no arrivals recorded at our stations of interest

dbnojoin permnet.origin assoc | dbdelete -

Remove the pesky location code for all instances of the 'chan' field.
dbset permnet chan "*" 'patsub(chan,"_00","")'

Set the arrival channel and chanid to be equal to the verical or horizontal component chanid for each station
dbsubset permnet.sitechan "vang=='0.0'" | dbtheta -c "sta==sitechan.sta" - arrival |
dbsubset - "iphase=='P'" | dbselect -s - arrival.chan:=sitechan.chan
dbsubset permnet.sitechan "vang=='0.0'" | dbtheta -c "sta==sitechan.sta" - arrival |
dbsubset - "iphase=='P'" | dbselect -s - arrival.chanid:=sitechan.chanid

dbsubset permnet.sitechan "hang=='90.0'" | dbtheta -c "sta==sitechan.sta" - arrival |
dbsubset - "iphase=='S'" | dbselect -s - arrival.chan:=sitechan.chan
dbsubset permnet.sitechan "hang=='90.0'" | dbtheta -c "sta==sitechan.sta" - arrival |
dbsubset - "iphase=='S'" | dbselect -s - arrival.chanid:=sitechan.chanid

6. Verify the database and fix any errors
dbverify permnet >&! verify.1

7. Merge this database with the NVT database