Building FreeREG

# There are 2 build rake tasks.

**Both the tasks (a and b) are dependent on 3 settings in the config environment development; mine are shown below**

1. **config.mongodb\_bin\_location = "d:/mongodb/bin/" where the Mongodb binary are located**
2. **config.datafiles = "e:/freereg9/" where the freereg data files are stored (Note the trailing ? is required.**
3. **config.dataset\_date = "13 Dec 2013" This is the date of the Freereg1 dataset we are using**

**It is also dependant on the addition of a file to the config.datafiles folder. A syndicates csv file holding the listing of coordinators for syndicates.**

* 1. **rake build:freereg [:type,:search\_records,:range1,:range2,:range3]**

This builds a freereg database by executing a number of rake tasks. Specifically it does the following:

* **setup** This deletes the log file and loads the emendations
* **save** It saves the Mater and Alias collections if the database is being **recreate**d. On an **add**ition it saves all collections.

**BEN.** I WONDER IF WE SHOULD ALWAYS SAVE ALL THE COLLECTIONS???????

* **drop** This task drops all the collections EXCEPT Master and Alias on a **recreate**.
* **create basic indexes** This task creates the minimum set of indices needed to have the processor run efficiently.
* **parallelp** This initiates and then waits for up to 3 separate rake task initiations of :process\_freereg1\_csv each with a different range of files. This could be generalized to n processors quite easily
* **process\_freereg1\_csv** initiates a single copy of the freereg1\_csv\_processor
* **parallel\_create\_search\_records** This initiates and then waits for up to 3 separate rake task initiations of :create\_search\_records each with a different set of files (the same set as for the processor. This could be generalized to n processors quite easily
* **create\_search\_records** initiates a single copy of the create search records
* **create\_userid\_docs** initiates the creation/update of the Userid Documents from the uDetails files. It needs the csv processors to have completed.
* **create\_syndicate\_docs** initiates the creation/update of the syndicates collection from a ***syndicates.csv file located in the freereg data files folder; has to be manually added to that folder at this time***
* **create\_county\_docs** uses the syndicates.csv file to create the county coordinators collection
* **create\_country\_docs** uses the syndicates.csv file to create the country coordinators collection
* **create\_freereg\_csv\_indexes** As its name implies we now create all of the indices we have defined

**The parameters.**

* **:type** This switch can be either **recreate** or anything else (I usually use **add** as the alternate). On a **recreate all previous records are wiped out** (Except Master and Alias). **On add** it only **processes those files that are newer** than those in the database. i.e. an update mode.
* **:search\_records** This **3 way switch** controls the way search records are created. A value of **create\_search\_records\_processor** tells the freereg1\_csv\_processor to create the search records as it processes the file. **create\_search\_records\_parallel** tells the freereg1\_csv\_processor not to worry and the search records are created latter in the build by the parallel processors. **no\_search\_records** or anything else tells the build to ignore the creation of search records. The reason for the 2 ways of creating search records is that on an update the freereg1\_csv\_processor is likely to have to do the creation of search records. But on a recreation I prefer to see the entries created and then the search records since the latter takes several days and my crash. Restarting with just the search records saves time.
* **range1, range2, range3** tell both the processor what actual files to process. It can take a number of different forms**. One of these is new**. Eg \*/\*.csv or \*/wry\*.csv or userid/\*.csv or userid/nfkaldba.csv or a-c or e-h. The first several of these are likely clear. a-c says process all csv files for all userids that start with a, b and c or e, f, g and h. This allows the segmentation of blocks of userids to be run by different copies of the processor.

**Full examples**

* **rake build:freereg[recreate,create\_search\_records\_parallel,\*/WRY\*.csv,\*/NFK\*.csv,\*/dev\*.csv]**

This creates a new database for the West Riding of Yorkshire, Norfolk and Devon.

* **rake build:freereg[recreate,create\_search\_records\_parallel,a-d,e-f,g-m]**

This creates a new database with all of the csv files for userids starting with a, b and c in the first process, e and f in the second and g h I j k l and m in the 3rd. Letters a-z and 0-9 are permitted

* **rake build:freereg[add,create\_search\_records\_processor,userid/wryconba.csv,useridb/nfkabsma.csv,useridc/devancbu.csv]**

This processes and add the 3 specific files if they are more recent that those in the database **and provided they have not been locked by the coordinator. Search records are created by the processor**

**In addition it is possible to run the freereg1\_ csv\_processor on its own as in the following example**

**rake build:process\_freereg1\_csv[:type,:search\_records,:range]**

All parameter have the same meaning and options except there is only 1 range

**Similarly it is possible to run the search\_record creation process on its own**

**rake build:create\_search\_records[:type,:search\_records,:range]**

* 1. **rake build:freereg\_from\_files,[:save, :drop, :reload\_from\_temp, :load\_from\_file, :index]**

This task allows one to rebuild the freereg database from files that the task creates OR from a previous set of files eg from a prior building or save. The parameters control the action of the individual tasks.

* Save Which collections do we save
* Drop Which collections do we drop
* Reload from temp Which collections do we load from the tmp folder
* Load from file Which collections do we load from our github respoitory or elsewhere
* Index Which indexes do we recreate

Each task is controlled by using a numbers for each of the various collections.

$collections[0] = "master\_place\_names"

$collections[1] = "alias\_place\_churches"

$collections[2] = "places"

$collections[3] = "churches"

$collections[4] = "registers"

$collections[5] = "freereg1\_csv\_files"

$collections[6] = "freereg1\_csv\_entries"

$collections[7] = "search\_records"

$collections[8] = "userid\_details"

$collections[9] = "syndicates"

$collections[10] = "counties"

$collections[11] = "countries"

Examples

* build:freereg\_from\_files["0/1/8/9/10/11","2/3/4/5/6/7", "0/1","2/3/4/5","0/1/2/3/4/5/8/9/10/11"]

this save the Master, Alias, Userid and Syndicates and collections to tmp, drops the other 6 collections, reloads the Master and Alias from the tmp, reloads places, churches, registers and files from the github respository and then indexes the main 8 collections. The database will not have entries and search records.

* build:freereg\_from\_files["","","","0/1","0/1"]

reloads the Master and Alias collections from Github and indexes them

* build:freereg\_from\_files["","","2/3/4/5/6/7/8/9","0/1","0/1/2/3/4/5/6/7/8/910/11"]

reloads saved versions of Places/Churches/Registers/Files/Entries/Search\_records from tmp and reloads the Master and Alias collections from Github and indexes them all.

How to use in practice.

Case 1

Let’s build from scratch and create a full database. First step is to seed the database with the Master and Alias

If there is no backup of the Master and Alias collection the we use the github collections to seed the database;

**rake build:freereg\_from\_files["","","","0/1","0/1"]**

If we have a previous backup that is more recent than the github we would use:

**rake build:freereg\_from\_files["","","0/1","","0/1"]**

Now we can build the database itself

**rake build:freereg[recreate,create\_search\_records\_parallel,a-f,g-l,m-9]**

Case 2

Lets update the database with a new set of csv files. The following will update only those files which have changed. Backups will be created first.

**rake build:freereg[add,create\_search\_records\_processor,a-f,g-l,m-9]**

**Case 3**

Lets produce a test database with complete entries, DAP and full places/churches/registers and Files but only one county of search records. This is something I suggest we do initially as it meets all of our testing needs. Assumes we have an existing database. If not we would first build from files as in Case 1.

**rake build:freereg[recreate,no\_search\_records,a-f,g-l,m-9]**

This creates everything except the search records; so now we do the following to create the search\_records for just WRY as an example.

**rake build:create\_search\_records[add,create\_search\_records\_parallel,\*/wry\*.csv]**

I use add on the assumption that there are no search\_records in the database as they were dropped by the first build, this way the search record creator does not bother to check for and delete the existing search\_records. This means that we can change the search records code and recreate the database search records by replacing add with recreate.