dbAutoMaker – User Guide

Introduction

dbAutoMaker is the collective name for a framework of Perl scripts designed to automatically: (1) download data files from online resources; (2) decompress data files if required; (3) convert data files to a format compatible for a database (if required); and (4) import the data files into a database.

This document explains how to install, run, and configure *dbAutoMaker*.

How to get started

There are 3 main ways to use *dbAutoMaker:*

1) Run the distribution version of *dbAutoMaker*.

You can choose to automatically create a customised SNP database for any of the following livestock species: *Bos taurus* (cow), *Gallus gallus* (chicken), *Sus scrofa* (pig), and *Ovis aries* (sheep). *Homo sapiens* (human) is also an option.

See section "**Installation**" and "**Program Execution**"

2) Configure *dbAutoMaker* for another species chosen from dbSNP

You can choose to configure *dbAutoMaker* to automatically created a customised SNP database for any species that contains SNPs in dbSNP (dbSNP is a database managed by National Center of Biotechnology Information).

See section "**Installation**" and "**Configuring *dbAutomaker***"

3) Configure *dbAutoMaker* to create your own local database

You can choose to configure *dbAutoMaker* to create a local database with data extracted from any type of online resource that provides data to the public.

See section "**Installation**" and "**Creating your own local database from scratch**"

Installation

Prerequisites:

Perl MUST be installed to run *dbAutoMaker*.

dbAutoMaker has been developed and tested on Perl 5.10.1 for Windows and Perl 5.8.8 for Linux

The following Perl modules MUST also be installed:

Config::Simple
LWP::Simple
Archive::Extract
File::Spec

Download:

A compressed *dbAutoMake*r directory containing all relevant programs and files can be downloaded from the following URL:

For Windows:

http://web4ftp.it.csiro.au/ftp4goo17a/dbAutoMaker/dbAutoMaker.zip

For Linux:

http://web4ftp.it.csiro.au/ftp4goo17a/dbAutoMaker/dbAutoMaker.tar.gz

Directory structure:

After decompressing the *dbAutoMaker* directory, the directory can be moved to any location of your choice. Figure 1 shows the directory structure for the distributed version of *dbAutoMaker*. The "?" in Figure 1 implies any pathname.

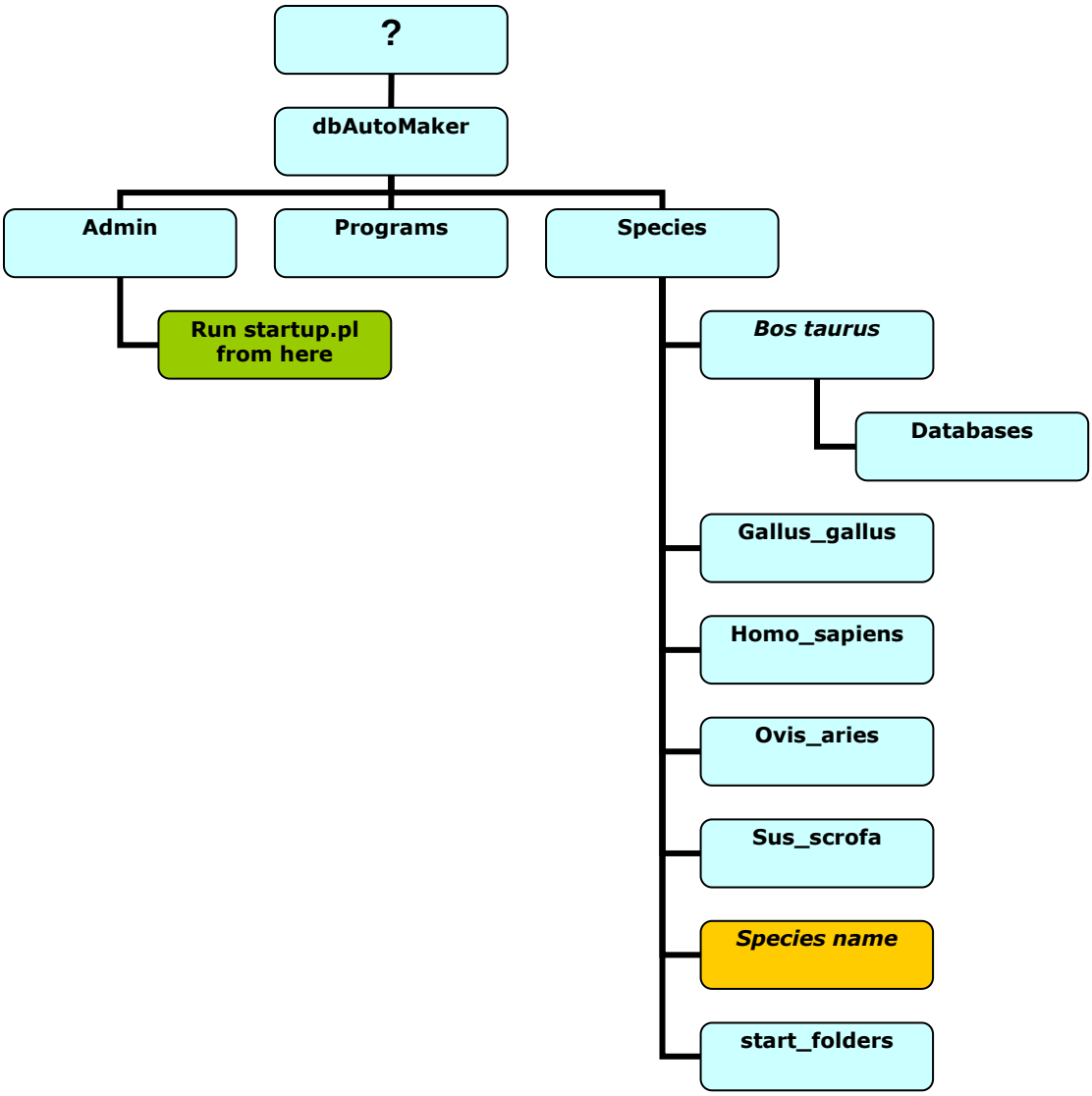

Figure 1: The *dbAutoMaker* directory structure for the distributed version

The contents of the directories are:

Admin - startup.pl (script to begin database creation process), startup.ini (a file containing the list of species configuration files), <species>.ini files (the database configuration files specific to a species), <species>.log (records the database creation stages, errors, and warnings. Used for debug purposes only).

Programs – contains all the Perl scripts used to create the database. It also contains the Windows and Linux versions of the SQLite executables.

Species - A separate directory is used for each species. It contains separate directories for each resource used to create the database. The "Databases" directory within each species directory is used to store a working database and the database being created. The "start_folders" directory contains the template directories. For example, if a database for a new species is to be created, "start_folders" is copied and renamed to the species name.

Program Execution

The database creation is invoked with a Perl script from a command prompt in Windows or within a Linux\Unix shell. The Perl script is called "startup.pl" and is located in the Admin directory (See Figure 1).

For example:

1. Use a Linux\Unix shell OR a command prompt in Windows
2. Change directory to Admin
3. Type: startup.pl bta <enter> (OR perl startup.pl bta <enter> if the ".pl" extension is not associated with Perl)

The startup.pl script requires one argument indicating which configuration file, associated with the required species, to use. The "bta" in step 3 above instructs the program to use 'bos_taurus.ini' as derived from a list of user defined configuration files in startup.ini (located in the Admin directory). Figure 2 shows the content of the default startup.ini configuration file.

```
#startup.ini – User defined configuration files
code<species<config<dir<url
bta<Bos taurus<bos_taurus.ini<Bos_taurus<http://web4ftp.it.csiro.au/ftp4goo17a/FunctSNP/btaSNP.zip
gga<Gallus gallus<gallus_gallus.ini<Gallus_gallus<http://web4ftp.it.csiro.au/ftp4goo17a/FunctSNP/ggaSNP.zip
hsa<Homo sapiens<homo_sapiens.ini<Homo_sapiens<http://web4ftp.it.csiro.au/ftp4goo17a/FunctSNP/hsaSNP.zip
oar<Ovis aries<ovis_aries.ini<Ovis_aries<http://web4ftp.it.csiro.au/ftp4goo17a/FunctSNP/oarSNP.zip
ssc<Sus scrofu<sus_scrofa.ini<Sus_scrofa<http://web4ftp.it.csiro.au/ftp4goo17a/FunctSNP/sscSNP.zip
```

Figure 2: startup.ini – contains a list of species configuration files.

It is expected that the "startup.ini" configuration file will only require modification *if* the desired species is not shown in the list *and* the user wishes to create their own species configuration file. The "startup.ini" configuration file contains 5 columns separated by a "<" character. The first column can be any number of characters and is used by the d*bAutoMaker* Perl scripts to make the association with the appropriate species configuration file. Column 2 is simply a description of the species and is not read by any Perl script. Column 3 is the species configuration filename (any user definable name) located in the Admin directory. Columns 4 and 5 are not used by dbAutoMaker and are used by a recommended database user interface called *FunctSNP* (refer http://www.csiro.au/science/FunctSNP.html). For example, if the user wishes to create a database for the horse species, they could add "eca<Horse<equus_callabus.ini".

Configuring dbAutoMaker

The core of *dbAutoMaker* is a species configuration file in a header-key format. Configuration files for the species *Bos taurus* (cow), *Gallus gallus* (chicken), *Sus scrofa* (pig), *Ovis aries* (sheep), and *Homo sapiens* (human) are supplied with the *dbAutoMaker* distribution. The distributed configuration files are called <species>.ini and can be found in the Admin directory (see Figure 1). SNP customised databases can be automatically created using the distributed species configuration files with no modification required. A customised SNP database can also be created for any species that contains SNPs in dbSNP (dbSNP is a database managed by NCBI- National Center of Biotechnology Information). In other words, any organism that is listed in the NCBI FTP site: ftp://ftp.ncbi.nih.gov/snp/organisms/.

Therefore, if the user wishes to create a SNP database for a species that has no provided configuration file, a new configuration file that is specific to the species is required. In such a case it is recommended the user copies an existing configuration file and modifies it accordingly.

The following section describes the content of a configuration file and presents an example of how a user can modify the contents in order to create a database for a new species. Appendix A shows a complete configuration file for the species *Bos taurus*, in this example we will modify a configuration file for the species *Equus callabus* (horse).

The first step is to copy "bos_taurus_ini" to "equus_callabus.ini" in the Admin directory. Then make a copy of the "start_folders" directory and call it "Equus_ callabus".

```
# Configuration file for bos_taurus    May 2009

[Resources]
name=dbSNP,GeneInfo,ProteinInfo,GO,KEGG,,UniProt,UniProt_ref,QTLdb,Homolo
,OMIA,Schema,Indexes,Extract,Build
```

Figure 3: Extract from bos_taurus.ini configuration file (Resources header)

Any line in the configuration file that begins with"#" is interpreted by the program as a comment only.

The distribution version of *dbAutoMaker* extracts data from the following online resources:

dbSNP - Single Nucleotide Polymorphism database (http://www.ncbi.nlm.nih.gov/).
GO - Gene Ontology (http://www.geneontology.org/).
KEGG - Kyoto Encyclopedia of Genes and Genomes (http://www.genome.jp/kegg/).
UniProt - Universal Protein Resource (http://www.uniprot.org/).
QTLdb - Animal Quantitative Trait Locus database (http://www.animalgenome.org/QTLdb/).
OMIA - Online Mendelian Inheritance in Animals (http://omia.angis.org.au/)
Homolo - HomoloGene (http://www.ncbi.nlm.nih.gov/homologene).

In the configuration file (see Figure 3), [Resources] is regarded as the header, and "name" is the key. Once *dbAutoMaker* is invoked, the program loops through the 4 steps - download, decompress, convert, and import - for each resource listed after the "name" key. The resource name in principle can be any name on the proviso that the name is used consistently throughout the rest of the configuration file. For example, instead of dbSNP one could use "NCBI SNP database". No modification under [Resources] is required for the Equus callabus example.

```
[General]
taxon_id=9913
species_code="bta"
species_dir="../Species/Bos_taurus"
program_dir="../../../Programs"
database_dir="../Databases"
log_file="bos_taurus_logfile.txt"
```

Figure 4: Extract from bos_taurus.ini configuration file (General header)

Under the General header (see Figure 4):

taxon_id:
: The taxonomy ID – change to 9796

species code:
: Any number of characters, however, the characters used must be consistent throughout the configuration file and with startup.ini. It is recommended that a 3 character code derived from the taxonomy name is used e.g. change to eca

species_dir:
: The root directory in which all data files and databases related to the species will be stored. The path entered can either be an absolute pathname (e.g., /dbAutoMaker/Species/Bos_taurus) or a relative pathname and relative to where startup.pl is invoked, which by default is in the Admin directory – change to "../Species/Equus_caballus"

program_dir:
: The directory that contains ALL Perl scripts and programs required for the creation of the database. The path entered can either be an absolute pathname (eg., /dbAutoMaker/Programs) or a relative pathname and relative to the resource directory e.g., relative to P:\dbAutoMaker\Species\Bos_taurus\dbSNP directory (in Windows)

database_dir:
: The directory that will contain a working database (working.db) and the species database being created (e.g. ecaSNP.db). The path entered can either be an absolute pathname (e.g., /dbAutoMaker/Species/Bos_taurus/Databases) or a relative pathname and relative to the resource directory e.g., relative to P:\dbAutoMaker\Species\Bos_taurus\dbSNP directory

log_file:
: A filename that will record the database creation stages (including encountered errors) and can be used for debug purposes – change to equus_caballus_logfile.txt

The changes for the *Equus caballus* species:

```
[General]
taxon_id=9796
species_code="eca"
species_dir="../Species/Equus_caballus"
program_dir="../../../Programs"
database_dir="../Databases"
log_file="equus_caballus_logfile.txt"
```

Figure 5: Extract from equus_caballus.ini configuration file (General header)

Each resource has a possible 4 sections defined by 4 headers – [<resourceName>], [<resourceName>_downloads], [<resourceName>_programs], and [<resourceName>_arguments].

Using Resource **dbSNP** as an example**:**

```
[dbSNP]
enable="YES"
working_dir="dbSNP"
decompress="YES"
#decompress_program="gzip.exe"
#decompress_arguments="-d *.gz"
import_program="sqlite3.exe"
import_arguments="working.db < master.sql"
```

Figure 6: Extract from bos_taurus.ini configuration file (dbSNP header)

Under the dbSNP header (see Figure 6):

enable:
: A “YES” or “NO” (not case dependent). A “YES” instructs the program to process the resource. A “No” instructs the program to ignore the processing of the resource.

working_dir:
: The directory into which all downloaded files will be saved and processed. The program expects the directory to be in the appropriate species directory.

decompress:
: A “YES” or “NO” (not case dependent). A “YES” instructs the program to automatically decompress all files with the suffixes tar, tgz, gz, and zip found in directory defined under working_dir. A “No” instructs the program not to decompress files.

decompress_program:
: It is possible for a user to use a different decompression program if required by entering the program name. The program needs to be stored in the directory defined under General - program_dir. No alternative decompression program is required in this example and the line is therefore commented out.

decompress_arguments:
: Arguments for the decompression program are entered here if required.

import_program:
: The program used to import data into the database. The program needs to be stored in the directory defined under General - program_dir. *dbAutoMaker* was designed with the relational database management systems (RDMS) SQLite and MySQL in mind. Both products are open source and provide batch mode functionality.

import_arguments:
: The arguments for the import program and in this case “working.db” is the name of the database to be created and stored in the directory defined under General - database_dir. The argument “< master.sql” instructs SQLite to run the SQL and SQLite commands contained in the file master.sql. “master.sql” is located in the directory defined under dbSNP – working_dir and is created by the Perl script “MySQL_SQLite.pl”.

dbAutoMaker processes the resource in the order: (1) download; (2) decompress; (3) convert; and (4) import as instructed through the value of the keys under the headers

[<resourceName>] , [<resourceName>_downloads] , [<resourceName>_programs] , [<resourceName>_arguments] .

1. Download: the Uniform Resource Locators (URLs) required are listed under the dbSNP_downloads header. The program downloads the files from the list one at a time into the directory defined under dbSNP – working_dir.

```
[dbSNP_downloads]
url0=ftp://ftp.ncbi.nih.gov/snp/database/shared_schema/dbSNP_main_table.sql.gz
url1=ftp://ftp.ncbi.nih.gov/snp/database/shared_data/SnpFunctionCode.bcp.gz
url2=ftp://ftp.ncbi.nih.gov/snp/organisms/cow_9913/database/organism_data/GeneIdToName.bcp.gz
url3=ftp://ftp.ncbi.nih.gov/snp/organisms/cow_9913/database/organism_data/SNP.bcp.gz
url4=ftp://ftp.ncbi.nih.gov/snp/organisms/cow_9913/database/organism_schema/cow_9913_table.sql.gz
url5=ftp://ftp.ncbi.nih.gov/snp/organisms/cow_9913/database/organism_data/b130_SNPContigLocusId_4_1.bcp.gz
url6=ftp://ftp.ncbi.nih.gov/snp/organisms/cow_9913/database/organism_data/b130_ContigInfo_4_1.bcp.gz
url7=ftp://ftp.ncbi.nih.gov/snp/organisms/cow_9913/database/organism_data/b130_SNPChrPosOnRef_4_1.bcp.gz
```

Figure 7: Extract from bos_taurus.ini configuration file (dbSNP_downloads header)

For the *Equus callabus* species change to:

```
[dbSNP_downloads]
url0=ftp://ftp.ncbi.nih.gov/snp/database/shared_schema/dbSNP_main_table.sql.gz
url1=ftp://ftp.ncbi.nih.gov/snp/database/shared_data/SnpFunctionCode.bcp.gz
url2=ftp://ftp.ncbi.nih.gov/snp/organisms/horse_9796/database/organism_data/GeneIdToName.bcp.gz
url3=ftp://ftp.ncbi.nih.gov/snp/organisms/ horse_9796/database/organism_data/SNP.bcp.gz
url4=ftp://ftp.ncbi.nih.gov/snp/organisms/ horse_9796/database/organism_schema/ horse_9796_table.sql.gz
url5=ftp://ftp.ncbi.nih.gov/snp/organisms horse_9796/database/organism_data/b130_SNPContigLocusId_2_1.bcp.gz
url6=ftp://ftp.ncbi.nih.gov/snp/organisms/ horse_9796/database/organism_data/b130_ContigInfo_2_1.bcp.gz
url7=ftp://ftp.ncbi.nih.gov/snp/organisms/ horse_9796/database/organism_data/b130_SNPChrPosOnRef_2_1.bcp.gz
```

Figure 8: Extract from equus_callabus.ini configuration file (dbSNP_downloads header)

If the file to download already exists locally, it will only be downloaded if the date of the file on the server has changed.

2. Decompress: The program automatically decompresses files with the suffixes tar, tgz, gz, and zip located in the directory defined under dbSNP – working_dir.

3. Convert: A Perl script called “MySQL_SQLite.pl” is used to convert the data files into a format that can be imported into SQLite.

```
[dbSNP_programs]
1="perl"

[dbSNP_arguments]
1="MySQL_SQLite.pl dbSNPbta.ini"
```

Figure 9: Extract from bos_taurus.ini configuration file (dbSNP_programs and dbSNP_arguments header)

The Perl script "MySQL_SQLite.pl" works in conjunction with a configuration file specific to the species and in this case dbSNPbta.ini for *Bos taurus*. How the script and the configuration file interact can be found in Appendix B. The script has been designed to be generic and the principal idea is that the user will only need to modify the appropriate configuration file when there is a need to convert a file in a Microsoft SQL Server or MySQL schema format to SQLite. For the *Equus callabus* species example, dbSNPbta.ini can be modified and called dbSNPeca.ini

4. Import: one of the outputs from "MySQL_SQLite.pl" is a file called "master.sql" that is used as an input to the SQLite program to import the converted data files into the SQLite database. "master.sql" contains SQL and SQLite commands to read and import data files.

The above 4 steps - download, decompress, convert, and import – are repeated for each resource in association with the 4 possible sections defined by the headers – [<resourceName>] , [<resourceName>_downloads] , [<resourceName>_programs] , [<resourceName>_arguments] . How *dbAutoMaker* processes each resource is the same as that described for the resource dbSNP. Appendix E shows all changes that are required to the configuration file for the *Equus caballus* species example.

Creating your own local database from scratch

There are essentially 4 manual steps in creating a local database from online resources: (1) download the data from the online resource; (2) decompress file(s) if required; (3) convert data into a format that can be imported to a database; and (4) import the data into the database. These 4 steps need to be repeated for each online resource. *dbAutoMaker* provides a framework in which these 4 steps are automatically repeated to create a local database with data extracted from any number of resources. Our principle motivation is to enable a user, with limited or no programming experience, to collect their data of interest from a range of resources into a unified depository.

The following is an example on how to configure *dbAutoMaker* to create a local database with data extracted from any type of online resource that provides data to the public. In the example we are going to create a local SQLite database that will contain data extracted from a fictitious online resource called "Biology Information Service".

Step1:

Install *dbAutoMaker* [See section **Installation**]

Step 2:

In the Admin directory create a text file called "bioinfo.ini" and add text as shown in Figure 10.

```
[Resources]
name=BioInfo

[General]
main_dir="../BioInfo"
program_dir="../../Programs"
database_dir="../database"
log_file="Bio_logfile.txt"
```

Figure 10: Extract from bioinfo.ini configuration file

Under the Resources header (see Figure 10):

name:
: The fictitious “Biology Information Service” resource is given the name “BioInfo”, which will be used as the reference to the resource in the rest of the configuration file.

Under the General header (see Figure 10):

main_dir:
: The root directory in which all data files and the database directory will be stored. The pathname is relative to where startup.pl is invoked, which by default is in the Admin directory.

program_dir:
: The directory that contains ALL programs required for the creation of the database. The pathname is relative to the working directory defined under BioInfo – working_dir (see later below).

database_dir:
: The directory that will contain the database - the pathname is relative to the working directory defined under BioInfo – working_dir (see later below).

```
[BioInfo_downloads]
url0=ftp://ftp.resource.for/BioInformation/datafile_1.txt.gz
url1=ftp://ftp.resource.for/BioInformation/datafile_2.txt.gz
url1=ftp://ftp.resource.for/BioInformation/datafile_2.txt.gz
```

Figure 11: Extract from bioinfo.ini configuration file (BioInfo_downloads header)

The prefix to the _downloads header, shown in Figure 11, must be the same as the resource name (“BioInfo” in this example).

Under the BioInfo_downloads header (see Figure 11):

: The list of files to download into the directory defined under BioInfo – working_dir (“data_files” in this example).

```
[BioInfo]
enable="YES"
working_dir="data_files"
decompress="YES"
import_program="sqlite3.exe"
import_arguments="bioinfo.db < input.txt"
```

Figure 12: Extract from bioinfo.ini configuration file (BioInfo header)

The header shown above in the text box (see Figure 12) must be the same as the resource name (“BioInfo” in this example).

Under the BioInfo header (see Figure 12):

enable:

A “YES” instructs the program to process the resource.

working_dir:

“data_files” is the directory into which all downloaded files will be saved and processed. The program expects the directory to be located in the main directory defined under General – main_dir (“BioInfo” in this example).

decompress:

The “YES” instructs the program to automatically decompress the files with the suffix gz found in the directory “data_files”.

import_program:

The program to be used to import data into the database will be sqlite3.exe, which needs to be stored in the directory defined under General - program_dir (“Programs” in this example).

import_arguments:

“bioinfo.db” is the name of the database to be created and stored in the directory defined under General - database_dir (“database” in this example). The argument “< input.txt” is a SQLite commands file to run within SQLite. “input.txt” is located in the directory defined under BioInfo – working_dir (“data_files”) and will be created by the program “convert.exe”.

```
[BioInfo_programs]
1=”convert.exe”
2="perl"

[BioInfo_arguments]
1=”*.txt”
2="bioinfo.pl”
```

Figure 13: Extract from bioinfo.ini configuration file (BioInfo_programs and BioInfo_arguments header)

The prefix to the headers shown Figure 13 must be the same as the resource name (“BioInfo” in this example).

There is currently no standardized format for transporting data from one database to the next. Some possible data formats that can be expected are Microsoft SQL Server, MySQL, SQLite, XML, BioMart, and ASCII text file (flat file). A user, when creating their own database, will need to specify the required program (or several programs to run consecutively) to convert the data files into an appropriate format compatible with their chosen database. It is expected that the user may need to develop their own programs to handle the conversion. If a conversion from a Microsoft SQL Server or MySQL schema format to SQLite is required, a generic Perl script, called MySQL_SQLite.pl is provided as part of the *dbAutoMaker* distribution. The Perl script is described in detail in Appendix B.

Under the BioInfo_programs header:

> "1=convert.exe" indicates that a program called convert.exe will be executed. Its task in this case is to convert the data files into a format that can be imported into SQLite. Immediately on the completion of the convert program, the Perl script "bioinfo.pl will execute. In theory, *any* number of programs and *any* command type programs listed under the "_programs" header can be run consecutively. The programs need not necessarily be used only for data file conversion. Any program task could theoretically be invoked through *dbAutoMaker*.

Under the BioInfo_arguments header:

> The arguments for each program are listed in the exact same order as the programs are listed. Therefore, "*.txt" is the argument for the program "convert.exe" and "bioinfo.pl" is the argument for the program "perl".

Step 3:
In the Admin directory (see Figure 1) add a new line to the startup.ini configuration file:

```
#startup.ini – User defined configuration files
bta<Bos taurus<bos_taurus.ini
gga<Gallus gallus<gallus_gallus.ini
hsa<Homo sapiens.ini<homo_sapiens.ini
oar<Ovis aries<ovis_aries.ini
ssc<Sus scrofu<sus_scrofa.ini
bioinfo< Biology Information Service<bioinfo.ini
```

Figure 14: startup.ini – contains a list of species configuration files.

Step 4:

Run the Perl called "startup.pl", located in the Admin directory (See Figure 1), within a Unix/Linux shell or a command prompt in Windows.

For example:

1) Run a command prompt in Windows
2) Change directory to Admin
3) Type: startup.pl bioinfo <enter> (OR perl startup.pl bioinfo <enter> if the ".pl" extension is not associated with Perl)

APPENDIX A

```
# Configuration file for bos_taurus        May 2009
[Resources]
name=dbSNP,GeneInfo,ProteinInfo,GO,KEGG,,UniProt,UniProt_ref,QTLdb,Homolo,OMIA,
Schema,Indexes,Extract,Build

[General]
taxon_id=9913
species_code="bta"
species_dir="../Species/Bos_taurus"
program_dir="../../../Programs"
database_dir="../Databases"
log_file="bos_taurus_logfile.txt"

[dbSNP]
enable="YES"
working_dir="dbSNP"
decompress="YES"
#decompress_program="gzip.exe"
#decompress_arguments="-d *.gz"
import_program="sqlite3.exe"
import_arguments="working.db < master.sql"

[dbSNP_programs]
1="perl"

[dbSNP_arguments]
1="MySQL_SQLite.pl dbSNPbta.ini"

[dbSNP_downloads]
url0=ftp://ftp.ncbi.nih.gov/snp/database/shared_schema/dbSNP_main_table.sql.gz
url1=ftp://ftp.ncbi.nih.gov/snp/database/shared_data/SnpFunctionCode.bcp.gz
url2=ftp://ftp.ncbi.nih.gov/snp/organisms/cow_9913/database/organism_data/GeneIdToName.bcp.gz
url3=ftp://ftp.ncbi.nih.gov/snp/organisms/cow_9913/database/organism_data/SNP.bcp.gz
url4=ftp://ftp.ncbi.nih.gov/snp/organisms/cow_9913/database/organism_schema/cow_9913_table.sql.gz
url5=ftp://ftp.ncbi.nih.gov/snp/organisms/cow_9913/database/organism_data/b130_SNPContigLocusId_4_1.bcp.gz
url6=ftp://ftp.ncbi.nih.gov/snp/organisms/cow_9913/database/organism_data/b130_ContigInfo_4_1.bcp.gz
url7=ftp://ftp.ncbi.nih.gov/snp/organisms/cow_9913/database/organism_data/b130_SNPChrPosOnRef_4_1.bcp.gz

[GeneInfo]
enable="YES"
working_dir="GeneInfo"
decompress="YES"
import_program="sqlite3.exe"
import_arguments="working.db < master.sql"

[GeneInfo_programs]
1="perl"
2="perl"

[GeneInfo_arguments]
1="GeneInfo_to_SQLite.pl bta gene_info"
2="GenePos.pl bta gene2accession"
```

```
[GeneInfo_downloads]
url0=ftp://ftp.ncbi.nlm.nih.gov/gene/DATA/gene_info.gz
url1=ftp://ftp.ncbi.nlm.nih.gov/gene/DATA/gene2accession.gz

[ProteinInfo]
enable="YES"
working_dir="Build"
decompress="YES"
run_programs="YES"

[ProteinInfo_programs]
1="perl"

[ProteinInfo_arguments]
1="ProteinInfo.pl rna.q"

[ProteinInfo_downloads]
url0=ftp://ftp.ncbi.nih.gov/genomes/Bos_taurus/mapview/rna.q.gz


[GO]
enable="YES"
working_dir="GO"
decompress="YES"
import_program="sqlite3.exe"
import_arguments="working.db < master.sql"

[GO_programs]
1="perl"

[GO_arguments]
1="Gene2GO.pl bta gene2go"

[GO_downloads]
url0=ftp://ftp.ncbi.nlm.nih.gov/gene/DATA/gene2go.gz

[KEGG]
enable="YES"
working_dir="KEGG"
decompress="NO"
import_program="sqlite3.exe"
import_arguments="working.db < master.sql"

[KEGG_programs]
1="perl"

[KEGG_arguments]
1="KEGG_to_SQLite.pl bta"

[KEGG_downloads]
url0=ftp://ftp.genome.jp/pub/kegg/genes/organisms/bta/bta_pathway.list
url1=ftp://ftp.genome.jp/pub/kegg/genes/organisms/bta/bta_ncbi-geneid.list

[UniProt]
enable="YES"
```

working_dir="UniProt"
decompress="YES"
import_program="sqlite3.exe"
import_arguments="working.db < master.sql"

[UniProt_programs]
1="perl"

[UniProt_arguments]
1="UniProt_To_SQLite.pl UniProt.ini"

[UniProt_downloads]
url17=ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/complete/uniprot_sprot.dat.gz

[UniProt_ref]
enable="YES"
working_dir="UniProt_ref"
decompress="YES"
import_program="sqlite3.exe"
import_arguments="working.db < master.sql"

[UniProt_ref_programs]
1="perl"

[UniProt_ref_arguments]
1="UniProt_ref.pl gene_refseq_uniprotkb_collab"

[UniProt_ref_downloads]
url0=ftp://ftp.ncbi.nlm.nih.gov/gene/DATA/gene_refseq_uniprotkb_collab.gz

[QTLdb]
enable="YES"
working_dir="QTLdb"
decompress="YES"
import_program="sqlite3.exe"
import_arguments="working.db < master.sql"

[QTLdb_programs]
1="perl"

[QTLdb_arguments]
1="QTLdb_to_SQLite.pl QTL_gff_BT"

[QTLdb_downloads]
url0=http://www.animalgenome.org/QTLdb/tmp/QTL_gff_BT.gz

[Homolo]
enable="YES"
working_dir="Homolo"
decompress="NO"
import_program="sqlite3.exe"
import_arguments="working.db < master.sql"

[Homolo_programs]

```
[Homolo_arguments]


[Homolo_downloads]
url0=ftp://ftp.ncbi.nlm.nih.gov/pub/HomoloGene/current/homologene.data
url1=ftp://ftp.ncbi.nlm.nih.gov/pub/HomoloGene/current/build_inputs/taxid_taxname

[OMIA]
enable="YES"
working_dir="OMIA"
decompress="YES"
import_program="sqlite3.exe"
import_arguments="working.db < master.sql"

[OMIA_programs]
1="perl"
2="perl"

[OMIA_arguments]
1="MySQL_SQLite.pl OMIA.ini"
2="OMIA.pl omia.sql OMIA.ini"


[OMIA_downloads]
url0=http://omia.angis.org.au/dumps/omia.sql.gz

[Schema]
enable="YES"
working_dir="Build"
decompress="NO"
import_program="sqlite3.exe"
import_arguments="working.db < schema.sql"

[Indexes]
enable="YES"
working_dir="Build"
decompress="NO"
run_programs="YES"
import_program="sqlite3.exe"
import_arguments="working.db < index_master.sql"

[Indexes_programs]
1="perl"
2="perl"

[Indexes_arguments]
1="create_index_master.pl"
2="create_extract_master.pl"

[Extract]
enable="YES"
working_dir="Build"
decompress="NO"
import_program="sqlite3.exe"
import_arguments="working.db < extract_master.sql"
```

```
[Build]
enable="YES"
working_dir="Build"
decompress="NO"
run_programs="YES"
import_program="sqlite3.exe"
import_arguments="btaSNP.db < SNP_master.sql"

[Build_programs]
1="perl"
2="perl"
3="perl"
4="perl"
5="perl"

[Build_arguments]
1="SNPID_To_QTL.pl"
2="snp_score.pl"
3="data_assembly.pl bta"
4="nearest_gene.pl"
5="create_SNP_master.pl"
```

APPENDIX B

MySQL_SQLite.pl - a Perl script to convert Microsoft SQL Server or MySQL to a SQLite compatible format.

A generic Perl script was developed to modify the dbSNP schema files into a format suitable for SQLite. The 'generic' idea is to ensure the program can handle MySQL, MSSQL or any SQL syntax different to SQLite. Another important requirement in the development was to create flexibility for future dbSNP schema changes. These requirements were achieved by the use of a header-key configuration file (Appendix C). MySQL_SQLite.pl works closely in conjunction with the configuration file. The following description explains the interaction of the Perl script and configuration file (Figure 15).

```
[Convert]
tables="YES"
data="YES"
batch_file="master.sql"

[Files_required]
tables="*_table.sql"
data="*.bcp"
log_tables="Converted_SQL_files.txt"
log_data="Converted_bcp_files.txt"

[Files_excluded]
1="Mod_"
2="master"
3=".gz"
```

Figure 15: Extract of configuration file for MySQL_SQLite.pl

Converting SQL table files: If [Convert.tables] key is equal to "YES", then the script searches for all files matching the specified mask in [Files_required.tables] in the working directory - in this example, the mask is "*_table.sql" and it assumes files with this suffix contains the SQL syntax for creating tables.. However, all masks listed under [Files_excluded] will be ignored. The program parsers all matching files and extracts only the lines associated with the tables listed under [Tables_required] as shown in Figure 16. The extracted lines are written to a new file called "Mod_<filename>_table.sql". The program obtains a list of SQL syntax to validate from [SQL_Exclusions_Begin] (Figure 16) and if any of the specified syntax occurs at the beginning of the extracted line, the line is ignored. . For example, any line that begins with "GO" is ignored. The program also obtains a list of invalid SQLite syntax from [SQL_invalid] (Figure 17) and if any of the syntax occurs anywhere within the line, the syntax is replaced with a blank.

```
[Tables_required]
1="SNP"
2=GeneIdToName
3=SnpFunctionCode
4=b128_SNPContigLocusId_3_1
5=b128_SNPChrPosOnRef_3_1
6=b128_ProteinInfo_3_1
7=b128_ContigInfo_3_1

[Table_Names]
1=SNP
2=GeneIdToName
3=SnpFunctionCode
4=SNPContigLocusId
5=SNPChrPosOnRef
6=ProteinInfo
7=ContigInfo

[SQL_Instructions]
DROP_required=YES
#Include perl escape characters if needed
Open_sep='\['
Close_sep='\]'
Data_sep='"       "'
```

Figure 16: Extract of configuration file for MySQL_SQLite.pl

The user can insert a "DROP TABLE <tablename>" SQL command into the modified schema files, if [SQL_Instructions.DROP_required] is set to "YES". The names of the tables to be used in the SQLite database do not need to be the same as in dbSNP. The user can change the name under [Table_Names] (Figure 16). As an example, the dbSNP table name b128_SNPContigLocusId_3_1 ([Tables_required] header - key 4 in Figure 16) can be changed to SNPContigLocusId ([Table_Names] header - key 4 in Figure 16). It is a requirement of the program that every table under [Tables_required] has a corresponding name (key) under [Table_Names] even if the name does not change.

In some SQL schema files the table name is enclosed by square brackets e.g., CREATE TABLE **[**table name**]** whereas in other files it is enclosed by other punctuation characters such as single quotation marks e.g., CREATE TABLE **'**table name**'**. In order for the program to extract the table name, it is a requirement that the opening and closing characters enclosing the table name is specified under [SQL_Instructions.Open_sep] and [SQL_Instructions.Close_sep] (Figure 16.) In Perl the square bracket is a Metacharacter and needs to be escaped. For example, the opening and closing character in the dbSNP schema files are squared brackets, hence Open_sep='\[' where \ is the character used by Perl so that the squared bracket is interpreted as a bracket and not a Perl instruction (Metacharacter)

All the modified files matching the specified mask are recorded for debugging purposes in the filename specified under [Files_required.log_tables] (Figure 15). In this example, the filename is "Converted_SQL_files.txt". The program also writes to a batch file named under [Convert.batch_file] (Figure 15) the SQLite command for reading from a file. For example, the batch file name is called master.sql and it will contain lines such as *.read <filename>*.

Where <*filename*> is the file containing the modified SQL commands for creating one table (refer Appendix D).

```
[SQL_Exlusions_Begin]
1=GO

[SQL_invalid]
1=Latin1_General_BIN
2= ON [PRIMARY]
```

Figure 17: Extract of configuration file for MySQL_SQLite.pl

Preparing data files for importing: If [Convert.data] key (Figure 15) is equal to "YES, the script searches for all files matching the specified mask in [Files_required.data] in the working directory - in this example, the mask is "*.bcp" and it assumes files with this suffix contain data with 1 line per database record. It is a requirement of the program that the character, separating each column in the record, is specified under [SQL_Instructions.Data_sep]. The separating character in the downloaded dbSNP data files is a tab (4 spaces), hence Data_sep='" "'(Figure 16). The SQLite command to indicate the separating character(s) is *.separator*. The program writes this command to the batch file named under [Conver.batch_file] (15). For dbSNP data files the command *.separator " "* is required.

All masks listed under [Files_excluded] will be ignored. All the files matching the specified mask are recorded for debugging purposes in the filename specified under [Files_required.log_data] (Figure 15). In this example, the filename is "Converted_bcp_files.txt". The program writes the SQLite command for importing data from a file to a batch file. For example, the batch file name is called master.sql and contains lines such as *.import* <*filename*> <*table name*>. Where <*filename*> is the file containing the data to be imported and <*table name*> is the name of the table receiving the data (refer Appendix E). If the data file is empty it causes an error when imported into SQLite. Therefore, for empty files, the program comments out the SQLite command using a "- -" within master.sql e.g. -- *.import* <*filename*> <*table name*>.

APPENDIX C

```
# Configuration file for converting MSSSQL to SQLite   June 2009
[Convert]
tables="YES"
data="YES"
batch_file="master.sql"

[Files_required]
tables="*_table.sql"
data="*.bcp"
log_tables="Converted_SQL_files.txt"
log_data="Converted_bcp_files.txt"

[Files_excluded]
1="Mod_"
2="master"
3=".gz"

[Tables_required]
1="SNP"
2=GeneIdToName
3=SnpFunctionCode
4=b128_SNPContigLocusId_3_1
5=b128_SNPChrPosOnRef_3_1
6=b128_ProteinInfo_3_1
7=b128_ContigInfo_3_1

[Table_Names]
1=SNP
2=GeneIdToName
3=SnpFunctionCode
4=SNPContigLocusId
5=SNPChrPosOnRef
6=ProteinInfo
7=ContigInfo

[SQL_Instructions]
DROP_required=YES
#Include perl escape characters if needed
Open_sep='\['
Close_sep='\]'
Data_sep='"       "'

[SQL_Exlusions_Begin]


[SQL_invalid]
```

APPENDIX D

The following is the master.sql file created by the program MySQL_to_SQLite.pl

```
-- Tables to read
.read Mod_cow_9913_table.sql
.read Mod_dbSNP_main_table.sql
-- Data to import
.separator "	"
.import b128_ContigInfo_3_1.bcp ContigInfo
.import b128_ProteinInfo_3_1.bcp ProteinInfo
.import b128_SNPChrPosOnRef_3_1.bcp SNPChrPosOnRef
.import b128_SNPContigLocusId_3_1.bcp SNPContigLocusId
.import GeneIdToName.bcp GeneIdToName
.import SNP.bcp SNP
.import SnpFunctionCode.bcp SnpFunctionCode
```

APPENDIX E

```
# Configuration file for Equus callabus Oct 2009

[Resources]
name=dbSNP,GeneInfo,ProteinInfo,GO,KEGG,,UniProt,UniProt_ref,Homolo,OMIA,Schema,I
ndexes,Extract,Build
#QTLdb has been removed

[General]
taxon_id=9796
species_code="eca"
species_dir="../Species/Equus_caballus"
program_dir="../../../Programs"
database_dir="../Databases"
log_file="equus_caballus_logfile.txt"

[dbSNP]
enable="YES"
working_dir="dbSNP"
decompress="YES"
import_program="sqlite3.exe"
import_arguments="working.db < master.sql"

[dbSNP_programs]
1="perl"

[dbSNP_arguments]
1="MySQL_SQLite.pl dbSNPeca.ini"

[dbSNP_downloads]
url0=ftp://ftp.ncbi.nih.gov/snp/database/shared_schema/dbSNP_main_table.sql.gz
url1=ftp://ftp.ncbi.nih.gov/snp/database/shared_data/SnpFunctionCode.bcp.gz
url2=ftp://ftp.ncbi.nih.gov/snp/organisms/horse_9796/database/organism_data/GeneIdToName.bcp.gz
url3=ftp://ftp.ncbi.nih.gov/snp/organisms/ horse_9796/database/organism_data/SNP.bcp.gz
url4=ftp://ftp.ncbi.nih.gov/snp/organisms/ horse_9796/database/organism_schema/ horse_9796_table.sql.gz
url5=ftp://ftp.ncbi.nih.gov/snp/organisms horse_9796/database/organism_data/b130_SNPContigLocusId_2_1.bcp.gz
url6=ftp://ftp.ncbi.nih.gov/snp/organisms/ horse_9796/database/organism_data/b130_ContigInfo_2_1.bcp.gz
url7=ftp://ftp.ncbi.nih.gov/snp/organisms/ horse_9796/database/organism_data/b130_SNPChrPosOnRef_2_1.bcp.gz


[GeneInfo]
enable="YES"
working_dir="GeneInfo"
decompress="YES"
import_program="sqlite3.exe"
import_arguments="working.db < master.sql"

[GeneInfo_programs]
1="perl"
2="perl"

[GeneInfo_arguments]
1="GeneInfo_to_SQLite.pl eca gene_info"
2="GenePos.pl eca gene2accession"

[GeneInfo_downloads]
url0=ftp://ftp.ncbi.nlm.nih.gov/gene/DATA/gene_info.gz
url1=ftp://ftp.ncbi.nlm.nih.gov/gene/DATA/gene2accession.gz
```

[ProteinInfo]
enable="YES"
working_dir="Build"
decompress="YES"
run_programs="YES"

[ProteinInfo_programs]
1="perl"

[ProteinInfo_arguments]
1="ProteinInfo.pl rna.q"

[ProteinInfo_downloads]
url0=ftp://ftp.ncbi.nih.gov/genomes/Bos_taurus/mapview/rna.q.gz

[GO]
enable="YES"
working_dir="GO"
decompress="YES"
import_program="sqlite3.exe"
import_arguments="working.db < master.sql"

[GO_programs]
1="perl"

[GO_arguments]
1="Gene2GO.pl eca gene2go"

[GO_downloads]
url0=ftp://ftp.ncbi.nlm.nih.gov/gene/DATA/gene2go.gz

[KEGG]
enable="YES"
working_dir="KEGG"
decompress="NO"
import_program="sqlite3.exe"
import_arguments="working.db < master.sql"

[KEGG_programs]
1="perl"

[KEGG_arguments]
1="KEGG_to_SQLite.pl eca"

[KEGG_downloads]
url0=ftp://ftp.genome.jp/pub/kegg/genes/organisms/eca/eca_pathway.list
url1=ftp://ftp.genome.jp/pub/kegg/genes/organisms/eca/eca_ncbi-geneid.list

[UniProt]
enable="YES"
working_dir="UniProt"
decompress="YES"
import_program="sqlite3.exe"
import_arguments="working.db < master.sql"

[UniProt_programs]
1="perl"

[UniProt_arguments]
1="UniProt_To_SQLite.pl UniProt.ini"

[UniProt_downloads]
url17=ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/complete/uniprot_sprot.dat.gz

[UniProt_ref]
enable="YES"
working_dir="UniProt_ref"
decompress="YES"
import_program="sqlite3.exe"
import_arguments="working.db < master.sql"

[UniProt_ref_programs]
1="perl"

[UniProt_ref_arguments]
1="UniProt_ref.pl gene_refseq_uniprotkb_collab"

[UniProt_ref_downloads]
url0=ftp://ftp.ncbi.nlm.nih.gov/gene/DATA/gene_refseq_uniprotkb_collab.gz

[Homolo]
enable="YES"
working_dir="Homolo"
decompress="NO"
import_program="sqlite3.exe"
import_arguments="working.db < master.sql"

[Homolo_programs]

[Homolo_arguments]

[Homolo_downloads]
url0=ftp://ftp.ncbi.nlm.nih.gov/pub/HomoloGene/current/homologene.data
url1=ftp://ftp.ncbi.nlm.nih.gov/pub/HomoloGene/current/build_inputs/taxid_taxname

[OMIA]
enable="YES"
working_dir="OMIA"
decompress="YES"
import_program="sqlite3.exe"
import_arguments="working.db < master.sql"

[OMIA_programs]
1="perl"
2="perl"

[OMIA_arguments]
1="MySQL_SQLite.pl OMIA.ini"
2="OMIA.pl omia.sql OMIA.ini"

[OMIA_downloads]
url0=http://omia.angis.org.au/dumps/omia.sql.gz

[Schema]
enable="YES"

working_dir="Build"
decompress="NO"
import_program="sqlite3.exe"
import_arguments="working.db < schema.sql"

[Indexes]
enable="YES"
working_dir="Build"
decompress="NO"
run_programs="YES"
import_program="sqlite3.exe"
import_arguments="working.db < index_master.sql"

[Indexes_programs]
1="perl"
2="perl"

[Indexes_arguments]
1="create_index_master.pl"
2="create_extract_master.pl"

[Extract]
enable="YES"
working_dir="Build"
decompress="NO"
import_program="sqlite3.exe"
import_arguments="working.db < extract_master.sql"

[Build]
enable="YES"
working_dir="Build"
decompress="NO"
run_programs="YES"
import_program="sqlite3.exe"
import_arguments="ecaSNP.db < SNP_master.sql"

[Build_programs]
1="perl"
2="perl"
3="perl"
4="perl"
5="perl"

[Build_arguments]
1="SNPID_To_QTL.pl"
2="snp_score.pl"
3="data_assembly.pl eca"
4="nearest_gene.pl"
5="create_SNP_master.pl"