Tutorial on resolving NCBI taxonomy using BioSQL

Umer Zeeshan Ijaz

Download BioSQL from http://biosql.org/DIST/biosql-1.0.1.tar.gz. Once the software is installed, setup a database and import the BioSQL schema. The following command line should create a new database on your own computer called bioseqdb, belonging to the root user account:

```
[uzi@quince-srv2 ~]$ mysqladmin -u root create bioseqdb
```

We can then tell MySQL to load the BioSQL scheme we downloaded above. Change to the scripts subdirectory from the unzipped BioSQL download, then:

```
[uzi@quince-srv2 ~]$ mysql -u root bioseqdb < biosqldb-mysql.sql
```

To update the NCBI taxonomy, change to the scripts subdirectory from the unzipped BioSQL download, then:

```
[uzi@quince-srv2 ~]$ ./load_ncbi_taxonomy.pl --dbname bioseqdb --driver mysql --dbuser root --download true
Loading NCBI taxon database in taxdata:
        ... retrieving all taxon nodes in the database
        ... reading in taxon nodes from nodes.dmp
        ... insert / update / delete taxon nodes
        ... (committing nodes)
        ... rebuilding nested set left/right values
        ... reading in taxon names from names.dmp
        ... deleting old taxon names
        ... inserting new taxon names
        ... cleaning up
Done.
```

Check which tables are imported

```
[uzi@quince-srv2 ~]$ mysql --user=root bioseqdb -e "show tables"
```

```
+----------------------------+
| Tables_in_bioseqdb         |
+----------------------------+
| biodatabase                |
| bioentry                   |
| bioentry_dbxref            |
| bioentry_path              |
| bioentry_qualifier_value   |
| bioentry_reference         |
| bioentry_relationship      |
| biosequence                |
| comment                    |
| dbxref                     |
| dbxref_qualifier_value     |
| location                   |
| location_qualifier_value   |
| ontology                   |
| reference                  |
| seqfeature                 |
| seqfeature_dbxref          |
| seqfeature_path            |
| seqfeature_qualifier_value |
| seqfeature_relationship    |
| taxon                      |
| taxon_name                 |
| term                       |
| term_dbxref                |
| term_path                  |
| term_relationship          |
| term_relationship_term     |
| term_synonym               |
+----------------------------+
[uzi@quince-srv2 ~]$ mysql --user=root bioseqdb
```

Check how many databases are there

```
mysql> show databases;
```

```
+--------------------+
| Database           |
+--------------------+
| information_schema |
| bioseqdb           |
| mysql              |
| test               |
+--------------------+

mysql> use bioseqdb
Database changed
```

Check the attributes of the table ‘taxon’

```
mysql> select column_name from information_schema.columns where table_name='taxon';
+-------------------+
| column_name       |
+-------------------+
| taxon_id          |
| ncbi_taxon_id     |
| parent_taxon_id   |
| node_rank         |
| genetic_code      |
| mito_genetic_code |
| left_value        |
| right_value       |
+-------------------+
```

Check the attributes of the table ‘taxon_name’

```
mysql> select column_name from information_schema.columns where table_name='taxon_name';
+-------------+
| column_name |
+-------------+
| taxon_id    |
| name        |
| name_class  |
```

```
+-------------+
```

Check if these tables are populated

```
mysql> SELECT ncbi_taxon_id from taxon LIMIT 10;
+---------------+
| ncbi_taxon_id |
+---------------+
|             1 |
|             2 |
|             6 |
|             7 |
|             9 |
|            10 |
|            11 |
|            13 |
|            14 |
|            16 |
+---------------+
```

Now we will run the following commands one by one to find the path of taxon id 54255 to the root node. We will do this by finding the parent taxon id of the given id, using this id to find the parents and repeating the procedure until we reach the root node

```
mysql> SELECT taxon_name.name, taxon.node_rank, taxon.parent_taxon_id FROM taxon, taxon_name where taxon.taxon_id =
taxon_name.taxon_id AND taxon_name.name_class='scientific name' AND taxon.ncbi_taxon_id=54255;
+--------------------+-----------+-----------------+
| name               | node_rank | parent_taxon_id |
+--------------------+-----------+-----------------+
| Thermofilum librum | species   |            1793 |
+--------------------+-----------+-----------------+

mysql> SELECT taxon_name.name, taxon.node_rank, taxon.parent_taxon_id FROM taxon, taxon_name where taxon.taxon_id =
taxon_name.taxon_id AND taxon_name.name_class='scientific name' AND taxon.taxon_id=1793;
+-------------+-----------+-----------------+
| name        | node_rank | parent_taxon_id |
```

```
+-------------+-----------+-----------------+
| Thermofilum | genus     |           86865 |
+-------------+-----------+-----------------+

mysql> SELECT taxon_name.name, taxon.node_rank, taxon.parent_taxon_id FROM taxon, taxon_name where taxon.taxon_id =
taxon_name.taxon_id AND taxon_name.name_class='scientific name' AND taxon.taxon_id=86865;
+---------------+-----------+-----------------+
| name          | node_rank | parent_taxon_id |
+---------------+-----------+-----------------+
| Thermofilaceae | family    |            1791 |
+---------------+-----------+-----------------+

mysql> SELECT taxon_name.name, taxon.node_rank, taxon.parent_taxon_id FROM taxon, taxon_name where taxon.taxon_id =
taxon_name.taxon_id AND taxon_name.name_class='scientific name' AND taxon.taxon_id=1791;
+-----------------+-----------+-----------------+
| name            | node_rank | parent_taxon_id |
+-----------------+-----------+-----------------+
| Thermoproteales | order     |          149927 |
+-----------------+-----------+-----------------+

mysql> SELECT taxon_name.name, taxon.node_rank, taxon.parent_taxon_id FROM taxon, taxon_name where taxon.taxon_id =
taxon_name.taxon_id AND taxon_name.name_class='scientific name' AND taxon.taxon_id=149927;
+-------------+-----------+-----------------+
| name        | node_rank | parent_taxon_id |
+-------------+-----------+-----------------+
| Thermoprotei | class     |           12664 |
+-------------+-----------+-----------------+

mysql> SELECT taxon_name.name, taxon.node_rank, taxon.parent_taxon_id FROM taxon, taxon_name where taxon.taxon_id =
taxon_name.taxon_id AND taxon_name.name_class='scientific name' AND taxon.taxon_id=12664;
+--------------+-----------+-----------------+
| name         | node_rank | parent_taxon_id |
+--------------+-----------+-----------------+
| Crenarchaeota | phylum    |            1705 |
+--------------+-----------+-----------------+
```

```
mysql> SELECT taxon_name.name, taxon.node_rank, taxon.parent_taxon_id FROM taxon, taxon_name where taxon.taxon_id =
taxon_name.taxon_id AND taxon_name.name_class='scientific name' AND taxon.taxon_id=1705;
+---------+-------------+-----------------+
| name    | node_rank   | parent_taxon_id |
+---------+-------------+-----------------+
| Archaea | superkingdom |          102425 |
+---------+-------------+-----------------+

mysql> SELECT taxon_name.name, taxon.node_rank, taxon.parent_taxon_id FROM taxon, taxon_name where taxon.taxon_id =
taxon_name.taxon_id AND taxon_name.name_class='scientific name' AND taxon.taxon_id=102425;
+--------------------+-----------+-----------------+
| name               | node_rank | parent_taxon_id |
+--------------------+-----------+-----------------+
| cellular organisms | no rank   |               1 |
+--------------------+-----------+-----------------+
```

I have written a small stored procedure to do the same thing as above. Make a new file with path_to_root_node.sql with the following contents:

```
DROP PROCEDURE IF EXISTS path_to_root_node;
DELIMITER //
CREATE PROCEDURE path_to_root_node(IN leaf INT )
BEGIN
      DECLARE parent_taxon_id INT;
      DECLARE taxon_name VARCHAR(255);
      DECLARE taxon_rank VARCHAR(255);
      DECLARE path_to_root VARCHAR(255);
      SET @path_to_root = '';
      SELECT taxon_name.name, taxon.node_rank, taxon.parent_taxon_id INTO @taxon_name, @taxon_rank, @parent_taxon_id
FROM taxon, taxon_name where taxon.taxon_id=taxon_name.taxon_id AND taxon_name.name_class='scientific name' AND
taxon.ncbi_taxon_id=leaf;
      SET @path_to_root=CONCAT(@path_to_root,@taxon_name,':',@taxon_rank);
      loop_label: LOOP
            SELECT taxon_name.name, taxon.node_rank, taxon.parent_taxon_id INTO @taxon_name, @taxon_rank,
      @parent_taxon_id FROM taxon, taxon_name where taxon.taxon_id=taxon_name.taxon_id AND
      taxon_name.name_class='scientific name' AND taxon.taxon_id=@parent_taxon_id;
```

```
            SET @path_to_root=CONCAT(@path_to_root,";",@taxon_name,':',@taxon_rank);
            IF (@parent_taxon_id <> 1) THEN
                  ITERATE loop_label;
            ELSE
                  LEAVE loop_label;
            END IF;
      END LOOP;
      SELECT @path_to_root;
END//
DELIMITER ;
```

Now load the stored procedure, call it, and it will give you the complete path from the given taxon id to root node

```
mysql> source path_to_root_node.sql
mysql> call path_to_root_node(54255);
+-------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------+| @path_to_root
|+------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------+| Thermofilum
librum:species;Thermofilum:genus;Thermofilaceae:family;Thermoproteales:order;Thermoprotei:class;Crenarchaeota:phylum
;Archaea:superkingdom;cellular organisms:no rank |+-----------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------+
```

You can also use the following one-liners on linux prompt

```
[uzi@quince-srv2 ~]$ mysql --user=root bioseqdb -e "source path_to_root_node.sql;call path_to_root_node(54255);" |
awk -F"|" '$0!~/^@path_to_root/{print $0}'
Thermofilum
librum:species;Thermofilum:genus;Thermofilaceae:family;Thermoproteales:order;Thermoprotei:class;Crenarchaeota:phylum
;Archaea:superkingdom;cellular organisms:no rank
[uzi@quince-srv2 ~]$ mysql --user=root bioseqdb -e "source path_to_root_node.sql;call path_to_root_node(9606);" |
awk -F"|" '$0!~/^@path_to_root/{print $0}'
Homo
sapiens:species;Homo:genus;Homininae:subfamily;Hominidae:family;Hominoidea:superfamily;Catarrhini:parvorder;Simiifor
mes:infraorder;Haplorrhini:suborder;Primates:order;Euarchontoglires:superorder;Eutheria:no rank;Theria:no
rank;Mammalia:class;Amniota:no rank;Tetrapoda:no rank;Sarcopterygii:no rank;Euteleostomi:no rank;Teleostomi:no
rank;Gnathostomata:superclass;Vertebrata:no rank;Craniata:subphylum;Chordata:phylum;Deuterostomia:no
```

```
rank;Bilateria:no rank;Eumetazoa:no rank;Metazoa:kingdom;Opisthokonta:no rank;Eukaryota:superkingdom;cellular
organisms:no rank
```

Say you have a list of accession numbers stored in a file accnos.list

```
[uzi@quince-srv2 ~]$ cat accnos.list
AB000389.1
AB000699.1
AB000700.1
AB000701.1
AB000702.1
AB001518.1
AB001724.1
AB001774.1
AB001775.1
AB001776.1
AB001777.1
AB001779.1
AB001781.1
AB001783.1
AB001784.1
AB001785.1
AB001791.1
AB001793.1
AB001797.1
AB001802.1
```

You are interested in finding the corresponding gID, taxa ID, and assignment at species and genus level for each accession number. You can get the gID and taxa ID from locally installed NT database using blastdbcmd and resolve the assignment at species and genus level using path_to_root_node.sql. For example, in the following script, we get the assignment at species level

```
[uzi@quince-srv2 ~]$ cat accnos.list | /home/opt/ncbi-blast-2.2.28+/bin/blastdbcmd -db /home/opt/ncbi-blast-
2.2.28+/db/nt -entry_batch - -outfmt "%a,%g,%T" | while IFS=$',' read -r -a myArray; do echo
${myArray[0]},${myArray[1]},${myArray[2]},$(mysql --user=root bioseqdb -e "source path_to_root_node.sql; call
path_to_root_node(${myArray[2]});" | awk -F "|" '$0!~/^@path_to_root/{print $0}' | perl -ne
'@a=split(/:/,join(/,/,grep {/species/} split(/;/,$_)));print $a[0]'); done
```

```
AB000389.1,11036396,81037,Pseudoalteromonas elyakovii
AB000699.1,3107908,153948,Nitrosomonas sp. AL212
AB000700.1,3107909,153949,Nitrosomonas sp. JL21
AB000701.1,3107910,153947,Nitrosomonas sp. GH22
AB000702.1,3107911,42353,Nitrosomonas sp.
AB001518.1,1871429,47467,Ixodes scapularis endosymbiont
AB001724.1,1902830,267859,Microcystis elabens
AB001774.1,1902837,85991,Chlamydia pecorum
AB001775.1,1902838,85991,Chlamydia pecorum
AB001776.1,1902839,85991,Chlamydia pecorum
AB001777.1,1902840,85991,Chlamydia pecorum
AB001779.1,1902842,83554,Chlamydia psittaci
AB001778.1,1902841,331636,Chlamydia psittaci
AB001780.1,1902843,83554,Chlamydia psittaci
AB001781.1,1902844,83554,Chlamydia psittaci
AB001782.1,1902845,83554,Chlamydia psittaci
AB001786.1,1902849,83554,Chlamydia psittaci
AB001787.1,1902850,83554,Chlamydia psittaci
AB001788.1,1902851,83554,Chlamydia psittaci
AB001789.1,1902852,83554,Chlamydia psittaci
AB001790.1,1902853,83554,Chlamydia psittaci
AB001812.1,1902875,83554,Chlamydia psittaci
AB001783.1,1902846,83555,Chlamydophila abortus
AB001784.1,1902847,83554,Chlamydia psittaci
AB001785.1,1902848,83556,Chlamydophila felis
AB001791.1,1902854,83554,Chlamydia psittaci
AB001793.1,1902856,83554,Chlamydia psittaci
AB001794.1,1902857,83554,Chlamydia psittaci
AB001800.1,1902863,83554,Chlamydia psittaci
AB001801.1,1902864,83554,Chlamydia psittaci
AB001803.1,1902866,83554,Chlamydia psittaci
AB001796.1,1902859,83554,Chlamydia psittaci
AB001797.1,1902860,83554,Chlamydia psittaci
AB001799.1,1902862,83554,Chlamydia psittaci
AB001802.1,1902865,83554,Chlamydia psittaci
```

Using “genus” as search pattern in grep command below gives assignment at genus level.

```
[uzi@quince-srv2 ~]$ cat accnos.list | /home/opt/ncbi-blast-2.2.28+/bin/blastdbcmd -db /home/opt/ncbi-blast-
2.2.28+/db/nt -entry_batch - -outfmt "%a,%g,%T" | while IFS=$',' read -r -a myArray; do echo
${myArray[0]},${myArray[1]},${myArray[2]},$(mysql --user=root bioseqdb -e "source path_to_root_node.sql; call
path_to_root_node(${myArray[2]});" | awk -F "|" '$0!~/^@path_to_root/{print $0}' | perl -ne
'@a=split(/:/,join(/,/,grep {/genus/} split(/;/,$_)));print $a[0]'); done
AB000389.1,11036396,81037,Pseudoalteromonas
AB000699.1,3107908,153948,Nitrosomonas
AB000700.1,3107909,153949,Nitrosomonas
AB000701.1,3107910,153947,Nitrosomonas
AB000702.1,3107911,42353,Nitrosomonas
AB001518.1,1871429,47467,
AB001724.1,1902830,267859,Microcystis
AB001774.1,1902837,85991,Chlamydia
AB001775.1,1902838,85991,Chlamydia
AB001776.1,1902839,85991,Chlamydia
AB001777.1,1902840,85991,Chlamydia
AB001779.1,1902842,83554,Chlamydia
AB001778.1,1902841,331636,Chlamydia
AB001780.1,1902843,83554,Chlamydia
AB001781.1,1902844,83554,Chlamydia
AB001782.1,1902845,83554,Chlamydia
AB001786.1,1902849,83554,Chlamydia
AB001787.1,1902850,83554,Chlamydia
AB001788.1,1902851,83554,Chlamydia
AB001789.1,1902852,83554,Chlamydia
AB001790.1,1902853,83554,Chlamydia
AB001812.1,1902875,83554,Chlamydia
AB001783.1,1902846,83555,Chlamydophila
AB001784.1,1902847,83554,Chlamydia
AB001785.1,1902848,83556,Chlamydophila
AB001791.1,1902854,83554,Chlamydia
AB001793.1,1902856,83554,Chlamydia
AB001794.1,1902857,83554,Chlamydia
AB001800.1,1902863,83554,Chlamydia
AB001801.1,1902864,83554,Chlamydia
AB001803.1,1902866,83554,Chlamydia
```

AB001796.1,1902859,83554,Chlamydia
AB001797.1,1902860,83554,Chlamydia
AB001799.1,1902862,83554,Chlamydia
AB001802.1,1902865,83554,Chlamydia