Exercise 3 : RepeatMasking sequences using the pipeline

You will need :

- An Ensembl database with loaded sequence and assembly
  (there is a working copy installed on your machine if you did not get all the way through Exercise 1 - it's called **mouse37_mini_ref**)
- MySQL client (installed in /usr/local/bin/mysql) and general knowledge of SQL queries
- RepeatMasker and CrossMatch programs (installed in /usr/local/ensembl/bin/)
- A checkout of ensembl, ensembl-analysis, ensembl-pipeline version 50
  (You'll find these checkouts here : home/training/ensembl-src/)
- To set your PERL5LIB environment by sourcing this script:

```
source /home/training/ensembl-src/ensembl-personal/usr/set_perl_path.sh
```

Overview of steps in this Exercise :

1. add RepeatMask analysis to the database
2. add rules for the analysis to the database, and make input_ids
3. fill out configuration files like BatchQueue.pm
4. check the setup by running a small dry run
5. run the pipeline on your machine locally
6. check the results by looking in the `repeat_feature` and `repeat_consensus` table of your database by connecting with the MySQL client.

Lets' start!

1. Add RepeatMask analysis to the database

(a) Adding an initial Submission-analysis to our database ...

- Analyses can be added to your database using the following analysis_setup.pl script. The script will read a file which contains the analysis configuration (see file repeatmask_ana.conf below).

- The script reads in commandline options like "`-dbhost < yourhost> -dbuser ens-training -dbpass workshop -dbport 3306`" so that it knows where your database lives.

- Further options for this script can be found by running :

```
$HOME/ensembl-src/ensembl-pipeline/scripts/analysis_setup.pl -help
```

- Besides options for database connection, the analysis_setup.pl script requires a file containing all the details of your analysis. This file is read by the script, parsed, and written into your database. An example file used as input for the analysis_setup.pl script looks like this :

```
cat $HOME/ensembl-src/ensembl-config/mouse/NCBIM37/pipe_conf/repeatmask_ana.conf
```

```
[RepeatMask]
db=repbase
db_version0129
db_file=repbase
program=RepeatMask
program_version=3.1.8
program_file=/path/to/repmasker/RepeatMask
parameters=-nolow -species mouse -s
module=RepeatMask
gff_source=RepeatMask
gff_feature=repeat
input_id_type=CONTIG
```

Instructions

- Now, add the RepeatMasker analysis to your database using the analysis_setup.pl script. You will find example files of how to set up different analyses in this folder :

```
$HOME/ensembl-src/ensembl-config/mouse/NCBIM37/pipe_conf/
```

- Have a look at the analysis table of your database with the MySQL client - you should see an entry with the logic_name "RepeatMask" in there.

(b) Adding an initial Submission-analysis to our database ...

The next thing we'll do is add a Submission analysis to your database. Here's some more info about Submission analyses, analysis rules and different types of input_ids.

- If we would like to run the analysis RepeatMask on the Contig level, then our RepeatMask analysis needs to have input_id_type of `CONTIG`. Check this by querying the input_id_type_analysis table.
- All analyses (eg. RepeatMask, Genscan, Exonerate) need a ‘dummy analysis’, also known as a Submission analysis, which has the same `input_id_type` as the analysis we’d like to run.
- Two analyses can share the same dummy analysis if they both have the same input_id_type. Example : if both RepeatMask and Pmatch are to be run on the contig level, then there only needs to be one Submission analysis in the analysis table - let's call this one "SubmitContig". However, if RepeatMask is to be run on Contig level and Pmatch is to be run on the Chromosome level, then they will each need their own dummy analysis – SubmitContig and SubmitChromosome respectively.

- Every analysis requires at least one rule that defines the conditions under which the analysis is allowed to run. RepeatMask is allowed to run if SubmitContig has been run. Other analyses (eg. alignment of proteins) might require both the dummy analysis and another analysis (eg. RepeatMask) to have been successful.
- Submission (dummy) analyses do not require rules
- Each analysis usually has its own 'accumulator', although this is not required. An accumulator for Pmatch would be called Pmatch_wait. It is only allowed to run after ALL Pmatch jobs have run successfully. Pmatch-dependent analyses will have Pmatch_wait as a condition. Thus, the accumulator prevents any jobs from Pmatch-dependent analyses from running until all Pmatch jobs are done.
- This information only scratches the surface to enable you to run the examples in this tutorial. You will find a lot more detailed information in the cvs files :

ensembl-docs/pipeline_docs/ the_genebuild_process.txt
ensembl-docs/pipeline_docs/ the_raw_computes.txt

- You'll find an example input file for the analysis_setup.pl script here :

```
vi \
$HOME/ensembl-src/ensembl-config/mouse/NCBIM37/pipe_conf/submit_contig_ana.conf
```

Instructions

- Use the analysis_setup.pl script from the previous section to now read the `submit_contig_ana.conf` to add a Submission analysis to your database.

```
perl $HOME/ensembl-src/ensembl-pipeline/scripts/analysis_setup.pl \
-dbhost localhost -dbuser ens-training -dbport 3306\
-dbname  mouse37_mini_ref\
-dbpass workshop -read \
-file $HOME/ensembl-src/ensembl-
config/mouse/NCBIM37/pipe_conf/submit_contig_ana.conf
```

- After you've added the Submission analysis, have another look into the analysis table of your database - it should now contain 2 entries.

2. Add rules for the analysis to the database, and make input_ids

(a) Add rules

- To combine the RepeatMask analysis with it's Submission analysis, we have to add `rule_conditions` and `rule_goals` to our database. We'll use the RuleHandler.pl script to do this. You'll find useful options by running this :

```
$HOME/ensembl-src/ensembl-pipeline/scripts/RuleHandler.pl -help
```

- Adding rules to your database with RuleHandler will write to the `rule_conditions` and `rule_goal` tables of your database :

```
perl $HOME/ensembl-src/ensembl-pipeline/scripts/RuleHandler.pl
-dbhost localhost -dbuser ens-training -dbport 3306\
-dbpass workshop\
-dbname mouse37_mini_ref -insert -goal RepeatMask \
-condition SubmitContig
```

- RuleHandler can also be used to list stored rules in the database table. Use the -rules option for the script - you should see something like this :

```
Rule ID:1
    conditions: SubmitContig (1)
    goal:       RepeatMask (2)
```

Instructions

- Add rules to your database for RepeatMasker and query your database to see what has been stored.

(b) Make input_ids to run the analysis on CONTIGS :

- Before we can run RepeatMasker we need to create input_ids.

- Each input_id becomes a job to run on our compute farm. By chopping the genome up into little bits (contigs, chromosomes, slices, or other input_id_types), we can run multiple jobs in parallel.

- If the input_id_type is CONTIG, then we create an entry in the input_id_analysis table for every contig.

- If the input_id_type is CHROMOSME, then we create an entry in the input_id_analysis table for every chromosome. If the input_id_type is 1MSLICE, then we create an entry in the input_id_analysis table for every slice of sequence in the genome of 1 megabase length.

- To create input_ids for Chromosomes, Contigs or Slices, we use the make_input_ids script. You'll find it here :

```
$HOME/ensembl-src/ensembl-pipeline/scripts/make_input_ids
```

- If you use the -help option, you'll find example commandlines to run this script on your database. The script will write into the `input_id_analysis` and `input_id_type_analysis` tables.

- Run the make_input_ids script on your database and check with the MySQL client if input_ids have been written to your database.

  Hint - an example commandline needs to include something like these options :

```
-logic_name SubmitContig -slice -coord_system contig
```

- After you've added the input_ids to the database, list the content of the input_id_analysis table with the MySQL client. You should see something like this ;

```
mysql -uens-training -pworkshop -hlocalhost -P3306 \
-D mouse37_mini_ref\
select ia.*,a.logic_name from input_id_analysis ia, analysis a where
ia.analysis_id = a.analysis_id ;

+-------------------------------+--------------+-------------+-----------------
---+---------+------------+--------+--------------+
| input_id                      | input_id_type | analysis_id |
created             | runhost | db_version | result | logic_name   |
+-------------------------------+--------------+-------------+-----------------
---+---------+------------+--------+--------------+
| contig::AC087062.25:1:224451:1 | CONTIG        |           2 | 2008-09-09
12:22:36 |         |            |      0 | SubmitContig |
| contig::AC138620.4:1:209846:1  | CONTIG        |           2 | 2008-09-09
12:22:36 |         |            |      0 | SubmitContig |
| contig::AC153919.8:1:264561:1  | CONTIG        |           2 | 2008-09-09
12:22:36 |         |            |      0 | SubmitContig |
| contig::AL589742.21:1:125641:1 | CONTIG        |           2 | 2008-09-09
12:22:36 |         |            |      0 | SubmitContig |
| contig::AL596185.12:1:248203:1 | CONTIG        |           2 | 2008-09-09
12:22:36 |         |            |      0 | SubmitContig |
| contig::AL732620.14:1:230116:1 | CONTIG        |           2 | 2008-09-09
12:22:36 |         |            |      0 | SubmitContig |
+-------------------------------+--------------+-------------+-----------------
---+---------+------------+--------+--------------+
```

3. Fill out configuration files like BatchQueue.pm

The configuration of the analyses you'd like to run and the pipeline infrastructure itself is split over several files. These files live here :

(i) /home/training/ensembl-src/ensembl-config/mouse/NCBIM37/Bio/EnsEMBL/Pipeline/Config
(ii) /home/training/ensembl-src/ensembl-config/mouse/NCBIM37/Bio/EnsEMBL/Analysis/Config

- The main file for the pipeline control is BatchQueue.pm. You'll find the BatchQueue.pm configuration file here :

```
/home/training/ensembl-src/ensembl-config/mouse/NCBIM37/Bio/EnsEMBL/Pipeline/Config/BatchQueue.pm
```

- BatchQueue.pm contains data for each analysis eg. Output directories, the batch size you like to use, and database resource requirements.

4. Check the setup by running a small dry run

- Before running any job on the farm, it's a good idea to first run a job in test_RunnableDB ('dry run'). This will catch any issues with your PERL5LIB and configuration setup.

- We have already set some of the config files for you, but it's worth having a look at them anyway. In particular, have a look at BatchQueue.pm and check that the output directory specified for RepeatMasker exists.

- If you use the -help option on $HOME/ensembl-src/ensembl-analysis/scripts/test_RunnableDB, you'll get more information about the script and example commandlines. An example commandline should use options like -analysis and -input_id as well as the standard parameters for dbhost, dbuser, dbpass and dbport.

```
perl $HOME/ensembl-src/ensembl-analysis/scripts/test_RunnableDB\
 -logic_name RepeatMask \
-input_id contig::AC087062.25:1:224451:1 \
-dbhost localhost -dbuser ens-training -dbport 3306 \
-dbname mouse37_mini_ref -dbpass workshop -verbose
```

5. Run the pipeline on your machine locally

```
perl  $HOME/ensembl-src/ensembl-pipeline/scripts/rulemanager.pl\
-logic_name RepeatMask \
-dbhost localhost -dbuser ens-training -dbport 3306 \
-dbname mouse37_mini_ref -dbpass workshop
```

- The repeatmasking analysis takes about 3.5 hours to run on an Intel 3.00 GHZ machine. To save time, we've already computed the `repeat_featues` for our contigs. You can load them into your database with the command :

```
mysql -u ens-training -pworkshop mouse37_mini_ref \
 < $HOME/ensembl-genebuild-data/sql_dumps/repeat_features.sql

We also need to add the input ids to the database :

mysql -u ens-training -pworkshop mouse37_mini_ref \
 < $HOME/ensembl-genebuild-data/sql_dumps/repeatmask_input_ids.sql
```

- Now check the `repeat_feature` and the `repeat_consensus` tables of your database to check that the features have been loaded into the database.

- Check the loaded sequence by dumping it out . The dumping takes about 60 seconds. You can use the sequence_dump.pl script which was used in Exercise 1.

```
mkdir -p /home/training/ensembl-data/genebuild/output/unmasked_seq

perl $HOME/ensembl-src/ensembl-analysis/scripts/sequence_dump.pl\
-dbhost localhost -dbuser ens-training -dbport 3306\
 -dbname  mouse37_mini_ref -mask -softmask -mask_repeat RepeatMask\
-dbpass workshop -coord_system_name chromosome \
-output_dir /home/training/ensembl-data/genebuild/output/softmasked_seq
```

  - Use the 'tail' command to look at some of your masked sequences in the output directory. Repeats should appear as small characters.

6. Check your results

Repeats from the RepeatMask run should now have finished running - hooray! Let's check how much sequence has been masked :

```
perl /home/training/ensembl-genebuild-data/scripts/repeat_coverage.pl\
-repeattypes RepeatMask\
-repeattypes RepeatMask -path NCBIM37\
-dbhost localhost -dbuser ens-training -dbport 3306 \
-dbname mouse37_mini_ref -dbpass workshop
```

- Look in the repeat_feature and repeat_consensus table:

```
mysql -u ens-workshop -p training -P 3306 -h locahost
```

* * * End of Exercise * * *