Birkbeck University of London
Bioinformatics and Systems Biology
Biocomputing II

# Group Project
# Documentation for database

Team: ChromAwesome_22
**Kostas Pantelidis**

One should run the files in the following order to run the database:

1) `parser.py`
2) `create_table_and_populate.py`

## `parser.py`

This is the script where l am parsing data from the GenBank file *chrom_CDS_22.gz*

The idea is to split the file in entries. It is noticed that every entry ends with "//", so this is where the splitting is done. By doing that I found that there are 1184 entries/genes in chromosome 22.

Then, I am starting parsing for every piece of data I need using Python Regular Expressions. In case of more than one data per entry/gene, the instructions are to take the very first one only.

**I was asked by my team (in the very first team meeting) to give back 10 specific data**:

accession
(it is unique for every entry, it's the main guide to check and find requested data)
gene_bp
(the number of pair bases per entry/gene)
gene_id
(gene name)
location
(the location of each entry/gene, getting it from *map* in GenBank file)
dna_seq
(the dna sequence extracted from *ORIGIN* in GenBank file)
cds
(returns the whole data under *CDS* (always the first one) in GenBank file)
complement
(if it is complement or not)
protein_id
(protein name)
product
(the product of every gene/entry)
translation
(the protein sequence)

After the parsing, the data is saved into lists
(eg accession data to *accession_list*, gene_id to *gene_id_list* etc.)
and then, **10 lists** of 1184 length each are created.

Finally, I create a list of these lists called **list_of_entries**, so that I can use it later to populate my database.

```
from parser.py import list_of_entries
```

## `create_table_and_populate.py`

The script where I create the table **"entries"** in the database. I only create one table.

I firstly connect in my own personal database in BBK's server.

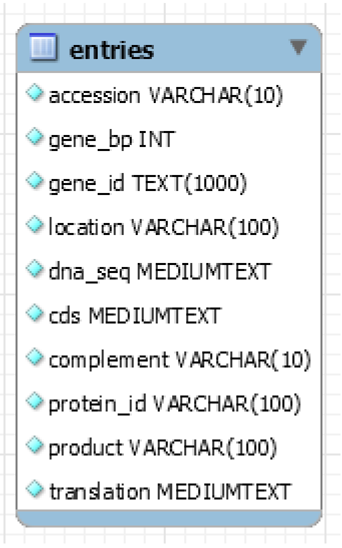
Using pymysql I make sql queries.

I create only one table that has 10 columns, one for each piece of data I was asked to give back.

The script uses *try* and tries to create a table but in case the table already exists then drops the table and create a new one.

After the table creation, the script populates the data using the **`list_of_entries`** from **`parser.py`** file

A print message is added for the user to ensure that the table has created and populated successfully.



```
entries
accession VARCHAR(10)
gene_bp INT
gene_id TEXT(1000)
location VARCHAR(100)
dna_seq MEDIUMTEXT
cds MEDIUMTEXT
complement VARCHAR(10)
protein_id VARCHAR(100)
product VARCHAR(100)
translation MEDIUMTEXT
```

Additional script/files:

## `RUN_database.sh`

A .sh script that the user can run in terminal and automatically can download chrom_CDS_22.gz file, then parse it and finally create and populate the table.

A `README.md` file