**Step-by-step Guidance for CTAMDB**

# Introduction

This document introduces the core database behind the CTAMDB and its related database population from Pescal++ result files and applications (statistical analyses and RESTful service). In this document, the base folder is the shared dropbox folder “**CTAM project**” which will be referenced as BASE\_DIR. For example, this document is in the **BASE\_DIR\documents**.

The conventions throughout this document are 1) all file names and folders are highlighted in bold, 2) all commands needed to be run are highlighted in bold and italic, 3) the source codes and contents of files are highlighted in italic and 4) comments are started with a “#” sign.

# Database design and creation

The chosen database system is MySQL and designed with MySQL Workbench1, within which the database schema file “**ctam database design.mwb**” can be opened. To convert the design into the executable SQL file, under “File” menu click “Export” menu item and then “Forward Engineer SQL CREATE script…” option, input the SQL file name, in this case it is called “**ctamdb creation.sql**”.

**Note**: all files mentioned in the “Database design and creation” section can be found in the folder **BASE\_DIR\documents\databases**.

Under the command prompt, in the **BASE\_DIR\documents\databases** folder use the following command to create the database assuming that there is a “root” user who has the full privilege to operate on the database.

***mysql –u root –p < "ctamdb creation.sql"***

# Connect to the database using Perl

Perl uses DBI (Database Interface) module as the standard database interface which defines a set of methods to provide a consistent database interface independent of the actual database system being used. DBD::mysql is the driver for connecting to MySQL with DBI.

To connect to a database, four pieces of information are required: host, database schema, user and password. In CTAM project, the database schema is fixed to “ctam”. The database credential is dealt with by an in-house Perl library, called **misc.pl**, which reads the credential information from a text file called **credential.txt** and manages to connect to the specified database. In the **credential.txt**, it is sectioned according to different database profiles. The profile name is wrapped in the [] and later used in **misc.pl**. In each profile three fields (host, user and password) are required, see the example below. A blank line is needed to separate sections.

*[localhost]*

*host=localhost*

*user=ctam*

*password=ctam*

This way avoids to contain such sensitive information in the Perl script to make it possible to host Perl scripts on public GitHub repository and provides a flexible method to connect to different database instances by simply changing the chosen profile.

To use the **misc.pl** library, in the Perl script, use the statement

*require “misc.pl”;*

*my $dbi = &getDBI(“localhost”);* #connect to the database using the profile titled with “localhost”

# Database population

## Proteins

Proteins are essential in the CTAM project. Swissprot2 is a well-curated and widely-accepted protein database. Canonical human protein sequences are downloaded from Swissprot search result3 as a single fasta file and parsed by the script “**parseFastaAndPopulateProteinTable.pl**” which extracts protein id, accession number, gene and description from the fasta header line and populates into the protein table.

This step is not necessary, but highly recommended because during the importing PSM step, an internet search will be executed if the identified protein is not found in the database. This step will dramatically reduce the number of such searches, hence increase the efficiency and avoid the possible error due to broken network connection. However the pre-populated protein table could have the issue of being out-of-date, which can be overcome by updating the protein table using the same script with a new release of fasta file.

“**pkinfam kinase database.csv**” given by Pedro is a CSV file containing known kinases with Uniprot accessions which is not table-ready. **parsePkinfamCSV.pl** extracts the information and output into another CSV file “**resultKinase.csv**” which contains three columns: common name, accession number and group. **populateKinaseTable.pl** parses the newly generated CSV file, inserts into kinase table and reports any protein which needs to retrieve related information from the internet.

The related commands in this sections are

***perl parseFastaAndPopulateProteinTable.pl “Swissprot human May 2017.fasta”***

***perl populateKinaseTable.pl***

## Metadata

The metadata include the cell lines, regulators and modifications used in the experiments. An excel file called “**metadata information.xlsx**” contains such information which is manually curated. As number of cell lines and modifications used is very limited, they are “hard-coded” directly as the SQL statements which are saved in “**populate database example.sql**”. For regulators, a separate TSV file is generated by copying the information contained in the “regulators” sheet without including the heading. A Perl script called “**populatedRegulatorTable.pl**” is developed to parse the TSV file and populate the regulator and regulator\_kinase tables.

#load the predefined cell lines and modifications

***mysql –u root –p < “populate database example.sql”***

#populate the regulator table

***perl populateRegulatorTable.pl***

Now the database is ready to accommodate experimental data outputted by Pescal++.

# Intermediate CSV files from Pescal++

Pescal++ is a PSI(Proteomics Standards Initiative) standards-compliant C++-based software used to quantify peptides in shotgun proteomics experiments. Peptide spectrum matches (PSMs) are read into memory by parsing a DAT file (Mascot identification file format) or an mzIdentML file containing peptide identifications for each mass spectrometry run. Unique PSMs from each run are concatenated to generate a complete list of unique peptide ions identified across the entire experiment. Using a peptide ion’s mass-to-charge (*m/z*) ratio and retention time, extracted ion chromatograms (XICs) are retrieved from the spectral files (Thermo proprietary RAW format) and the peptide ion features (which contain multiple isotopic peaks in *m/z* space) are quantified by summing peak areas across *m/z* and retention time space. Some peptides may not have been identified in a given run, and so retention times are calculated for these peptide ions based on their retention times when present in other runs, therefore making it possible to retrieve an XIC and perform quantitation for every identified peptide in every sample.

All files mentioned in this section can be found in the folder **BASE\_DIR\documents\Pescal++**, most of which are simplified (i.e. incomplete) sample files, i.e. the program may encounter errors purely using these sample files.

## Experiment list

The concept of “experiment” here is a collection of MS runs among which the identified peptides are aligned and quantitation is calculated. It is a logical concept which allows the same run(s) to appear in multiple experiments.

The file “**experiment list.csv.sample**”, as its name suggests, is a sample file within which the experiments to be imported are listed. The “sample” suffix indicates that the file is used as the sample only. When it is actually to be used, it needs to remove the sample suffix, e.g. the experiment file will be renamed to “**experiment list.csv**” because this file name is hard-coded in the script. This renaming business applies to all other files with “sample” suffix in the folder. In the experiment list file, the first line is the header line within which there are six columns: experiment name, folder name (defining where to look for experiment-specific files which will be described below), submitter, affiliation, date, and description. The experiments can be toggled by adding/removing the leading “#” sign of each line. Note the file separator used in the folder name column is “//” which is supposed to be both windows and linux compatible. All six columns except date column allow free text while date column must use the format of “*DD-MM-YY*” which is same as the allowed date type in MySQL and maybe need to be changed if other database system is used.

## Experiment-specific files

In the folder specified by the “folder name” in the “**experiment list.csv**”, three types of experiment-specific files are expected: 1) run list, 2) PSM-related files, and 3) quantitation-related files. 2) and 3) are

the result files generated by Pescal++, they are briefly introduced here for completeness, please refer to the documents of Pescal++ for more details.

### Run list

A run is an MS run represented by a spectral file, on which the peptide spectrum matching (PSM) is performed by a search engine. In the current CTAM project, the search engine is limited to Mascot which generates .DAT file as its search result file containing list of PSMs with related measurements. The .DAT files then are converted into mzIdentML files. The run list defines the relationship between spectral files and corresponding identification files. On top of that, the group of the run, cell line and regulator(s) used, search engine and search parameters are included in the run list file (“**metadata.csv.sample**”). The group can be assigned based on anything, e.g. the time series, treatment vs regulators etc. There is a conserved group name, “control”. If a column contains multiple entities, the entities are concatenated with commas and wrapped between a pair of quotation marks to form the string for that column, for example, “Oxidation(M), Phospho (ST)”.

### PSM-related files

Information of PSMs is extracted from the identification file either in DAT or mzIdentML format and saved in the F\*\*\*\*\*\*.csv, where the asterisks represent six digits and match to the original DAT file. In one identification file, one peptide could be identified by several spectra, the spectrum with the highest score is treated as the “best PSM” by Pescal++ and listed in the pF\*\*\*\*\*\*.csv where F\*\*\*\*\*\* part matches the corresponding PSM CSV file.

### Quantitation-related files

In CTAM project, a unique peptide is determined by the combination of peptide sequence, protein, charge and modification. The peptides identified from best PSMs across all runs within the same experiment are aligned according to the retention time. All identified peptides from the experiment with observed retention time and other measurements are listed in “**combiPeptData.csv**”, in which the “*db ID*” column defines the peptide id which only applies within the experiment, i.e. same peptide in different experiments could have different peptide id. And the calculated (after alignment) retention time and quantitation values are listed in “**calculatedRTs.csv**” and “**peakAreas.csv**” respectively in which the first column contains the peptide id serving as the references among the three quantitation-related files.

### Populate the database

“**populatePescalPPintoMySQL\_final.pl**” reads the “**experiment list.csv**” to understand what experiments are going to be imported. For each experiment, the script reads all files under the folder which is specified under the “folder name” in the “**experiment list.csv**”. Firstly it reads the “**metadata.csv**” to establish the relationship between spectral files and identification files and other metadata. Then it parses all PSM files specified in the “**metadata.csv**” and populates the *psm* table, followed by analyse corresponding best PSM files. When the PSM-related files are done, “**combiPeptData.csv**” is parsed to populate *peptide\_in\_experiment* table and *observed\_RT* table and extract the database id for each peptide. These peptide ids are facilitated to populate *calculated\_RT* and *peak\_area* tables while reading “**calculatedRTs.csv**” and “**peakAreas.csv**”. The command to run this script is simple as all file names are fixed.

***perl populatePescalPPintoMySQL.pl***

# Statistical analysis

After populate the quantitative data, the database is ready to determine which phosphosites have significant quantitation value change. The basic algorithm is 1) for the given experiment, read all runs from that experiments and the corresponding group and cell lines 2) for each available combination of group and cell line, retrieve the quantitation data 3) within each combination calculate count of non-zero values, mean and standard derivation for each peptide 4) normalization either by quantile method or by original Pedro’s method 5) calculate the fold change and p-value for each peptide within every group other than control group against the corresponding control group which is labelled with the conserved group name “control” from the same cell line 6) output all phosphosites into a CSV file. These steps are implemented in the file “**ctam statisticas.r**”.

To populate the phosphosites into the database, a separate Perl script “**savePhosphoSite.pl**” has been developed which reads in the CSV file generated in the previous step for the given experiment, finds the corresponding peptide id and cell line id, counts the identified times of all peptides in both treatment and control group and populates the *final\_result* table. Both files can be found under the **BASE\_DIR\documents\statistics** folder. The commands are

***RScript “ctam statistics.r” 17 column***#17 is the experiment id and the value of “column” indicates using Pedro’s method

***perl savePhosphoSite.pl 17***

# RESTful service

Representational State Transfer (REST) or RESTful service allows users to access and manipulate textual representations of web resources (in our case, the content of CTAMDB) using a uniform and predefined set of operations. For the security reason, only access operations are allowed. The chosen web service framework is Dancer24 which is a simple but powerful framework for Perl. The web service is hosted at ctamdb.sbcs.qmul.ac.uk and the /api route lists all available APIs (Application Programming Interface). All routes generate results in the TSV format. More details can be found in a separate document **API\_Docs.docx** under the folder **BASE\_DIR\documents** which demonstrates how to use CTAMDB RESTful service.

The CTAMDB RESTful service is generated by the command ***dancer2 -a CtamWeb*** and followed by modifying the auto-generated **CtamWeb\lib\CtamWeb.pm** to add routes. For convenience a copy of **CtamWeb.pm** (version 0.8.2 last modified on 29 Apr 2017) is stored under the folder **BASE\_DIR\documents\RESTful**.

To start the RESTful service on gio2 server, the commands are

***cd ~junf/CtamWeb*** #the main CtamWeb.pm is under ~junf/CtabWeb/lib

***plackup –E deployment –s Starman --workers =10 –p 5000 -r -R lib bin/app.psgi -a bin/app.psgi*** #allow 10 connections to access the server at the same time using Starman (the perl module *Plack::Handler::Starman* needs to be installed first) –r auto reload –R tell which folder to monitor

The above two commands are recommended to be executed under the “screen5” command, which allows the users to run scripts in their own virtual window within the terminal, to keep the RESTful service continuously running.