RetroMiner to RTPEA - Populating the databases

Mohamed Nazrath Mohamed Nawaz | September 25, 2018

templates	input	script	output	database	archive

PeptideShaker output (.cpsx)

100 — custom export
- PeptideShaker's CLI
- runs custom report exportation
- filters results: LINE / HERV files

small text files (.txt)
- per sample
- per dataset
- per protein

Paper

Experimental Design (.txt)

300 — parser
- extracts protein level results for RT

filters

small text files (.txt)
- samples combined
- per dataset
- per protein
- with exp design

50 — output table

combined table (.txt)
- single super table
- samples combined
- datasets combined
- proteins combined

→ archive

metadata (.txt)

size data (.txt)

100 — add metadata
- add manually collated metadata to results

combined table + metdata (.txt)
- diseases
- tissue types
- study

→ archive

example structure file (.json)

300 — json conversion
- complete results in easy form
- creates result table jsons

counter

many json files (.json)
- data for table
- per dataset
- samples combined

→ archive

sequence information (.fasta)

example structure file (.json)

250 — protvista
- aligns variant to consensus sequences
- extracts variants from result jsons
- creates new jsons with variation information

many json files (.json)
- per variant
- per dataset
- per protein
- per family

Database → archive

example structure file (.json)

50 — combine protvista
- to view all information on protein page
- all variants etc.

protein json (.json)
- per protein

Database → archive

Reference Genome (.fasta)

sequence information (.fasta)

example structure file (.json)

100 — ideogram
- generate chromosome level info

chr json (.json)
- one file with all chromosomes

Database → archive

PRIDE API (.json)

size data (.txt)

100 — sizes
- sum spetral file sizes for bar chart

size table (.txt)
- table with dataset and sizes

Database

200-300 — 3-5 more
- archiving
- getting list of id's
- small file manipulations
- bar chart
- error catching
- organising structure