

S2 Text. Examples description

Files needed by each example were placed in example directories:

- *Example_1*
- *Example_2*
- *Example_3*
- *Example_4*
- *Example_5*

Each example directory contains subdirectories: *input_data*, *queries* and *results*.

Subdirectory *input_data* contains all input data needed to do all example tasks.

Subdirectory *queries* contains all GVC queries used by example and may be used when user do not want to manually (or do not know how to) properly fill query forms to filter data.

Subdirectory *results* contain results for each example stage allowing to fast jump to any other example stage without need to conduct all stages (some of them are time-consuming).

Example 1 contains set of easy, unassociated, one stage tasks demonstrating unique and most useful features of software using very simple artificial dataset. It may be used as reference when conducting other examples. Datasets are so small and simple that it was possible to show input and output data in the form of color tables to make understanding more easy.

Other examples contain real, scientific datasets and are more complex (many stages) and were presented as flowcharts with descriptions and notes.

In order to make text more readable some conventions were used:

- file names and paths are written using *mono font and coloured darkgreen*.
- software menu options were *italicized and mono font was used*.
- data tables column names were in *coloured purple*.
- rejected data rows are ~~strikethrough~~
- stage descriptions on flowcharts are in blue rectangles and notes are in gray rectangles.

Example 1

Directory structure:

Example_1 – main directory containing example files

Example_1/input_data/ – input files for individual stages

Example_1/results – output files for individual stages

Example_1/queries – query files for individual stages

```
.
├── input_data
│   ├── example_input_file1.txt
│   ├── example_input_file2a.txt
│   ├── example_input_file2b.txt
│   ├── example_input_file3.txt
│   ├── example_input_file4a.txt
│   └── example_input_file4b.txt
├── queries
│   ├── q1.que
│   ├── q2.que
│   ├── q3.que
│   ├── q4.que
│   ├── q5.que
│   ├── q6.que
│   ├── q7.que
│   ├── q8.que
│   └── q9.que
└── results
    ├── r1.txt
    ├── r2.txt
    ├── r3.txt
    ├── r4.txt
    ├── r5.txt
    ├── r6.txt
    ├── r7.txt
    ├── r8.txt
    └── r9.txt
```

Filtering on the base of samples (column SAMPLE) where only proteins (column PROTEIN) present in > 50% of samples are selected

SAMPLE	PROTEIN	SCORE
sample1	protein1	0.1
sample1	protein2	0.5
sample1	protein3	0.8
sample1	protein4	0.9
sample2	protein1	0.6
sample2	protein4	0.8
sample2	protein8	1.0
sample2	protein9	0.3
sample3	protein1	0.4
sample3	protein12	0.6
sample4	protein13	0.6
sample4	protein14	1.1

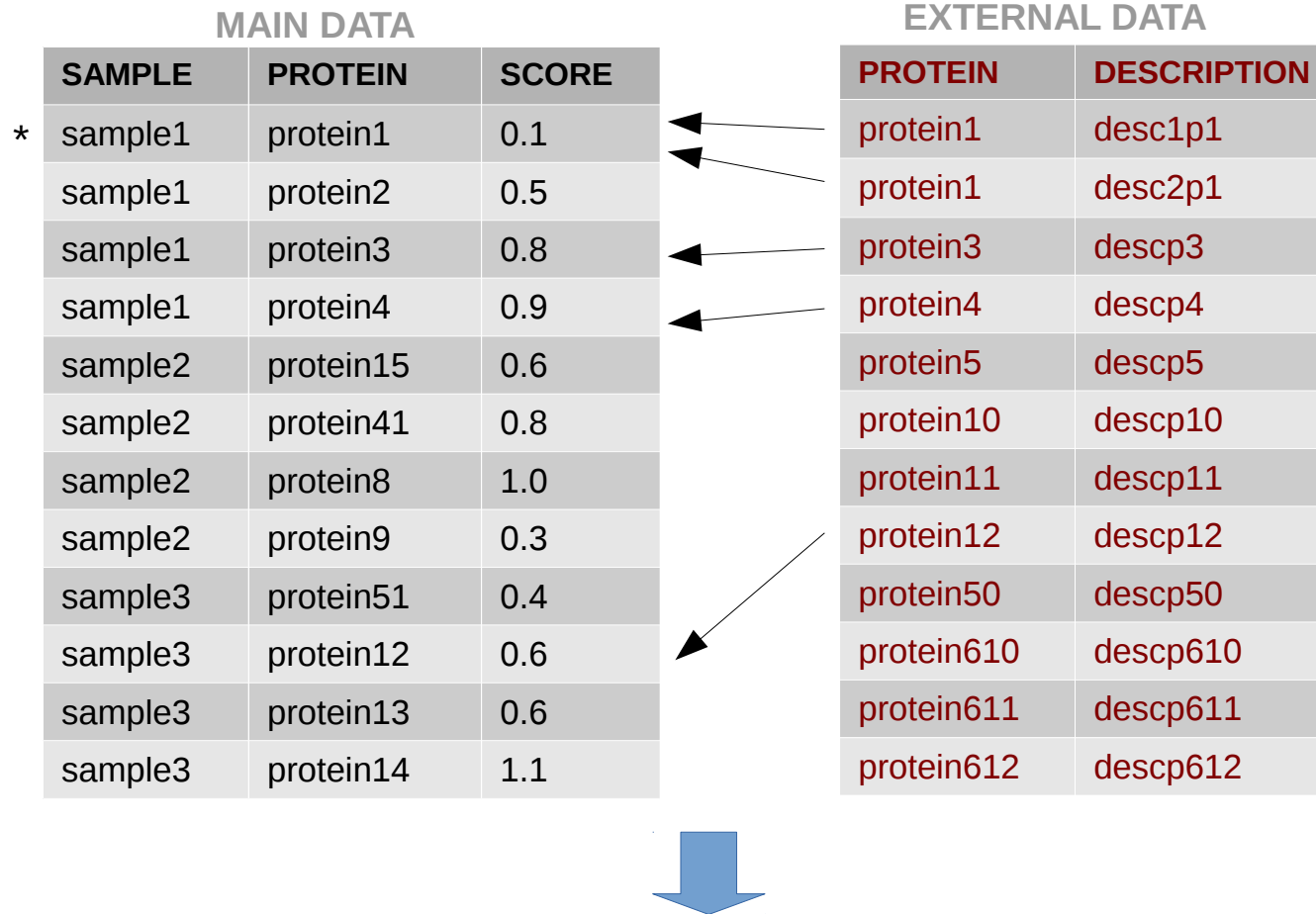


SAMPLE	PROTEIN	SCORE
sample1	protein1	0.1
sample1	protein2	0.5
sample1	protein3	0.8
sample1	protein4	0.9
sample2	protein1	0.6
sample2	protein4	0.8
sample2	protein8	1.0
sample2	protein9	0.3
sample3	protein1	0.4
sample3	protein12	0.6
sample4	protein13	0.6
sample4	protein14	1.1

menu: *File* -> *Open File* -> Example_1/input_data/example_input_file1.txt
 menu: *View* -> *Advanced mode*
 menu: *Queue* -> *Add query* -> *Sample Filter:*
 Sample columns: **SAMPLE**
 Analysed columns: **PROTEIN**
 IS PRESENT IN 50% OF SAMPLES

Note:
 Query is saved as file
 Example_1/queries/q1.que
 Saved result file is
 Example_1/results/r1.txt

Horizontal joining of the files on the base of one column (column PROTEIN)



*- „REPEAT DATA ROW WHEN MORE THAN ONE EXTERNAL DATA ROWS MATCH” option active

* - system columns from external data not showed



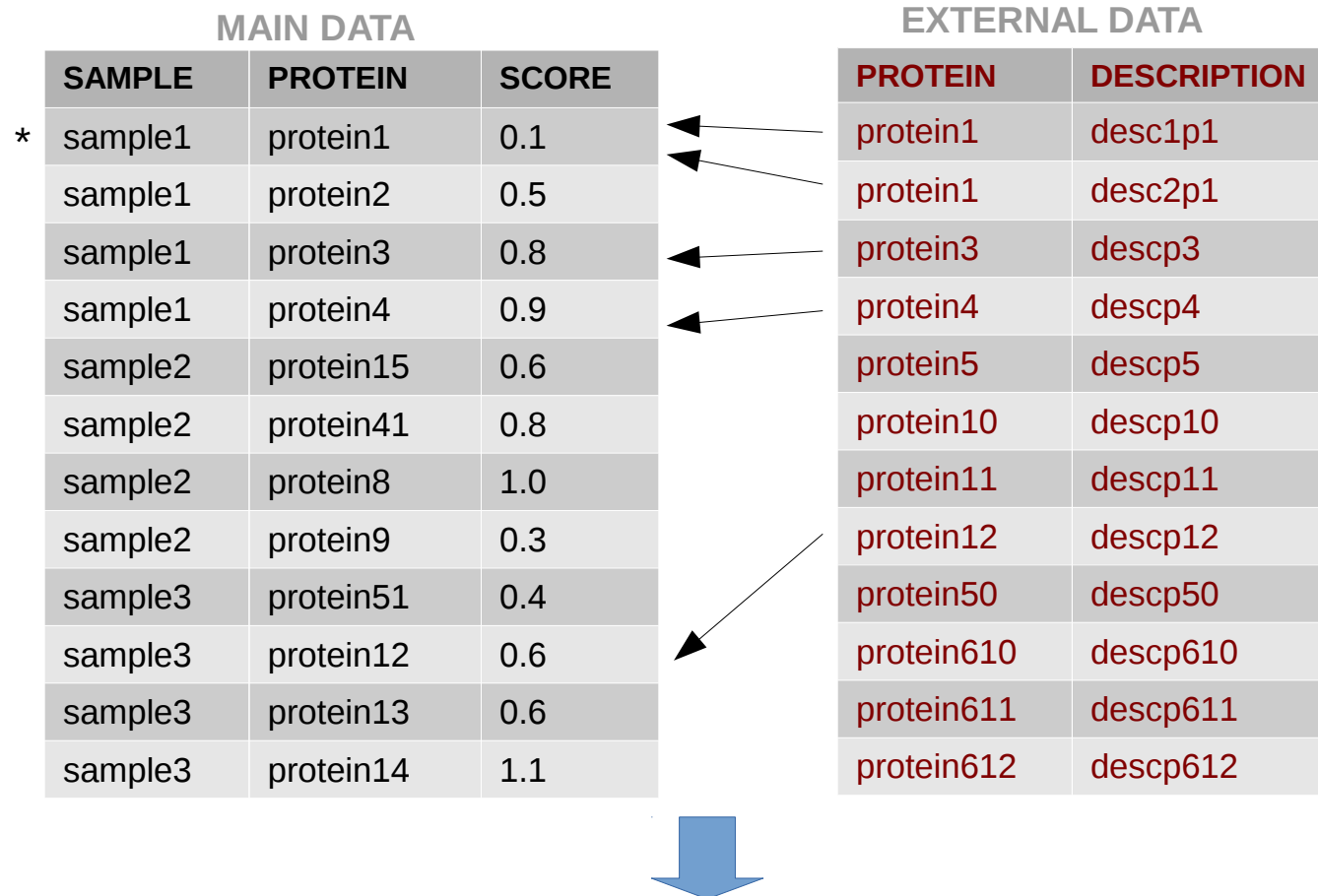
RESULT

SAMPLE	PROTEIN	SCORE	PROTEIN	DESCRIPTION
sample1	protein1	0.1	protein1	desc1p1
sample1	protein1	0.1	protein1	desc2p1
sample1	protein2	0.5		
sample1	protein3	0.8	protein3	descp3
sample1	protein4	0.9	protein4	descp4
sample2	protein1	0.6		
sample2	protein4	0.8		
sample2	protein8	1.0		
sample2	protein9	0.3		
sample3	protein51	0.4		
sample3	protein12	0.6	protein12	descp12
sample3	protein13	0.6		
sample3	protein14	1.1		

menu: *File* -> *Open File* -> `Example_1/input_data/example_input_file2a.txt`
 menu: *File* -> *Open Second File* -> `Example_1/input_data/example_input_file2b.txt`
 menu: *View* -> *Advanced mode*
 menu: *Queue* -> *Add query* -> *External Data Filter/Merger*:
 Select columns and condition: Columns: Column: **PROTEIN**
 Select columns and condition: Columns [external data]: Column: **PROTEIN**
 Select columns and condition: Condition: =
 Select action: Add all columns from external data matching rows
 Select action: do not filter - merge only
 Select action: repeat data row when more than one external data rows match

Note:
 Query is saved as file
`Example_1/queries/q2.que`
 Saved result file is
`Example_1/results/r2.txt`

Horizontal joining of the files on the base of one column (column PROTEIN)
with filtering



*- „REPEAT DATA ROW WHEN MORE THAN ONE
EXTERNAL DATA ROWS MATCH” option active

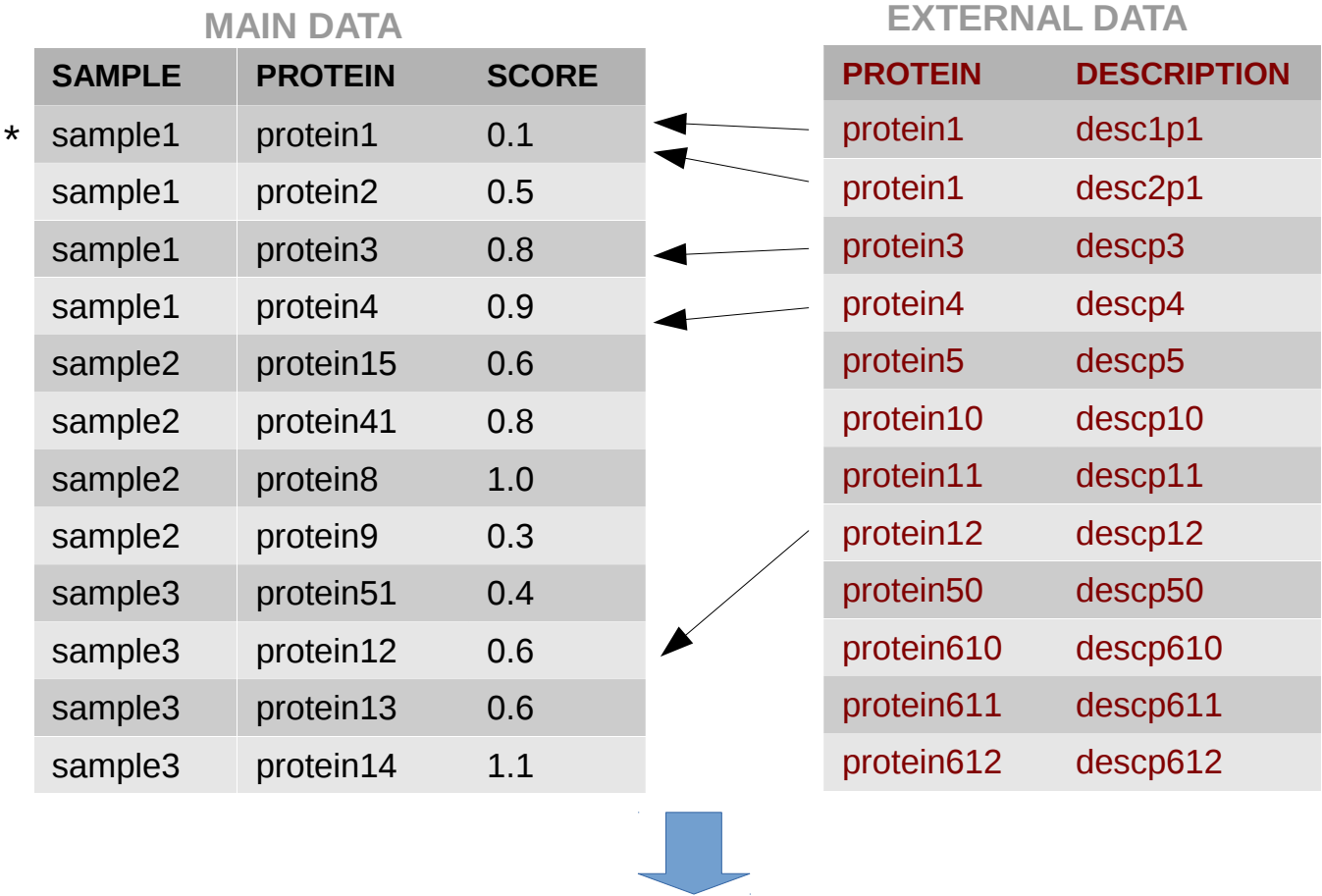


SAMPLE	PROTEIN	SCORE	PROTEIN	DESCRIPTION
sample1	protein1	0.1	protein1	desc1p1
sample1	protein1	0.1	protein1	desc2p1
sample1	protein3	0.8	protein3	descp3
sample1	protein4	0.9	protein4	descp4
sample3	protein12	0.6	protein12	descp12

menu: *File* -> *Open File* -> `Example_1/input_data/example_input_file2a.txt`
menu: *File* -> *Open Second File* -> `Example_1/input_data/example_input_file2b.txt`
menu: *View* -> *Advanced mode*
menu: *Queue* -> *Add query* -> *External Data Filter/Merger*:
 Select columns and condition: Columns: Column: **PROTEIN**
 Select columns and condition: Columns [external data]: Column: **PROTEIN**
 Select columns and condition: Condition: =
 Select action: Add all columns from external data matching rows
 Select action: repeat data row when more than one external data rows match

Note:
Query is saved as file
`Example_1/queries/q3.que`
Saved result file is
`Example_1/results/r3.txt`

Filtering one file using another file on the base of one column (column PROTEIN) without joining



*- „REPEAT DATA ROW WHEN MORE THAN ONE EXTERNAL DATA ROWS MATCH” option active



SAMPLE	PROTEIN	SCORE
sample1	protein1	0.1
sample1	protein3	0.8
sample1	protein4	0.9
sample3	protein12	0.6

menu: *File* -> *Open File* -> *Example_1/input_data/example_input_file2a.txt*
menu: *File* -> *Open Second File* -> *Example_1/input_data/example_input_file2b.txt*
menu: *View* -> *Advanced mode*
menu: *Queue* -> *Add query* -> *External Data Filter/Merger*:
 Select columns and condition: Columns: Column: **PROTEIN**
 Select columns and condition: Columns [external data]: Column: **PROTEIN**
 Select columns and condition: Condition: =
 Select action: Filter only

Note:
Query is saved as file
Example_1/queries/q4.que
Saved result file is
Example_1/results/r4.txt

Selecting unique rows on the base of two columns
(columns PROTEIN and REGION)

SAMPLE	PROTEIN	REGION
sample1	protein1	20
sample1	protein2	5
sample1	protein3	8
sample1	protein4	9
sample2	protein1	6
sample2	protein4	9
sample2	protein8	1
sample2	protein9	33
sample3	protein1	4
sample3	protein12	0
sample3	protein9	33
sample3	protein9	33



SAMPLE	PROTEIN	REGION
sample1	protein1	20
sample1	protein2	5
sample1	protein3	8
sample1	protein4	9
sample2	protein1	6
sample2	protein4	9
sample2	protein8	1
sample2	protein9	33
sample3	protein1	4
sample3	protein12	0
sample3	protein8	33
sample3	protein9	33

menu: *File* -> *Open File* -> `Example_1/input_data/example_input_file3.txt`

menu: *View* -> *Advanced mode*


menu: *Queue* -> *Add query* -> *Data preprocessing:*

Select unique rows in all data

On the base of columns: **PROTEIN**
REGION

Note:
Query is saved as file
`Example_1/queries/q5.que`
Saved result file is
`Example_1/results/r5.txt`

Selecting unique rows within each sample (column „SAMPLE”) on the base of two columns (PROTEIN and REGION)*.

SAMPLE	PROTEIN	REGION		SAMPLE	PROTEIN	REGION
sample1	protein1	20		sample1	protein1	20
sample1	protein2	5		sample1	protein2	5
sample1	protein3	8		sample1	protein3	8
sample1	protein4	9		sample1	protein4	9
sample2	protein1	6		sample2	protein1	6
sample2	protein4	9		sample2	protein4	9
sample2	protein8	1		sample2	protein8	1
sample2	protein9	33		sample2	protein9	33
sample3	protein1	4		sample3	protein1	4
sample3	protein12	0		sample3	protein12	0
sample3	protein9	33		sample3	protein9	33
sample3	protein9	33		sample3	protein9	33

menu: *File* -> *Open File* -> `Example_1/input_data/example_input_file3.txt`

menu: *View* -> *Advanced mode*

menu: *Queue* -> *Add query* -> *Data preprocessing:*

Select unique rows in each sample

Sample columns: **SAMPLE**


On the base of columns: **PROTEIN**
REGION

Note:
Query is saved as file
`Example_1/queries/q5.que`
Saved result file is
`Example_1/results/r5.txt`

*- the result data set is identical with selecting unique rows on the base of three columns „SAMPLE”, „PROTEIN” and „REGION”.

Result of filtering on the base of genome location

REGION	CHR	START	STOP
region1	chr1	20	500
region2	chr2	500	1000
region3	chr3	400	4000
region4	chr1	30	500
region5	chr2	500	1000
region6	chr3	400	4000



REGION	CHR	START	STOP
region1	chr1	20	500
region2	chr2	500	1000
region3	chr3	400	4000
region4	chr1	30	500
region5	chr2	500	1000
region6	chr3	400	4000

menu: *File* -> *Open File* -> *Example_1/input_data/example_input_file4a.txt*

menu: *View* -> *Advanced mode*

menu: *Queue* -> *Add query* -> *Location Filter*:

Select columns or create locus: Multiple Column Locus

Select columns or create locus: Chromosome column: **CHR**

Select columns or create locus: Start position column: **START**

Select columns or create locus: Stop position column: **STOP**

Locus search: Chromosome: 1

Locus search: Start pos.: 30

Locus search: Stop pos.: 500


Locus search: Length:

Locus search: Overlap at least [%]: 0 (means any overlap percent is allowed)

Note:
Query is saved as file
Example_1/queries/q7.que
Saved result file is
Example_1/results/r7.txt

Result of filtering on the base of genome location

REGION	CHR	START	STOP
region1	chr1	20	500
region2	chr2	500	1000
region3	chr3	400	4000
region4	chr1	30	500
region5	chr2	500	1000
region6	chr3	400	4000



REGION	CHR	START	STOP
region1	chr1	20	500
region2	chr2	500	1000
region3	chr3	400	4000
region4	chr1	30	500
region5	chr2	500	1000
region6	chr3	400	4000

menu: *File* -> *Open File* -> `Example_1/input_data/example_input_file4a.txt`

menu: *View* -> *Advanced mode*

menu: *Queue* -> *Add query* -> *Location Filter*:

Select columns or create locus: Multiple Column Locus

Select columns or create locus: Chromosome column: **CHR**

Select columns or create locus: Start position column: **START**

Select columns or create locus: Stop position column: **STOP**

Locus search: Chromosome: 1

Locus search: Start pos.: 30

Locus search: Stop pos.: 500

Locus search: Length:

Locus search: Overlap at least [%]: 100 (means only 100 % overlap are selected)

Note:
Query is saved as file
`Example_1/queries/q8.que`
Saved result file is
`Example_1/results/r8.txt`

Horizontal joining of the files on the base of genome location

MAIN DATA

REGION	CHR	START	STOP
region1	chr1	20	500
region2	chr2	500	1000
region3	chr3	400	4000
region4	chr1	30	500
region5	chr2	500	1000
region6	chr3	400	4000

EXTERNAL DATA

FEATURE	CHR	START	STOP	DESCRIPTION
promoterX	chr1	10	200	Description of promotor X...
regulatory sequenceY	chr1	100	1000	Description of regulatory sequence Y...
promoterZ	chr2	10	100	Description of promotor Y...
regulatory sequenceQ	chr2	390	460	Description of regulatory sequence Q...
promoterW	chr3	10	200	Description of promotor Y...
regulatory sequenceW	chr3	10	300	Description of regulatory sequence Q...





REGION	CHR	START	STOP	FEATURE	DESCRIPTION	external_data_found_rows
region1	chr1	20	500	promoterX	Description of promotor X...	2
region4	chr1	30	500	promoterX	Description of promotor X...	2
region1	chr1	20	500	regulatory sequenceY	Description of regulatory sequence Y...	2
region4	chr1	30	500	regulatory sequenceY	Description of regulatory sequence Y...	2

menu: *File* -> *Open File* -> `Example_1/input_data/example_input_file4a.txt`
 menu: *File* -> *Open Second File* -> `Example_1/input_data/example_input_file4b.txt`
 menu: *View* -> *Advanced mode*
 menu: *Queue* -> *Add query* -> *External Data Filter /Merger (Location):*

Select columns or create locus: Multiple Column Locus
 Select columns or create locus: Chromosome column: **CHR**
 Select columns or create locus: Start position column: **START**
 Select columns or create locus: Stop position column: **STOP**

Select columns or create locus (external data): Multiple Column Locus
 Select columns or create locus (external data): Chromosome column: **CHR**
 Select columns or create locus (external data): Start position column: **START**
 Select columns or create locus (external data): Stop position column: **STOP**

Condition: overlaps at least 0% of main data location

Select action: ADD SELECTED COLUMNS FROM EXTERNAL DATA MATCHING ROWS
 Select columns: FEATURE
 DESCRIPTION

„REPEAT DATA ROW WHEN MORE THAN ONE EXTERNAL DATA ROWS MATCH” option active
 „ADD BASIC STATISTICS” option active

Note:
 Query is saved as file
 Example_1/queries/q9.que
 Saved result file is
 Example_1/results/r9.txt

Vertical joining of the files with partially the same column names

menu: *File* -> *Open directory* ->
Example_1/input_data/input_files5a5b
menu: *Data* -> *Show system columns* (unchecked)

REGION	CHR	START	STOP
region1	chr1	20	500
region2	chr2	500	1000
region3	chr3	400	4000
region4	chr1	30	500

REGION	CHR	START	STOP	DESCRIPTION
region1	chr1	2000	50000	DESC1
region2	chr2	50000	100000	DESC2
region3	chr3	40000	400000	DESC3
region4	chr1	3000	50000	DESC4



REGION	CHR	START	STOP	DESCRIPTION
region1	chr1	20	500	
region2	chr2	500	1000	
region3	chr3	400	4000	
region4	chr1	30	500	
region1	chr1	2000	50000	DESC1
region2	chr2	50000	100000	DESC2
region3	chr3	40000	400000	DESC3
region4	chr1	3000	50000	DESC4

Note:
Saved result file is
Example_1/results/r10.txt

Example 2

Directory structure:

Example_2 – main directory containing example files

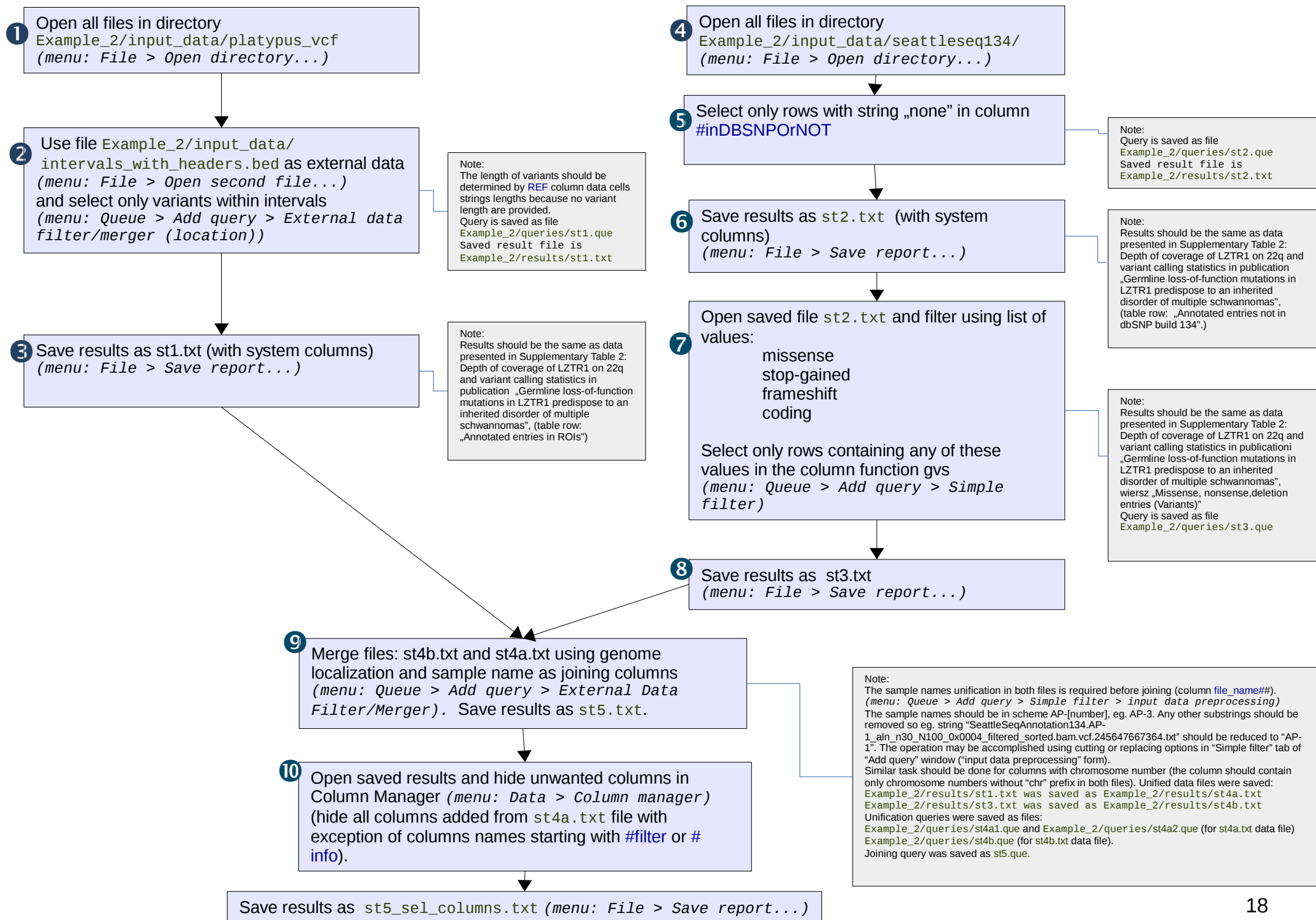
Example_2/input_data/platypus_vcf – vcf files produced by Platypus (<http://www.well.ox.ac.uk/platypus>)

Example_2/input_data/seattleseq134 – files produced by SeattleSeq (<http://snp.gs.washington.edu/>)

Example_2/results – output files for individual stages

Example_2/queries – query files for individual stages

```
├── input_data
│   ├── intervals_with_col_headers.bed
│   ├── platypus_vcf
│   │   ├── AP-1_aln_n30_N100_0x0004_filtered_sorted.bam.vcf
│   │   ├── AP-2_aln_n30_N100_0x0004_filtered_sorted.bam.vcf
│   │   ├── AP-3_aln_n30_N100_0x0004_filtered_sorted.bam.vcf
│   │   ├── AP-4_aln_n30_N100_0x0004_filtered_sorted.bam.vcf
│   │   ├── AP-5_aln_n30_N100_0x0004_filtered_sorted.bam.vcf
│   │   ├── AP-6_aln_n30_N100_0x0004_filtered_sorted.bam.vcf
│   │   ├── AP-7_aln_n30_N100_0x0004_filtered_sorted.bam.vcf
│   │   └── AP-8_aln_n30_N100_0x0004_filtered_sorted.bam.vcf
│   └── seattleseq134
│       ├── SeattleSeqAnnotation134.AP-1_aln_n30_N100_0x0004_filtered_sorted.bam.vcf.245647667364.txt
│       ├── SeattleSeqAnnotation134.AP-1_aln_n30_N100_0x0004_filtered_sorted.bam.vcf.245705740173.txt
│       ├── SeattleSeqAnnotation134.AP-2_aln_n30_N100_0x0004_filtered_sorted.bam.vcf.245649049586.txt
│       ├── SeattleSeqAnnotation134.AP-2_aln_n30_N100_0x0004_filtered_sorted.bam.vcf.245706569206.txt
│       ├── SeattleSeqAnnotation134.AP-3_aln_n30_N100_0x0004_filtered_sorted.bam.vcf.245650093533.txt
│       ├── SeattleSeqAnnotation134.AP-3_aln_n30_N100_0x0004_filtered_sorted.bam.vcf.245710002010.txt
│       ├── SeattleSeqAnnotation134.AP-4_aln_n30_N100_0x0004_filtered_sorted.bam.vcf.245650643836.txt
│       ├── SeattleSeqAnnotation134.AP-4_aln_n30_N100_0x0004_filtered_sorted.bam.vcf.245712178025.txt
│       ├── SeattleSeqAnnotation134.AP-5_aln_n30_N100_0x0004_filtered_sorted.bam.vcf.245651221106.txt
│       ├── SeattleSeqAnnotation134.AP-5_aln_n30_N100_0x0004_filtered_sorted.bam.vcf.245714132870.txt
│       ├── SeattleSeqAnnotation134.AP-6_aln_n30_N100_0x0004_filtered_sorted.bam.vcf.245652094290.txt
│       ├── SeattleSeqAnnotation134.AP-6_aln_n30_N100_0x0004_filtered_sorted.bam.vcf.245716307661.txt
│       ├── SeattleSeqAnnotation134.AP-7_aln_n30_N100_0x0004_filtered_sorted.bam.vcf.245652712335.txt
│       ├── SeattleSeqAnnotation134.AP-7_aln_n30_N100_0x0004_filtered_sorted.bam.vcf.245716589511.txt
│       ├── SeattleSeqAnnotation134.AP-8_aln_n30_N100_0x0004_filtered_sorted.bam.vcf.245653191154.txt
│       └── SeattleSeqAnnotation134.AP-8_aln_n30_N100_0x0004_filtered_sorted.bam.vcf.245717027188.txt
├── results
│   ├── st1.txt
│   ├── st2.txt
│   ├── st3.txt
│   ├── st4a.txt
│   ├── st4b.txt
│   ├── st5_sel_columns.txt
│   └── st5.txt
└── queries
    ├── st1.que
    ├── st2.que
    ├── st3.que
    ├── st4a.que
    ├── st4b1.que
    ├── st4b.que
    └── st5.que
```



Example 3

Directory structure:

Example_3 – main directory containing example files

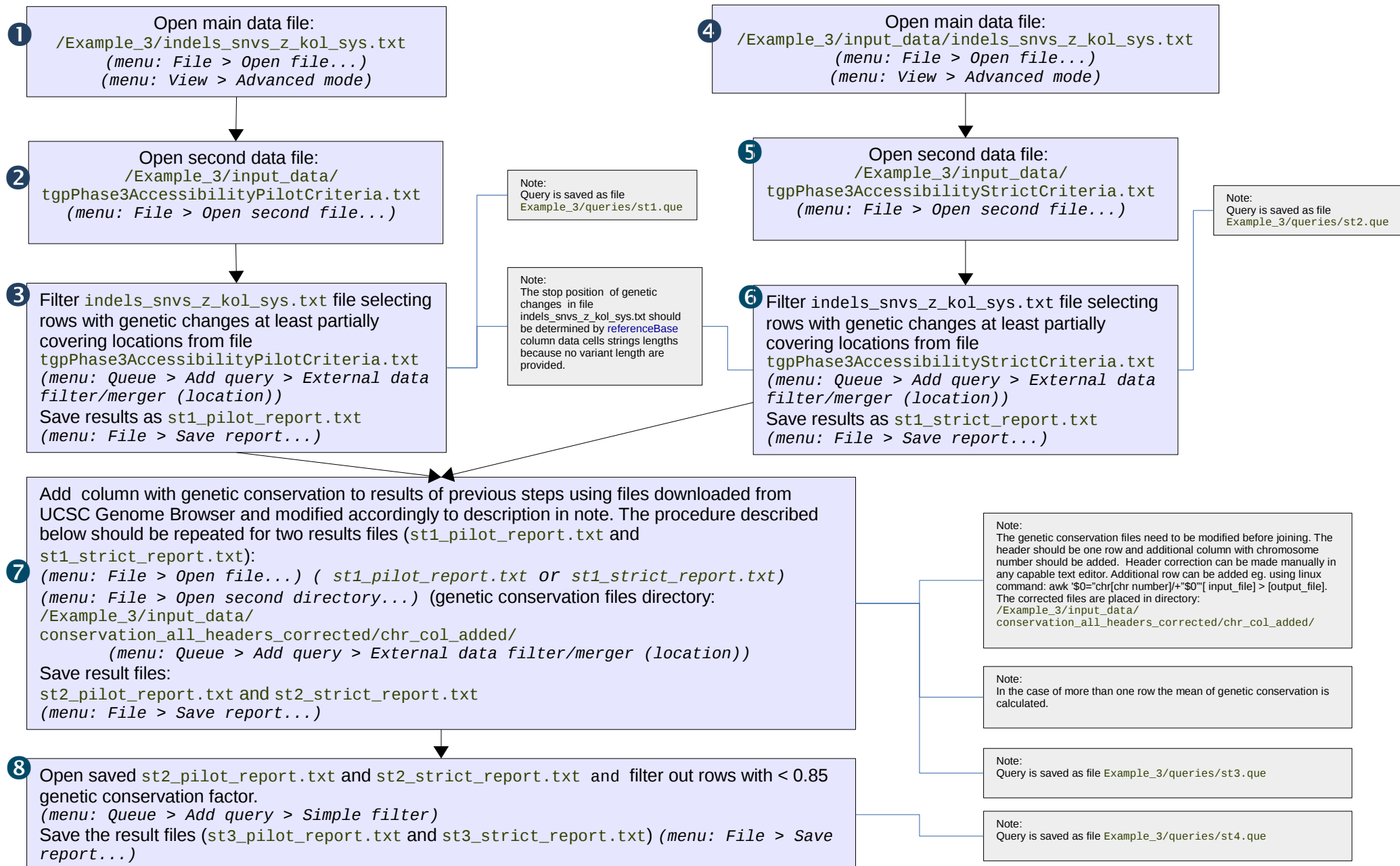
Example_3/input_data/conservation_all_headers_corrected/

– evolutionary conservation files (subdirectory *chr_col_added* contain the same files extended by required columns with chromosome name)

Example_3/results – output files for individual stages

Example_3/queries – query files for individual stages

```
.
├── input_data
│   ├── conservation_all_headers_corrected
│   │   ├── chr22_16050001-26657233_phastVertebrateCons46way.txt
│   │   ├── chr22_26657234-36737763_phastVertebrateCons46way.txt
│   │   ├── chr22_36737764-46804985_phastVertebrateCons46way.txt
│   │   ├── chr22_46804986-51304566_phastVertebrateCons46way.txt
│   │   └── chr_col_added
│   │       ├── 2_chr22_16050001-26657233_phastVertebrateCons46way.txt
│   │       ├── 2_chr22_26657234-36737763_phastVertebrateCons46way.txt
│   │       ├── 2_chr22_36737764-46804985_phastVertebrateCons46way.txt
│   │       └── 2_chr22_46804986-51304566_phastVertebrateCons46way.txt
│   ├── indels_snvs_z_kol_sys.txt
│   ├── tgpPhase3AccessibilityPilotCriteria.txt
│   └── tgpPhase3AccessibilityStrictCriteria.txt
├── results
│   ├── st1_report_pilot.txt
│   ├── st1_report_strict.txt
│   ├── st2_report_pilot.txt
│   ├── st2_report_strict.txt
│   ├── st3_report_pilot.txt
│   └── st3_report_strict.txt
└── queries
    ├── st1.que
    ├── st2.que
    ├── st3.que
    └── st4.que
```



Example 4

Directory structure:

Example_4 – main directory containing example files

Example_4/input_data/vcf – vcf files

Example_4/results – output files for individual stages

Example_4/saved_queries – query files for individual stages

```
.
├── input_data
│   ├── Supporting_Table_S2.csv
│   └── vcf
│       ├── 608P.bam_PASS_filter_only.txt.vcf
│       ├── 642P.bam_PASS_filter_only.txt.vcf
│       ├── 903P_PASS_filter_only.bam.vcf
│       ├── 915P.bam_PASS_filter_only.txt.vcf
│       ├── 916P.bam_PASS_filter_only.txt.vcf
│       └── 924P_PASS_filter_only.bam.vcf
├── queries
│   ├── st1.que
│   ├── st2.que
│   └── st3.que
├── results
│   ├── final
│   │   ├── 608h_608h.txt
│   │   ├── 608h_608m.txt
│   │   ├── 608h_608t.txt
│   │   ├── 608h_all.txt
│   │   ├── 608h_null.txt
│   │   ├── 642h_642h.txt
│   │   ├── 642h_642m.txt
│   │   ├── 642h_642t.txt
│   │   ├── 642h_all.txt
│   │   ├── 642h_null.txt
│   │   ├── 903h_903m.txt
│   │   ├── 903h_903t.txt
│   │   ├── 903h_all.txt
│   │   ├── 903h_null.txt
│   │   ├── 915h_915h.txt
│   │   ├── 915h_915m.txt
│   │   ├── 915h_915t.txt
│   │   ├── 915h_all.txt
│   │   ├── 915h_null.txt
│   │   ├── 916h_916h.txt
│   │   ├── 916h_916m.txt
│   │   ├── 916h_916t.txt
│   │   ├── 916h_all.txt
│   │   ├── 916h_null.txt
│   │   ├── 924h_924h.txt
│   │   ├── 924h_924m.txt
│   │   ├── 924h_924t.txt
│   │   ├── 924h_all.txt
│   │   └── 924h_null.txt
│   ├── st1.txt
│   ├── st2.txt
│   ├── st3.txt
│   └── Supporting_Table_S2_st1.txt
```

1 Open all files in directory
Example_4/input_data/vcf
(menu: File > Open directory...).
Save result file as st1.txt

2 The sample names unification is
required in column vcf_sample_name
(see note). Save result file as st2.txt
(menu: File > Save report...)

Note:
The sample names unification in st1.txt file is
required (column vcf_sample_name).
(menu: Queue > Add query > Simple filter
> input data preprocessing)
The sample names should be in scheme [number]H,
eg. 924H. Any other substrings should be removed so
eg. string "galaxy41-[ap924p_sam-to-
bam_on_data_23_converted_bam]" should first be
reduced to "924p" and next the letter "p" in resultant
sample name string should be replaced by letter "H"
giving string "924H". The operation may be
accomplished using replacing options in "Simple filter"
tab of "Add query" window ("input data preprocessing"
form).
(menu: Queue > Add query > Simple
filter)
Unification queries were saved as file:
Example_4/queries/st1.que

3 Open file
Example_4/input_data/Supporting_Table_S1.csv
Remove numbers "1" or "1,2" at the end of strings in
column patient so that cell content was eg. 924T
instead 924T1,2
(menu: View > Advanced mode)
(menu: Queue > Add query > Simple filter)
Save result file as Supporting_Table_S2_st1.txt
(menu: File > Save report...)

Note:
Query is saved as file
Example_4/queries/st2.que

4 Open file st2.txt as main data
(menu: View > Advanced mode)
(menu: File > Open file...)
Open file Supporting_Table_S2_st1.txt as external data
(menu: File > Open second file...)
Merge the files using genome localization.
(menu: Queue > Add query > External data filter/merger (location)).
Save result file as st3.txt
(menu: File > Save report...)

Note:
Query is saved as file
Example_4/queries/st3.que

5 Open file st3.txt
Each sample need operation described below (on the example of sample 608H)
(menu: View > Basic mode)
(button: Filter)
Select only rows containing string "608H" in column vcf_sample_name
(button: Filter)
Save result file 608h_all.txt
Open result file 608h_all.txt
(button: Filter)
Select only rows containing string "608H" in column patient
(button: Filter)
Save result file 608h_608h.txt
Select only rows containing string "608T" in column patient
(button: Filter)
Save result file 608h_608t.txt
Select only rows containing string "608M" in column patient
(button: Filter)
Save result file 608h_608m.txt
Select only rows containing string "null" in column patient
(button: Filter)
Save result file 608h_null.txt

Note:
Saved result files are in
directory
Example_4/results/final/

Example 5

Directory structure:

Example_5 – main directory containing example files

Example_5/results – output files for individual stages

```
.
├── input_data
│   ├── 1A_S1_hg38.bam.vcf
│   ├── 540237_41071_Area1_2012-06-25_25000bp_avg_segMNT.gff
│   └── 540237_41071_Area1_2012-06-25_unavg_segMNT.gff
└── results
    ├── 1A_S1_hg38.bam.txt
    ├── 540237_41071_Area1_2012-06-25_25000bp_avg_segMNT_s1.gff
    └── 540237_41071_Area1_2012-06-25_unavg_segMNT_s1.gff
```

1

Open file

Example_5/input_data/1A_S1_hg38.bam.vcf

Remove all columns with exception:

```
#_CHROM
_POS
_ID
_REF
_ALT
_QUAL
_FILTER
_INFO@_TC
_INFO@_TR
```

(menu: Data > Column manager)

Uncheck check boxes near the names of columns which should be deleted

Save result file as 1A_S1_hg38.bam.txt
(menu: File > Save report...)

Note:
The result file is saved as tab delimited text file.
The result file size was reduced to 14,6% of input file size.

2

Open file

Example_5/input_data/540237_41071_Area1_2012-06-25_25000bp_avg_segMNT.gff

Remove columns NimbleScan:segMNT and
540237_41071_Area1_2012-06-25_25000bp_segMNT

(menu: Data > Column manager)

Uncheck check boxes near the names of columns
Save result file as 540237_41071_Area1_2012-06-25_25000bp_avg_segMNT_s1.gff
(menu: File > Save report...)

Note:
The result file size was reduced to 43,5% of input file size.

3

Open file

Example_5/input_data/540237_41071_Area1_2012-06-25_unavg_segMNT.gff

Remove columns NimbleScan:segMNT and
540237_41071_Area1_2012-06-25_25000bp_segMNT

(menu: Data > Column manager)

Uncheck check boxes near the names of columns
Save result file as 540237_41071_Area1_2012-06-25_unavg_segMNT_s1.gff
(menu: File > Save report...)

Note:
The result file size was reduced to 36% of input file size.

Note:
These files does not contain column headers so the first row is treated as column headers row. This fact do not affect the actual results.