Database mining with biomaRt

Steffen Durinck

Illumina Inc.

BioC 2009

Overview

- The BioMart software suite
- biomaRt package
- biomaRt installation
- biomaRt example queries to show the variety of different data types/questions that can be retrieved/answered for many organisms

BioMart 0.7

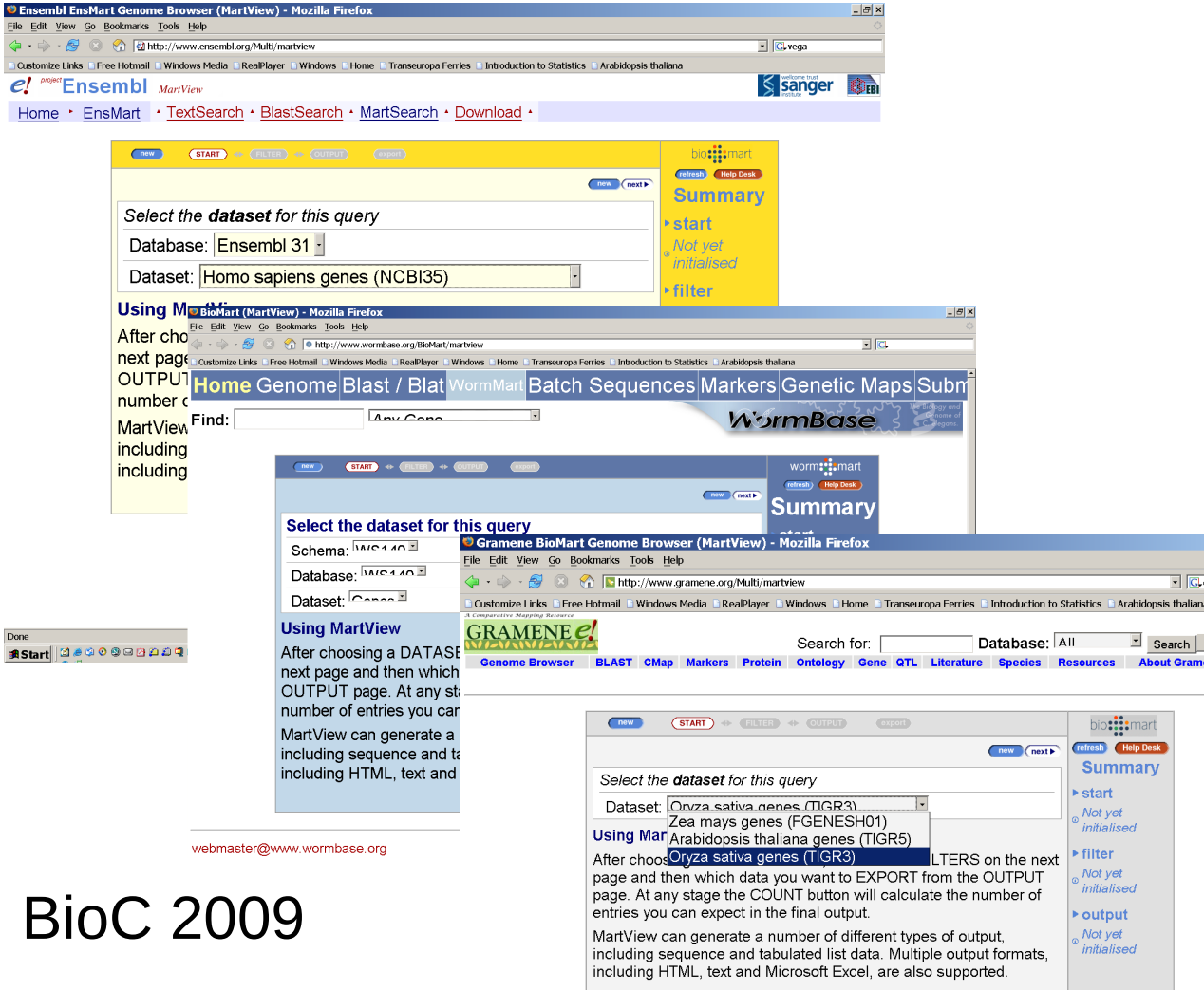
- BioMart is a query-oriented data management system developed jointly by the European Bioinformatics Institute (EBI) and Cold Spring Harbor Laboratory (CSHL).
- Originally developed for the Ensembl project but has now been generalized

BioMart 0.7

- BioMart data can be accessed using either web, graphical, or text based applications, or programmatically using web services or software libraries written in Perl and Java.
- http://www.biomart.org

Example BioMart databases

- Ensembl
- Wormbase
- Reactome
- Gramene
- .....



BioC 2009

BioMart databases

- De-normalized
- Tables with ‘redundant’ information
- Query optimized
- Fast and flexible

- Well suited for batch querying

biomaRt

- R interface to BioMart databases
- Performs online queries
- Current release version 2.0.0
- Depends on Rcurl and XML packages

Installing biomaRt & GenomeGraphs

- Platforms on which biomaRt has been installed:
  - Linux (curl http://curl.haxx.se)
  - OSX (curl)
  - Windows

Installing biomaRt & GenomeGraphs

> source("http://www.bioconductor.org/biocLite.R")

> biocLite('GenomeGraphs')

Running biocinstall version 2.4.11 with R version 2.9.1

Your version of R requires version 2.4 of Bioconductor.

also installing the dependencies 'bitops', 'XML', 'RCurl', 'biomaRt'

List available BioMart databases

```
> library(biomaRt)
Loading required package: XML
Loading required package: Rcurl
> listMarts()
```

List available BioMarts

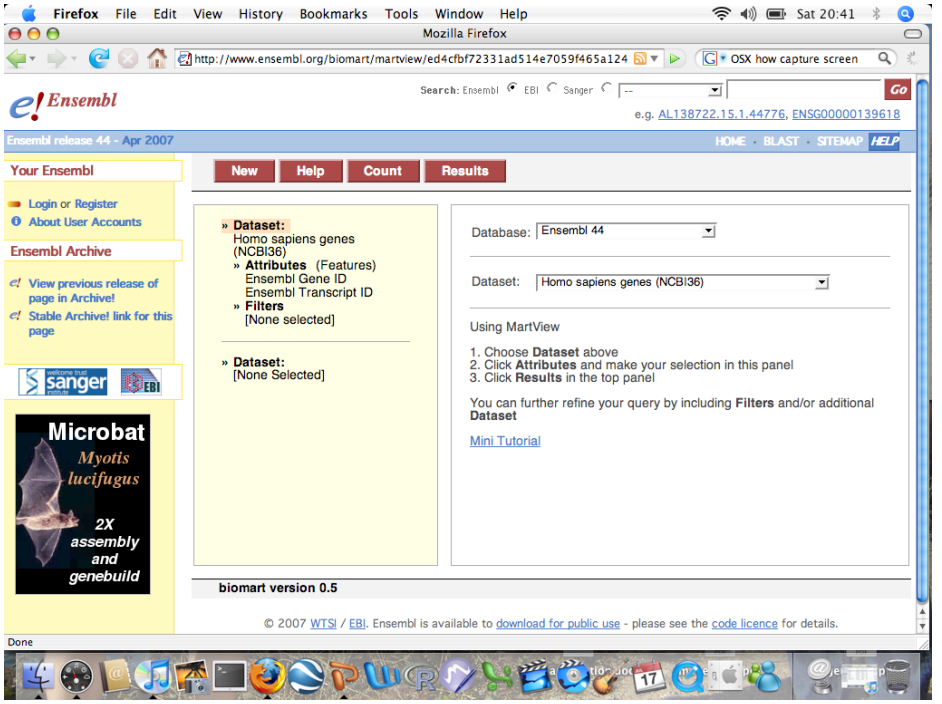
	biomart	version
1	ensembl	ENSEMBL 55 GENES (SANGER UK)
2	snp	ENSEMBL 55 VARIATION (SANGER UK)
3	functional_genomics	ENSEMBL 55 FUNCTIONAL GENOMICS
4	vega	VEGA 35 (SANGER UK)
5	msd	MSD PROTOTYPE (EBI UK)
6	htgt	HIGH THROUGHPUT GENE TARGETING AND TRAPPING
7	QTL_MART	GRAMENE 29 QTL DB (CSHL US)
8	ENSEMBL_MART_ENSEMBL	GRAMENE 29 GENES
9	ENSEMBL_MART_SNP	GRAMENE 29 SNPs
10	GRAMENE_MARKER_29	GRAMENE 29 MARKERS

Ensembl

- Ensembl is a joint project between EMBL - European Bioinformatics Institute (EBI) and the Wellcome Trust Sanger Institute (WTSI)
- A software system which produces and maintains automatic annotation on selected eukaryotic genomes.
- http://www.ensembl.org

Ensembl - BioMart

> *ensembl=useMart("ensembl")*



Ensembl - Datasets

> *listDatasets(ensembl)*

Returns:

- name: *hsapiens_gene_ensembl*
- description: *Homo sapiens genes*
- version: *(GRCh37)*

Ensembl currently contains 50 datasets~species

Ensembl - Datasets

A dataset can be selected using the useMart function

```
> ensembl = useMart("ensembl",
   dataset="hsapiens_gene_ensembl)
```

```
Checking attributes ... ok
Checking filters ... ok
```

biomaRt query: Attributes

- Attributes define the values which the user is interested in.
- Conceptually equal to output of the query
- Example attributes:
  - chromosome_name
  - band

biomaRt query: Filters

- Filters define restrictions on the query
- Conceptually filters are inputs

- Example filters:
  - entrezgene
  - chromosome_name

biomaRt query

Attributes (e.g., chromosome and band)

Filters (e.g., “entrezgene”)

Values (e.g., EntrezGene identifiers)

biomaRt query

Three main biomaRt functions

- *listFilters*
  - Lists the available filters
- *listAttributes*
  - Lists the available attributes
- *getBM*
  - Performs the actual query and returns a data.frame

Microarrays & Ensembl

- Ensembl does an independent mapping of array probe sequences to genomes (Affymetrix, Illumina, Agilent,…)
- If there is no clear match then that probe is not assigned to a gene

TASK 1 - Ensembl

- Annotate the following Affymetrix probe identifiers from the human u133plus2 platform with hugo gene nomenclature symbol (hgnc_symbol) and chromosomal location information:

211550_at, 202431_s_at, 206044_s_at

TASK 1 - Ensembl

- Filters: affy_hg_u133_plus_2
- Attributes:

  affy_hg_u133_plus_2, chromosome_name, start_position, end_position, band, strand

- Values:

  211550_at, 202431_s_at, 206044_s_at

TASK 1 - Ensembl

```
> affyids =
  c("211550_at","202431_s_at","206044_s_at")
> annotation =
  getBM(attributes=c("affy_hg_u133_plus_2","ensembl_gene_id","hgnc_symbol","chromosome_name","start_position","end_position","band","strand"),
  filters="affy_hg_u133_plus_2", values=affyids,
  mart = ensembl)
```

TASK 1 - Ensembl

>annotation

	affy_hg_u133_plus_2	ensembl_gene_id	hgnc_symbol	chromosome_name
1	202431_s_at	ENSG00000136997	MYC	8
2	206044_s_at	ENSG00000157764	BRAF	7
3	211550_at	ENSG00000146648	EGFR	7

	start_position	end_position	band	strand
	128748316	128753671	q24.21	1
	140433817	140624564	q34	-1
	55086714	55324313	p11.2	1

TASK 1* - Ensembl

Retrieve GO annotation for the following Illumina human_wg6_v2 identifiers:

ILMN_1728071, ILMN_1662668

TASK 1* - Ensembl

Retrieve GO annotation for the following Illumina human_wg6_v2 identifiers:

ILMN_1728071, ILMN_1662668

```
> illuminaIDs = c("ILMN_1728071","ILMN_1662668")
> goAnnot = getBM(c("illumina_humanwg_6_v2", "go_biological_process_id","go_biological_process_linkage_type"), filters="illumina_humanwg_6_v2", values=illuminaIDs, mart = ensembl)
```

TASK 1* - Ensembl

	illumina_humanwg_6_v2	go_biological_process_id	go_biological_process_linkage_type
1	ILMN_1662668	GO:0000281	IMP
2	ILMN_1662668	GO:0006461	IDA
3	ILMN_1662668	GO:0006974	IDA
4	ILMN_1662668	GO:0007026	IDA
5	ILMN_1662668	GO:0007050	IDA

Using more than one filter

- getBM can be used with more than one filter
- Filters should be given as a vector
- Values should be a list of vectors where the position of each vector corresponds with the position of the associated filter in the filters argument

TASK 2 - Ensembl

Retrieve all genes that are involved in Diabetes Mellitus Type I or Type II and have transcription factor activity

TASK 2 - Ensembl

1. Diabetes Mellitus type I MIM accession: 222100
2. Diabetes Mellitus type II MIM accession: 125853
3. GO id for “transcription factor activity”: GO:0003700

TASK 2 - Ensembl

```
diab=getBM(c("ensembl_gene_id","hgnc_symbol"),
           filters=c("mim_morbid_accession","go"),
          values=list(c("125853","222100"),"GO:0003700"),
          mart=ensembl)
```

TASK 2 - Ensembl

	ensembl_gene_id	*hgnc_symbol*
1	*ENSG00000139515*	*PDX1*
2	*ENSG00000108753*	*HNF1B*
3	*ENSG00000148737*	*TCF7L2*
4	*ENSG00000106331*	*PAX4*
5	*ENSG00000162992*	*NEUROD1*
6	*ENSG00000135100*	*HNF1A*

Boolean filters

- Filters can be either numeric, string or boolean
- Boolean filters should have either TRUE or FALSE as values
  - TRUE: return all information that comply with the given filter (e.g. return only genes that have a hgnc_symbol)
  - FALSE: return all information that doesn't comply with the given filter (e.g. with no hgnc_symbol)

Boolean filters/ *filterType*

The function *filterType* allows you to figure out which type each filter is (this function is currently only available in the devel version of biomaRt)

> filterType("affy_hg_u133_plus_2", mart=ensembl)

[1] “id_list

>filterType("with_affy_hg_u133_plus_2", mart=ensembl)

[1] "boolean_list"

TASK 3 - Ensembl

Retrieve all miRNAs known on chromosome 13 and their chromosomal locations

TASK 3 - Ensembl

```
>miRNA =
    getBM(c("mirbase","ensembl_gene_id","start_position",
"chromosome_name"),
    filters=c("chromosome_name","with_mirbase"),
    values=list(13,TRUE), mart=ensembl)
> miRNA[1:5,]
```

TASK 3 - Ensembl

	mirbase	ensembl_gene_id	start_position	chromosome_name
1	MI0008190	ENSG00000211491	41301964	13
2	MI0003635	ENSG00000207652	41384902	13
3	MI0000070	ENSG00000208006	50623109	13
4	MI0000069	ENSG00000207718	50623255	13
5	MI0003636	ENSG00000207858	90883436	13

attributePages

- attributePages gives brief overview of available attribute pages (useful for displaying subset of attributes)

> attributePages(ensembl)
[1] "feature_page" "structure" "snp" "homologs" "sequences"

>listAttributes(ensembl, page = *"feature_page"*)

Additional help to figure out which filter and attribute names to use

- Go to www.biomart.org and select BioMart you use
- Select attributes and filters
- Press to XML button to get their names

FilterOptions function: enumerates all possible values for a filter (if available)

TASK 4 - Ensembl

Retrieve all entrezgene identifiers on chromosome 22 that have a non-synonymous coding SNP

TASK 4 - Ensembl

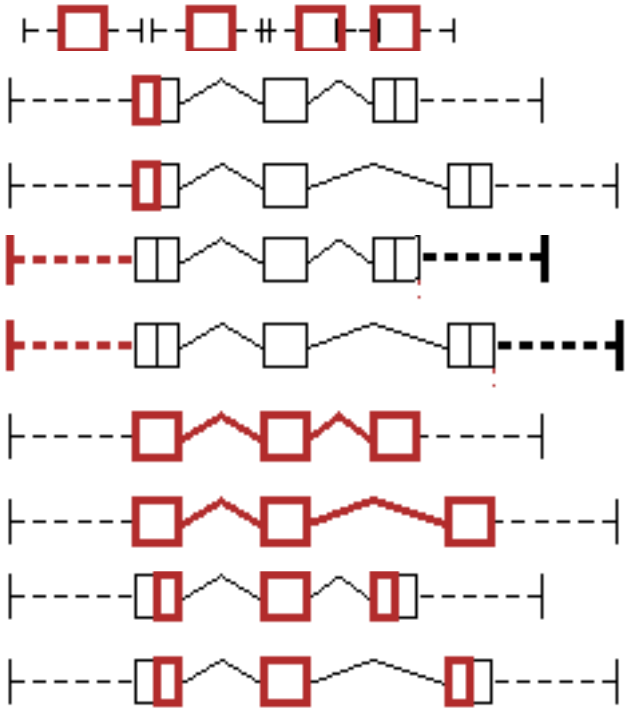
```
> filterOptions("snptype_filters",ensembl)
[1] "[STOP_GAINED,STOP_LOST,COMPLEX_INDEL,FRAMESHIFT_CODING,
NON_SYNONYMOUS_CODING,STOP_GAINED,SPLICE_SITE,STOP_LOST,SPLI
    CE_SITE,FRAMESHIFT_CODING,SPLICE_SITE,NON_SYNONYMOUS_CODI
    NG,SPLICE_SITE,SYNONYMOUS_CODING,SPLICE_SITE,SYNONYMOUS_C
    ODING,5PRIME_UTR,SPLICE_SITE,5PRIME_UTR,3PRIME_UTR,SPLICE_SIT
    E,3PRIME_UTR,INTRONIC,ESSENTIAL_SPLICE_SITE,INTRONIC,SPLICE_SI
    TE,INTRONIC,UPSTREAM,DOWNSTREAM]“
> entrez =
    getBM("entrezgene",filters=c("chromosome_name","snptype_filters"),
    values=list(22,"NON_SYNONYMOUS_CODING"),mart=ensembl)
> entrez[1:5,]
> [1]  23784  81061 150160 150165 128954
```

getSequence

- Retrieving sequences from Ensembl can be done using the *getBM* function or the *getSequence* wrapper function
- Output of *getSequence* can be exported to FASTA file using the *exportFASTA* function

getSequence

- Available sequences in Ensembl:
  - Exon
  - 3'UTR
  - 5'UTR
  - Upstream sequences
  - Downstream sequences
  - Unspliced transcript/gene
  - Coding sequence
  - Protein sequence



getSequence

- Arguments of getSequence:
  - *id*: identifier
  - *type*: type of identifier used e.g. hgnc_symbol or affy_hg_u133_plus_2
  - *seqType*: sequence type that needs to be retrieved e.g. gene_exon, coding, 3utr, 5utr,
  - *upstream/downstream*: specify number of base pairs upstream/downstream that need to be retrieved

TASK 5 - Ensembl

Retrieve all exons of CDH1

TASK 5 - Ensembl

```
> seq = getSequence(id="CDH1",
    type="hgnc_symbol",seqType="gene_exon", mart = ensembl)
> seq[1,]

                                                        gene_exon
1
    TACAAGGGTCAGGTGCCTGAGAACGAGGCTAACGTCGTAATCAC
    CACACTGAAAGTGACTGATGCTGATGCCCCCAATACCCCAGCGT
    GGGAGGCTGTATACACCATATTGAATGATGATGGTGGACAATTTG
    TCGTCACCACAAATCCAGTGAACAACGATGGCATTTTGAAAACAG
    CAAAG
  hgnc_symbol
1       CDH1
```

TASK 6 - Ensembl

Retrieve 2000bp sequence upstream of the APC and CUL1 translation start site

TASK 6 - Ensembl

```
>promoter=getSequence(id=c("APC","CUL1"),type=
   "hgnc_symbol",
   seqType="coding_gene_flank",upstream =2000,
   mart=ensembl)
```

```
> promoter
```

Homology - Ensembl

- The different species in Ensembl are interlinked
- biomaRt takes advantage of this to provide homology mappings between different species

Linking two datasets

- Two datasets (e.g. two species in Ensembl) can be linked to each other by using the *getLDS* (get linked dataset) function
- One has to connect to two different datasets and specify the linked dataset using *martL*, *filtersL*, *attributesL*, *valuesL* arguments

TASK 7 - Ensembl

Retrieve human gene symbol and affy identifiers of their homologs in chicken for the following two identifiers from the human affy_hg_u95av2 platform: 1434_at, 1888_s_at

TASK 7 - Ensembl

```
> human=useMart("ensembl", dataset="hsapiens_gene_ensembl")
  Checking attributes and filters ... ok
> chicken=useMart("ensembl", dataset="ggallus_gene_ensembl")
  Checking attributes and filters ... ok
>out = getLDS(attributes=c("affy_hg_u95av2","hgnc_symbol"),
    filters="affy_hg_u95av2",
    values=c("1888_s_at",“1434_at"),mart=human,
    attributesL="affy_chicken", martL=chicken)
> out
       V1   V2              V3
1   1434_at PTEN GgaAffx.25913.1.S1_a
2 1888_s_at  KIT     Gga.606.1.S1_at
```

Variation BioMart

- dbSNP mapped to Ensembl

```
> snp = useMart("snp", dataset="hsapiens_snp"))
```

TASK 8 - Variation

Retrieve all refsnp_ids and their alleles and position that are located on chromosome 8 and between bp 148350 and 158612.

TASK 8 - Variation

```
>out=getBM(attributes=c("refsnp_id","allele","chrom_start"),
    filters=c("chr_name","chrom_start","chrom_end"),
    values=list(8,148350, 158612), mart=snp)
> out[1:5,]
     refsnp_id allele chrom_start
1 ENSSNP4490669    C/G      148729
2 ENSSNP5558526    T/C      148909
3 ENSSNP4089737    T/A      149060
4 ENSSNP9060169    C/T      149245
5 ENSSNP4351891    C/G      149250
```

Ensembl Archives

- Provide alternate host

>listMarts(host="may2009.archive.ensembl.org/biomart/martservice/")

	biomart	version
1	*ENSEMBL_MART_ENSEMBL*	*Ensembl 54*
2	*ENSEMBL_MART_SNP*	*Ensembl Variation 54*
3	*ENSEMBL_MART_VEGA*	*Vega 35*
4	*REACTOME*	*Reactome(CSHL US)*
5	*wormbase_current*	*WormBase (CSHL US)*
6	*pride*	*PRIDE (EBI UK)*

>ensembl54=useMart("ENSEMBL_MART_ENSEMBL", host="may2009.archive.ensembl.org/biomart/martservice/")

Ensembl Archives

- Access to archives by setting archive=TRUE or connect to specific host (Note that this is currently not up to date in the central repository)

```
>listMarts(archive=TRUE)
                biomart                    version
1     ensembl_mart_51          Ensembl 51
2     snp_mart_51              SNP 51
3     vega_mart_51                   Vega 32
4     ensembl_mart_50          Ensembl 50
1     snp_mart_50              SNP 50
```

```
> ensembl51 = useMart("ensembl_mart_51", archive=TRUE,
    dataset="hsapiens_gene_ensembl")
```

Gramene

- Gramene is a curated, open-source, data resource for comparative genome analysis in the grasses.
- Rice, Maize and Arabidopsis

TASK 9 - Gramene

Retrieve affy ATH1 ids and CATMA ids that map to the *Arabidopsis thaliana* chromosome 1 between basepair 30.000 and 41.000

TASK 9 - Gramene

```
>gramene =
  useMart("ENSEMBL_MART_ENSEMBL",
  dataset="athaliana_gene_ensembl")
>getBM(c("affy_ath1_id","catma_tigr5_id"),
  filters=c("chromosome_name","start","end")
  , values=list("1", "30000","41000"),
  mart=gramene)
```

TASK 9 - Gramene

	affy_ath1_id	catma_tigr5_id
1	261579_at	CATMA1a00040
2	261569_at	CATMA1a00045
3	261569_at	CATMA1a00045
4	261569_at	CATMA1a00045
5	261576_at	CATMA1a00050
6	261576_at	CATMA1a00050

Wormbase

- Database on the genetics of C elegans and related nematodes.

TASK 10 - Wormbase

Determine the RNAi ids and the observed phenotypes for the gene with wormbase gene id: WBGene00006763

TASK 10 - Wormbase

```
> worm = useMart("wormbase176",
                 dataset="wormbase_rnai")

> pheno =
  getBM(c("rnai","phenotype_primary_name"),
  filters="gene", values="WBGene00006763",
  mart=worm)
```

TASK 10 - Wormbase

```
>pheno
  rnai            phenotype_primary_name
1  WBRNAi00021278                slow_growth
2  WBRNAi00021278
   postembryonic_development_abnormal
3  WBRNAi00021278             embryonic_lethal
4  WBRNAi00021278               larval_lethal
5  WBRNAi00021278               larval_arrest
6  WBRNAi00021278             maternal_sterile
7  WBRNAi00021278                    Abnormal
8  WBRNAi00021278              sterile_progeny
9  WBRNAi00026915                slow_growth
```

Discussion

- Using biomaRt to query public web services gets you started quickly, is easy and gives you access to a large body of metadata in a uniform way
- Need to be online
- Online metadata can change behind your back; although there is possibility of connecting to a particular, immutable version of a dataset

Reporting bugs

- Check with MartView if you get the same output
  - Yes: contact database e.g.

    helpdesk@ensembl.org

  - No: contact me - sdurinck@gmail.com

Acknowledgements

- EBI
  - Rhoda Kinsella
  - Arek Kasprzyk
  - Ewan Birney
- EMBL
  - Wolfgang Huber

Bioconductor users