Overview

- Brief intro:
  - The BioMart software suite
  - biomaRt package
  - biomaRt installation

- Tour of BioMart databases available through 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.
BioMart is a query-oriented data management system developed jointly by the Ontario Institute for Cancer Research (OICR) and the European Bioinformatics Institute (EBI).

The system can be used with any type of data and is particularly suited for providing ‘data mining’ like searches of complex descriptive data. BioMart comes with an ‘out of the box’ website that can be installed, configured and customised according to user requirements. Further access is provided by graphical and text based applications or programmatically using web services or API written in Perl and Java. BioMart has built-in support for query optimisation and data federation and in addition can be configured to work as a DAS 1.5 Annotation server. The process of converting a data source into BioMart format is fully automated by the tools included in the package. Currently supported RDBMS platforms are MySQL, Oracle and Postgres.

BioMart is completely Open Source, licensed under the LGPL, and freely available to anyone without restrictions.

Powered by BioMart software:

- BioMart Central Portal
- ICGC Data Portal
- Ensembl
- Ensembl Bacteria
- Ensembl Metazoa
- Ensembl Protists
- Ensembl Plants
- Ensembl Fungi
- Phytozome
- Gramene
- EuroPhenome
- UniProt
- InterPro
- HGNC
- Rice-Map
- iKMC
- Wormbase
- DroStRe
- ArrayExpress
- Eurexpress
- HapMap
- Dietybase
- COSMIC
- iOGen
- Rat Genome Database
- GermOnLine
- PRIDE
- PepSeeker
- VectorBase
- HTGT
- Cldb
- Pancreatic Expression Database
- Reactome
- EU Rat Mart
- Paramoecium DB
- International Potato Center (CIP)
- Mouse Genome Informatics (MGI)
- Cyanone

Third party software with BioMart Plugin:

- Bioclipse
- biomaRT-BioConductor
- Cytoscape
- Galaxy
- Gtools
- Ruby API
- Taverna
- WebLab
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 2.4.0
- Imports Rcurl and XML packages
Package Download Stats

![Package Download Stats Chart](chart.png)
Installing biomaRt

- Platforms on which biomaRt has been installed:
  - Linux (curl http://curl.haxx.se)
  - OSX (curl)
  - Windows
Wiki with code example for this tutorial

http://biomart2010.wikispaces.com/
Installing biomaRt

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

> biocLite('biomaRt')

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 (currently 42 BioMarts)

<table>
<thead>
<tr>
<th>biomart</th>
<th>version</th>
</tr>
</thead>
<tbody>
<tr>
<td>ensembl</td>
<td>ENSEMBL GENES 58 (SANGER UK)</td>
</tr>
<tr>
<td>snp</td>
<td>ENSEMBL VARIATION 58 (SANGER UK)</td>
</tr>
<tr>
<td>functional_genomics</td>
<td>ENSEMBL FUNCTIONAL GENOMICS 58 (SANGER UK)</td>
</tr>
<tr>
<td>vega</td>
<td>VEGA 38 (SANGER UK)</td>
</tr>
<tr>
<td>bacterial_mart_5</td>
<td>ENSEMBL BACTERIA 5 (EBI UK)</td>
</tr>
<tr>
<td>fungal_mart_5</td>
<td>ENSEMBL FUNGAL 5 (EBI UK)</td>
</tr>
<tr>
<td>.....</td>
<td></td>
</tr>
</tbody>
</table>
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
A dataset can be selected using the `useMart` function

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

`Checking attributes ... ok`

`Checking filters ... ok`
biomaRt query: 3 parts

Attributes (e.g., chromosome and band)  Filters (e.g., entrezgene)  Values (e.g., list of entrezgene ids)

biomaRt query
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
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,

cart = ensembl)
### TASK 1 - Ensembl

```plaintext
>annotation

<table>
<thead>
<tr>
<th>start_position</th>
<th>end_position</th>
<th>gene</th>
<th>chromosome_name</th>
<th>strand</th>
</tr>
</thead>
<tbody>
<tr>
<td>128748316</td>
<td>128753671</td>
<td>202431_s_at</td>
<td>MYC</td>
<td>1</td>
</tr>
<tr>
<td>140433817</td>
<td>140624564</td>
<td>206044_s_at</td>
<td>BRAF</td>
<td>-1</td>
</tr>
<tr>
<td>55086714</td>
<td>55324313</td>
<td>211550_at</td>
<td>EGFR</td>
<td>1</td>
</tr>
</tbody>
</table>
```
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

<table>
<thead>
<tr>
<th>illumina_humanwg_6_v2</th>
<th>go_biological_process_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILMN_1662668</td>
<td>GO:0000281</td>
</tr>
<tr>
<td>ILMN_1662668</td>
<td>GO:0006461</td>
</tr>
<tr>
<td>ILMN_1662668</td>
<td>GO:0006974</td>
</tr>
<tr>
<td>ILMN_1662668</td>
<td>GO:0007026</td>
</tr>
<tr>
<td>ILMN_1662668</td>
<td>GO:0007050</td>
</tr>
</tbody>
</table>

**go_biological_process_linkage_type**

- IMP
- IDA
- IDA
- IDA
- 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
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

<table>
<thead>
<tr>
<th>ensembl_gene_id</th>
<th>hgnc_symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENSG00000139515</td>
<td>PDX1</td>
</tr>
<tr>
<td>ENSG00000108753</td>
<td>HNF1B</td>
</tr>
<tr>
<td>ENSG00000148737</td>
<td>TCF7L2</td>
</tr>
<tr>
<td>ENSG00000106331</td>
<td>PAX4</td>
</tr>
<tr>
<td>ENSG00000162992</td>
<td>NEUROD1</td>
</tr>
<tr>
<td>ENSG00000135100</td>
<td>HNF1A</td>
</tr>
</tbody>
</table>
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)

```r
> 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_id","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

> miRNA[1:5,]

<table>
<thead>
<tr>
<th>mirbase_id</th>
<th>ensembl_gene_id</th>
<th>start_position</th>
<th>chromosome_name</th>
</tr>
</thead>
<tbody>
<tr>
<td>hsa-mir-622</td>
<td>ENSG00000207858</td>
<td>90883436</td>
<td>13</td>
</tr>
<tr>
<td>hsa-mir-19a</td>
<td>ENSG00000207610</td>
<td>92003145</td>
<td>13</td>
</tr>
<tr>
<td>hsa-mir-92a-1</td>
<td>ENSG00000207968</td>
<td>92003568</td>
<td>13</td>
</tr>
<tr>
<td>hsa-mir-18a</td>
<td>ENSG00000199180</td>
<td>92002997</td>
<td>13</td>
</tr>
<tr>
<td>hsa-mir-320d-1</td>
<td>ENSG00000211491</td>
<td>41301964</td>
<td>13</td>
</tr>
</tbody>
</table>
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](http://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,SPLICE_SITE,F
RAMESHIFT_CODING,SPLICE_SITE,NON_SYNONYMOUS_CODING,SPLICE_SITE,SYN
ONYMOUS_CODING,SPLICE_SITE,SYNONYMOUS_CODING,
5PRIME_UTR,SPLICE_SITE,5PRIME_UTR,3PRIME_UTR,SPLICE_SITE,
3PRIME_UTR,INTRONIC,ESSENTIAL_SPLICE_SITE,INTRONIC,SPLICE_SITE,INTRONIC,
UPSTREAM,DOWNTSTREAM]"

> 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. *hgnce_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

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

> seq[1,]

gene_exon
1
TACAAGGGTCAGGTGCCTGAGAACGAGGCTAACGTCGTAATCACCACACTGA
AAGTGACTGATGCTGATGCCCTGAGAAGCGAGGCCCTAACGTCGTAATCACCACACTGA
AAGTGACTGATGCTGATGCCCTGAGAAGCGAGGCCCTAACGTCGTAATCACCACACTGA
AAGTGACTGATGCTGATGCCCTGAGAAGCGAGGCCCTAACGTCGTAATCACCACACTGA
AAGTGACTGATGCTGATGCCCTGAGAAGCGAGGCCCTAACGTCGTAATCACCACACTGA
AAGTGACTGATGCTGATGCCCTGAGAAGCGAGGCCCTAACGTCGTAATCACCACACTGA
AAGTGACTGATGCTGATGCCCTGAGAAGCGAGGCCCTAACGTCGTAATCACCACACTGA
AAGTGACTGATGCTGATGCCCTGAGAAGCGAGGCCCTAACGTCGTAATCACCACACTGA
hgcnc_symbol
1  CDH1
```
TASK 6 - Ensembl

Retrieve 2000bp sequence upstream of the APC and CUL1 translation start site
Task 6 - Ensembl

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

> promoter

coding_gene_flank
1 TTGTTTCATCTGAAGAGTTGATTTTTTTATTCCTGTAATA
2 TCCGTAGCAGTTGAATGTG

hgnc_symbol
1 APC
2 CUL1
```
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_at
2 1888_s_at KIT  Gga.606.1.S1_at
Ensembl Archives

- Provide alternate host

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

<table>
<thead>
<tr>
<th>biomart</th>
<th>version</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 ENSEMBL_MART_ENSEMBL</td>
<td>Ensembl 54</td>
</tr>
<tr>
<td>2 ENSEMBL_MART_SNP</td>
<td>Ensembl Variation 54</td>
</tr>
<tr>
<td>3 ENSEMBL_MART_VEGA</td>
<td>Vega 35</td>
</tr>
<tr>
<td>4 REACTOME</td>
<td>Reactome(CSHL US)</td>
</tr>
<tr>
<td>5 wormbase_current</td>
<td>WormBase (CSHL US)</td>
</tr>
<tr>
<td>6 pride</td>
<td>PRIDE (EBI UK)</td>
</tr>
</tbody>
</table>
```

```r
> 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)

```r
> 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
    5    snp_mart_50    SNP 50

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

```r
> out = getBM(attributes = c("refsnip_id", "allele", "chrom_start"),
  filters = c("chr_name", "chrom_start", "chrom_end"),
  values = list(8, 148350, 158612), mart = snp)

> out[1:5,]

  snp_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
```
HapMap

- public resource that will help researchers find genes associated with human disease and response to pharmaceuticals

- Task 9:
  Retrieve the alleles and allele frequencies of all non-synonymous coding SNPs on chromosome 19 in the Yoruban population
HapMap

```r
> hapmap = useMart("HapMap_rel27", dataset="hm27_variation_yri")
> yri = getBM(c("chrom","start","alleles","ref_allele","ref_allele_freq","other_allele_freq"), filters=c("chrom","coding_nonsynon"), values=list("chr19",TRUE), mart=hapmap)
```
**HapMap**

```r
> head(yri)

<table>
<thead>
<tr>
<th>chrom</th>
<th>start</th>
<th>alleles</th>
<th>ref_allele</th>
<th>ref_allele_freq</th>
<th>other_allele_freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>chr19</td>
<td>244828</td>
<td>C/G</td>
<td>G</td>
<td>0.458</td>
<td>0.542</td>
</tr>
<tr>
<td>chr19</td>
<td>244913</td>
<td>C/T</td>
<td>C</td>
<td>0.721</td>
<td>0.279</td>
</tr>
<tr>
<td>chr19</td>
<td>244934</td>
<td>A/G</td>
<td>A</td>
<td>0.429</td>
<td>0.571</td>
</tr>
<tr>
<td>chr19</td>
<td>278923</td>
<td>C/T</td>
<td>C</td>
<td>0.996</td>
<td>0.004</td>
</tr>
<tr>
<td>chr19</td>
<td>285441</td>
<td>C/T</td>
<td>C</td>
<td>1.000</td>
<td>0.000</td>
</tr>
<tr>
<td>chr19</td>
<td>313283</td>
<td>C/T</td>
<td>C</td>
<td>0.947</td>
<td>0.053</td>
</tr>
</tbody>
</table>
```

BioC 2010
COSMIC

- Catalogue Of Somatic Mutations In Cancer (Sanger)

Note:
Need devel version of biomaRt (>= 2.5.1)
TASK 10 - COSMIC

Retrieve all known mutations in the following two cell lines: MCF7 and BT474

Attributes to query:

sample_name
gene_name
aa_mut_syntax
mut_type_cds
zygosity
> cosmic = useMart("CosmicMart", dataset = "COSMIC47")
> mut = getBM(c("sample_name","gene_name","aa_mut_syntax","aa_mut_start","mut_type_cds","zygosity"), filters = "sample_name", values = c("MCF7", "BT474"), mart = cosmic)
> mut[1:10,]

<table>
<thead>
<tr>
<th>sample_name</th>
<th>gene_name</th>
<th>aa_mut_syntax</th>
<th>mut_type_cds</th>
<th>zygosity</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCF7</td>
<td>ERBB2</td>
<td></td>
<td>Unknown</td>
<td></td>
</tr>
<tr>
<td>MCF7</td>
<td>CDKN2A</td>
<td>p.o?</td>
<td>Deletion</td>
<td></td>
</tr>
<tr>
<td>MCF7</td>
<td>RB1</td>
<td></td>
<td>Homozygous</td>
<td></td>
</tr>
<tr>
<td>MCF7</td>
<td>CDH1</td>
<td></td>
<td>Unknown</td>
<td></td>
</tr>
<tr>
<td>MCF7</td>
<td>STK11</td>
<td></td>
<td>Unknown</td>
<td></td>
</tr>
<tr>
<td>MCF7</td>
<td>FBXW7</td>
<td></td>
<td>Unknown</td>
<td></td>
</tr>
<tr>
<td>MCF7</td>
<td>PIK3CA</td>
<td>p.E545K</td>
<td>Unknown</td>
<td></td>
</tr>
<tr>
<td>MCF7</td>
<td>SMAD4</td>
<td></td>
<td>Unknown</td>
<td></td>
</tr>
<tr>
<td>MCF7</td>
<td>BRAF</td>
<td></td>
<td>Unknown</td>
<td></td>
</tr>
<tr>
<td>MCF7</td>
<td>EGFR</td>
<td></td>
<td>Unknown</td>
<td></td>
</tr>
</tbody>
</table>
Reactome

- Reactome is an open-source and manually curated pathway database that provides pathway analysis tools for life science researcher
- [http://www.reactome.org](http://www.reactome.org)
- Task 11:
  Retrieve uniprot ids for human genes involved in the following pathways: DNA Repair, Signaling by WNT, Muscle contraction
Reactome

```r
> reactome = useMart("REACTOME", dataset="pathway")
> ids = getBM(c("referencedatabase_uniprot","_displayname"), filters=c("_displayname","species_selection"), value=list(c("DNA Repair","Signaling by WNT","Muscle contraction"), "Homo sapiens"), mart=reactome)
```
Reactome

> head(ids)

<table>
<thead>
<tr>
<th>referencedatabase_uniprot_displayname</th>
<th>1</th>
<th>P62988</th>
<th>DNA Repair</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>P52435</td>
<td>DNA Repair</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>P36954</td>
<td>DNA Repair</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>P30876</td>
<td>DNA Repair</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>O15514</td>
<td>DNA Repair</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>P62487</td>
<td>DNA Repair</td>
<td></td>
</tr>
</tbody>
</table>
Gramene

- Gramene is a curated, open-source, data resource for comparative genome analysis in the grasses.
- Rice, Maize and Arabidopsis
TASK 12 - Gramene

Retrieve the ensembl gene id, external gene id, a description and the start positions of all genes from *Arabidopsis thaliana* that are located on chromosome 1 between basepair 30.000 and 41.000
TASK 12 - Gramene

```r
> gramene = useMart("ENSEMBL_MART_ENSEMBL", dataset=""athaliana_eg_gene"")
> getBM(c("ensembl_gene_id","external_gene_id","description","start_position","end_position"), filters=c("chromosome_name","start","end"), values=list("1","30000","41000"), mart=gramene)
```
> getBM(c("ensembl_gene_id","external_gene_id","description","start_position", "end_position"), filters=c("chromosome_name","start","end"), values=list("1", "30000", "41000"), mart=gramene)

<table>
<thead>
<tr>
<th>ensembl_gene_id</th>
<th>external_gene_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT1G01050</td>
<td>TAIR-G</td>
</tr>
<tr>
<td>AT1G01070</td>
<td>TAIR-G</td>
</tr>
<tr>
<td>AT1G01040</td>
<td>TAIR-G</td>
</tr>
<tr>
<td>AT1G01060</td>
<td>TAIR-G</td>
</tr>
</tbody>
</table>

description
1 pyrophosphorylase 1; inorganic diphosphatase; Encodes a soluble protein with inorganic pyrophosphatase activity that is highly specific for Mg-inorganic pyrophosphate.

2 nodulin MtN21 family protein; nodulin MtN21 family protein; LOCATED IN: membrane; EXPRESSED IN: 17 plant structures; EXPRESSED DURING: 7 growth stages; CONTAINS InterPro DOMAIN/s: Protein of unknown function DUF6, transmembrane (InterPro:IPR000620); BEST Arabidopsis thaliana protein match is: nodulin MtN21 family protein (TAIR:AT1G11460.1); Has 1705 Blast hits to 1692 proteins in 315 species: Archaea - 18; Bacteria - 780; Metazoa - 4; Fungi - 6; Plants - 641; Viruses - 0; Other Eukaryotes - 256 (source: NCBI BLink).

3 DCL1 (DICER-LIKE 1); ATP-dependent helicase/double-stranded RNA binding / protein binding / ribonuclease III; Encodes a Dicer homolog. Dicer is a RNA helicase involved in microRNA processing. Mutations in this locus can result in embryo lethality. Embryo shape at seed maturity is globular-elongate. Other mutants convert the floral meristems to an indeterminate state, others yet show defects in ovule development. DCL1 is able to produce miRNAs and siRNAs.

4 DNA binding / transcription factor; LHY encodes a myb-related putative transcription factor involved in circadian rhythm along with another myb transcription factor CCA1.
Wormbase

- Database on the genetics of C elegans and related nematodes.
TASK 13 - Wormbase

Determine the RNAi ids and the observed phenotypes for the gene with wormbase gene id: WBGene00006763
TASK 13 - Wormbase

> worm = useMart("wormbase195",
                   dataset="wormbase_rnai")

> pheno = getBM(c("rnai" , "phenotype_primary_name" ),
                filters="gene", values="WBGene00006763",
                mart=worm)
### TASK 13 - Wormbase

```r
> pheno

<table>
<thead>
<tr>
<th></th>
<th>rnai</th>
<th>phenotype_primary_name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WBRNAio00021278</td>
<td>slow_growth</td>
</tr>
<tr>
<td>2</td>
<td>WBRNAio00021278</td>
<td>postembryonic_development_abnormal</td>
</tr>
<tr>
<td>3</td>
<td>WBRNAio00021278</td>
<td>embryonic_lethal</td>
</tr>
<tr>
<td>4</td>
<td>WBRNAio00021278</td>
<td>larval_lethal</td>
</tr>
<tr>
<td>5</td>
<td>WBRNAio00021278</td>
<td>larval_arrest</td>
</tr>
<tr>
<td>6</td>
<td>WBRNAio00021278</td>
<td>maternal_sterile</td>
</tr>
<tr>
<td>7</td>
<td>WBRNAio00021278</td>
<td>Abnormal</td>
</tr>
<tr>
<td>8</td>
<td>WBRNAio00021278</td>
<td>sterile_progeny</td>
</tr>
<tr>
<td>9</td>
<td>WBRNAio00026915</td>
<td>slow_growth</td>
</tr>
<tr>
<td>10</td>
<td>WBRNAio00026915</td>
<td>postembryonic_development_abnormal</td>
</tr>
<tr>
<td>11</td>
<td>WBRNAio00026915</td>
<td>embryonic_lethal</td>
</tr>
<tr>
<td>12</td>
<td>WBRNAio00026915</td>
<td>larval_lethal</td>
</tr>
</tbody>
</table>
```
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
- Sometimes server can be down
Reporting bugs

- Check if [http://www.biomart.org](http://www.biomart.org) is online
- Check with MartView if you get the same output
  - Yes: contact database e.g. helpdesk@ensembl.org
  - No: contact me
    sdurinck@lbl.gov
Acknowledgements

- EBI
  - BioMart Team
    - Arek Kasprzyk
    - Syed Haider
  - Ensembl Team
    - Rhoda Kinsella
    - Ewan Birney
- EMBL
  - Wolfgang Huber