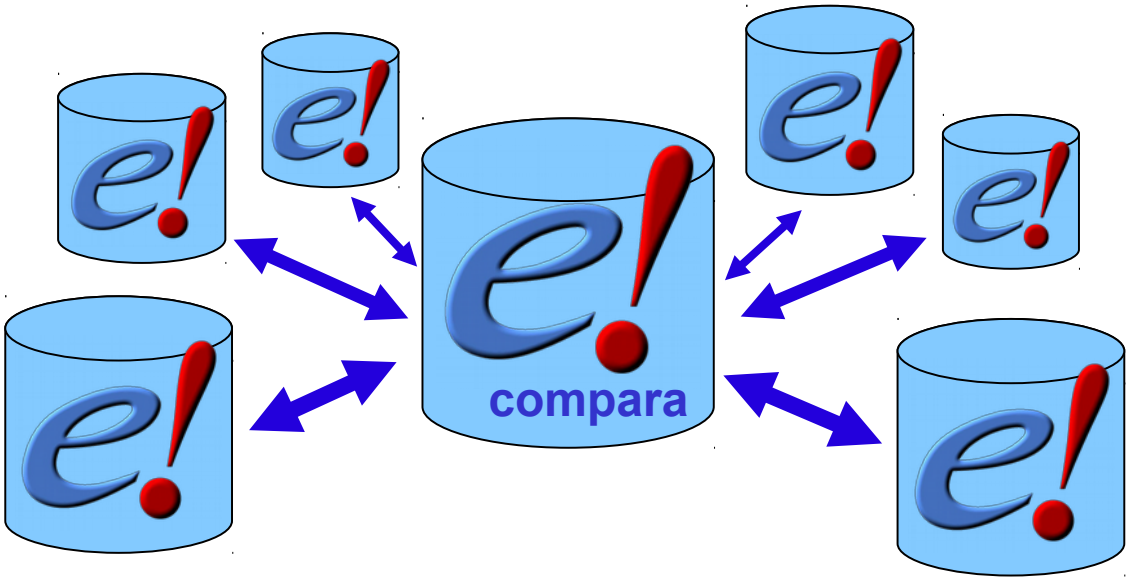
# Ensembl Compara Perl API




*Wasiu Akanni & Matthieu Muffato*

*API workshop - Cambridge*

Jun 2016

# Outline of the course

- Introduction about Compara
  - Resources
  - API
- Inputs
  - Species, Chromosomes, Genes
- Outputs
  - Gene analyses
  - Genome analyses

# Outline of the course

- Introduction about Compara
  - Resources
  - API
- Inputs
  - Species, Chromosomes, Genes
- Outputs
  - Gene analyses
  - Genome analyses

# What is Ensembl Compara?

A single database which contains precalculated comparative genomics data and which is linked to all the Ensembl Species (69 in e84) databases.

Access via perl API and mysql

A production system for generating that database
*(i.e. pipelines and SOPs, not in this presentation)*

# The genomes in Ensembl

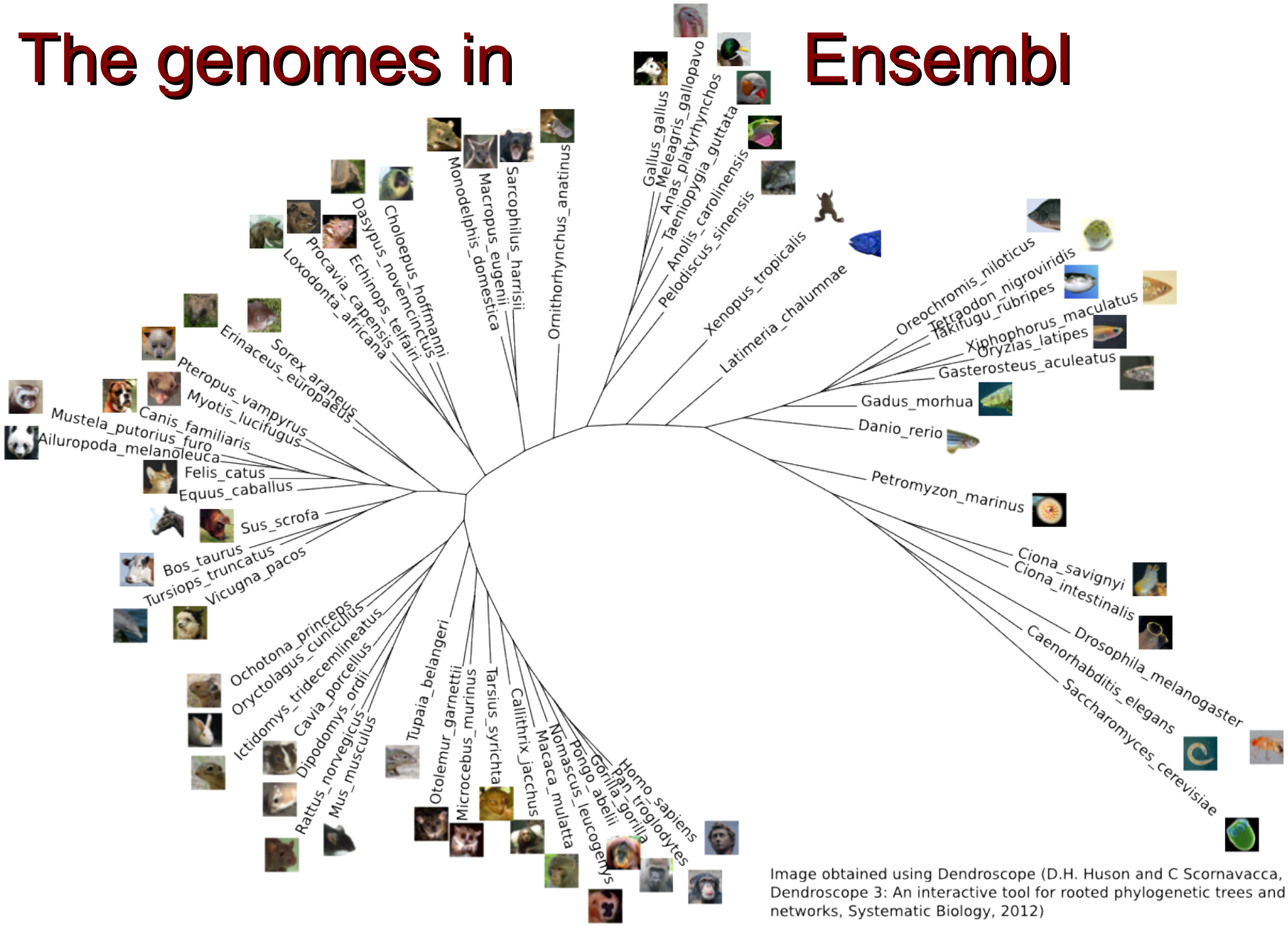



Image obtained using Dendroscope (D.H. Huson and C Scornavacca, Dendroscope 3: An interactive tool for rooted phylogenetic trees and networks, Systematic Biology, 2012)

# Compara data

## Genome level

Whole genome alignments (pairwise and multiple)

Constrained elements (based on multiple align.)

Syntenic regions (based on pair-wise align.)

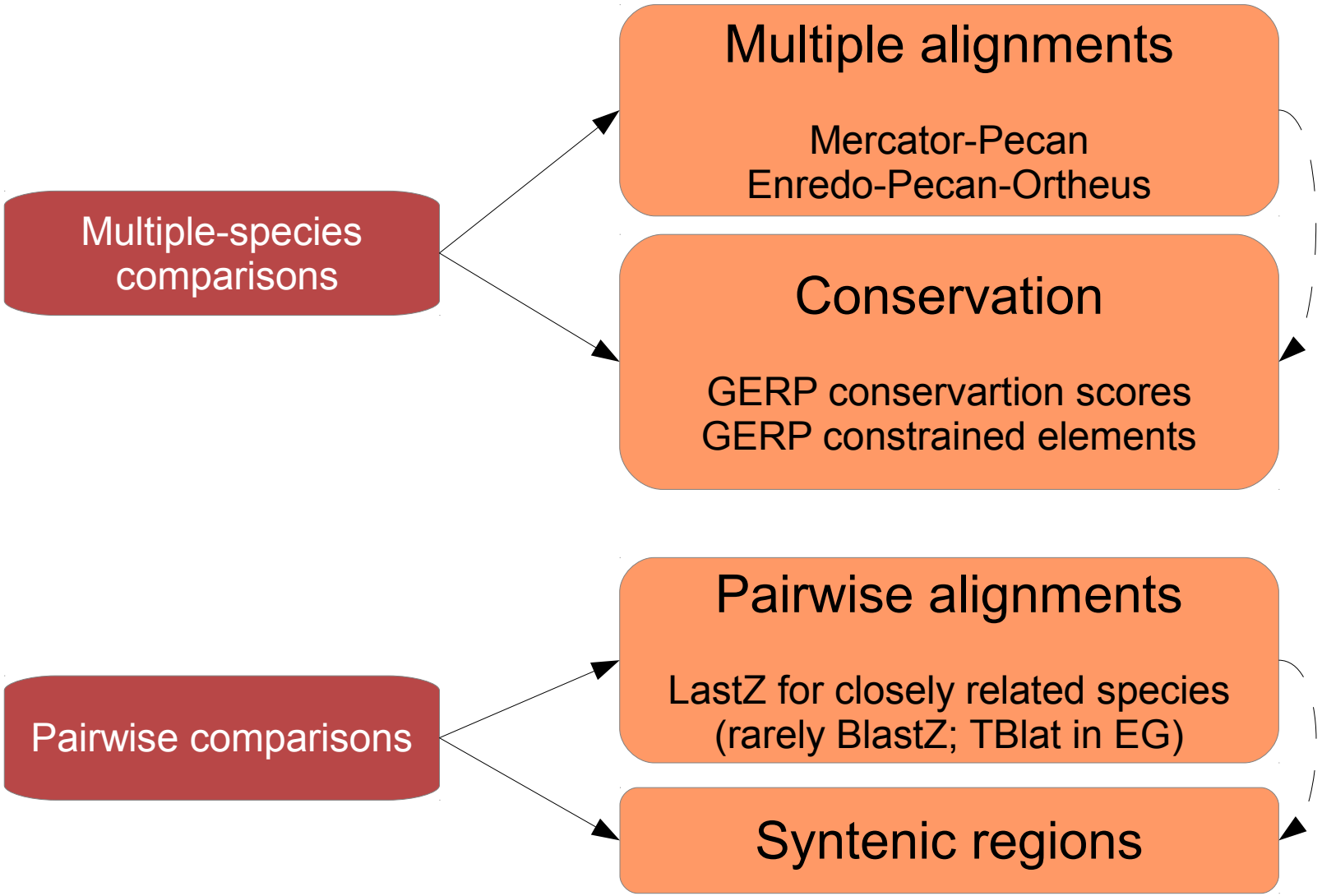
## Gene level

Families (clusters of proteins + multiple align.)

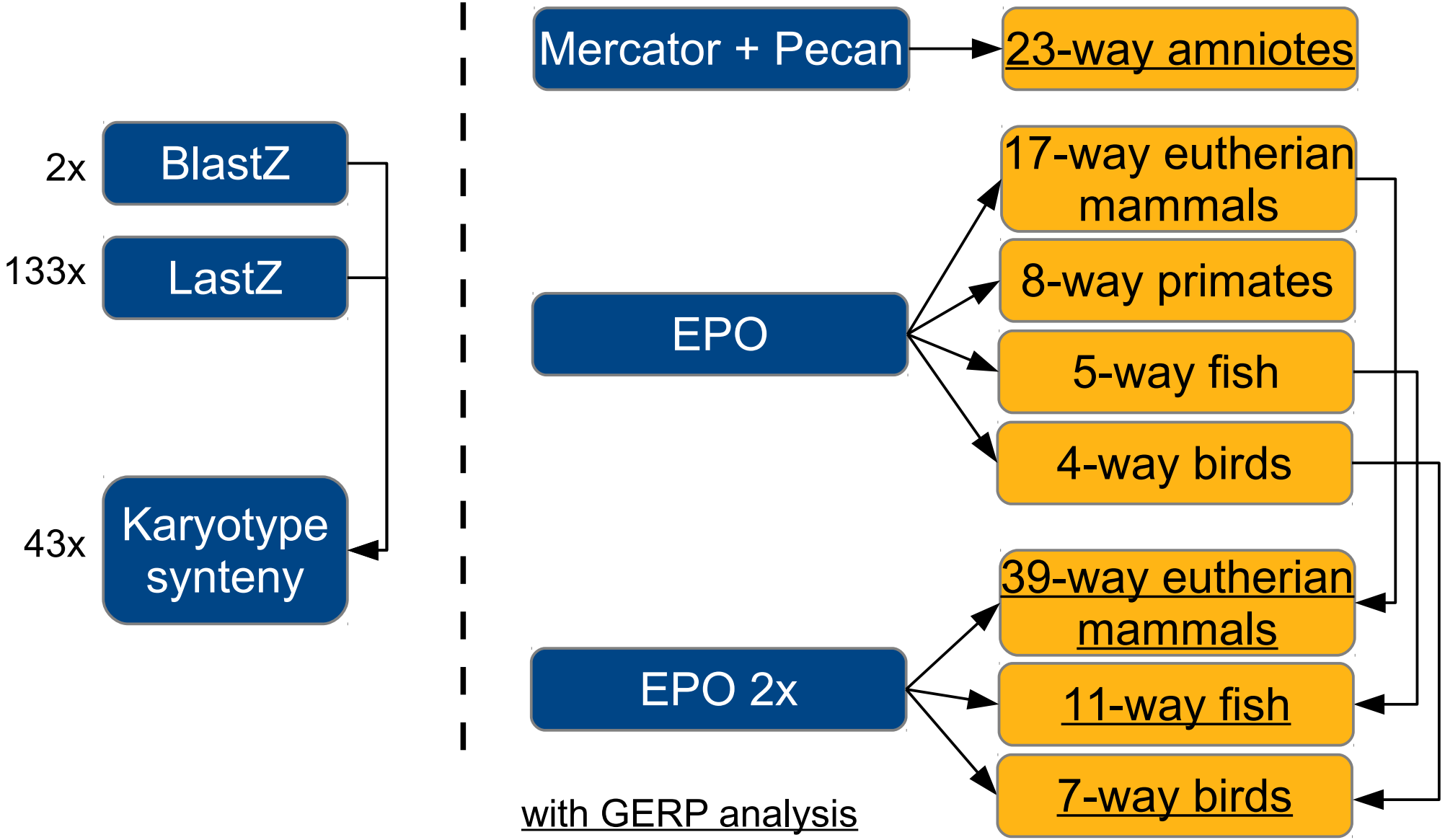
Gene trees (proteins, non-coding RNAs)

Gene orthology / paralogy predictions

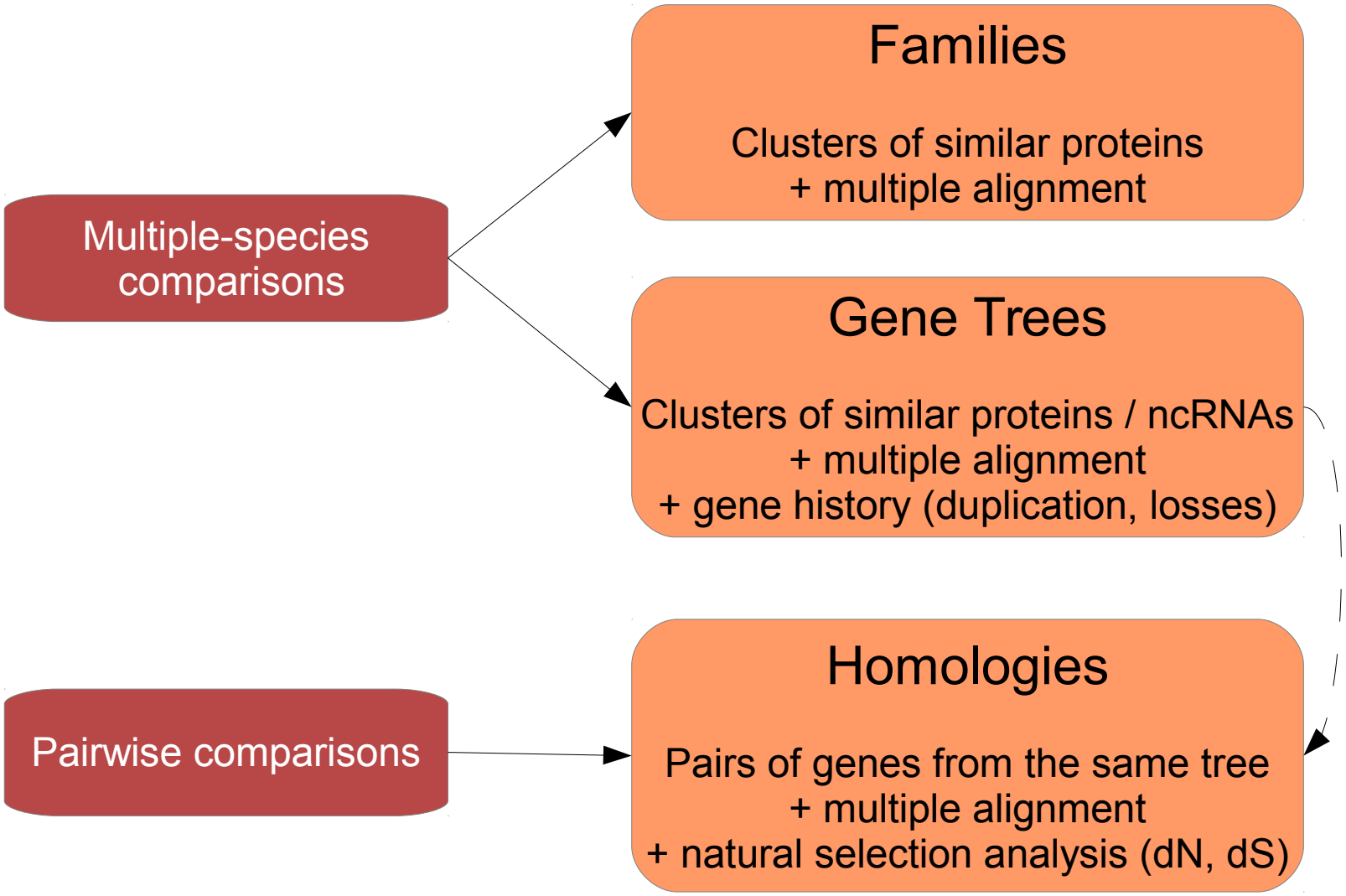
# Nucleotide sequence analyses



# Nucleotide sequence analyses in e!84



# Gene analyses



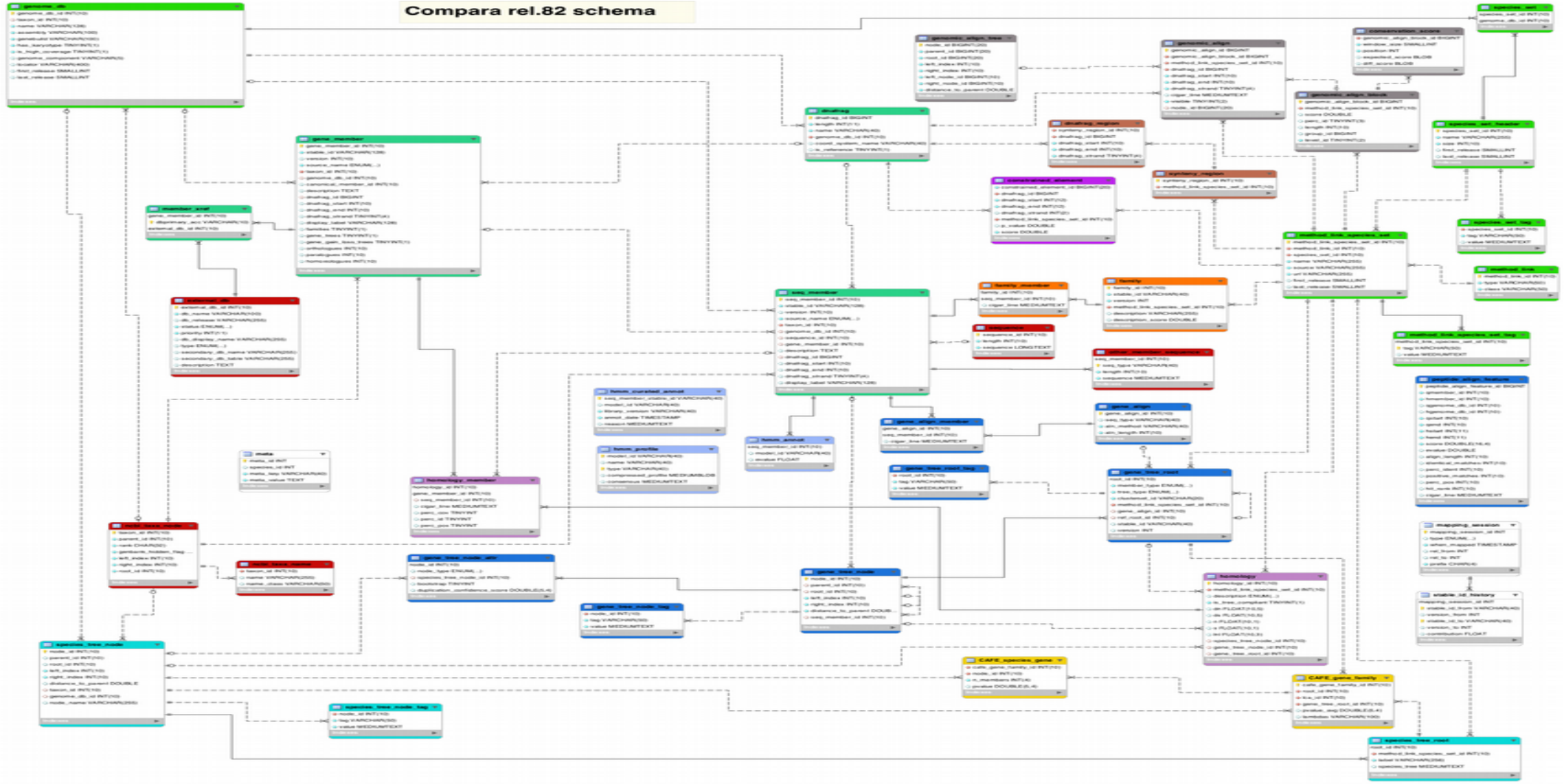
# Outline of the course

- Introduction about Compara
  - Resources
  - API

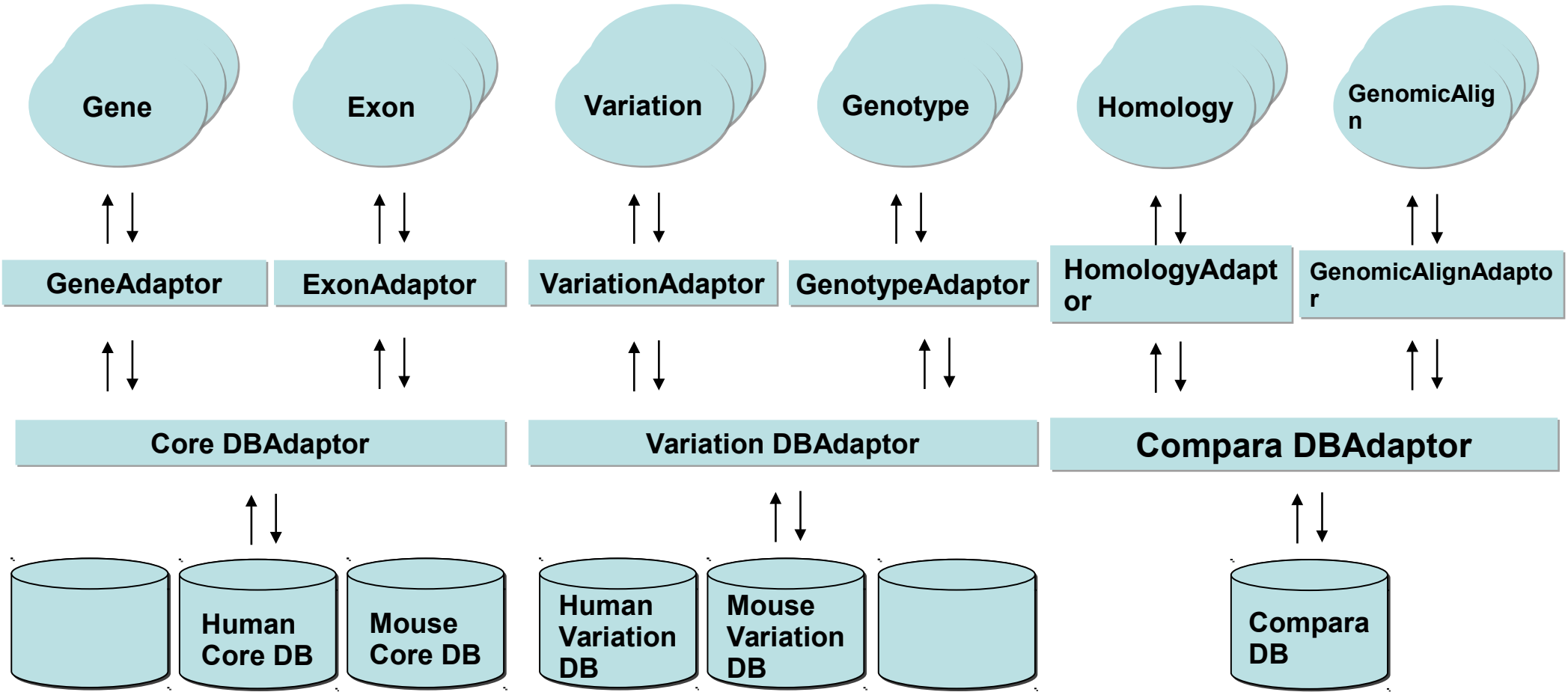
- Inputs
  - Species, Chromosomes, Genes
- Outputs
  - Gene analyses
  - Genome analyses

# The Compara Perl API

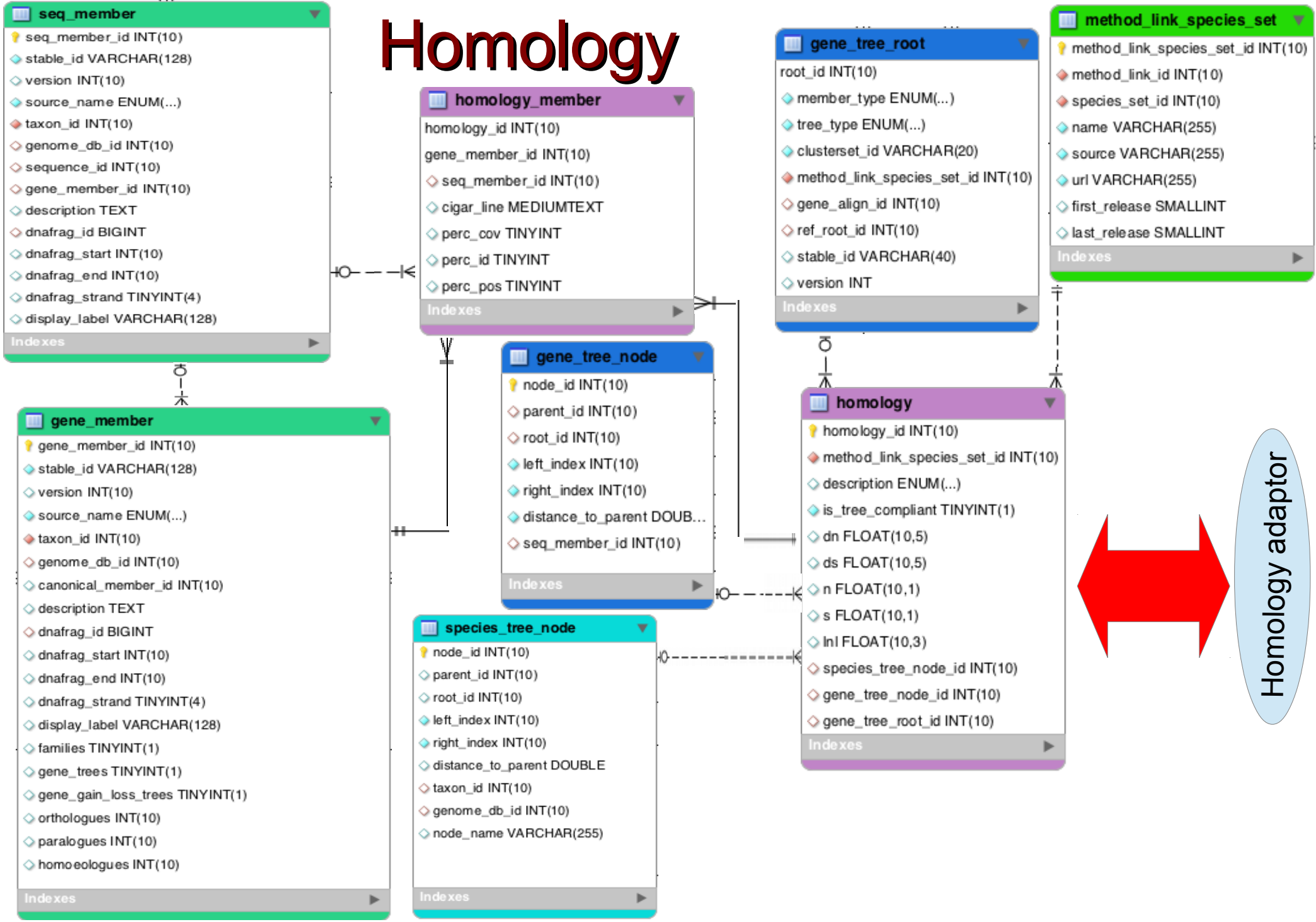
- Written in Object-Oriented Perl
- Used to retrieve data from and store data into the Ensembl Compara database
- Generalized to extend to non-Ensembl genomic data (Uniprot)
- Follows same ‘Object Adaptor’ & ‘Data Object’ design as the Core API


Compara rel.82 schema

# Ensembl API Architecture



# Homology


seq_member
seq_member_id INT(10)
stable_id VARCHAR(128)
version INT(10)
source_name ENUM(...)
taxon_id INT(10)
genome_db_id INT(10)
sequence_id INT(10)
gene_member_id INT(10)
description TEXT
dnafrag_id BIGINT
dnafrag_start INT(10)
dnafrag_end INT(10)
dnafrag_strand TINYINT(4)
display_label VARCHAR(128)
Indexes
homology_member
homology_id INT(10)
gene_member_id INT(10)
seq_member_id INT(10)
cigar_line MEDIUMTEXT
perc_cov TINYINT
perc_id TINYINT
perc_pos TINYINT
Indexes
gene_tree_root
root_id INT(10)
member_type ENUM(...)
tree_type ENUM(...)
clusterset_id VARCHAR(20)
method_link_species_set_id INT(10)
gene_align_id INT(10)
ref_root_id INT(10)
stable_id VARCHAR(40)
version INT
Indexes
method_link_species_set
method_link_species_set_id INT(10)
method_link_id INT(10)
species_set_id INT(10)
name VARCHAR(255)
source VARCHAR(255)
url VARCHAR(255)
first_release SMALLINT
last_release SMALLINT
Indexes
gene_tree_node
node_id INT(10)
parent_id INT(10)
root_id INT(10)
left_index INT(10)
right_index INT(10)
distance_to_parent DOUB...
seq_member_id INT(10)
Indexes
gene_member
gene_member_id INT(10)
stable_id VARCHAR(128)
version INT(10)
source_name ENUM(...)
taxon_id INT(10)
genome_db_id INT(10)
canonical_member_id INT(10)
description TEXT
dnafrag_id BIGINT
dnafrag_start INT(10)
dnafrag_end INT(10)
dnafrag_strand TINYINT(4)
display_label VARCHAR(128)
families TINYINT(1)
gene_trees TINYINT(1)
gene_gain_loss_trees TINYINT(1)
orthologues INT(10)
paralogues INT(10)
homoeologues INT(10)
Indexes
homology
homology_id INT(10)
method_link_species_set_id INT(10)
description ENUM(...)
is_tree_compliant TINYINT(1)
dn FLOAT(10,5)
ds FLOAT(10,5)
n FLOAT(10,1)
s FLOAT(10,1)
lnl FLOAT(10,3)
species_tree_node_id INT(10)
gene_tree_node_id INT(10)
gene_tree_root_id INT(10)
Indexes
species_tree_node
node_id INT(10)
parent_id INT(10)
root_id INT(10)
left_index INT(10)
right_index INT(10)
distance_to_parent DOUBLE
taxon_id INT(10)
genome_db_id INT(10)
node_name VARCHAR(255)
Indexes
Homology adaptor

# Compara template script

```
use strict;
use Bio::EnsEMBL::Registry;
my $reg = "Bio::EnsEMBL::Registry";

# Auto-configure the registry
$reg->load_registry_from_db(
    -host => "ensembldb.ensembl.org",
    -user => "anonymous"
);


# Get the adaptor object for the data type you want
# e.g. GeneTree
my $xx_adaptor =  $reg->get_adaptor("Multi", "compara", "XX");


# Fetch the data objects using the adaptor
# e.g. get all the genes in a given gene tree
my $all_interesting_xx = $xx_adaptor->fetch_all_by_YY();


print "All XX objects from E!Compara :\n";
foreach my $this_xx (@$all_interesting_xx) {
  # Do some stuff with the data object
  print "\t", $this_xx->stable_id, "\n";
}
```

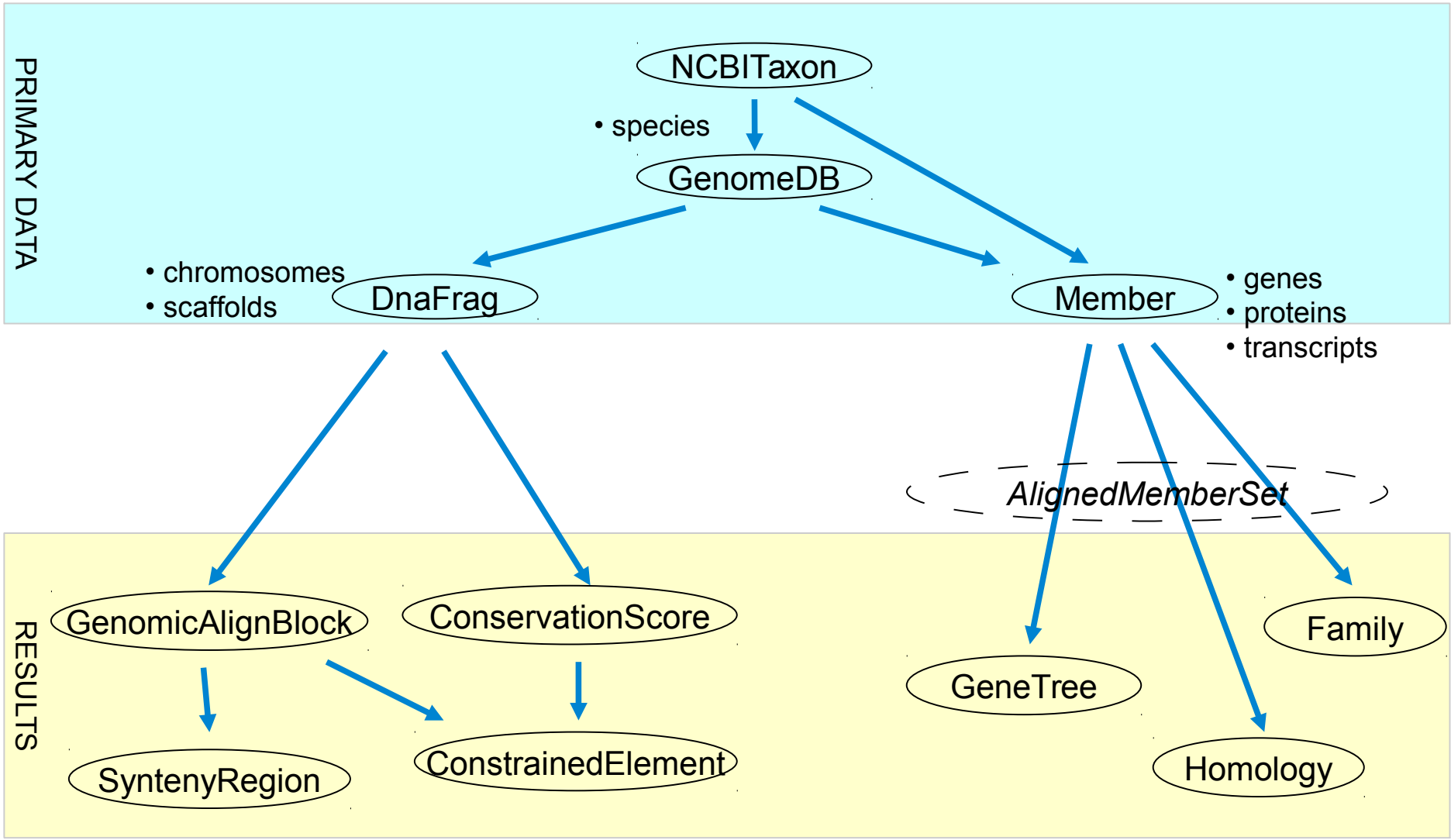
# Help & Useful documentation

- perldoc – Viewer for offline API documentation
  - `shell> perldoc Bio::EnsEMBL::Compara::GenomeDB`
  - `shell> perldoc Bio::EnsEMBL::Compara::DBSQL::DnaFragAdaptor`

- Online documents (website)
  - http://e84.ensembl.org/info/docs/Doxygen/compara-api/index.html
  - http://e84.ensembl.org/info/docs/api/compara/index.html

- Mailing lists:
  - dev@ensembl.org
  - helpdesk@ensembl.org

# Compara object model overview



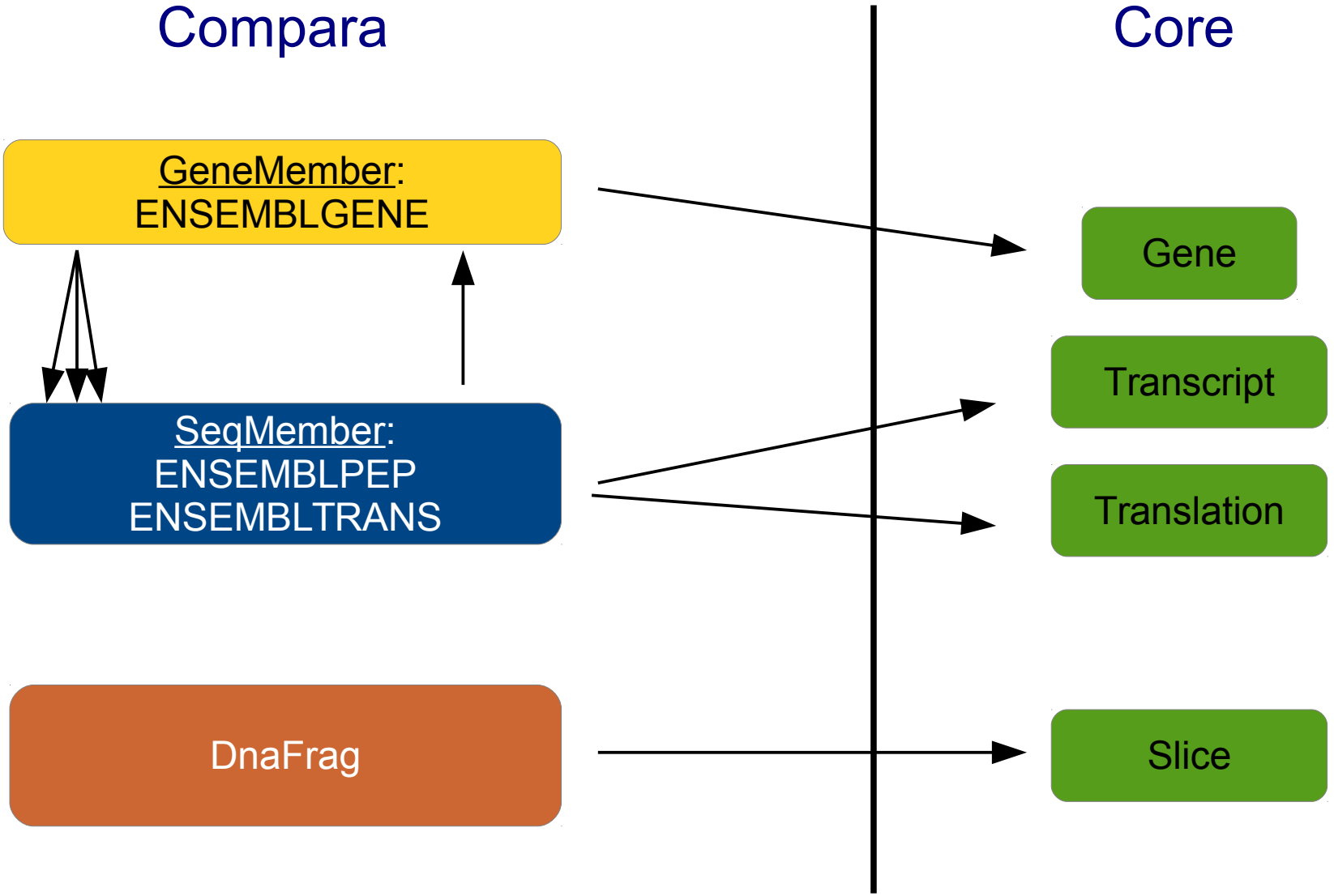
# Outline of the course

- Introduction about Compara
  - Resources
  - API
- Inputs
  - Species, Chromosomes, Genes 
- Outputs
  - Gene analyses
  - Genome analyses

# Links between Compara and Core

- Compara only stores references to the Core objects
- The full data lies in the core databases

# Links between Compara and Core



# GenomeDB

- Represents a species
  Links the Compara database to the Core species databases

| Attributes | Methods |
|---|---|
| Species name | `$genomedb->name()` |
| Assembly | `$genomedb->assembly()` |
| Gene build | `$genomedb->genebuild()` |
| Taxon | `$genomedb->taxon_id()` |
| **Adaptor methods** | |
| `$genomedb_adaptor->fetch_all(...)` | |
| `$genomedb_adaptor->fetch_by_registry_name(...)` | |

# DnaFrag

- Represents a top-level region in the Compara database.
- Equivalent to a whole-sequence Slice

| Attributes | Methods |
|---|---|
| Region name | *$dnafrag->name()* |
| Region type | *$dnafrag->coord_system_name()* |
| **Adaptor methods** | |
| `$dnafrag_adaptor->fetch_all_by_GenomeDB_region(...)` | |
| `$dnafrag_adaptor->fetch_by_Slice(...)` | |
| `$nafrag_adaptor->fetch_by_GenomeDB_and_name(...)` | |

# Code Examples

## 1. GenomeDB

```
my $genome_db_adaptor = Bio::EnsEMBL::Registry->get_adaptor( "Multi", "compara", "GenomeDB");
my $list_ref_of_gdbs = $genome_db_adaptor->fetch_all();

foreach my $genome_db( @{ $list_ref_of_gdbs } ){
   print join( "\t", $genome_db->dbID(), $genome_db->name(), $genome_db->assembly() ), "\n";
}
```

## 2. DnaFrag

```
my $dnafrag_adaptor = $reg->get_adaptor("Multi", "compara", "DnaFrag");
my $gorilla_chr_dnafrags = $dnafrag_adaptor→
                           fetch_all_by_GenomeDB_region( $gorilla_genome_db, 'chromosome' );

foreach my $dnafrag (@{ $gorilla_chr_dnafrags }){
   print "Chromsome ", $dnafrag->name(), " contains ", $dnafrag->length(), " bp.\n";
}
```

# Exercises – GenomeDB & DnaFrag

- Print the name, assembly version and genebuild version for all the GenomeDBs in the compara database

- Print all the chromosomes (DnaFrags) for chimpanzee

# *GeneMember* and *SeqMember*

- GeneMember for genes
  - source_name: ENSEMBLGENE
- SeqMember for RNAs and proteins
  - source_name: ENSEMBLPEP, ENSEMBLTRANS, Uniprot/SPTREMBL, Uniprot/SWISSPROT

| Attributes | Methods |
|---|---|
| Stable ID | `$member->stable_id()` |
| Coordinates | `$member->dnafrag->name()`<br>`$member->dnafrag_start()` … |
| Sequence (SeqMember only) | `$member->sequence()` |
| Function | `$member->description()` |
| **Adaptor methods** | |
| `$seq_member_adaptor->fetch_by_stable_id(…)` | |
| `$gene_member_adaptor->fetch_all_by_GenomeDB(…)` | |

# HOWTO: get an Ensembl ID from a gene symbol

- Compara only references genes by their Ensembl stable ID
- From a gene symbol, you first have to use the core API to get the stable id(s)
- Gene symbols may not be unique (for instance: U6)

```
# Get the Human gene adaptor
my $hg_adaptor = $reg->get_adaptor("human", "core", "Gene");

# Get all the genes
my $all_genes = $hg_adaptor->fetch_all_by_external_name(XX);

# For each gene
foreach my $gene (@{$all_genes}) {
   do some stuff with $gene->stable_id();
}
```

# Code Example - Member

```
my $seq_member_adaptor = $reg->get_adaptor("Multi", "compara", "SeqMember");
my $human_seq_members =
                    $seq_member_adaptor->fetch_all_by_GenomeDB($gorilla_genome_db);

# print 10 protein members and 10 transcript members
my ($prot_count, $trans_count) = (0, 0);
foreach my $seq_mem ( @{ $human_seq_members } ) {
   my $type = $seq_mem->source_name();
   if ( $type =~ m/PEP/ && $prot_count < 10 ){
      print $seq_mem->stable_id(), ":", $seq_mem->source_name(), "\n";
      $prot_count++;
   }
   elsif ( $type =~ m/TRANS/ && $trans_count < 10 ){
      print $seq_mem->stable_id(), ":", $seq_mem->source_name(), "\n";
      $trans_count++;
   }
   elsif ( $prot_count >= 10 && $trans_count >= 10 ) {
      last;
   }
}
```

# Exercises - *Member*

- Print the sequence of the Member corresponding to SwissProt protein O93279

- Find and print the sequence of all the protein Members corresponding to the human protein-coding gene(s) FRAS1

# Outline of the course

- Introduction about Compara
  - Resources
  - API
- Inputs
  - Species, Chromosomes, Genes
- Outputs
  - Gene analyses
  - Genome analyses

# *AlignedMemberSet* object

- Base object that represents a set of members aligned together, e.g. a multiple alignment of proteins / ncRNAs
- “Applied” in gene trees, families, and homologies
- No specific adaptor

| Attributes | Methods |
|---|---|
| List of members | `$aln->get_all_Members()`<br>`$aln->get_all_GeneMembers()` |
| Alignment (BioPerl object) | `$aln->get_SimpleAlign()` |
| Description (if available) | `$aln->description()` |
| Stable ID (if available) | `$aln->stable_id()` |

# HOWTO: print a BioPerl alignment

- Compara objects return alignments as BioPerl instances

```
$aln->get_SimpleAlign()
```

- BioPerl provides an AlignIO object to format the actual output in various formats (fasta, clustalw, phylip … )

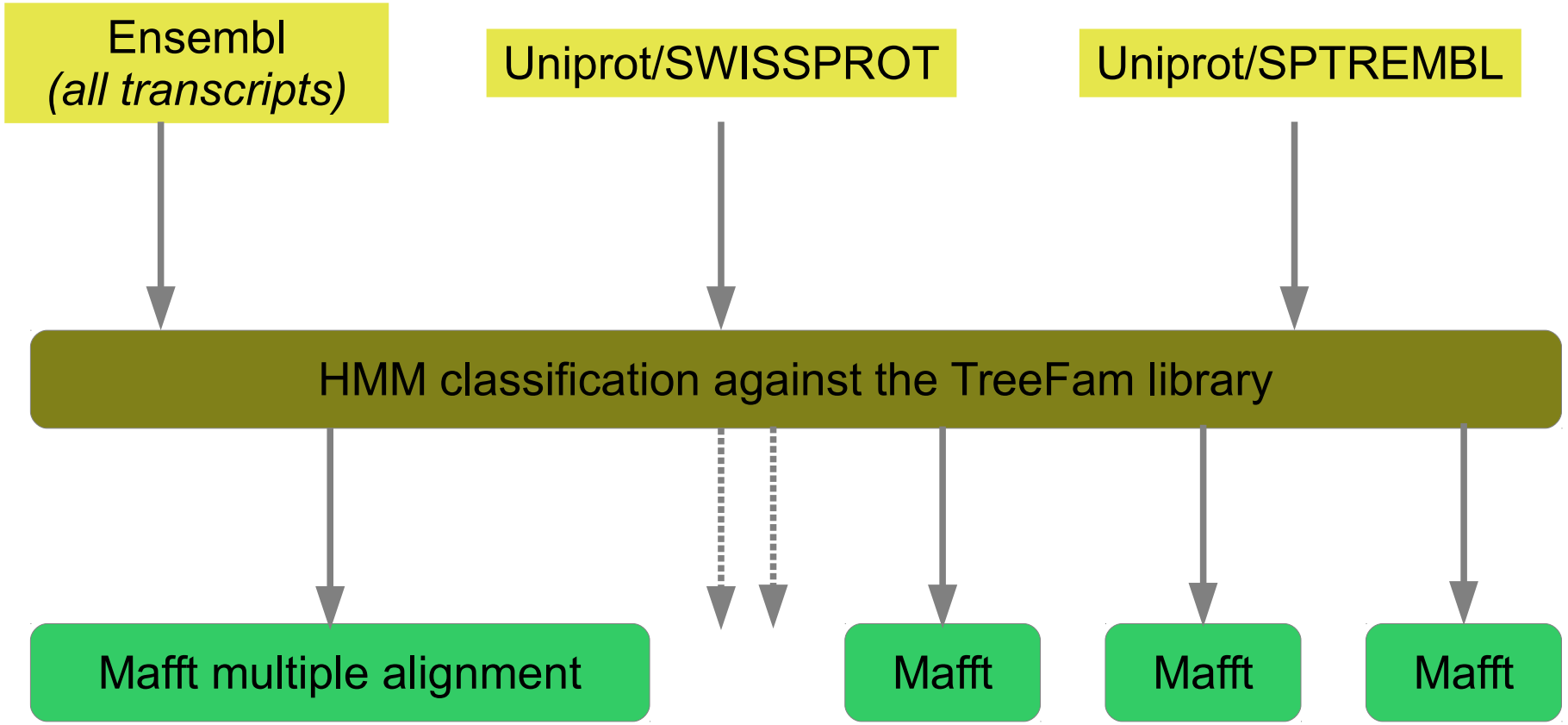
```
use Bio::AlignIO;

# Get the alignIO object from BioPerl
my $alignIO = Bio::AlignIO->newFh(-format => "fasta");

# Print the alignment
print $alignIO $aln;
```

# Families

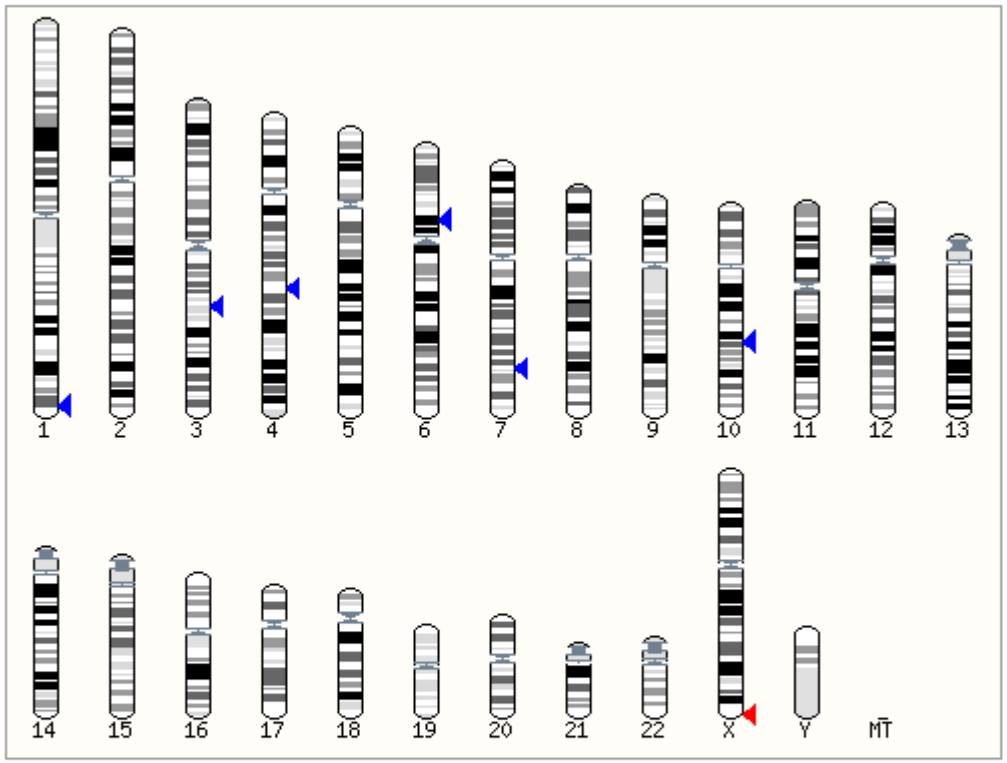
## Families are clusters of similar proteins



# Example: PTHR24240 (opsin) in Human

## HUMAN genes in this family

**Ensembl genes containing proteins in family PTHR24240**




| Gene ID and Location | Gene Name | Description (if known) |
|---|---|---|
| ENSG00000054277<br>Chromosome 1: 241.59m | OPN3 | opsin 3 [Source:HGNC Symbol;Acc:HGNC:14007] |
| ENSG00000163914<br>Chromosome 3: 129.53m | RHO | rhodopsin [Source:HGNC Symbol;Acc:HGNC:10012] |
| ENSG00000180245<br>Chromosome 4: 109.83m | RRH | retinal pigment epithelium-derived rhodopsin homolog Symbol;Acc:HGNC:10450] |

# *Family* object

- (almost) the same methods as in *AlignedMemberSet*
- Alternative transcripts can belong to different families !

| Attributes | Methods |
|---|---|
| Alignment | `$family->get_SimpleAlign()` |
| Biological function | `$family->description()` |
| Gene content | `$family->get_all_Members()` |
| **Adaptor methods** | |
| `$family_adaptor->fetch_all_by_GeneMember(…)`<br>`$family_adaptor->fetch_by_SeqMember(…)` | |
| `$family_adaptor->fetch_by_stable_id(…)` | |

# Code Example - Family

```
my $family_adaptor = $reg->get_adaptor("Multi", "compara", "Family");
my $ddx_families = $family_adaptor→
                              fetch_by_description_with_wildcards('dead box', 1);

# print first 10 family descriptions
my $c = 0;
foreach my $fam ( @{ $ddx_families } ) {
   print $fam->description(), "\n";
   $c++;
   last if $c >= 10;
}
```

# Exercises - Families

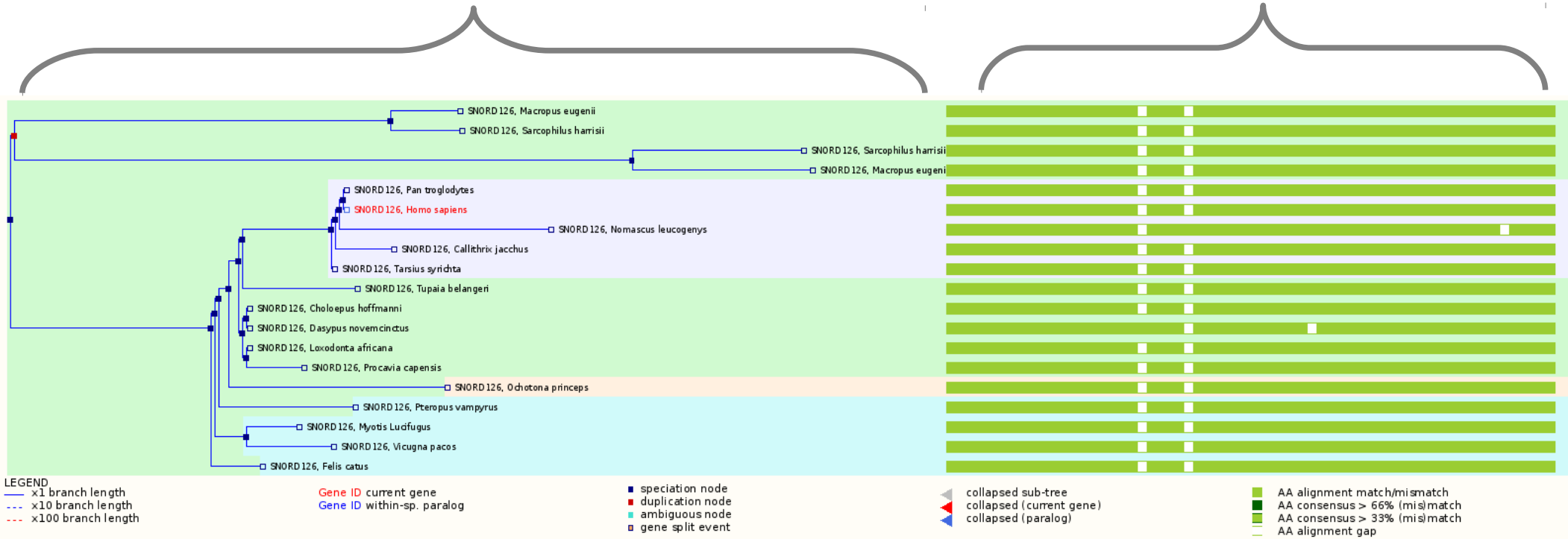
- Get the multiple alignment corresponding to the family with the stable id PTHR10740_SF4

- Get the families predicted for the human gene ENSG00000283087. What do you notice ?

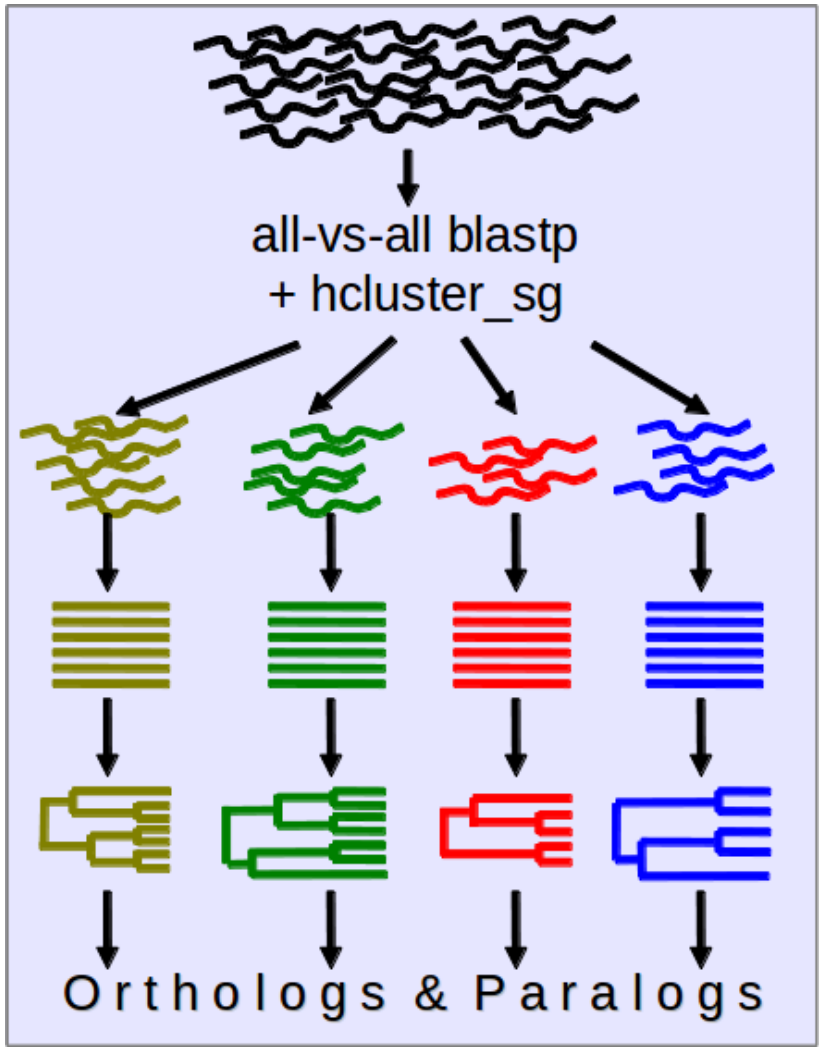
# GeneTree example on the website

Tree

Multiple alignment



# Protein-Tree pipeline overview




All *e!* genes – canonical prot.

BLAST

hcluster_sg

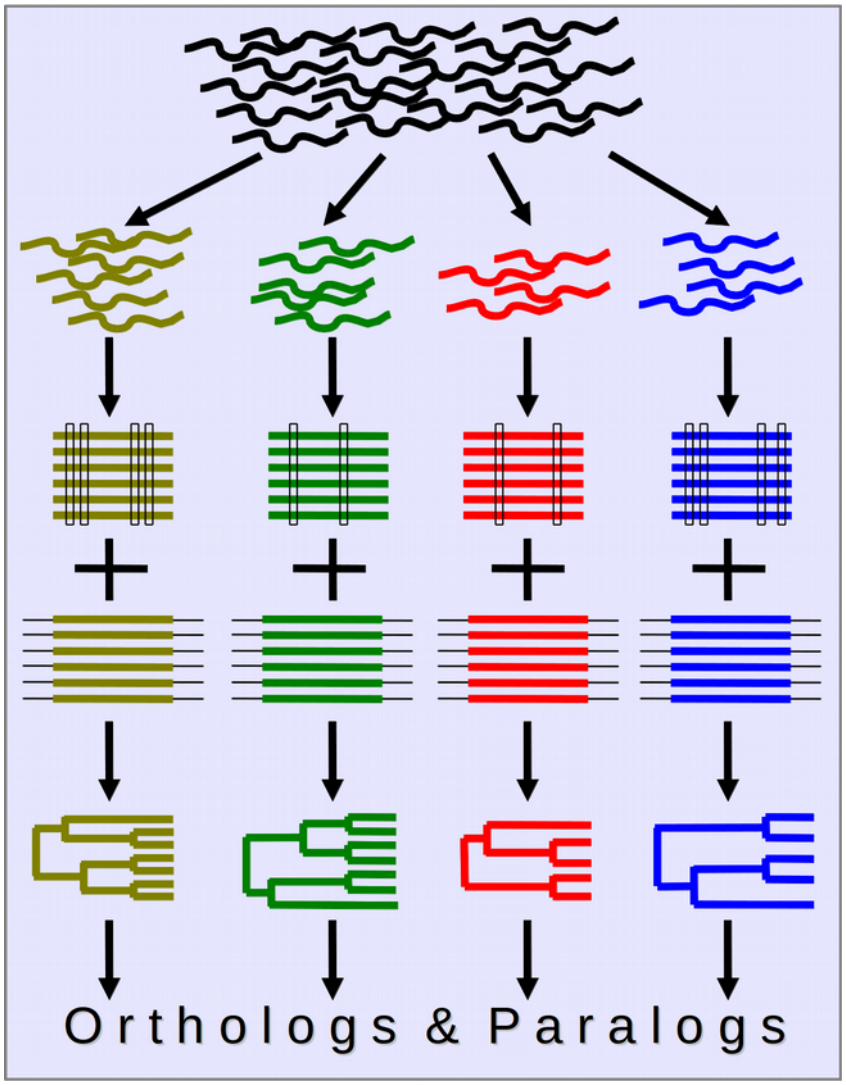
MCoffee: MSA

TreeBeST: (+ reconciliation)

Ortholog/Paralog inference

Vilella et al., Genome Res. 2009

# ncRNA-Tree pipeline overview




All *e!* ncRNA genes

Grouped in Family Models - RFAM

Infernal alignment + RaxML trees

PRANK alignment + NJ/ML trees

TreeBeST (tree reconciliation)

Ortholog/Paralog inference

Pignatelli et al., in preparation

# GeneTree object

- *fetch_all** methods may require some more arguments:

```
-clusterset_id => 'default'
-tree_type => 'tree'
-member_type => 'protein' or 'ncrna'
```

| Attributes | Methods |
|---|---|
| Alignment | *$family->get_SimpleAlign()* |
| Tree export | *$tree->newick_format('simple')*<br>*$tree->nhx_format('full')*<br>*$tree->print_tree()* |
| Stable ID | *$tree->stable_id()* |
| **Adaptor methods** | |
| `$genetree_adaptor->fetch_by_stable_id(…)` | |
| `$genetree_adaptor->fetch_default_for_Member(…)` | |

# Exercises – Protein and ncRNA trees

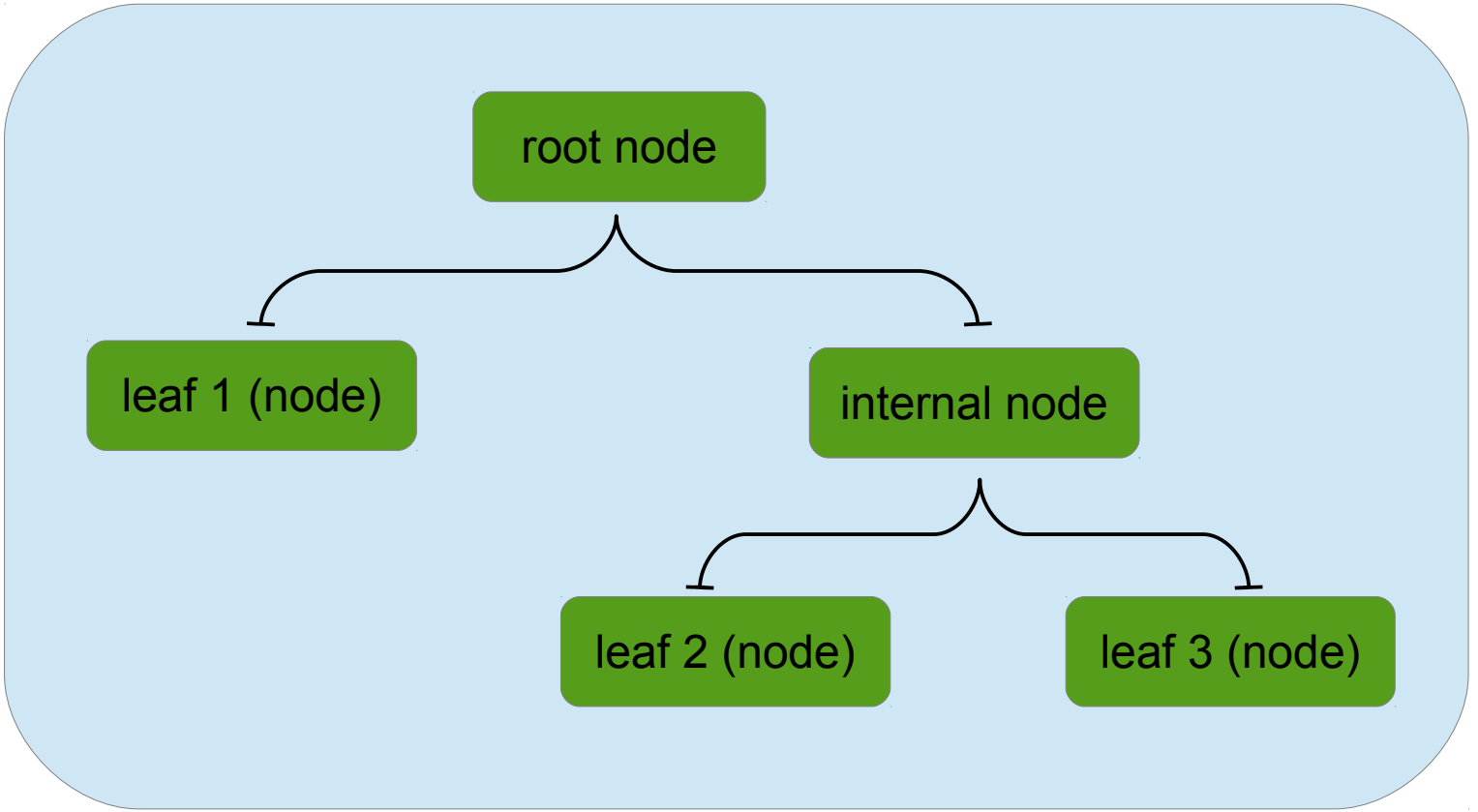
- Print the protein tree with the stable id ENSGT00390000003602

- Print all the members of the tree containing the human ncRNA gene ENSG00000238344, and their alignment

# *GeneTreeNode* object

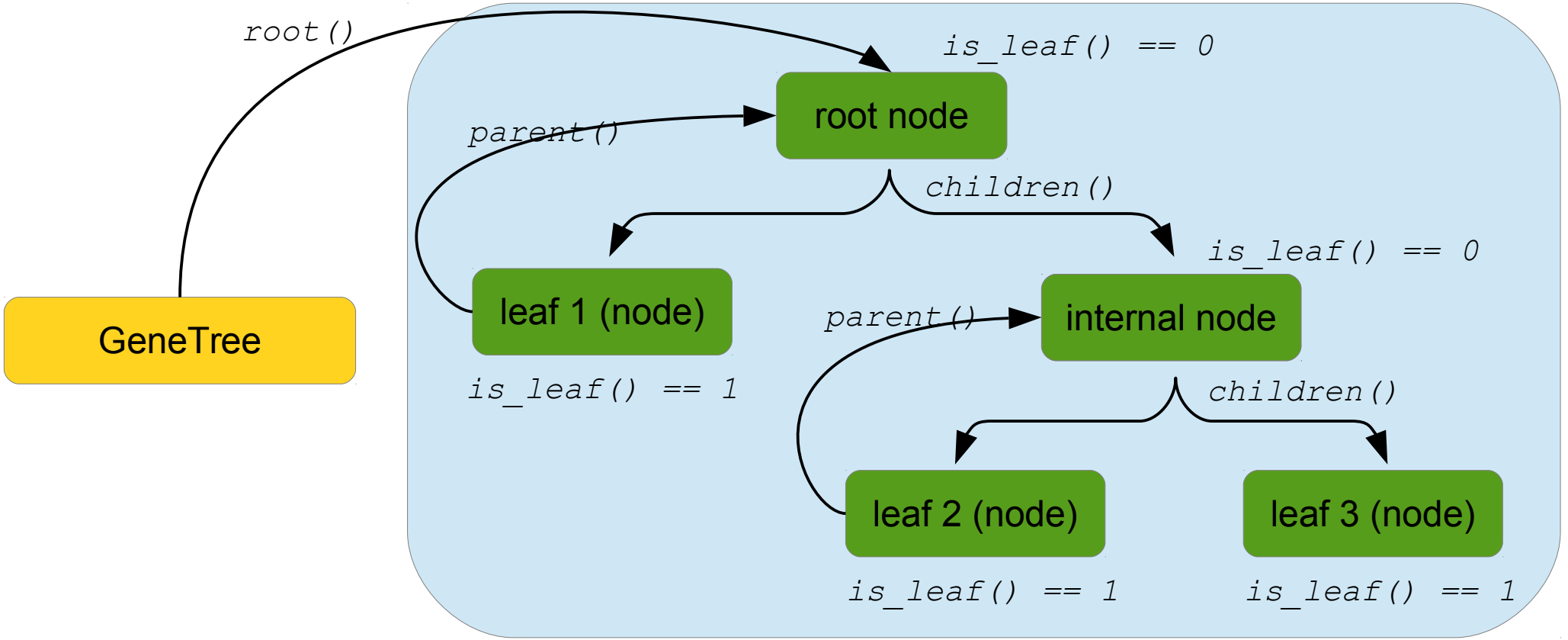
The actual tree structure is a hierarchy of *GeneTreeNode* objects

GeneTree



# *GeneTreeNode* object

The actual tree structure is a hierarchy of *GeneTreeNode* objects




## Extra information

```
$node->node_type()                  $node->duplication_confidence_score()
$node->taxonomy_level()             $node->bootstrap()
```

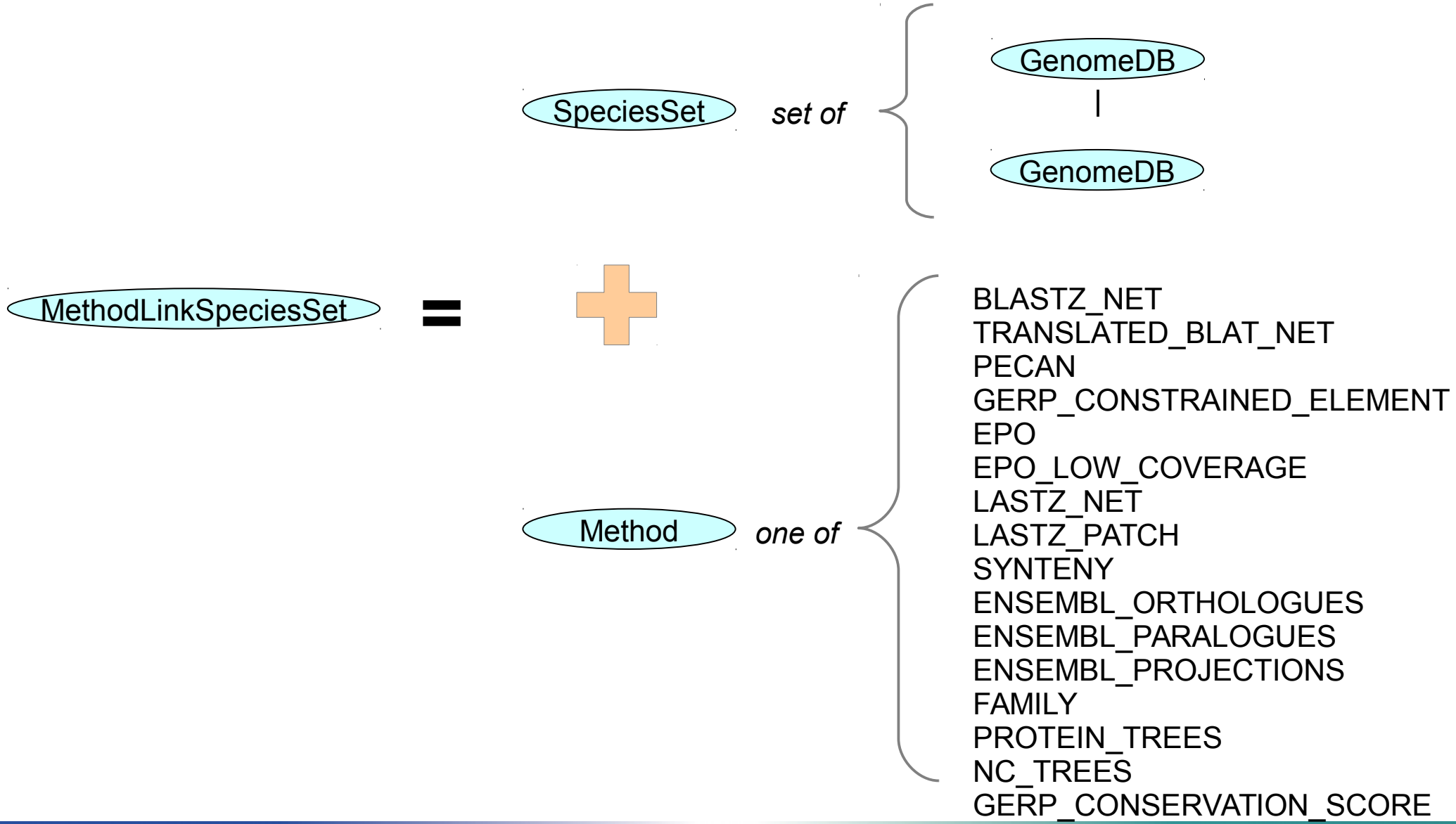
# Outline of the course

- Introduction about Compara
  - Resources
  - API
- Inputs
  - Species, Chromosomes, Genes
- Outputs
  - Gene analyses
  - Genome analyses

# MethodLinkSpeciesSet object

- The Compara database contains **lots** of cross-species comparisons
- There are multiple comparisons of the same type (pairwise alignments, homologies, etc)
- We need a way of defining which analysis is performed on which genomes
- Many adaptor methods require a MethodLinkSpeciesSet

# MethodLinkSpeciesSet object



# MethodLinkSpeciesSet

- Links a method (an analysis) to a set of species

| Attributes | Methods |
|---|---|
| Name | `$mlss->name()` |
| Type of analysis | `$mlss->method()->type()` |
| List of GenomeDBs | `$mlss->species_set()` |
| **Adaptor methods** | |
| `$mlss_adaptor->fetch_by_method_link_type_registry_aliases()` | |
| `$mlss_adaptor->fetch_by_method_link_type_species_set_name()` | |

# Example Code – MethodLinkSpeciesSet

```
my $mlss_adaptor = $reg->get_adaptor("Multi", "compara", "MethodLinkSpeciesSet");
my $gorilla_mlss_list = $mlss_adaptor->fetch_all_by_GenomeDB( $gorilla_genome_db );

my $c = 0;
foreach my $mlss ( @{ $gorilla_mlss_list } ) {
   print join( "\t", $mlss->dbID(), $mlss->method->type() ), "\n";
   $c++;
   last if $c >= 10;
}
```

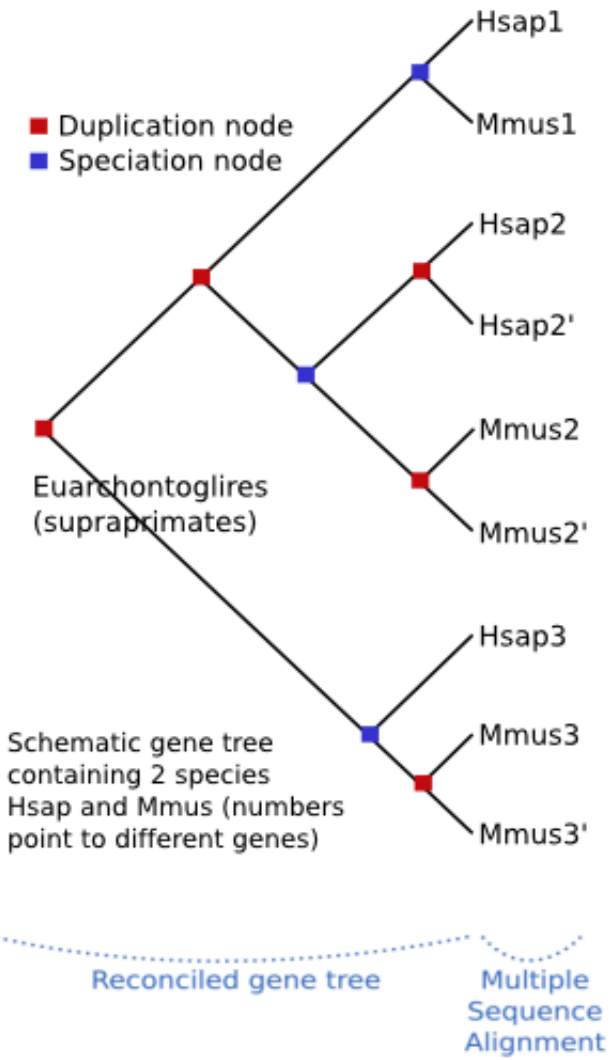
# Exercises – MethodLinkSpeciesSet

- Print the total number of MethodLinkSpeciesSet entries stored in the database

- Print a unique list of method_link_types and a count of their number in the database.

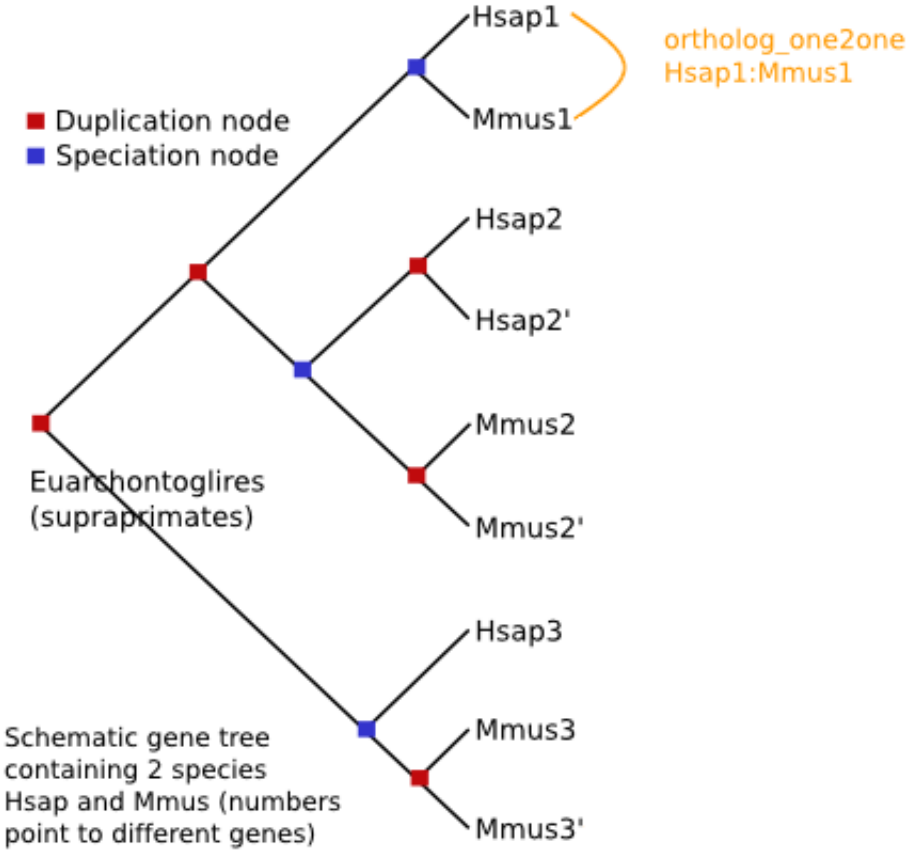
- Print the list of the species for the 17 eutherian mammals EPO alignments

# Homology inference

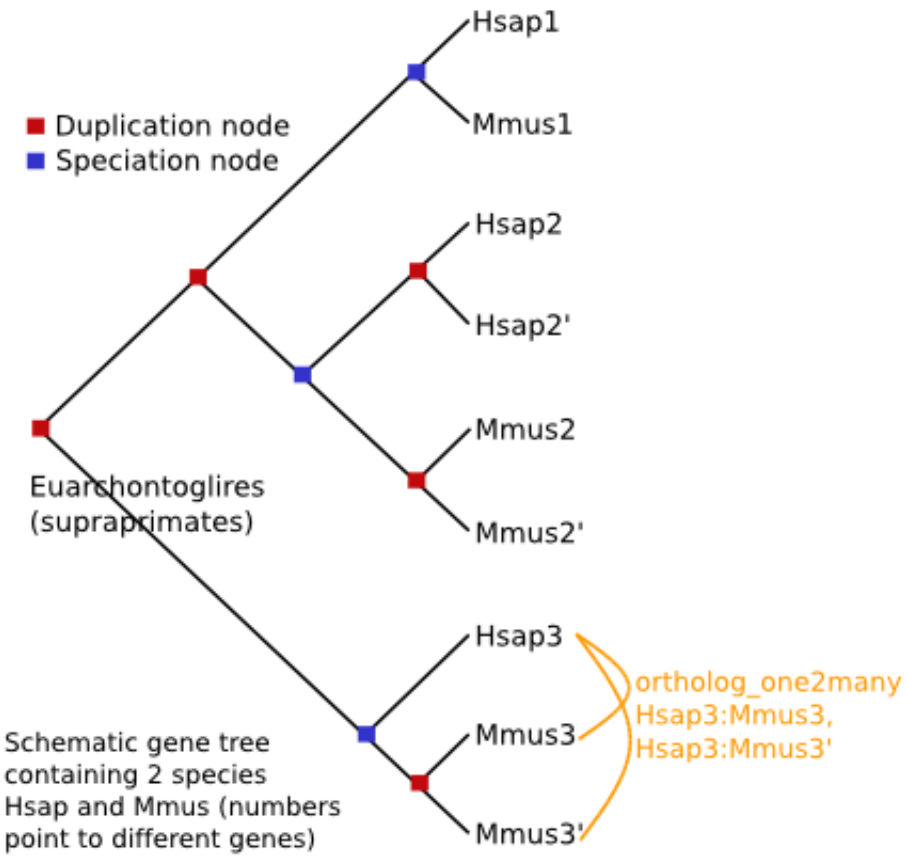



Consists in tagging the pairs of genes of all the trees with a relation type, depending on the tree topology.

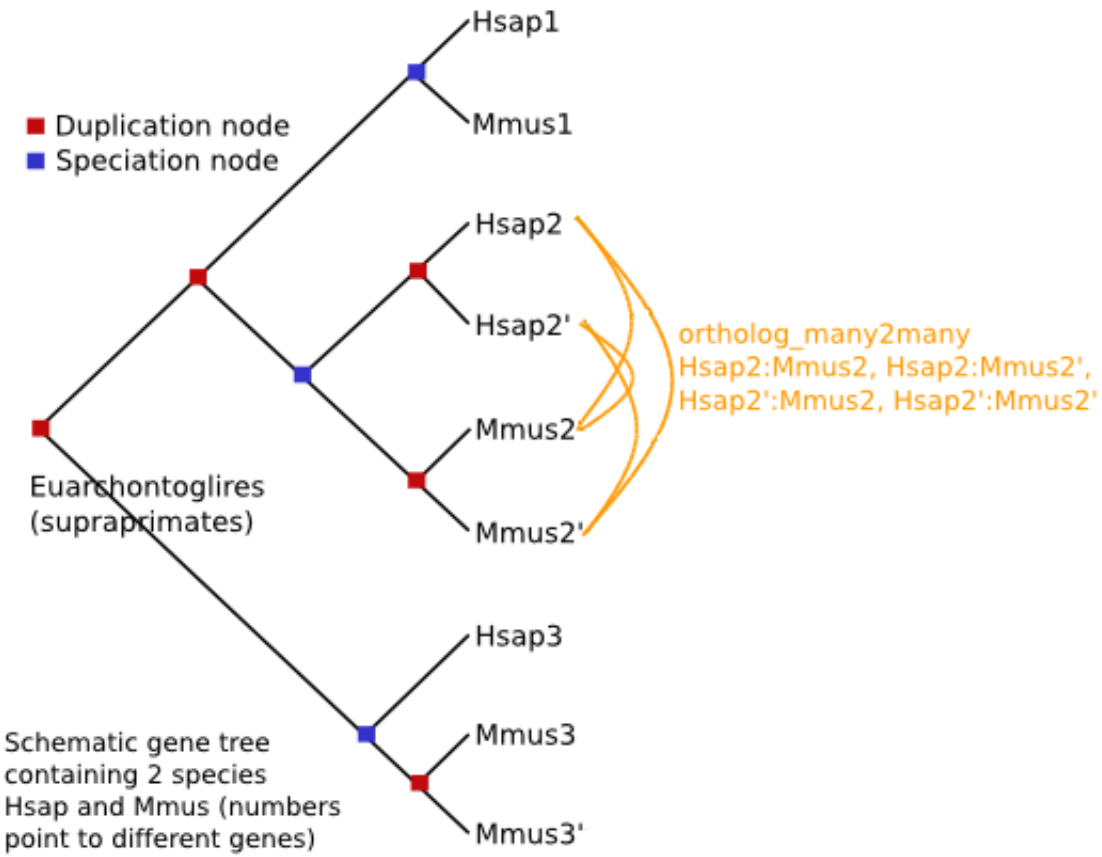
# Homology inference



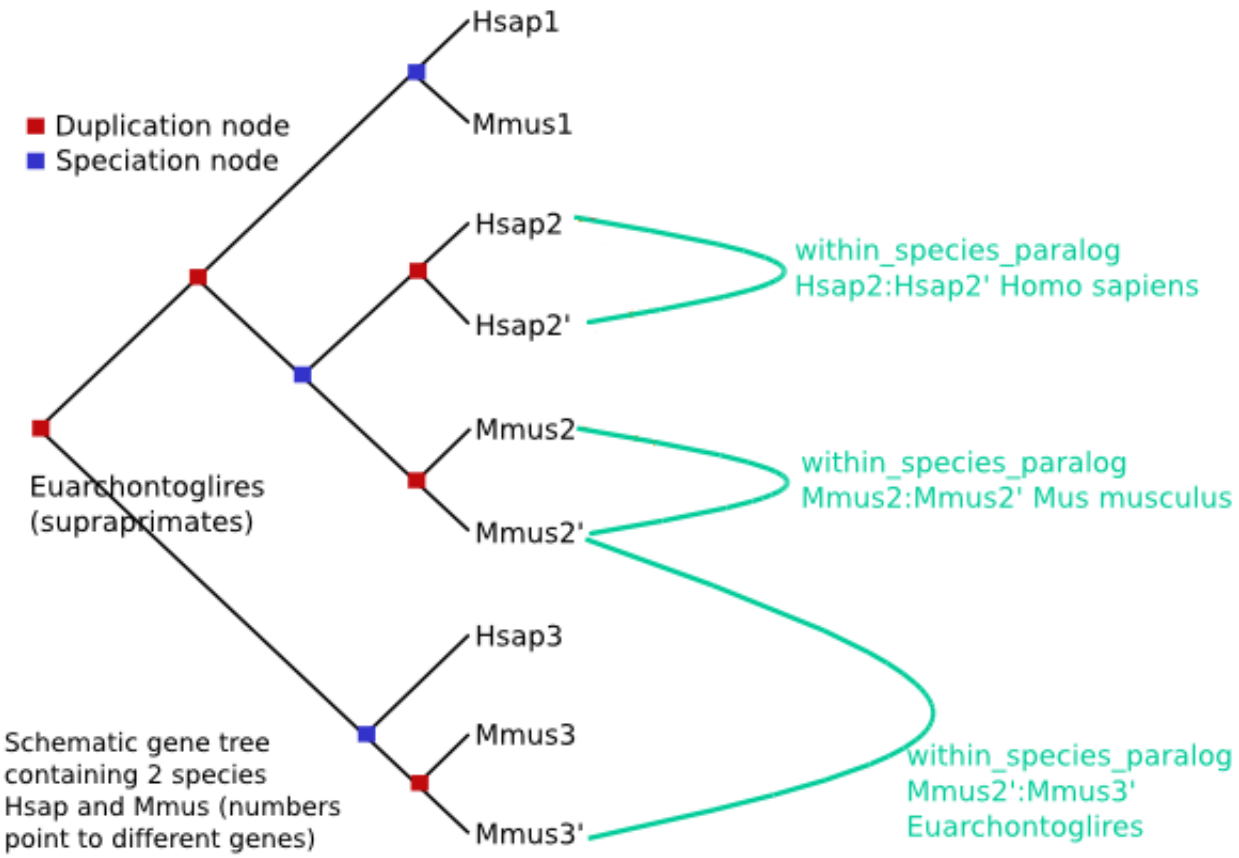
# Homology inference



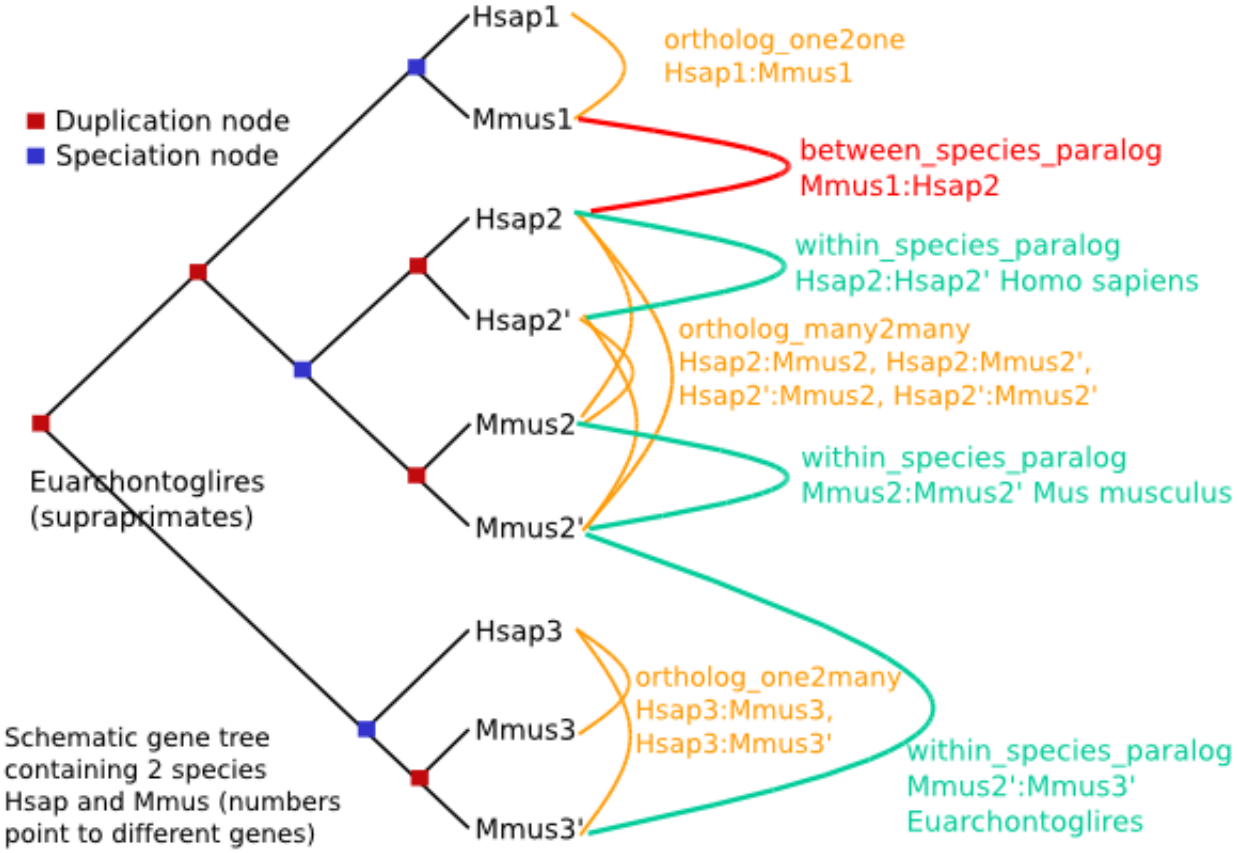
# Homology inference



# Homology inference



# Homology inference



# *Homology* object

- An Homology object links two genes together
- One-to-many relationships are split:
  - "H ortholog to M1" and "H ortholog to M2" are different objects

| Attributes | Methods |
|---|---|
| Alignment | `$homology->get_SimpleAlign()` |
| Natural selection | `$homology->dn() / $homology->ds()` |
| Gene content | `$homology->get_all_GeneMembers()` |
| Homology characteristics | `$homology->description()`<br>`$homology->taxonomy_level()` |
| **Adaptor methods** | |
| `$homology_adaptor->fetch_all_by_Member(…)` | |
| `$homology_adaptor->fetch_all_by_MethodLinkSpeciesSet(…)` | |

# Code Example - Homology

```
my $mlss_adaptor = $reg->get_adaptor("Multi", "compara", "MethodLinkSpeciesSet");
my $homology_adaptor = $reg->get_adaptor("Multi", "compara", "Homology");

my $mlss = $mlss_adaptor->fetch_by_method_link_type_registry_aliases(
      'ENSEMBL_ORTHOLOGUES', ["human", "gorilla"]
);
my $orthologs = $homology_adaptor->fetch_all_by_MethodLinkSpeciesSet($mlss);

my $c = 0;
foreach my $orth ( @{ $orthologs } ){
   print $orth->toString(), "\n";
   $c++;
   last if $c >= 10;
}
```

# Exercises – Homologies (and MethodLinkSpeciesSet)

- Get all the homologues for the human gene ENSG00000229314
- Count the number of “one2one” homologues between human and mouse
- Find the human orthologues of ENSMUSG00000004843 and ENSMUSG00000025746. For each homology, display the alignment and the dn value. Comment on the divergence

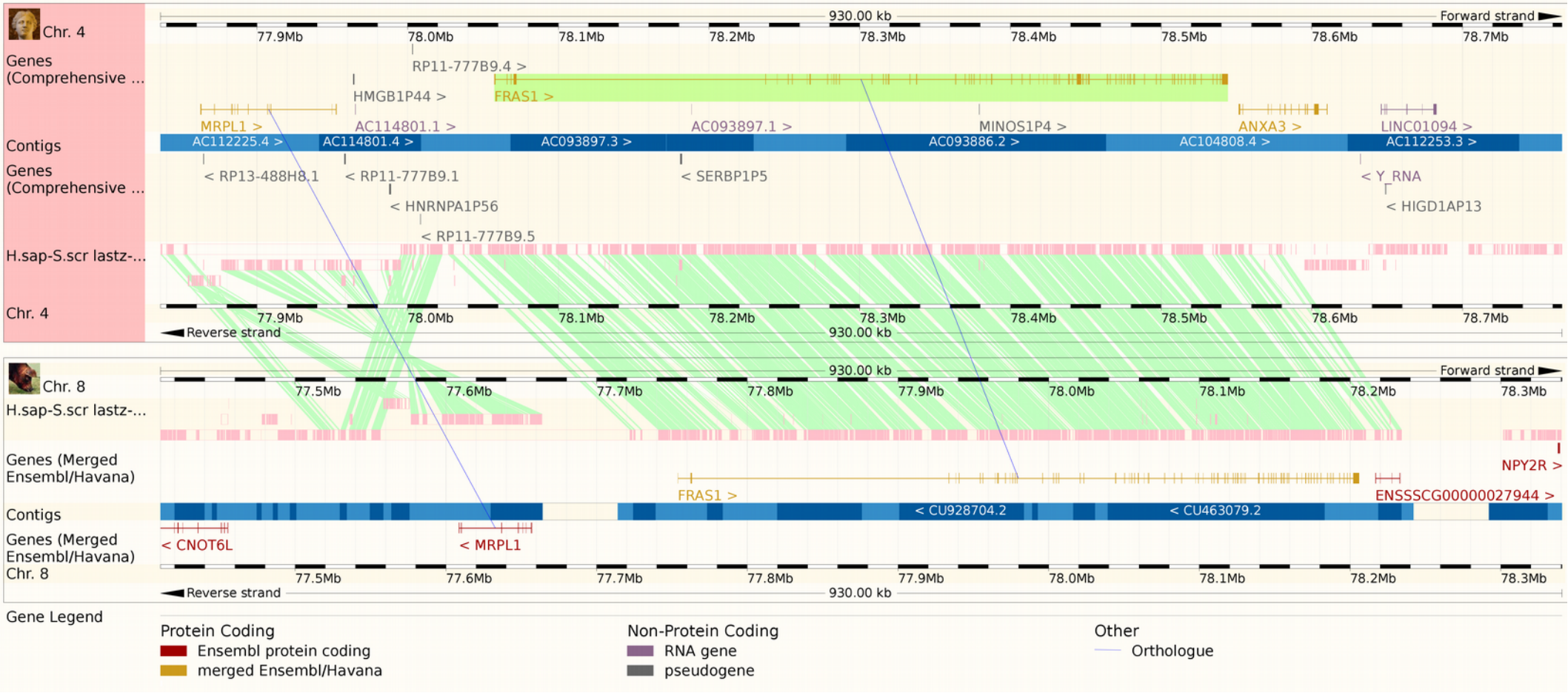
# Outline of the course

- Introduction about Compara
  - Resources
  - API
- Inputs
  - Species, Chromosomes, Genes
- Outputs
  - Gene analyses
  - Genome analyses

# Whole-genome alignments

**Alignments at the DNA level**

Example: Human vs pig



# How are alignments stored ?

A small example :

```
gorilla_gorilla/MT/935-953    gacat-ttaactaaaac-ccc
macaca_mulatta/MT/1469-1488   aacatcttaactaaacg-ccc
pan_troglodytes/MT/934-953    gatac-ttaacttaaaccccc
pongo_pygmaeus/MT/940-958     actac-ctaactaaaac-ccc
homo_sapiens/MT/1516-1534     gacat-ttaactaaaac-ccc
                                 *    ***** **    ***
```

```
GACATTTAACTAAAACCCC       5MD11MD3M
AACATCTTAACTAAACGCCC      17MD3M
GATACTTAACTTAAACCCCC      5MD15M
ACTACCTAACTAAAACCCC       5MD11MD3M
GACATTTAACTAAAACCCC       5MD11MD3M
```

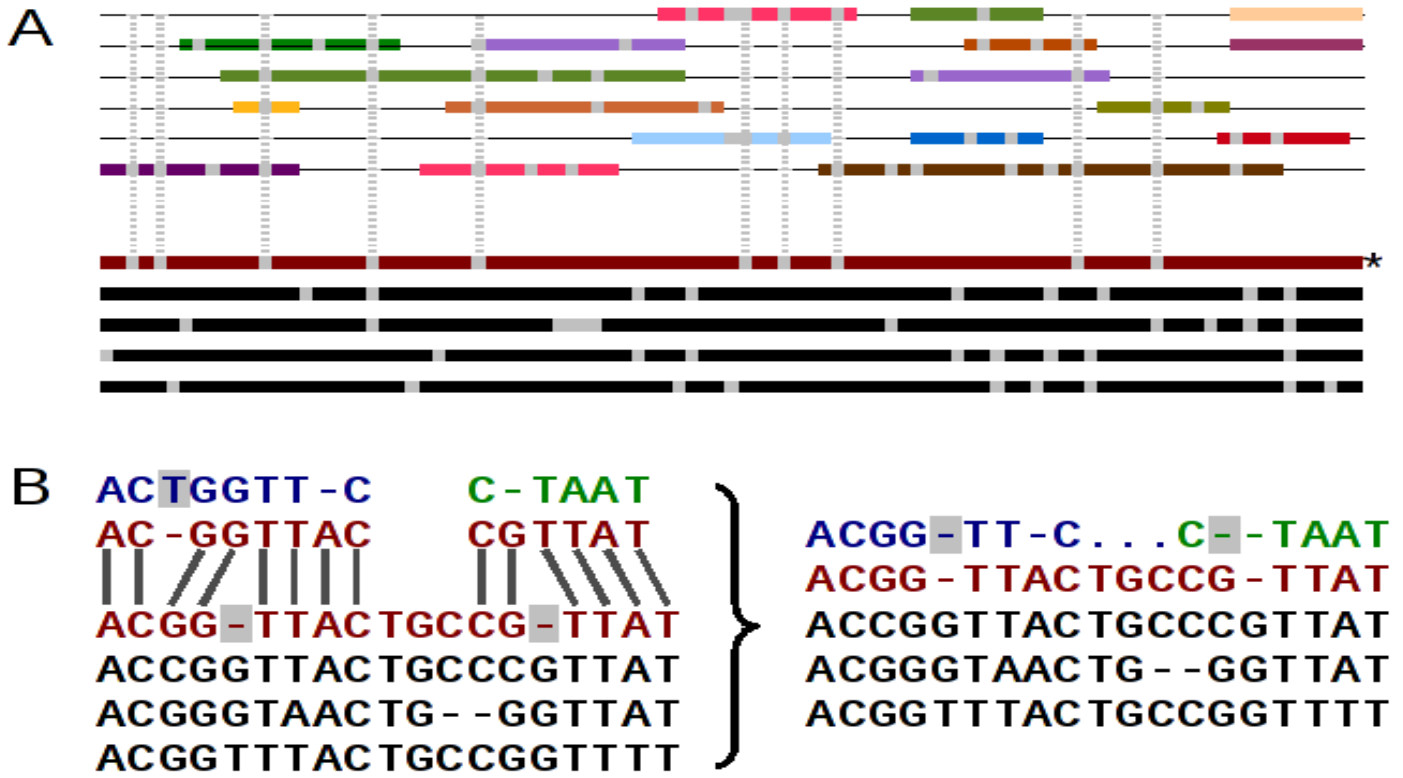
5 *genomic_align* entries
1 *genomic_align_block* entry

Sequences from core

CIGAR line in Compara

# Adding low-coverage genomes

- Low coverage genomes cannot be fully assembled
- Resulting assembly is too scattered to be used with Enredo
- Run EPO on high-coverage genomes only
- Map 2X genomes using pairwise alignments on a reference species



# Objects on the genomic side

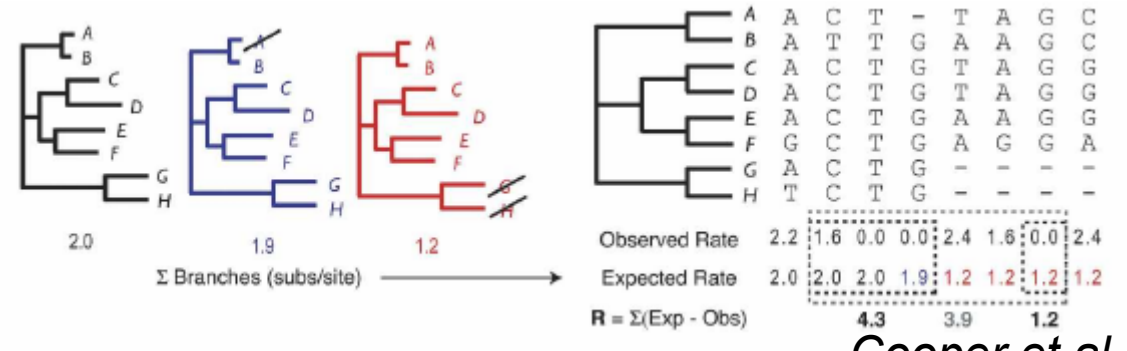
- A GenomicAlignBlock represents an alignment between two or more regions of genomic DNA. Within these blocks every region of genomic DNA is represented by a GenomicAlign object.

- A ConstrainedElement represent regions in the multiple alignment which appear to be under functional constraint.

- Synteny blocks are derived from Lastz-net alignments
  - group syntenic alignments closer than 200Kb
  - link syntenic groups closer than 3Mb
  - minimum length of the syntenic block: 100 kb

# GERP Constrained Elements

- Stretches of the alignment with a high conservation




*Cooper et al. Genome Research, 2005*

- Constrained elements and coding exons
  - 74% of coding exons are associated with constr. elem.
  - 22% of constr. elem. are associated with coding exons

# GenomicAlignBlock

- An alignment-block (across 2 or more sequences)
- The adaptor returns the blocks that **overlap** the query region
  → Call restrict_between_reference_positions()

| Attributes | Methods |
|---|---|
| BioPerl alignment object | *$gab->get_SimpleAlign()* |
| Aligned sequences | *$gab->get_all_GenomicAligns()* |
| (Restrict the block) | *$gab->restrict_between_reference_positions()* |
| **Adaptor methods** | |
| `$gab_adaptor->fetch_all_by_MethodLinkSpeciesSet_Slice()` | |

- GenomicAlign has a similar interface to Members, e.g. $ga→dnafrag, $ga→dnafrag_start, etc

# Code Example - GenomicAlignBlock

```
my $mlss_adaptor = $reg->get_adaptor("Multi", "compara", "MethodLinkSpeciesSet");
my $genomic_align_block_adaptor = $reg->get_adaptor("Multi", "compara",
"GenomicAlignBlock");

my $epo_mlss_list = $mlss_adaptor->fetch_all_by_method_link_type("EPO");

foreach my $epo_mlss ( @{ $epo_mlss_list } ){
   my $genomic_align_blocks = $genomic_align_block_adaptor->
                                    fetch_all_by_MethodLinkSpeciesSet( $epo_mlss );
   print $epo_mlss->name() . " has " . scalar( @{ $genomic_align_blocks } ) .
            " alignment blocks\n";
}
```

# Exercises – Genomic Alignments

- Print the LASTZ-NET alignments for pig chromosome 15 with cow (using pig coordinates 105734307 and 105739335).

- Change the above example so that it prints the 17-way eutherian mammal (EPO) multiple alignments.

- Print the constrained element alignments from the above pig locus (use the constrained elements generated from the EPO_LOW_COVERAGE mammals alignments)

# Exercises – Synteny

- Print the pig-cow synteny map using pig chromosome 15 as a reference

  *web reference:*
  http://www.ensembl.org/Sus_scrofa/Location/Synteny?r=15&otherspecies=Bos_taurus

# Acknowledgements





Leo Mateus Matthieu Carla Aj

*D710–D716 Nucleic Acids Research, 2016, Vol. 44, Database issue* *Published online 19 December 2015*
*doi: 10.1093/nar/gkv1157*

**Ensembl 2016**

**Andrew Yates[1], Wasiu Akanni[1], M. Ridwan Amode[1], Daniel Barrell[1,2], Konstantinos Billis[1], Denise Carvalho-Silva[1], Carla Cummins[1], Peter Clapham[2], Stephen Fitzgerald[1], Laurent Gil[1], Carlos García Girón[1], Leo Gordon[1], Thibaut Hourlier[1], Sarah E. Hunt[1], Sophie H. Janacek[1], Nathan Johnson[1], Thomas Juettemann[1], Stephen Keenan[1], Ilias Lavidas[1], Fergal J. Martin[1], Thomas Maurel[1], William McLaren[1], Daniel N. Murphy[1], Rishi Nag[1], Michael Nuhn[1], Anne Parker[1], Mateus Patricio[1], Miguel Pignatelli[1], Matthew Rahtz[2], Harpreet Singh Riat[1], Daniel Sheppard[1], Kieron Taylor[1], Anja Thormann[1], Alessandro Vullo[1], Steven P. Wilder[1], Amonida Zadissa[1], Ewan Birney[1], Jennifer Harrow[2], Matthieu Muffato[1], Emily Perry[1], Magali Ruffier[1], Giulietta Spudich[1], Stephen J. Trevanion[1], Fiona Cunningham[1], Bronwen L. Aken[1], Daniel R. Zerbino[1] and Paul Flicek[1,2,*]**

[1]European Molecular Biology Laboratory, European Bioinformatics Institute, Wellcome Genome Campus, Hinxton, Cambridge CB10 1SD, UK and [2]Wellcome Trust Sanger Institute, Wellcome Genome Campus, Hinxton, Cambridge, CB10 1SA, UK

## Funding

wellcome trust

National Human Genome Research Institute

BLUEPRINT epigenome

EMBL

BBSRC bioscience for the future

EURATRANS

Co-funded by the European Union