EGAD v0.01 Documentation

Evolutionary Genomic Architecture Database

Zachery Mielko

Table of Contents

Introduction and logging in:

The first live release will be hosted on an OIT virtual machine, vcm-10665.vm.duke.edu. Documentation for the release is available on https://github.com/zmielko/DB_dev. In order to access the database locally, you will want to install anaconda. Instructions for anaconda installation can be found on the github page, but essentially it is a package manager that is "batteries included" and makes installing the extensions needed for SQL easier.

You should have received a username and password, which you will need in order to log into the database. The examples I have online are for a local version, so the login information is general. You will only need to include the login information once, at the top of the jupyter notebook.

In addition to the notebooks, you can login through a variety of other application including: psql, RStudio, and Excel. For RStudio and Excel, the driver setup for the connections tab is not very intuitive (in my opinion) so we will start with the jupyter notebooks. There are other applications you can use as well, including pgAdmin, DBSchema, and Postico (Mac).

Schema Overview

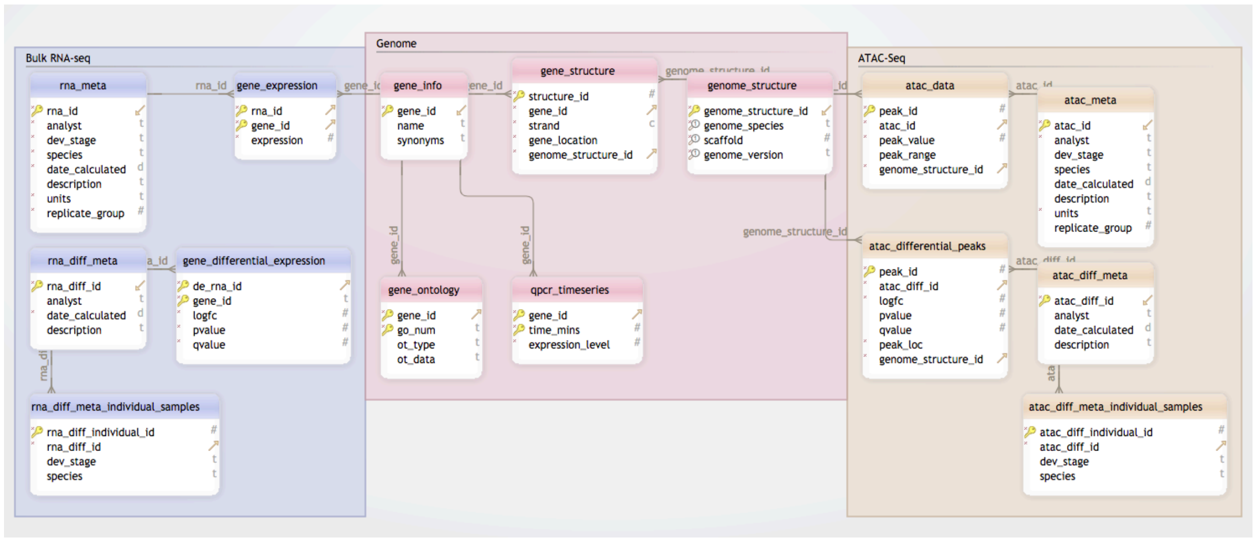


Figure 1: Schema for EGAD v 0.01. Drawn with DBSchema

Currently, the schema is set-up to handle RNA and ATAC-seq data. There isn’t any differential RNA-seq data uploaded yet, so I don’t have that portion connected yet. In general, the schema is designed so that every dataset you need can start with a many-to-many connection to the meta data (Genome group). The meta data is designed to link information about genes, with where the genes are located in genomes. Here is an example:

The table gene_info has a list of genes, each of which can be expressed so they are related to gene expression data. Each gene can be in multiple places in a genome (ie a duplicate gene) so it is a one-to-many relation. The location of a gene is relative to a genome assembly, so it has a relation to a table that outlines the structure of the genome (scaffolds, genome version, genome species). That scaffold in a particular genome is where you could have open chromatin regions, so it is related to the atac-seq data.

Some of the tables for meta data are used directly from Echinobase. The benefit is that instead of having separate databases for each species and each assembly, you can now ask questions across species and use positional information from chromosomes (ie. ATAC-seq). As the schema develops, it will become more different but it should be (generally) compatible to updates when Echinobase is updated.

Initial datasets

The database comes with the following data pre-loaded:

1. RNA-seq data from *Israel et al.* for *L.variegatus, H.tuberculata,H.erythrogamma*
    a. Time course per developmental stage
2. RNA-seq data from *Materna et al.,* for S.purpuratus
    a. Time course every hour
3. QPCR data from *Materna et al.,* for S.purpuratus
4. Genomic data for lva 2.2 and spu 3.1 assemblies
5. ATAC-seq data from Phillip Davidson for *L.variegatus, H.tuberculata,H.erythrogamma*
6. ATAC-seq data from *Shashikant et al.* for *S.purpuratus*

Using tidy (long) format

One of the properties of databases is that the typically use a tidy format. In short, having data in this format makes the tables scalable. Say for example you have a time course experiment with time points like this:

spu_id	sp_name	hr_0	hr_1	hr_2	hr_3
SPU_028148	Sp-Ac/Sc	15	13	13	13
SPU_025302	Sp-Alx1	4	4	4	5
SPU_000129	Sp-Arnt	73	71	71	71
SPU_017348	Sp-Atbf1/z30	249	241	252	260
SPU_026905	Sp-Atf2	352	341	343	331
SPU_027235	Sp-Blimp1a	36	35	36	35
SPU_027235	Sp-Blimp1b	101	94	101	106

Each time point is in its own column, making it wide format. The benefit to this is that it is easy to make comparisons by eye. But say you wanted to add a different experiment to the table and it looks like this:

gene	LVeggA	LVeggB	LVeggC	LV4cellA	LV4cellB
SPU_000003	238	144	424	484	591
SPU_000007	5	0	0	0	0
SPU_000008	0	1	4	2	3
SPU_000009	198.1	306.73	366	853.44	722.25
SPU_000011	66	1542	2373.28	1076.72	1235.44
SPU_000012	0	17.81	1	0	3
SPU_000014	21	51	47	132	81
SPU_000021	6	15	17	38	38.99
SPU_000022	30	39	124	74	67

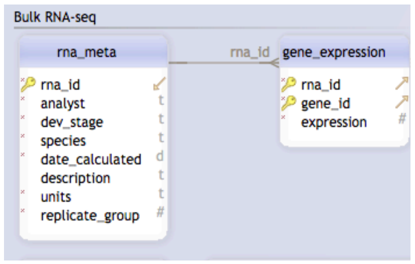
If we were to add them together, we would have to make additional columns for every new way we measure a time point. In addition, if we measure different genes, then we need to add rows where for some datasets the value will be null. Having a tidy dataset means that we will only need to add rows to a table when adding new information. It also standardizes queries, so for example instead of having to know all the specific time points and the different normalized units they might have for each dataset, you can just ask for the values given a specific time point and unit in 3 columns (time point, value, unit). For more information check out these documents:

- https://kiwidamien.github.io/long-vs-wide-data.html
- https://vita.had.co.nz/papers/tidy-data.pdf

Transforming data to fit in a database

One of the differences between flat files and databases is that the information in a typical file can be spread out among multiple tables. Here is an example for RNA-seq data. In the database there are 2 tables, one for data about the experiment and other for the expression data. The Israel et al. data comes in a wide format and we want to place it into the database.

	LVeggA	LVeggB	LVeggC	LV4cellA	LV4cellB
SPU_000003	6.127094667	5.37362271	6.604116301	5.901095683	6.19527004
SPU_000007	1.291220947	0	0	0	0
SPU_000008	0	0.35723346	0.93231105	0.313603164	0.450890347
SPU_000009	5.865797451	6.447197478	6.394243862	6.708152919	6.480534029
SPU_000011	4.329484684	8.762386421	9.076460405	7.041680864	7.248247631
SPU_000012	0	2.598700682	0.295232395	0	0.450890347
SPU_000014	2.82351982	3.938222153	3.545089065	4.089609388	3.44703111
SPU_000021	1.452501098	2.38253086	2.28107642	2.488861395	2.528442855
SPU_000022	3.275618749	3.579870211	4.865832005	3.319601663	3.200650732
SPU_000026	3.7178406	3.360633319	3.653199218	4.254683126	3.913020518
SPU_000034	8.677516747	7.665445136	8.676995678	8.1632136	8.117020708
SPU_000035	4.004721508	5.893390455	5.122478541	1.432774716	3.60036506



We make 2 tables, one of the meta data and the other of the expression.

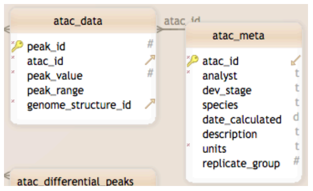
rna_id	analyst	dev_stage	species	date_calculated	description	units	replicate_group
0	Israel et al.	egg	L.variegatus	3/4/16	Log2-transformed, normalized data	counts	1
1	Israel et al.	egg	L.variegatus	3/4/16	Log2-transformed, normalized data	counts	2
2	Israel et al.	egg	L.variegatus	3/4/16	Log2-transformed, normalized data	counts	3
3	Israel et al.	4cell	L.variegatus	3/4/16	Log2-transformed, normalized data	counts	1
4	Israel et al.	4cell	L.variegatus	3/4/16	Log2-transformed, normalized data	counts	2
5	Israel et al.	4cell	L.variegatus	3/4/16	Log2-transformed, normalized data	counts	3
6	Israel et al.	16cell	L.variegatus	3/4/16	Log2-transformed, normalized data	counts	1
…	…	…	…	…	…	…	…

rna_id	gene_id	expression
42	SPU_000003	7.135225014
54	SPU_000003	6.924823574
44	SPU_000003	6.817785041
55	SPU_000003	6.7130452
43	SPU_000003	6.662212979
2	SPU_000003	6.604116301
56	SPU_000003	6.484295086
58	SPU_000003	6.230956042
4	SPU_000003	6.19527004
0	SPU_000003	6.127094667
…	…	…

Our meta data table contains all of the information about the experiment and each rna_id value relates to the rna_id values in the expression table. The benefit to this is that it is scalable, we can add any type of expression data and no matter which genes were looked at or what units were used or time points taken, they can all be represented in the same tables. The gene_id (in blue/grey) relates to the gene_info table.

The same principles apply to ATAC-seq data.

chr	start_peak	finish_peak	He16a	He64a	He128a	He256a
Scaffold9430	1377	2336	0	0	0	0
Scaffold9430	2493	4213	0	1.726796602	0	0
Scaffold9430	4266	5469	861.9582506	818.5015895	1104.313448	1191.576867
Scaffold9430	1377	2336	0	0	0	0
Scaffold9430	2493	4213	0	1.726796602	0	0
Scaffold9430	4266	5469	861.9582506	818.5015895	1104.313448	1191.576867
Scaffold20135	2232	2606	0	3.453593205	6.310362562	0
Scaffold517	2818	3319	0	0	0	0
Scaffold517	13040	13471	4.939588829	12.08757622	6.310362562	11.95562074
Scaffold517	15289	15812	4.939588829	12.08757622	3.155181281	3.985206912



atac_id	analyst	dev_stage	species	date_calculated	description	units	replicate_group
0	Phillip Davidson	16cell	H.erythrogramma	5/18/18	normalized read counts	CPM	1
1	Phillip Davidson	64cell	H.erythrogramma	5/18/18	normalized read counts	CPM	1
2	Phillip Davidson	128cell	H.erythrogramma	5/18/18	normalized read counts	CPM	1
3	Phillip Davidson	256cell	H.erythrogramma	5/18/18	normalized read counts	CPM	1
4	Phillip Davidson	512cell	H.erythrogramma	5/18/18	normalized read counts	CPM	1
5	Phillip Davidson	32cell	H.erythrogramma	5/18/18	normalized read counts	CPM	2
6	Phillip Davidson	64cell	H.erythrogramma	5/18/18	normalized read counts	CPM	2
7	Phillip Davidson	128cell	H.erythrogramma	5/18/18	normalized read counts	CPM	2
8	Phillip Davidson	256cell	H.erythrogramma	5/18/18	normalized read counts	CPM	2
9	Phillip Davidson	512cell	H.erythrogramma	5/18/18	normalized read counts	CPM	2

atac_id	**peak_value**	**peak_range**	**genome_structure_id**
0	4.939588829	[12919,13305]	32008
0	2.469794414	[32038,32290]	32008
0	2.469794414	[58240,58720]	32008
0	9.879177657	[67542,68733]	32008
0	7.409383243	[77372,77734]	32008
0	24.69794414	[77929,78919]	32008
0	2.469794414	[87000,87204]	32008
0	2.469794414	[90919,91120]	32008
0	2.469794414	[110737,111903]	32008
0	9.879177657	[139776,140676]	32008
0	2.469794414	[141908,142425]	32008
0	24.69794414	[151551,152888]	32008
0	9.879177657	[160072,160752]	32008
0	14.81876649	[160759,161251]	32008
1	1.726796602	[7453,7654]	32008
1	3.453593205	[12556,12773]	32008
1	1.726796602	[12919,13305]	32008
1	3.453593205	[16468,16774]	32008
1	1.726796602	[32038,32290]	32008

Here, the information for the peaks relate to genome_structure as an integer, rather than the more informative gene_id for rna-seq data. Having an integer key instead of multiple columns (scaffold, genome version, genome species) means that the joins between tables are faster, since the database only needs to match numbers in one column. The downside is that one needs to transform the data into a format that references the data in the genome_structure table.

genome_structure_id	genome_species	scaffold	genome_version
0	S.purpuratus	1	3.1
1	S.purpuratus	2	3.1
2	S.purpuratus	3	3.1
3	S.purpuratus	4	3.1
4	S.purpuratus	5	3.1
5	S.purpuratus	6	3.1
6	S.purpuratus	7	3.1

Here, the genome_structure_id is a unique value that corresponds to a scaffold in the genome. The benefit of this is that we can add additional columns without interfering with the relation between ATAC-seq data and the rest of the meta data. It also ensures that the data uploaded is aligned to the correct genome, so if the data contains scaffold # 1,000,000,000 there is no genome_structure_id to match it to.