sramongo Documentation

Release 1.0.0

Justin Fear

Jul 02, 2020

Contents:

sramongo is a python library and command line tool `sra2mongo` that queries NCBI's sequence read archive(SRA) and dumps all relevant information into a mongo database. Mongo is a popular document based database, which stores information as `key:value` pairs; unlike a typical relational database (e.g., SQL) which stores data as rows in multiple related tables. One major advantage of a document based database to a relation database is that there is no need a defined schema; attributes can be arbitrarily added or removed and don't need to be the same across records. What this means is that you can run `sra2mongo` to query and populate your database, then freely modify or add new fields/documents as part of a processing pipeline.

Roughly speaking sramongo is made up of 3 parts. The first is a parser for SRA XML, the second is an object relational mapper to allow easy interface with mongo, and the third is a command line utility which uses Biopython and Entrez utilities to query the SRA and download the resulting XML.

Warning: Please use this tool responsibly, querying the SRA and dumping large amounts of data can be taxing on their system and may result in blacklisting of your IP address.

CHAPTER 1

Quick Start

1.1 Installation

1.1.1 MongoDB

sramongo requires a working version of MongoDB community server >=3.4.1. Please download the appropriate version here. There are also a number of online hosts that will run a mongoDB server for free (small databases) or relatively cheaply.

1.1.2 sramongo

sramongo can be installed using pip:

```
pip install git+https://github.com/jfear/sramongo
```

1.2 sra2mongo Usage

`sra2mongo` is the command line tool provided by sramongo. To get a full set of options run `sra2mongo -h`. A simple query would look like:

```
sra2mongo \
    --email john.smith@example.com \
    --query '"Drosophila melanogaster"[orgn]'
```

The \ allows for breaking the command on multiple lines. This command will query the SRA for `"Drosophila melanogaster"[orgn]`, download the XML for all of the runs, and parse the XML into a database named 'sramongo'.

A (see *sramongo mappings*) for a list of database fields.

Note: The query string is passed directly to SRA, so any query options such as [orgn], [pid], or [author] will work. Also queries can include boolean operators (i.e., AND, OR).

1.3 Querying the Database

Todo: Add section about querying the database using mongoengine and pymongo. Until then follow mongoengines docs

CHAPTER 2

sramongo mappings

The database created by `sra2mongo` consists of a single document that is organized hierarchically:

This can be thought of as a giant JSON or python dict which various levels can be accessed by indexing through (e.g., `ncbi.sra.run.run_id`). MongoDB has a very nice querying system which allows easy searching through the document.

Note: One downside of storing all of this information as a single document is that mongoDB has a max document size of 16 MB. This is more than enough for storing metadata and text, but if you start adding data tables you may hit this limit.

2.1 ncbi

This is the top level document. Information from each database is stored under its name. As I add data normalization steps I intend to aggregate data from the different databases and store them up in this top level document.

2.1.1 sra

This stores all from the Sra. There are also a couple of summary fields that are stored at this level. Each section of the SRA record are represented as subdocuments.

`class sramongo.models.SraDocument(*args, **values)`

organization

`class sramongo.models.Organization(*args, **kwargs)`

Organization embedded document.

An organization contains information about the group that submitted to sra. For example, all data submitted to GEO are submitted to SRA using the GEO credentials.

`organization_type`

Weather this organization is a center or individual or some other kind of group.

Type str

`abbreviation`

A short name for the organization.

Type str

`name`

Name of the organization.

Type str

`emai`

Contact email address.

Type str

`first_name`

First name of the person who submitted the data.

Type str

`last_name`

First name of the person who submitted the data.

Type str

submission

study

`class sramongo.models.Study(*args, **kwargs)`

The contents of a SRA study.

A study consists of a set of experiments designed with an overall goal in mind. For example, this could include a control experiment and a treatment experiment with the goal being to identify expression differences resulting from the treatment. The SRA study is the top level of the submission hierarchy.

`accn`
The primary identifier for a study. Identifiers begin with SRP/ERP/DRP depending on which database they originate from.

Type mongoengine.StringField

`bioproject`
The associated BioProject identifier.

Type mongoengine.StringField

`geo`
The associated GEO identifier.

Type mongoengine.StringField

`geo`
The associated Pubmed identifiers.

Type mongoengine.StringField

`title`
The title of the study.

Type mongoengine.StringField

`abstract`
Abstract of the study.

Type mongoengine.StringField

`center_name`
Name of the submitting center.

Type mongoengine.StringField

`center_project_name`
Center specific identifier for the study.

Type mongoengine.StringField

`description`
Additional text describing the study.

Type mongoengine.StringField

run

class `sramongo.models.`**`Run`**`(`**args*, ***kwargs*`)`
Run Document.

A Run describes a dataset generated from an Experiment. For example if a Experiment is sequenced on multiple lanes of a Illumina flowcell then data from each lane are considered a Run.

`srr`
The primary identifier for a run. Identifiers begin with SRR/ERR/DRR depending on which database they originate from.

Type mongoengine.StringField

nspots
The total number of spots on a Illumina flowcell.

Type mongoengine.IntField

nbases
The number of bases.

Type mongoengine.IntField

nreads
The number of reads.

Type mongoengine.IntField

read_count_r1
Some Runs have additional information on reads. This is the number of reads from single ended or the first read pair in pair ended data.

Type mongoengine.FloatField

read_len_r1
This is the average length of reads from single ended or the first read pair in pair ended data.

Type mongoengine.FloatField

read_count_r2
This is the number of reads from the second read pair in pair ended data.

Type mongoengine.FloatField

read_len_r2
This is the avearge length of reads from the second read pair in pair ended data.

Type mongoengine.FloatField

release_date
Release date of the Run. This information is from the runinfo table and not the XML.

Type mongoengine.DateTimeField

load_date
Date the Run was uploaded. This information is from the runinfo table and not the XML.

Type mongoengine.DateTimeField

size_MB
Size of the Run file. This information is from the runinfo table and not the XML.

Type mongoengine.IntField

sample

class sramongo.models.**Sample**(**args*, ***kwargs*)
The contents of a SRA sample.

A sample is the biological unit. An individual sample or a pool of samples can be use in the SRA Experiment. This document contains information describing the sample ranging from species information to detailed descriptions of what and how material was collected.

accn
The primary identifier for a sample. Identifiers begin with SRS/ERS/DRS depending on which database they originate from.

Type mongoengine.StringField

biosample
The associated BioSample identifier.

Type mongoengine.StringField

geo
The associated GEO identifier.

Type mongoengine.StringField

title
The title of the sample.

Type mongoengine.StringField

taxon_id
The NCBI taxon id.

Type mongoengine.IntField

scientific_name
The scientific name.

Type mongoengine.StringField

common_name
The common name.

Type mongoengine.StringField

attributes
A set of key:value pairs describing the sample. For example tissue:ovary or sex:female.

Type mongoengine.DictField

2.1.2 biosample

Information from the BioSample database is stored here.

class sramongo.models.**BioSample**(**args*, ***kwargs*)
The contents of a BioSample.

BioSample is another database housed at NCBI which records sample metadata. This information should already be present in the Sra.sample information, but to be safe we can pull into the BioSample for additional metadata.

accn
The primary identifier for a BioSample. These are the accession number which begin with SAM.

Type mongoengine.StringField

id
The primary identifier for a BioSample. These are the id number.

Type mongoengine.IntField

title
A free text description of the sample.

Type mongoengine.StringField

description
A free text description of the sample.

Type mongoengine.StringField

publication_date
Date the sample was published.

Type mongoengine.StringField

last_update
Last time BioSample updated sample information.

Type mongoengine.StringField

submission_date
Date the sample was submitted

Type mongoengine.StringField

attributes
A list of dictionaries containing key:value pairs describing the experiment. The stored dictionaries are of the form {'name': value, 'value': value}. This was done to make querying easier.

Type mongoengine.ListField of mongoengine.DictField

2.1.3 bioproject

Information from the BioProject database is stored here.

class sramongo.models.**BioProject**(**args*, ***kwargs*)
The contents of a BioProject.

BioProject is another database housed at NCBI which records project metadata. This information should already be present in the SRA information, but to be safe we can pull into the BioProject for additional metadata.

accn
The primary identifier for a BioProject. These are the accession number which begin with PRJ.

Type mongoengine.StringField

id
The primary identifier for a BioProject. These are the id numbers.

Type mongoengine.IntField

name
A brief name of the project.

Type mongoengine.StringField

title
The title of the project.

Type mongoengine.StringField

description
A short description of the project.

Type mongoengine.StringField

last_date
Last date the BioProject was updated.

Type mongoengine.DateTimeField

submission_date
Date the BioProject was submitted.

Type mongoengine.DateTimeField

2.1.4 pubmed

Information from the Pubmed is stored here.

class `sramongo.models.`**`Pubmed`**`(`**args*, ***kwargs*`)`

The contents of a Pubmed document.

This document contains specific information about publications.

`accn`

The primary identifier for Pubmed. These are the accession number which begin with PMID.

Type mongoengine.StringField

`title`

Title of the paper.

Type mongoengine.StringField

`abstract`

Paper abstract.

Type mongoengine.StringField

`authors`

List of authors.

Type mongoengine.ListField

`citation`

Citation information for the paper.

Type mongoengine.StringField

`date_created`

Date the pubmed entry was created.

Type mongoengine.DateTimeField

`date_completed`

Date the pubmed entry was completed.

Type mongoengine.DateTimeField

`date_revised`

Date the pubmed entry was last updated.

Type mongoengine.DateTimeField

CHAPTER 3

SRA Constants

Using the XML schema from SRA I developed a list of expected constants. These constants are used to validate data coming from the SRA.

3.1 Study Types

- Cancer Genomics
- Epigenetics
- Exome Sequencing
- Metagenomics
- Other
- Pooled Clone Sequencing
- Population Genomics
- Synthetic Genomics
- Transcriptome Analysis

- Whole Genome Sequencing

3.2 Library Strategy

- AMPLICON
- Bisulfite-Seq
- ChIP-Seq
- CLONE
- CLONEEND
- CTS
- DNase-Hypersensitivity
- EST
- FAIRE-seq
- FINISHING
- FL-cDNA
- MBD-Seq
- MeDIP-Seq
- miRNA-Seq
- MNase-Seq
- MRE-Seq
- ncRNA-Seq
- OTHER
- POOLCLONE
- RIP-Seq
- RNA-Seq
- Synthetic-Long-Read
- SELEX
- Tn-Seq
- WCS
- WGA
- WGS
- WXS

3.3 Library Source

- GENOMIC
- METAGENOMIC

- METATRANSCRIPTOMIC
- NON GENOMIC
- OTHER
- SYNTHETIC
- TRANSCRIPTOMIC
- VIRAL RNA

3.4 Library Selection

- 5-methylcytidine antibody
- CAGE
- cDNA
- CF-H
- CF-M
- CF-S
- CF-T
- ChIP
- DNAse
- HMPR
- Hybrid Selection
- MBD2 protein methyl-CpG binding domain
- MDA
- MF
- MNase
- MSLL
- Oligo-dT
- other
- padlock probes capture method
- PCR
- PolyA
- RACE
- RANDOM
- RANDOM PCR
- Reduced Representation
- Restriction Digest
- RT-PCR

- size fractionation
- unspecified

3.5 Library Layout

- PAIRED
- SINGLE

3.6 Platforms

- ABI_SOLID
- CAPILLARY
- COMPLETE_GENOMICS
- HELICOS
- ILLUMINA
- ION_TORRENT
- LS454
- OXFORD_NANOPORE
- PACBIO_SMRT

3.7 Instrument Models

- 454 GS
- 454 GS 20
- 454 GS FLX
- 454 GS FLX+
- 454 GS FLX Titanium
- 454 GS Junior
- AB 310 Genetic Analyzer
- AB 3130 Genetic Analyzer
- AB 3130xL Genetic Analyzer
- AB 3500 Genetic Analyzer
- AB 3500xL Genetic Analyzer
- AB 3730 Genetic Analyzer
- AB 3730xL Genetic Analyzer
- AB 5500 Genetic Analyzer
- AB 5500xl Genetic Analyzer

- AB SOLiD 3 Plus System
- AB SOLiD 4hq System
- AB SOLiD 4 System
- AB SOLiD PI System
- AB SOLiD System
- AB SOLiD System 2.0
- AB SOLiD System 3.0
- Complete Genomics
- Helicos HeliScope
- Illumina Genome Analyzer
- Illumina Genome Analyzer II
- Illumina Genome Analyzer IIx
- Illumina HiScanSQ
- Illumina HiSeq 1000
- Illumina HiSeq 1500
- Illumina HiSeq 2000
- Illumina HiSeq 2500
- Illumina HiSeq 3000
- Illumina HiSeq 3500
- Illumina HiSeq 4000
- Illumina HiSeq X Five
- Illumina HiSeq X Ten
- Illumina MiSeq
- Illumina MiniSeq
- Ion Torrent PGM
- Ion Torrent Proton
- NextSeq 500
- NextSeq 550
- MinION
- PacBio RS
- unspecified

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Index