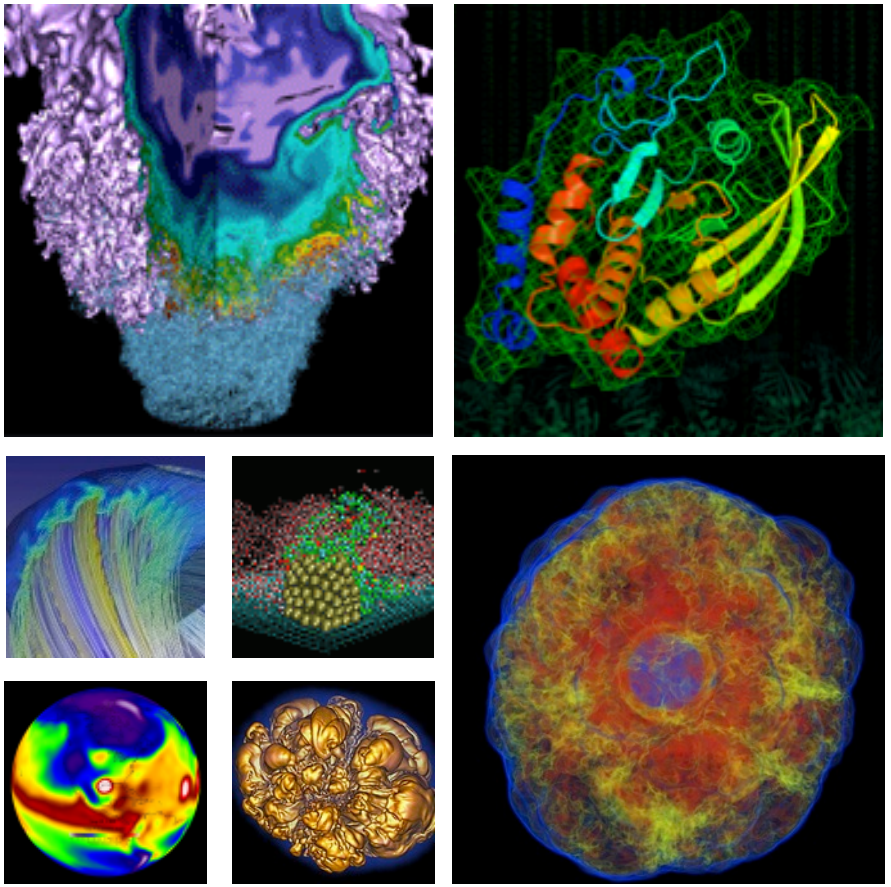
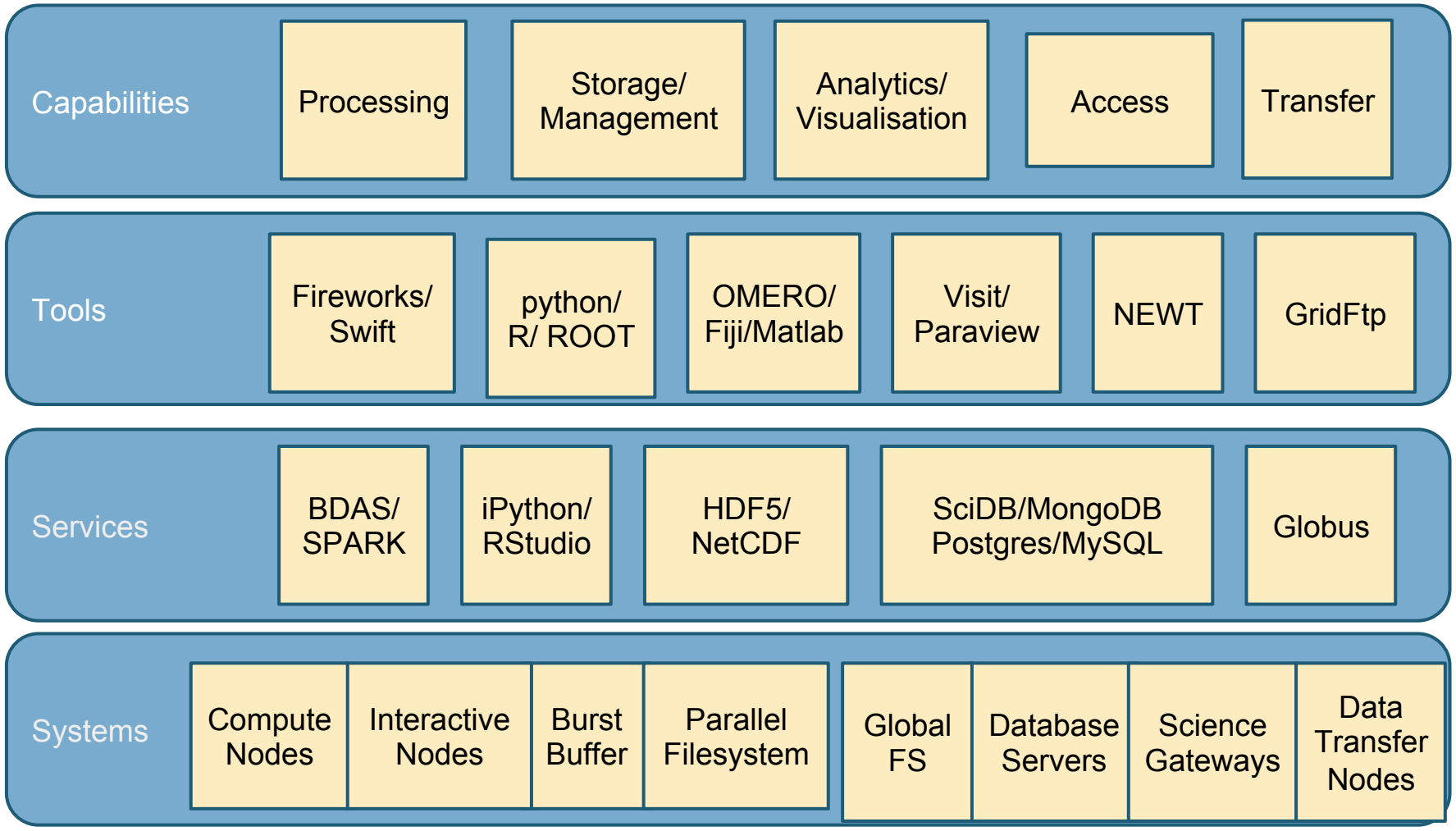
Databases and I/O



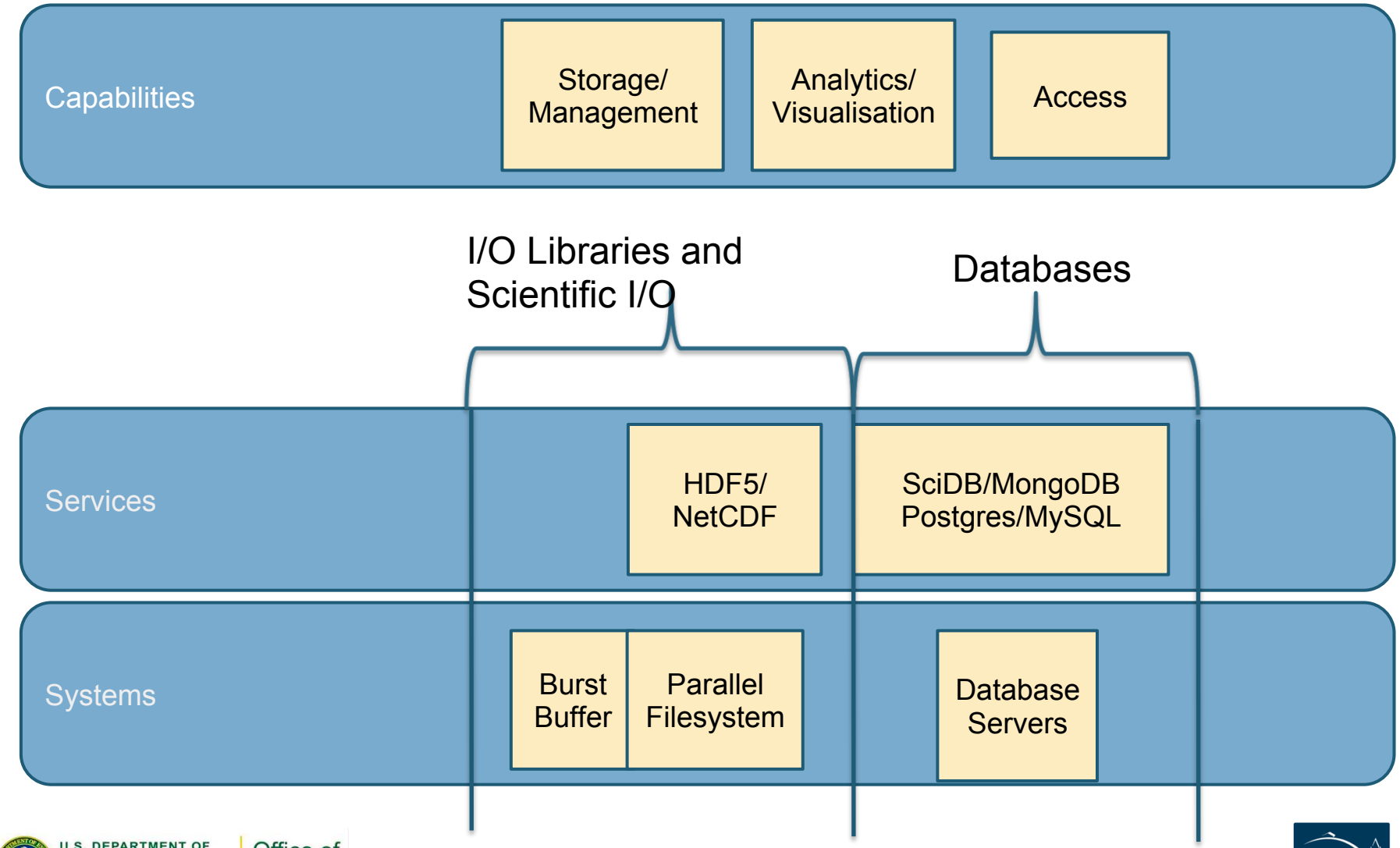
Wahid Bhimji (wbhimji@lbl.gov)

https://www.nersc.gov/users/data-analytics/data-management/

NERSC Data and Analytics Stack



Databases and I/O Libraries



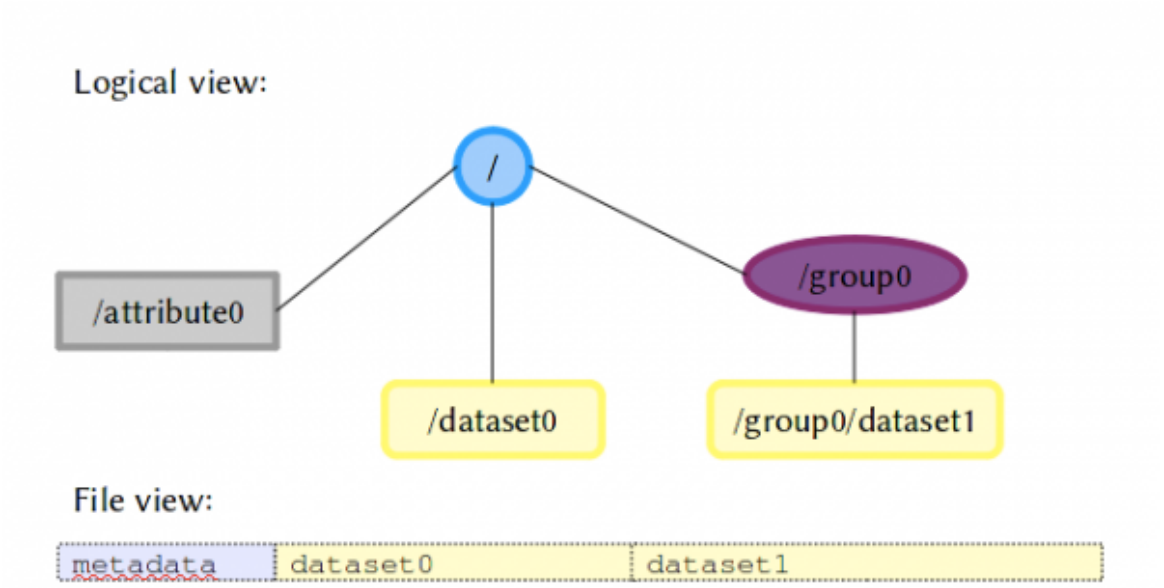
I/O Libraries

I/O Libraries

- Storing large amounts of data in files on disk?
- Use a high-level I/O library:
  - Compressed, self-describing, binary files
- You can use your own format of course
- But some examples heavily used in science are **HDF5, NetCDF and ROOT** – we 'support' these
- **HDF5 (Hierarchical Data Format v5)** probably the most widely-used / general purpose:
  - Data model allows for complex data objects.
  - Portable file format across machines.
  - Software libraries; performance features and tools.

HDF5

- **Hierarchical:** *groups* and *datasets* like folders and files
- **Datasets:** Have *headers* (that describe type, dimensionality, and layout) and *data array*
- ***Attribute***: Name, Value pairs



HDF5: APIs

- Load the relevant module. E.g parallel HDF5 on Edison

```
module swap PrgEnv-pgi PrgEnv-cray
module load hdf5-parallel
```

- Include header / library. E.g. in C++, create file:

```
#include <hdf5.h>
file_id = H5Fcreate("myfile.h5", H5F_ACC_TRUNC, H5P_DEFAULT, H5P_DEFAULT)
```

- For python can use *Pandas (PyTables library):*

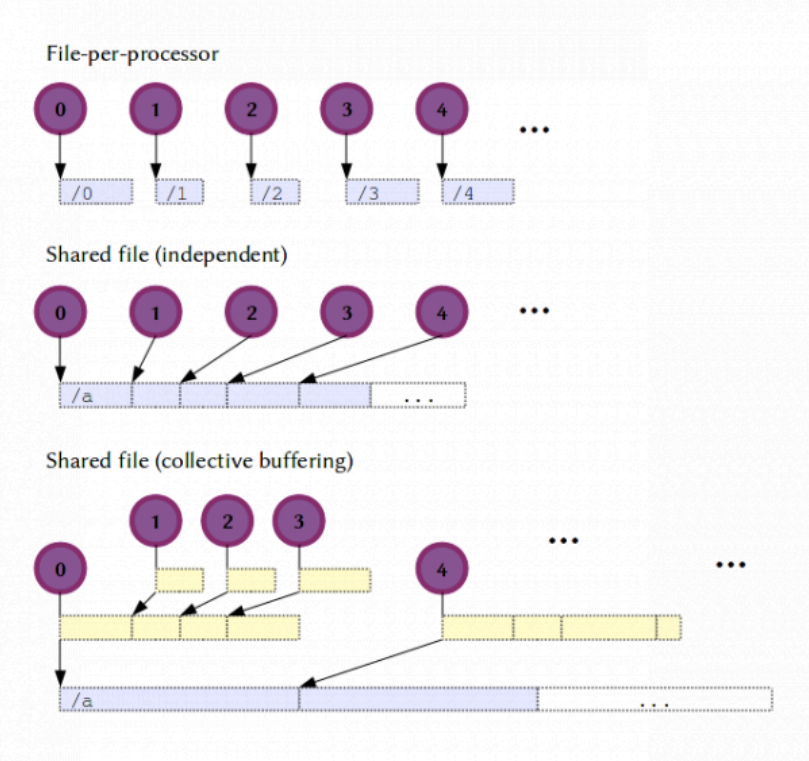
```
store = HDFStore('store.h5') ; store['df'] = DataFrame(data,index, columnames)
```

Also supports top-level API to_hdf, read_hdf similar to to_csv

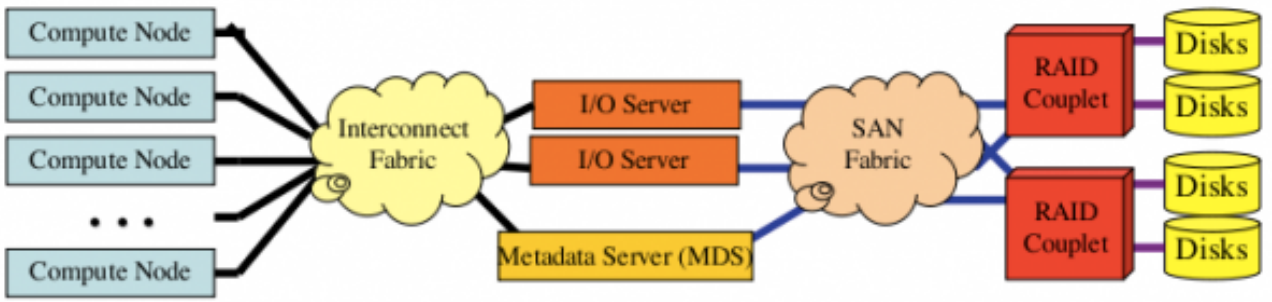
```
dataframe.to_hdf('store_tl.h5','table',append=True)
```

Parallel I/O

- For tips on parallel HDF5, examples of scientific I/O and Lustre optimizations see: https://www.nersc.gov/users/training/online-tutorials/introduction-to-scientific-i-o/
- **Parallel HDF5** supports MPI and **MPI-I/O**
  - To achieve highest bandwidth write one shared file in parallel
  - And use collective parallel I/O



Lustre Striping



- **Lustre** stores files *striped* across storage devices – Object Storage Targets (OSTs):
  - Larger files can be striped across many OSTs

```
lfs setstripe [file,dir] -c [count] -s [size]
```

Or `stripe medium` [helper wrapper script]

Size	Collective shared-file	File per processor
< 1GB	Use default	stripe_fpp or default
1-10GB	stripe_small	stripe_fpp or default
10-100GB	stripe_medium	stripe_fpp or default
100GB-1TB	stripe_large	Contact us

Databases

Databases

- **Relational / SQL Databases**
  - NERSC host/support MySQL and PostgreSQL DBs for users
- **NoSQL / Schema-less Databases**
  - MongoDB
- **Array Databases**
  - SciDB

To request a database:
https://www.nersc.gov/users/science-gateways/science-gateway-databases/

SQL Databases

- **Good for:**
  - Your data is structured (you have a 'Schema')
  - Relational (tables of rows and columns)
  - Mid-Size, <=several GB in total
  - transactional operations (ensuring DB is consistent)
- Single databases on single servers so multiple connections not served in parallel.
- **PostgreSQL:** Object relational, some powerful features and extensions as well as SQL standards
- **MySQL:** Very popular, open-source, relational database

Accessing SQL DBs

- **Postgres:**

  ```
  psql –h scidb1.nersc.gov yourdb –U dbuser
  ```

- **Mysql:**

  ```
  mysql yourdb –u dbuser –h scidb1.nersc.gov -p
  ```

- **Very basic sql:**

  SET PASSWORD = PASSWORD('password');

  USE yourdb;

  CREATE TABLE yourtable (a_id INTEGER PRIMARY KEY ,b VARCHAR(10));

  SELECT * from yourtable WHERE yourtable.b = 'bob';

- Learning SQL try **http://sqlzoo.net/**

MongoDB

- **'NoSQL', document-oriented database**
- JSON-like documents (key: value)
- Queries are javascript expressions
- Memory-mapped files – queries can be fast
- Though not configured for very frequent/ high-volume writes or very many connections
- **Good For:**
  - Un-Structured Data ('Schema-less')
  - Mid-Size to Large, e.g. 10 GB of Text

Accessing MongoDB

- Use mongo client

```
mongo -u yourdb_admin –p password mongodb01.nersc.gov/yourdb
```

– Create a collection ; put a document in it and find it

```
doc1 = {name: “bob”, friends:5 }
yourdb.acollection.insert(doc1)
db.customers.find({name:”bob"})
```

- Use pymongo for Python:

```
import pymongo
client= pymongo.MongoClient(‘mongodb01.nersc.gov’)
client.admin.authenticate(yourdb_admin, args.passwd)
client.yourdb.acollection.insert([{"name": ”bob”, ”friends": 5}])
```

SciDB

- **'Array' Database: Time series / Spectra / Images etc.**
- Scalable to big data (TBs+) – 'horizontally'
- Perform statistical analyses in parallel without HPC programming using queries
- **Good For:**
  - Array Structured Data (Dense or Sparse)
  - Large, e.g. 10+ GB
  - You want to push array type computations on big data and return statistical summaries (not extract the big data)
- **Boutique service – contact us via web form if interested:** https://www.nersc.gov/users/science-gateways/science-gateway-databases/

Accessing databases (and data)

- Can use command line or APIs
- Or via Webpages – e.g. Science Gateways or

Omero:

- **Visualization, management and analysis of (biological microscope) images**
  - CLI ('dropbox' folder) to upload images
  - Read any image format
  - Connect via web portal or query…
  - Perform operations (averages, max , min etc.)
- Request PostgreSQL via web form and specify Omero in comments

Parting thoughts on I/O and databases

- **Files and Databases (some personal observations)**
  - Massively parallel HPC programs -> Files
  - Instrument data distributed around the world -> Files
  - Large 100-1000 user collaborations -> Files
  - metadata (e.g. about conditions in which data was collected) -> Database (SQL if schema known)
  - Multi-source, aggregated, instrument metadata -> NoSQL DB
  - Array analytics -> SciDB

- **Database and file I/O documentaton:**
  - https://www.nersc.gov/users/data-analytics/data-management/

- **Database Request Form**
  - https://www.nersc.gov/users/science-gateways/science-database-request-form/