Princeton workshop 2019

Data Joint

DataJoint Primer for MATLAB - building your project's data pipeline

Presented by
Shan Shen
Feb 13, 2019

Overview of this workshop

- 2/13 1 - 3 pm: DataJoint basics and build your own pipeline from scratch.
- 2/14 10 am - 12 pm: Introduction to several canonical pipelines.
    - Colony management
    - Behavior & Ephys
    - Two-photon
- 2/14 1 - 4 pm: Individual sessions
- 2/15 9 am -12 pm: more individual sessions and topics on demand.
    - Git & Docker
    - Python version of DataJoint

Topics today

- Session 0: Getting access to materials and DataJoint
- Session 1: Getting started with DataJoint: create, query and fetch a data pipeline
- Session 2: Imported and Computed table
- Session 3: Common design patterns and advanced queries

DataJoint

Session 0:
Getting access to materials and DataJoint

Access to the materials

- Go to website: https://datajoint.io/workshops
- Download materials from: https://github.com/vathes/princeton-workshop-2019
- Start MATLAB and go to the directory of the download materials.

Setting up DataJoint for MATLAB is simple!

- Start MATLAB
- Home —> Add-Ons —> Get Add-Ons
- Search “DataJoint”
- Click on “DataJoint” in the search result
- Click Add —> Add to MATLAB —> OK
- In your console, type in “dj.version”
- You are all set and ready to start!

Data
Joint

Session 1:
Getting started with DataJoint

Session 1 Goals

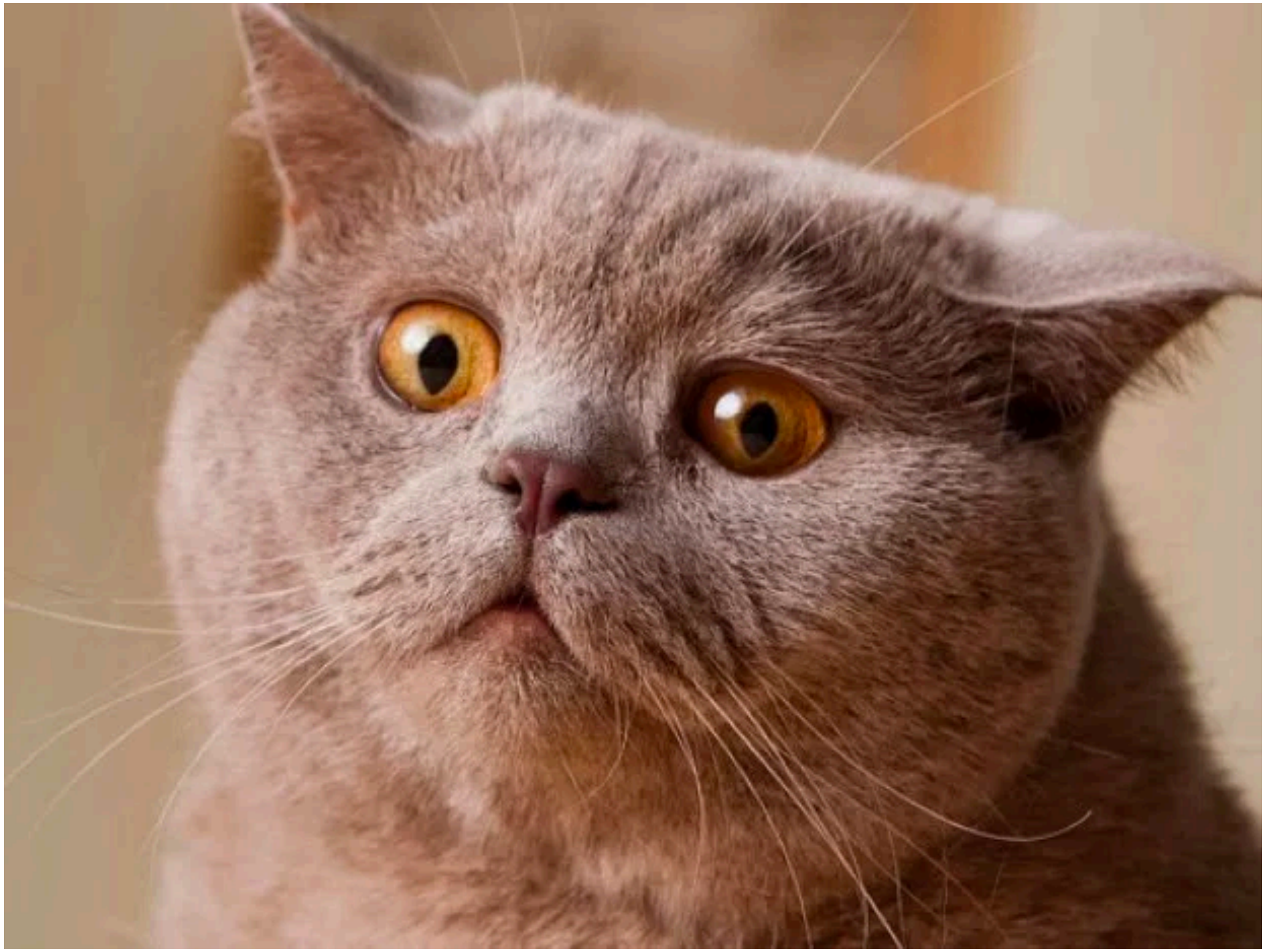
1. Learn what a pipeline is
2. Design and create our first pipeline in DataJoint
3. Insert data
4. Query and fetch data

What is a data pipeline?

“A data pipeline is a sequence of steps (more generally a directed acyclic graph) with integrated storage at each step. These steps may be thought of as nodes in a graph”

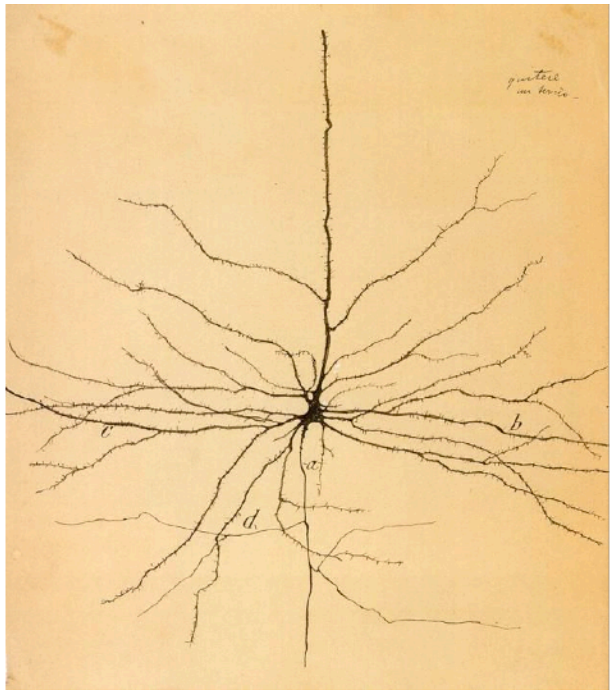
~ from DataJoint documentation (https://docs.datajoint.io)

...?



Data pipeline are about “things” in your experiment!

mouse

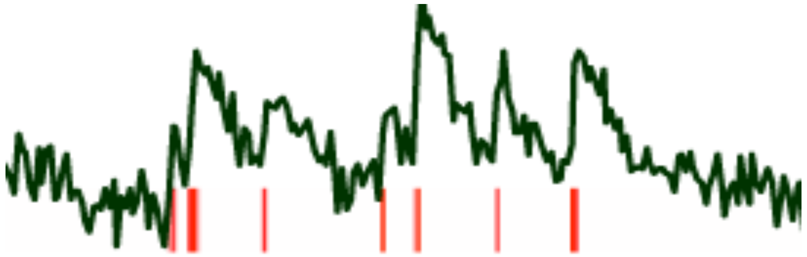


neuron

experimental session

Θ

parameter

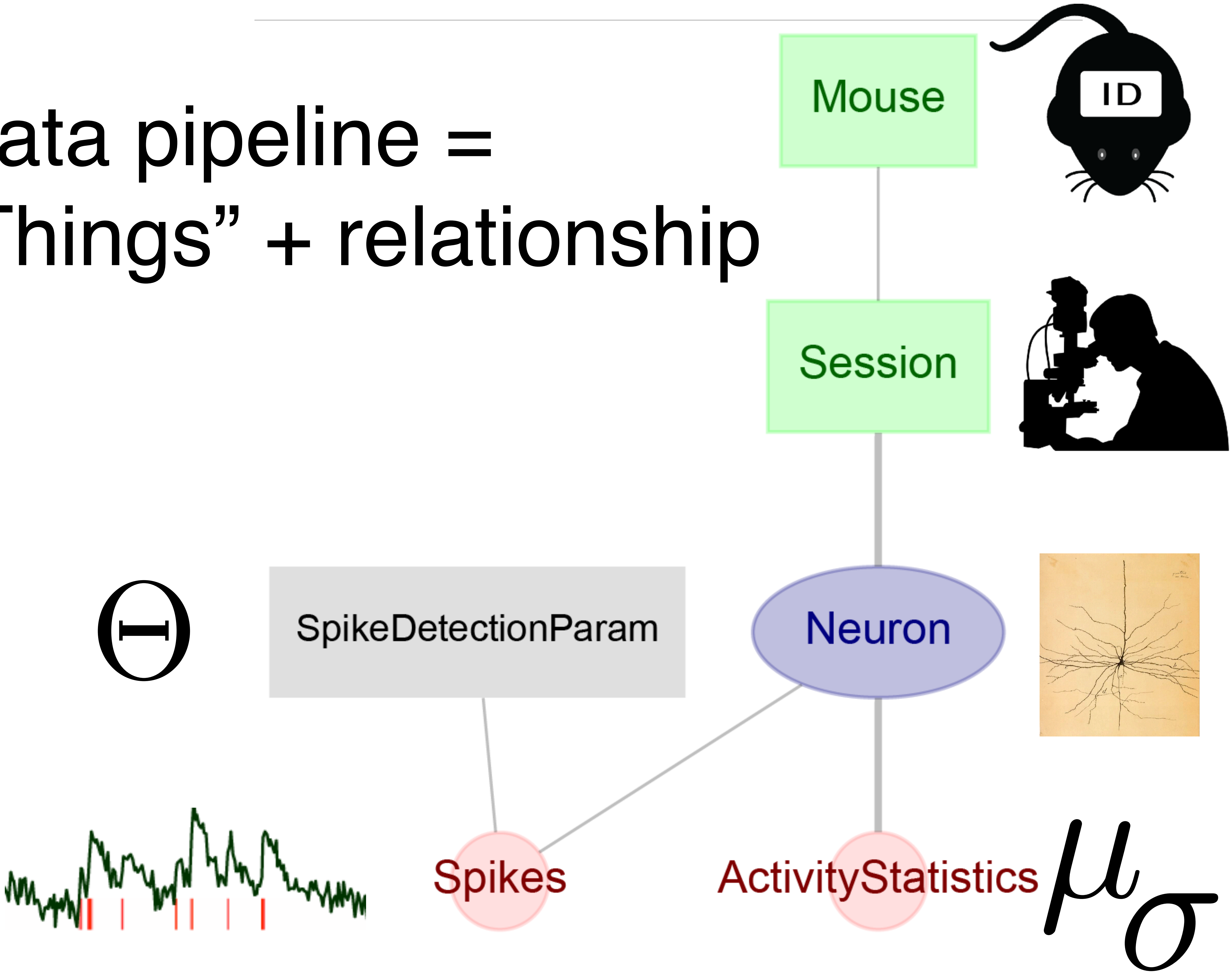


spikes

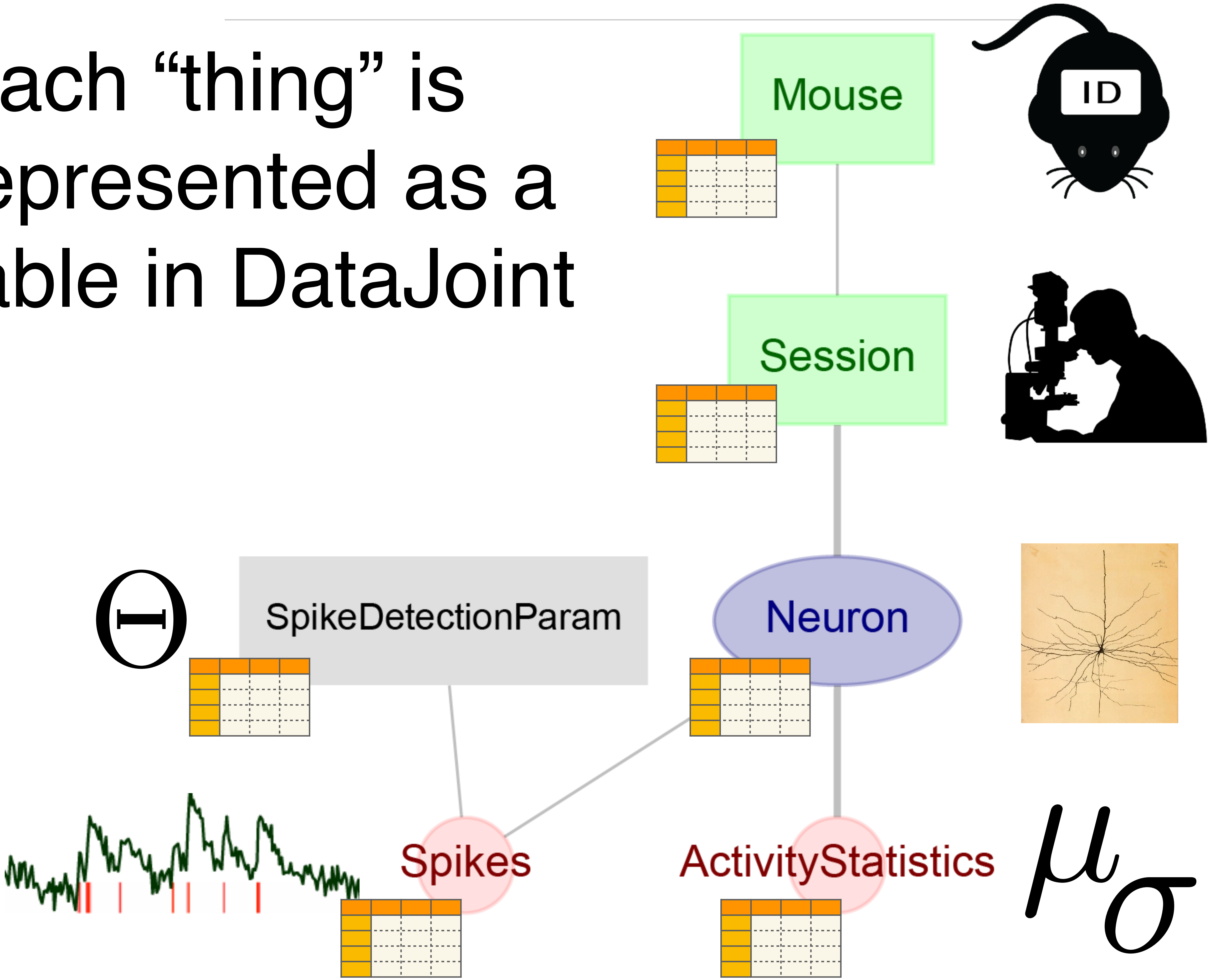
μ_{σ}

statistics

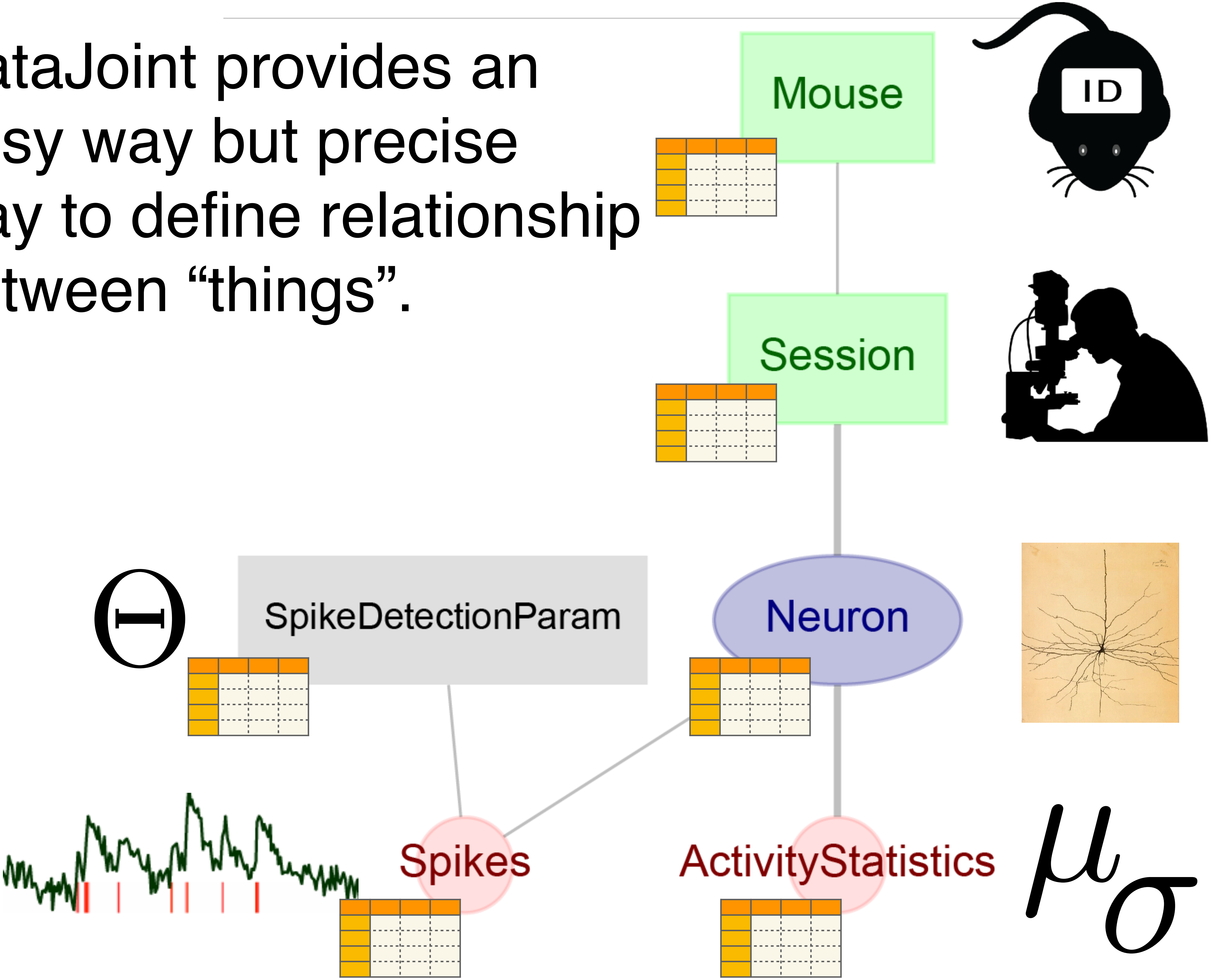
Data pipeline = “Things” + relationship



Each “thing” is represented as a table in DataJoint



DataJoint provides an easy way but precise way to define relationship between “things”.



Building your first pipeline

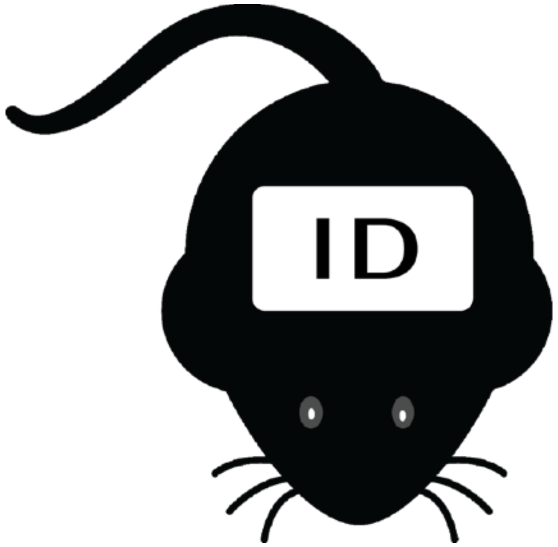
- “Things” in our project:
  - Mouse
  - Experimental session
  - Neuron
  - Spikes

Building your first pipeline

- “Things” in our project:
  - **Mouse**
  - Experimental session
  - Neuron
  - Spikes

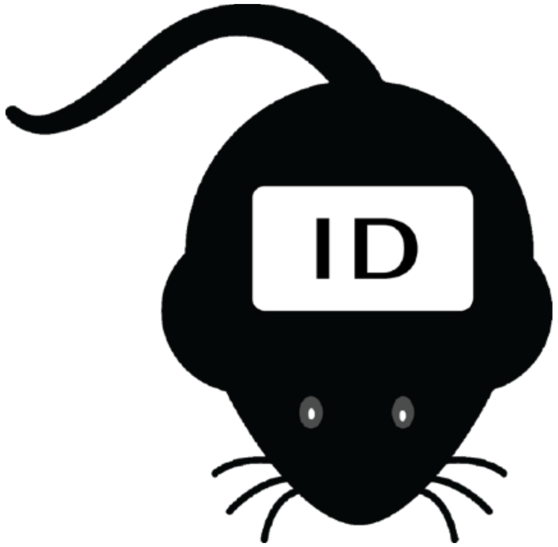
Representing a mouse

What would uniquely identify a mouse?



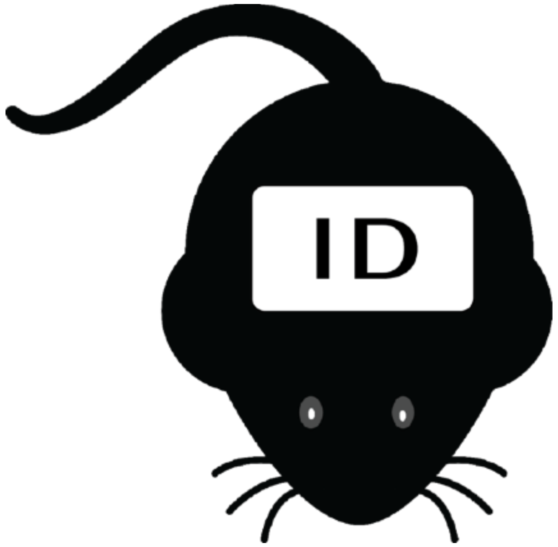
Representing a mouse

mouse_id!



mouse_id*		
12123		
15302		
1243		

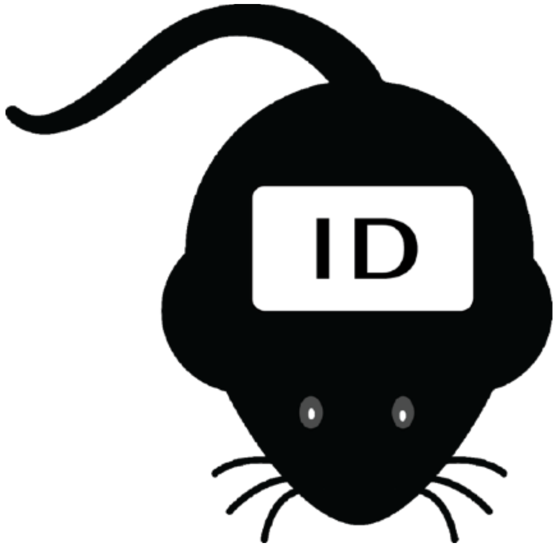
Representing a mouse



This is the **primary key**

mouse_id*		
12123		
15302		
1243		

Representing a mouse

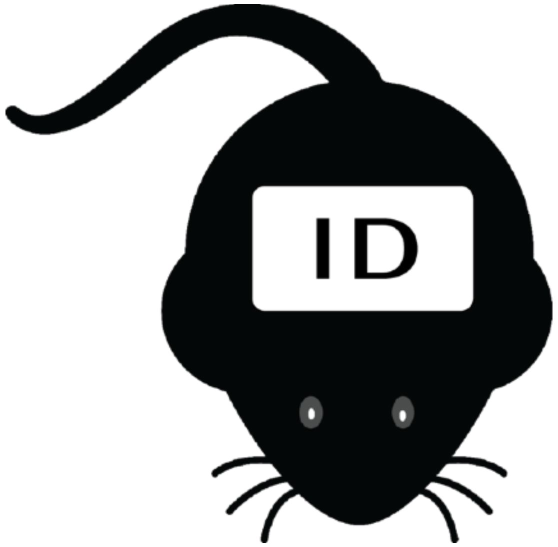


mouse_id*		
12123		
15302		
1243		

Each row is **a mouse**

Representing a mouse

Adding other attributes (columns) **about each mouse**



mouse_id*	dob	sex
12123	2017-01-12	M
15302	2018-01-01	F
1243	2016-03-05	Unknown

Let’s now go build the pipeline in DataJoint!

So far…

- All tables have been manual …
- What about tables for recordings that need loading from external data files?
- What about some analysis tables that need computation?
- DataJoint provides nice support for auto computation and insertion.

DataJoint

Session 2:
Imported and Computed tables

Session 2 Goals

1. import neuron activity data from data files into an **Imported table**
2. compute various statistics for each neuron by defining a **Computed table**
3. define a **Lookup table** to store parameters for computation
4. define another **Computed table** to perform spike detection and store the detected spikes
5. automatically trigger computations for all missing entries with **populate**

DataJoint

Session 3: Common design patterns and advanced queries

Session 3 Goals

1. highlight common design patterns found in our data pipeline

2. some more complex DataJoint queries

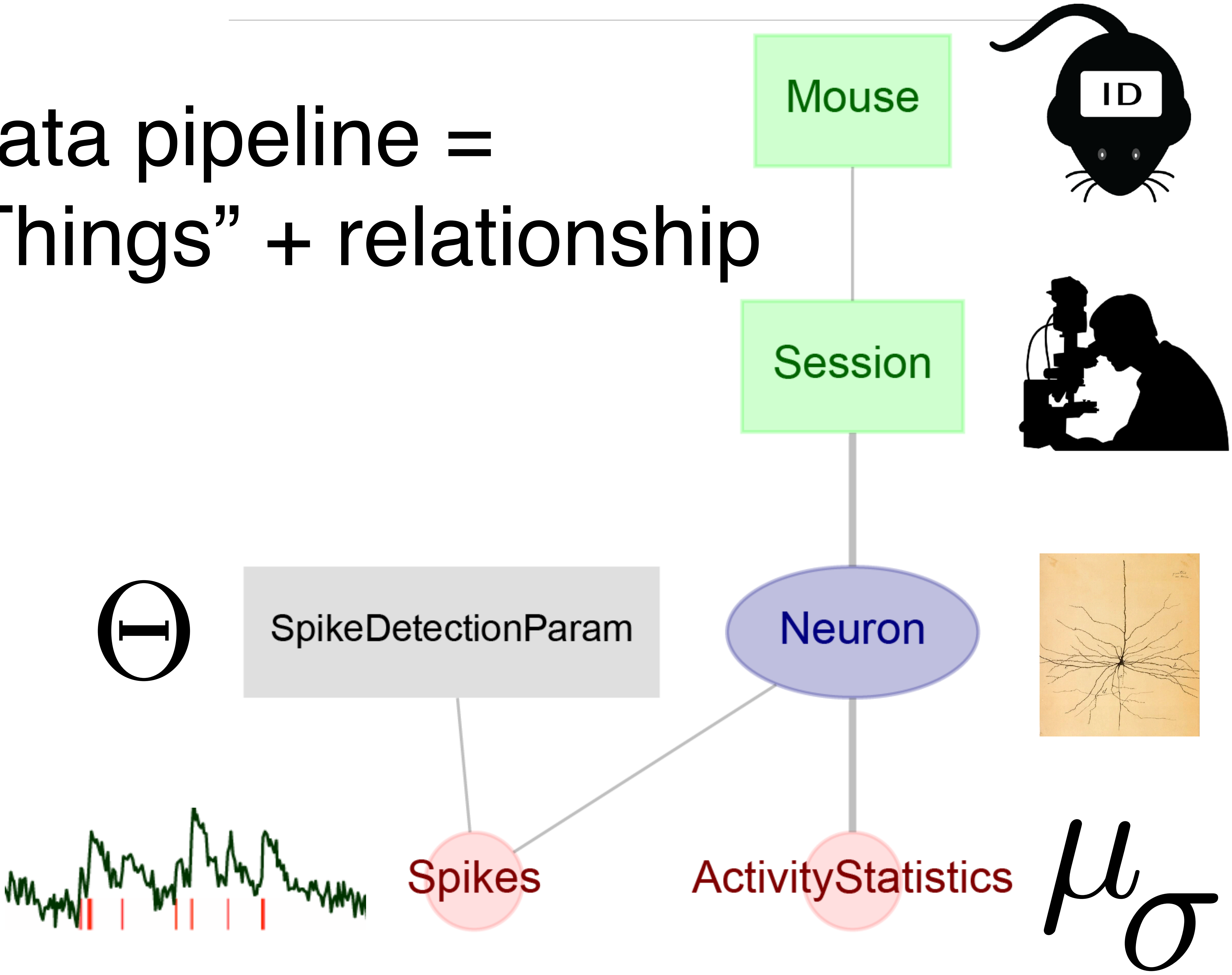
DataJoint

Recap of Today

We covered a lot!

- Designed our first data pipeline
- Learned to insert, query and fetch data
- Learned to define computations as tables in data pipeline
    - Computing “statistics”
    - Detected “spikes”
- Learned to use `make` and `populate` logic to automatically “populate” tables
- Studies common design patterns in data pipeline

Data pipeline = “Things” + relationship



Mouse

Session

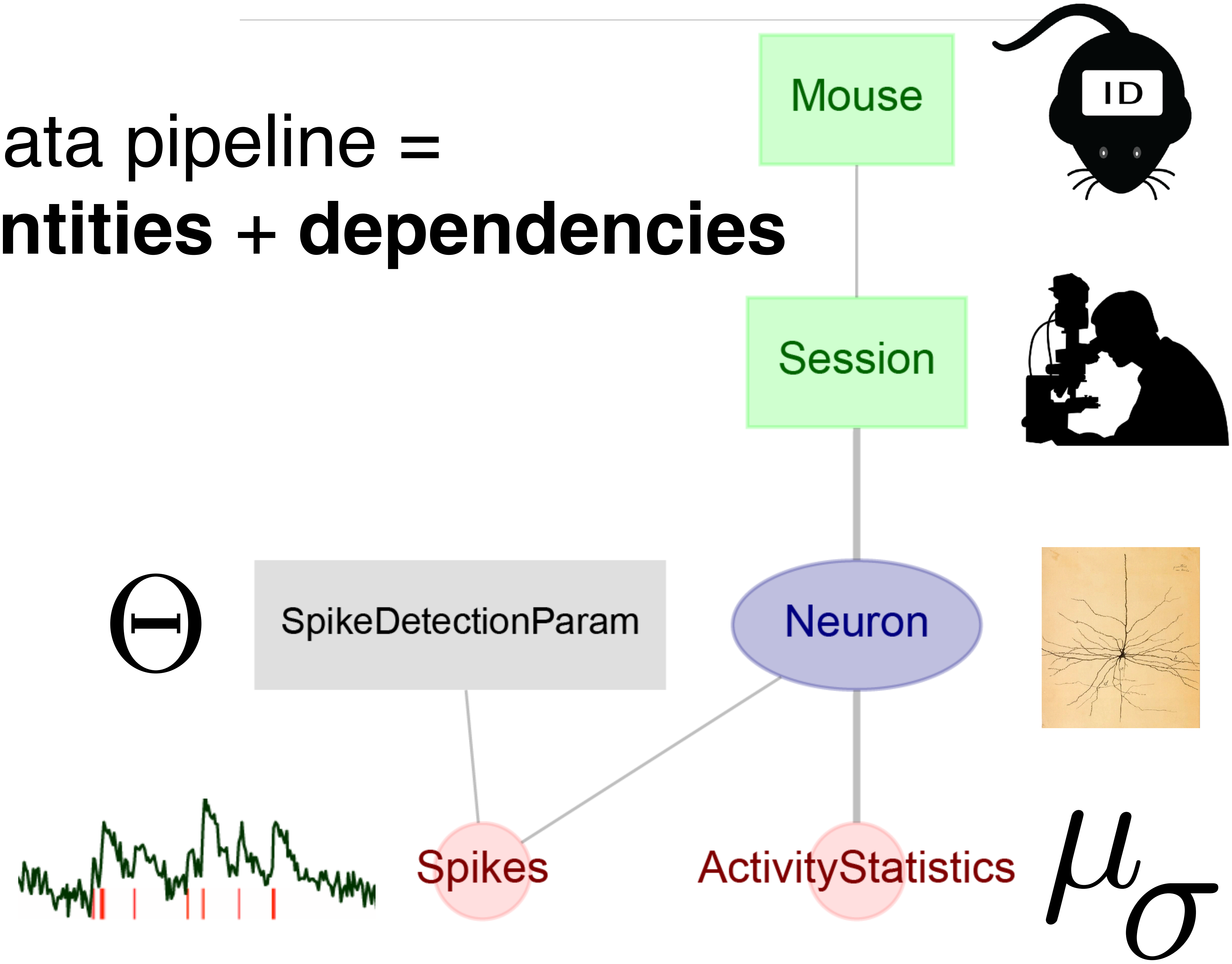
Θ SpikeDetectionParam

Neuron

Spikes

ActivityStatistics μ_{σ}

Data pipeline =
Entities + dependencies



Mouse

Session

Neuron

SpikeDetectionParam

Spikes

ActivityStatistics

Additional learning resources

- Visit https://datajoint.io for more information about DataJoint - the free open-source libraries for Python 3 and MATLAB

- Documentation and tutorials are available at https://docs.datajoint.io and https://tutorials.datajoint.io

- DataJoint Slack group is an excellent place to interact with developers and other users.

- More learning resources are up and coming!

Thanks for attending!