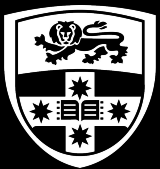


FrameTree: mapping data trees onto frames

Thomas Close

National Imaging Facility Informatics Fellow
School of Biomedical Engineering

Australian Imaging Service - Analysis Pipelines Lead



THE UNIVERSITY OF
SYDNEY



**NATIONAL
IMAGING
FACILITY**

In-place analysis of large datasets

- Biomedical imaging datasets are getting bigger
- Clinical trials require robust reproducible analysis
- End-to-end pre-processing and analysis workflows
 - From images to results
 - Reproduce studies on other datasets
- Derivatives can be shared along with the acquired data

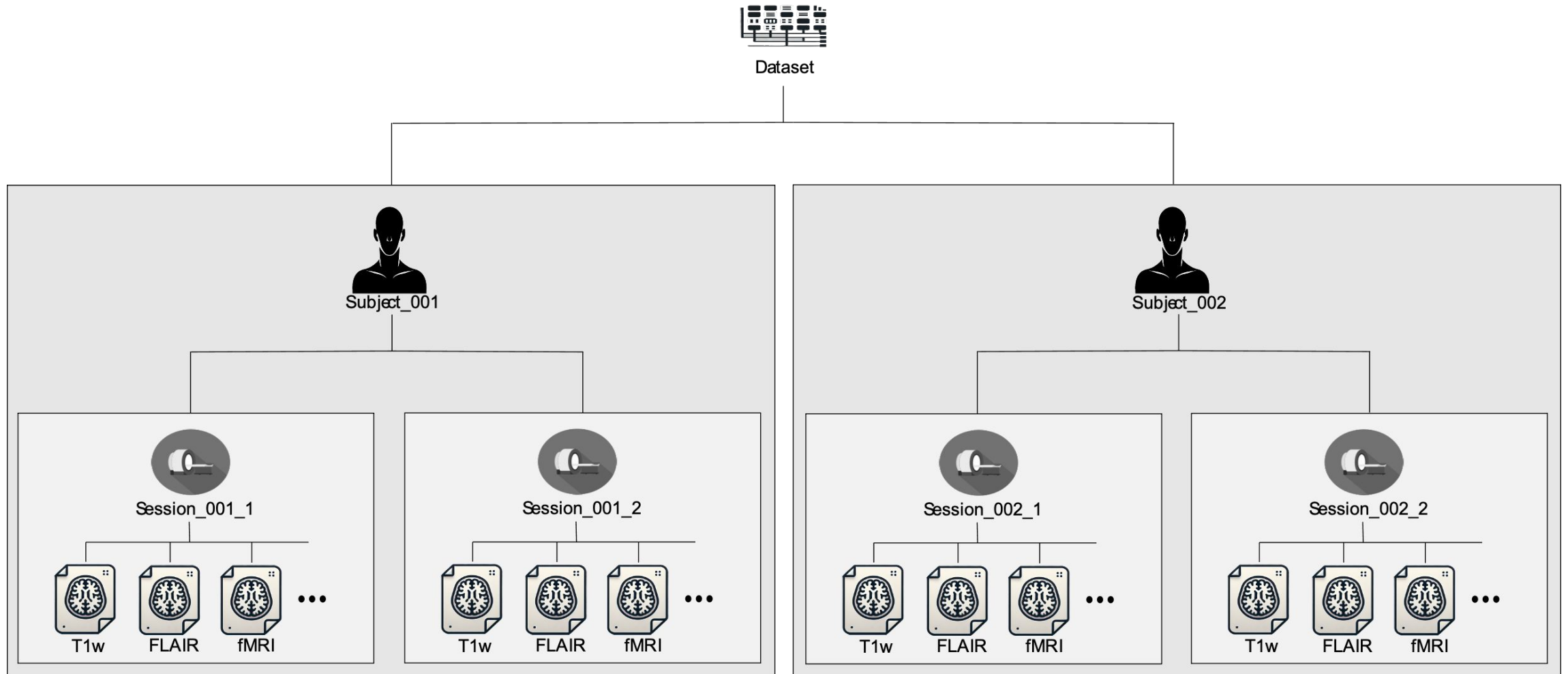
The screenshot displays a web application interface for managing biomedical data. The top navigation bar includes links for Browse, New, Upload, Administer, Tools, and Help, along with a search bar and user information (Logged in as: Thomas Close, Auto-logout in: 0:59:34, renew, Logout).

The main content area is divided into two sections. The top section shows the details of a dataset, with tabs for Details, Access, Manage, Pipelines, and Audit Trail. The Details tab is active, displaying fields for ID, Description, Keywords, PI, and Investigators. A large black redaction box covers the description and keywords. Below these fields are buttons for Edit Details, Delete, and Manage Custom Variables. To the right of the details section is an Actions menu with options: Add, Add to Favorites, Download XML, XSync, Download Images, Full Audit Trail, Processing Dashboard, Manage Files, Project Settings, Scan Type Cleanup, Upload Images, and View Prearchive.

The bottom section shows a table of subjects. The table has columns for Subject, M/F, Hand, YOB, View, MR Sessions, and Neural Mobilization Sessions. The table is paginated, showing 1 of 6 Pages (1004 Rows). The first few rows of the table are visible:

Subject	M/F	Hand	YOB	View	MR Sessions	Neural Mobilization Sessions
ASD1116	U					
ASD1891	U					
FR	U				1	
FR_	U					
FR1147	U					
FR1764	U					
FR1795	U					
FR20072	U				4	

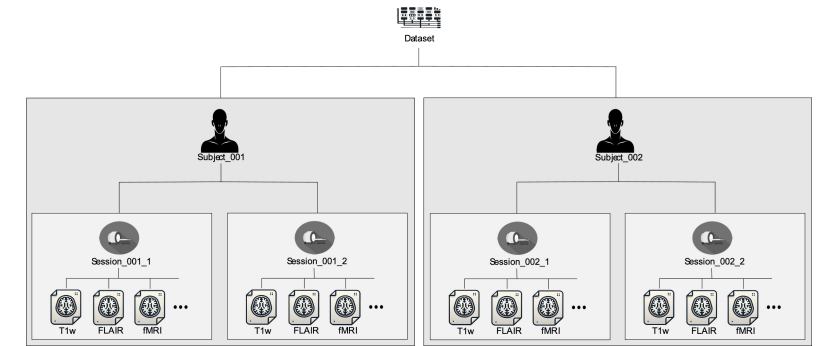
Imaging data organised within trees



Dataset > Subject > Session > Scans

Imaging data is ultimately analysed in frames

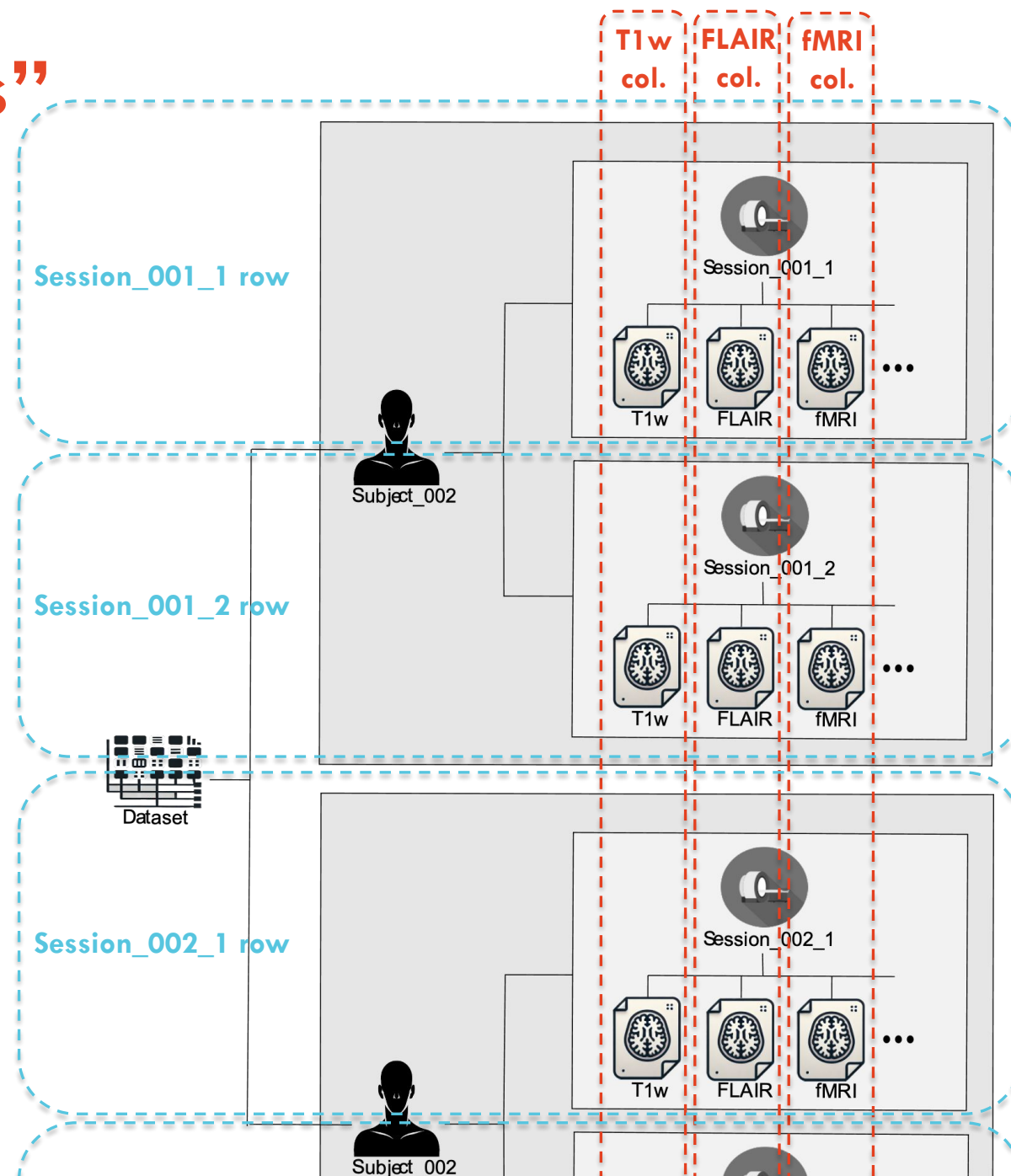
- Imaging data stored in categorical trees
 - e.g. projects, groups, subjects, sessions
- Biomarkers statistically analysed in data frames
- End-to-end workflows need to incorporate this mapping



[23]:	frame			
[23]:	average grey matter thickness (mm)	WM lesion volume (mm^3)	corpus callosum CSA (mm^2)	CST axonal density (1/mm^2)
0	1.934130	9.720063	46.231689	10348.816408
1	2.025898	9.494445	45.245623	9900.701140
2	1.915613	9.134742	62.471290	10300.549746
3	2.083231	9.941202	35.301097	10623.688702
4	2.154176	9.794590	45.607787	9828.329515
5	2.103030	9.770396	63.552230	9402.715681
6	1.873429	9.965063	33.576352	9754.472359
7	2.122523	10.489721	54.458515	9425.989576
8	2.127151	9.820211	46.912216	10779.800399
9	2.030857	9.372148	51.175165	10451.553316
10	1.995499	10.280969	49.552575	9654.425094
11	1.955350	10.337629	47.074623	10049.232598
12	2.029961	9.834188	52.973365	10442.379256

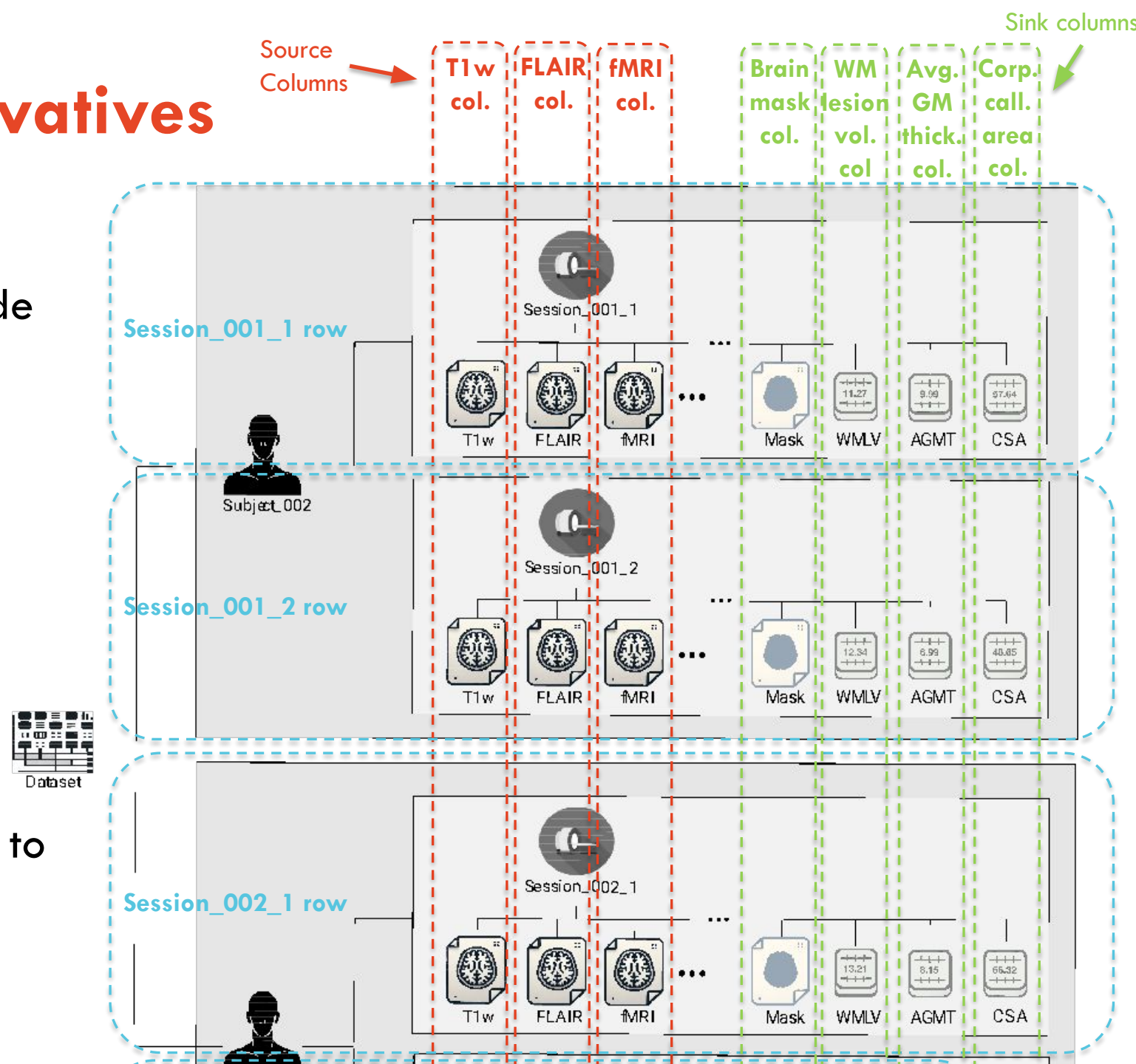
Overlay virtual "columns"

- Columns select images across different sessions
- Selection criteria are combinations of
 - Resource name (regex)
 - File format
 - Metadata
 - Order they appear, i.e. first scan called "mprage"
- Sessions form the "rows"



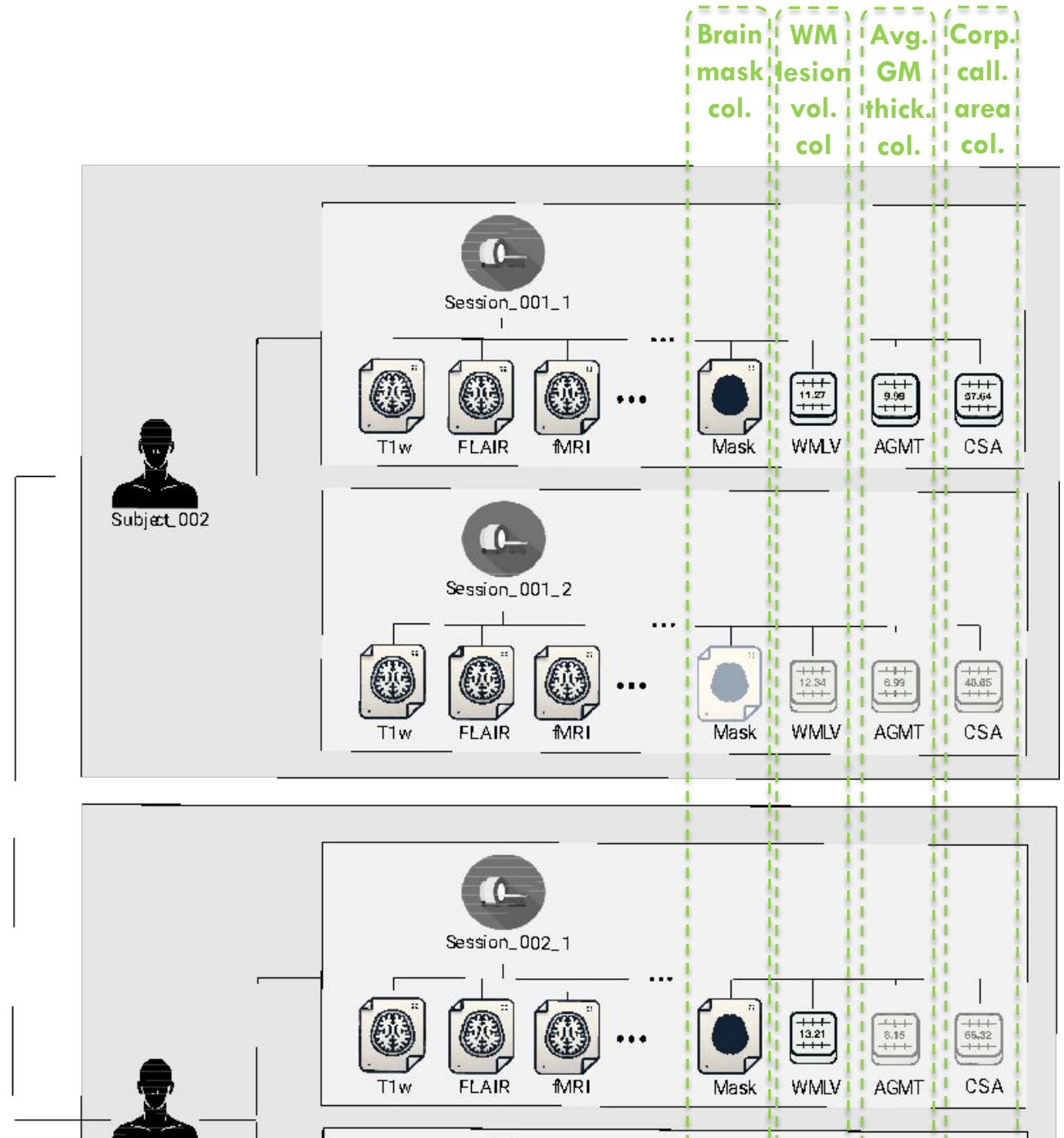
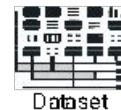
Sink columns for derivatives

- Derivatives are stored alongside source data
- “Sink columns” specify output location
- Intermediate derivatives also stored
 - QA
 - Reuse
- Field columns can be exported to data frames



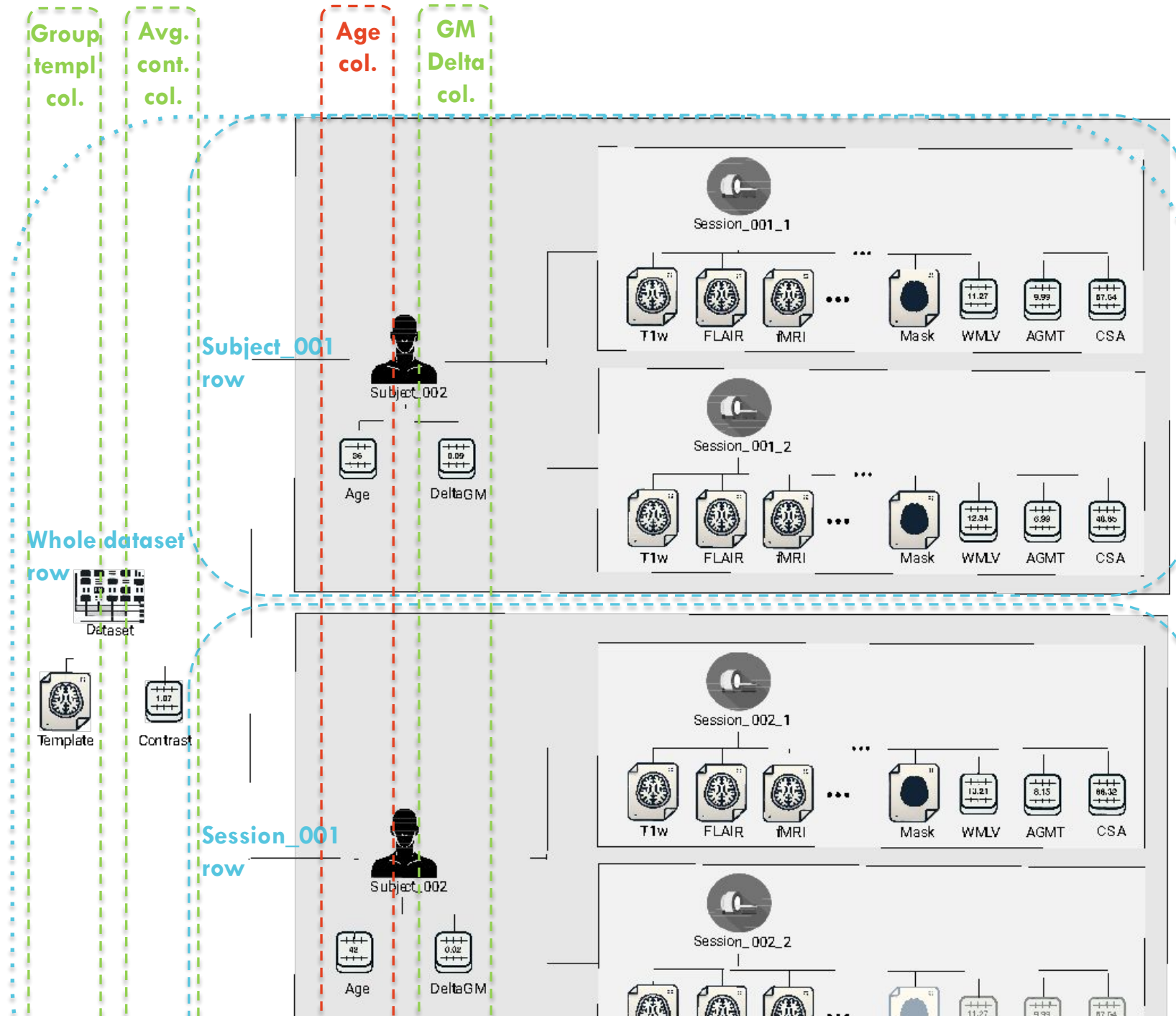
Incremental processing

- Derivatives can be incrementally processed/quality controlled
 - Detect which sessions are yet to be processed
- Rows that fail QC can be omitted from subsequent analysis



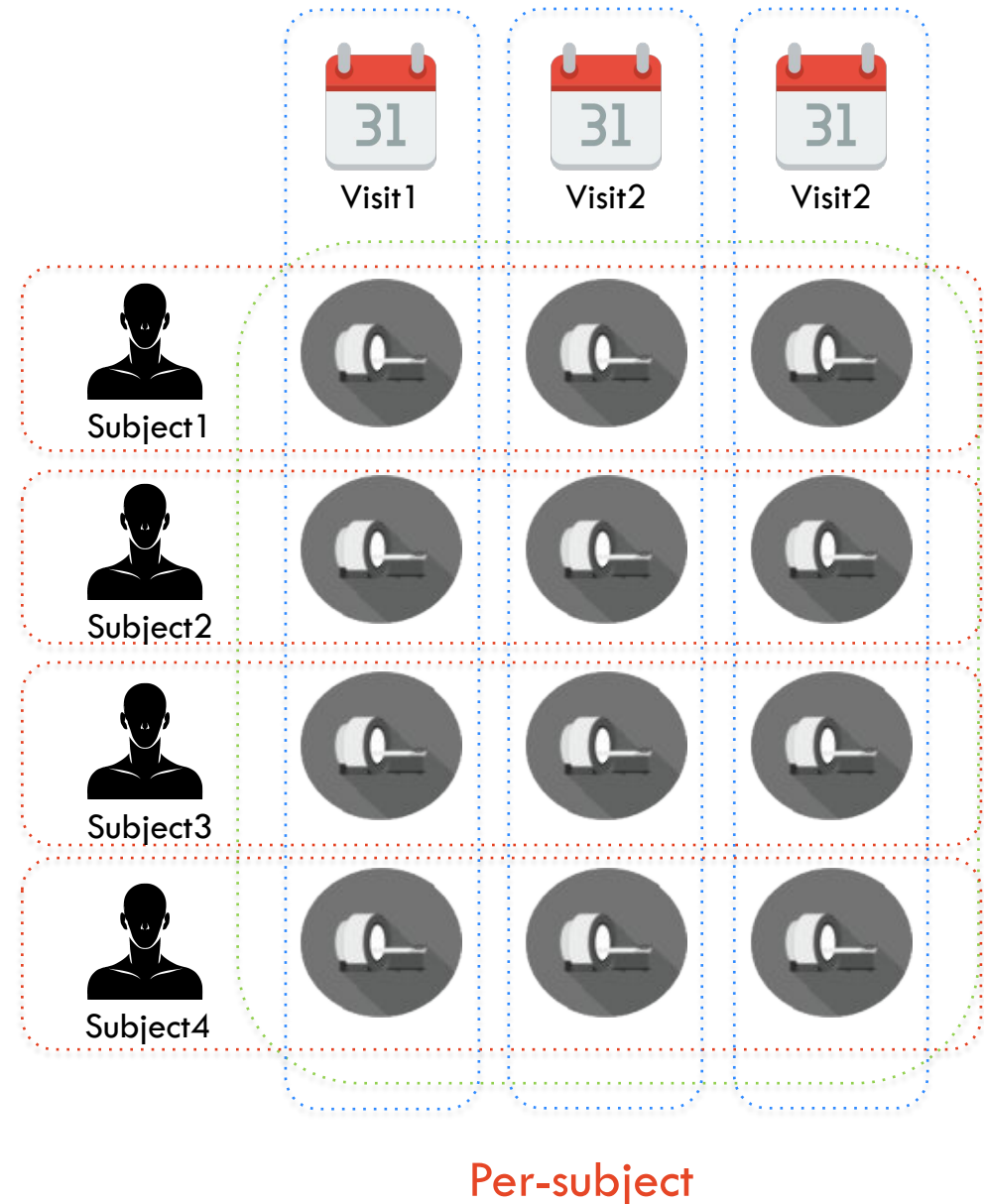
Subject rows

- Some data doesn't sit within sessions
- Rows have different “frequencies”
 - Subject-level
 - Dataset-level
- Rather than 1 frame there is a frame for each row frequency, e.g.
 - per-session
 - per-subject
 - per-visit



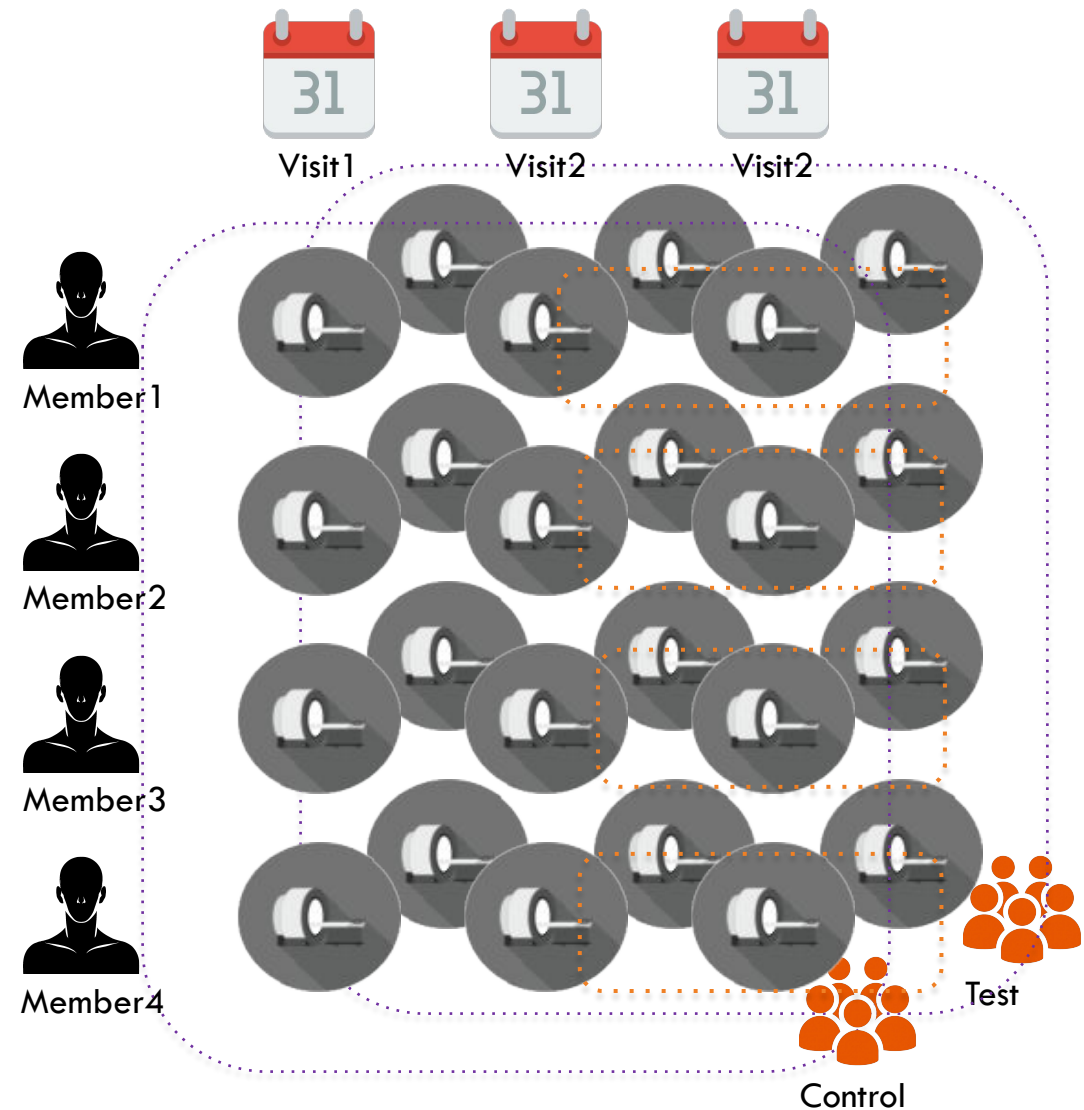
Row frequencies

- “Data frame rows” can be any combination of categorical variables
- Data from multiple sessions are aggregated
- Data frame rows can correspond to any slice of the n-d session array
 - i.e. 2^N row frequencies
- Arbitrary categorical variables, i.e. can appear in tree or not



Row frequencies

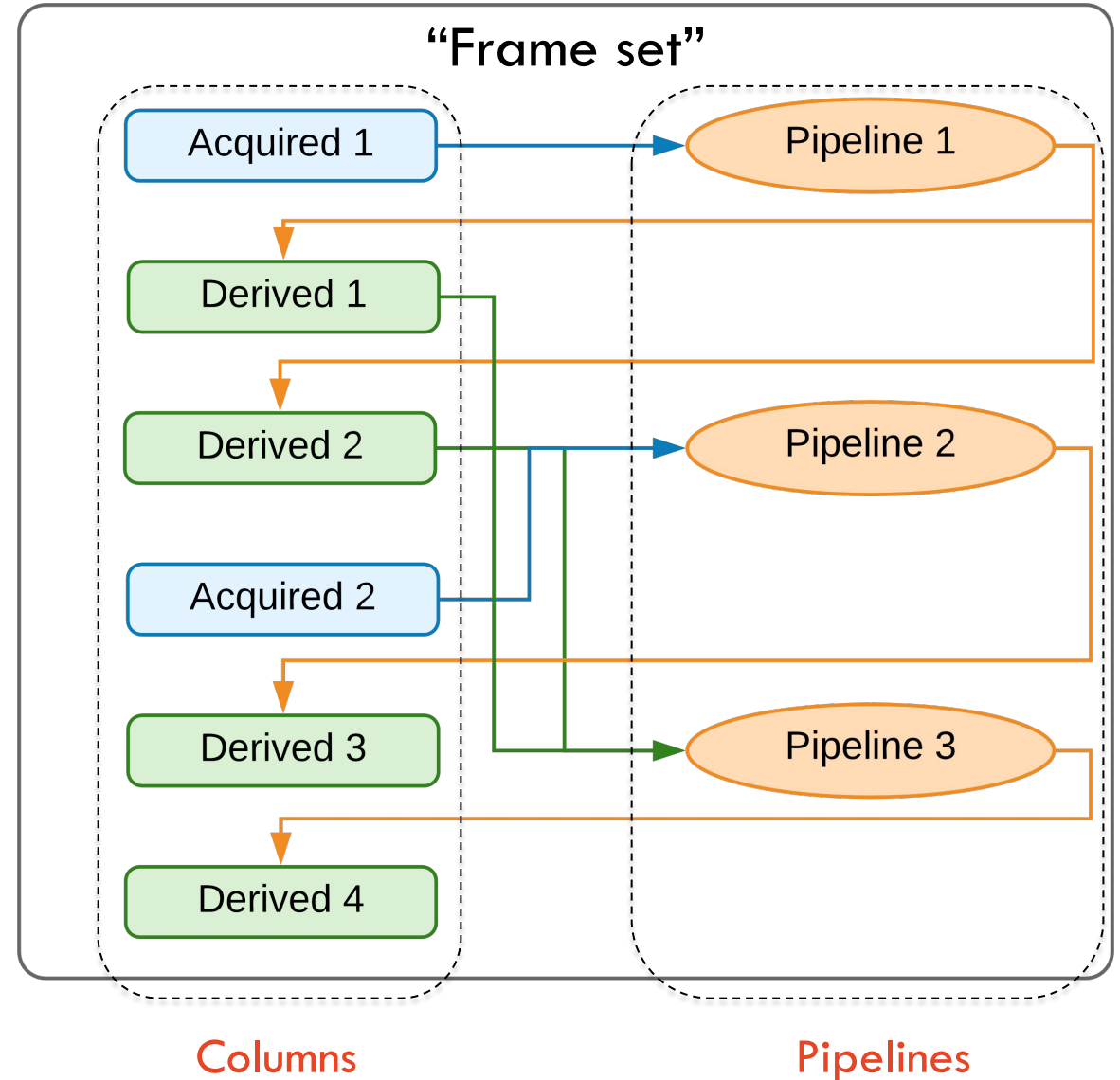
- “Data frame rows” can be any combination of categorical variables
- Data from multiple sessions are aggregated
- Data frame rows can correspond to any slice of the n-d session array
 - i.e. 2^N row frequencies
- Arbitrary categorical variables, i.e. can appear in tree or not



Per-matched-pair

Pipelines chain columns together

- Pipelines take source or sink columns as inputs and sink columns as outputs
- On request, required pipelines are pushed onto an execution stack
- Common preprocessing derivatives are reused
- Pipelines can be run incrementally as new data is added/QC'd
- Provenance is stored with derivatives, and checked before reusing



“FrameSet” definitions

- Columns and pipelines are defined in a “FrameSet” definition JSON file
 - Stored at project level of the tree
- Can be specified using `frametree` command line tool
 - Plan/hope to write a simple UI to edit it
- Parameters are stored with so analysis can be resumed from any machine
- FrameSet definitions to be used to check protocols matches the study design (in dev.)
 - Check for missing images and incorrect metadata
 - Pipeline to be run on data ingest
- File formats of the column data are specified by “MIME-like” as defined by the *FileFormats Python* package
 - Used in automatic conversions

FileFormats: file identification/handling in Python

- Python package for identifying file types
 - A class corresponding to each file type
 - Checks file extensions and magic numbers where applicable
 - Full coverage of official MIME file types
 - Customisable for arbitrary formats, e.g. JSON side cars, directory formats
- Mapping to/and from MIME-type or MIME-like strings
 - Types without a MIME type can be converted to MIME-like string in novel namespaces, e.g. `medimage/nifti`, `datascience/hdf5`
- Streamlines file copying and hashing for multi-file formats
- Hooks for converters and data reading/manipulating methods can be defined and implemented in separate packages



FileFormats

FileFormats and ontologies

- Extended fileformat syntax to allow the contents of the files to be classified
- In MIME strings this is specified using the '+' operator, e.g.
 - `image/png+zip` is a zip file containing a PNG
- The Radlex ontology (<https://radlex.org>) can be used as classifiers for the medimage namespace
 - `medimage/t1weighted.brain+dicom-series`
- Ontological information can be stored in the FrameSet definitions

Acknowledgements

- Hakim Achtenberg (Erasmus MC Rotterdam)
- Francesco Sforazzini (Monash University, Melbourne)
- Arkiev D'Souza (The University of Sydney)
- Ryan Sullivan (The University of Sydney)
- Fernando Calamante (The University of Sydney)

