PPP 20/10/2022

What term should we use to define a group of files with the same metadata?

- There doesn't seem to be a single term used by all frameworks.
- There is some merit to the fact that same files in the same "dataset" have the same schema and might need to use the same metadata.
- People are used to the fact that a "sample" means data that uses the same metadata. The fact there are one or more files is not highly relevant, those files could have been merged with hadd in principle and the logic would still work the same.
- Maybe "sampleset"?

Takeaway: The term "group" doesn't feel right. Most used ones are either "sample" or "dataset".

Should we let users define friend trees globally or within the same "group"?

- If one can define the specification such that there are friend files divided by "group", then we have to foresee that some "groups" may have friends while others won't. Thus, we have to be able to call `Define` with a default value to be used in case we are processing part of the full dataset without trees (missing columns).
- Example of using "global" friends (i.e. aligned w.r.t. the main chain)?
    - ML inference: Keep the inferred values in a column then `Snapshot` to a file. That file will afterwards be aligned w.r.t. the full dataset.
    - Maybe there should be some utility to annotate the output to know that it came from inference.
    - Maybe we could refactor `Snapshot` to save to different files according to the group. The advantage would be keeping the same dataset (specification) layout also in consecutive analyses.

Takeaway: much more common to have friend trees "per-group". Also more natural to write the list of files of friends together with the list of files of the main chain within the same "group".

About entry ranges

Takeaway: It should be possible to define entry ranges on each "group", i.e. how many entries should be read from the files that belong to that "group". Motivation: if you want to run on less data, you still want to keep info from all the different samples. Also, this allows discarding a whole group if needed.

Using the same file in different "groups"

Takeaway: Seemingly not super important, non-issue with one event loop per "group".

Dividing the main event loop in "per-group" event loops

This is a point that was raised many times while discussing. It seems that, due to how all operations would be linked to the fact they happen in a certain group, it could be better to implement the RDF event loop into separate event loops, one per "group". This is only from the scheduling point of view, API and computation graph kept as they are.

Conversely, if one needs really separate computation graphs, it's always possible to create separate RDF objects.

Going forward

There was a discussion about creating a core schema for the specification of the full dataset in a semi-structured format. It would be nice if there was a shared set of keys that could be accepted by the interfaces of the various frameworks. A github issue will be prepared to keep track of the discussion.

Miscellaneous

- Assign a sequential integer ID to each "group" that is being added so that it can be retrieved within a `DefinePerSample` call.
- Within the new API (`RDatasetSpec`,`RMetaData`,`RSpecBuilder`) it would be nice to also have a class to store all the info of a certain "group", which can then be passed programmatically to `RSpecBuilder::AddGroup`. Motivation: for cleaner code, easier to just have a list of `RDatasetGroup` and then add them all at once, so that if some have to be added/removed it's easier to track them.
- It is very important to be able to use `DefinePerSample` to define a categorical axis for a histogram, so that it can be indexed over "groups".
- Maybe `DefinePerSample` should just work per "group", not per tree
- Whenever the feature lands, tutorials should show separate histograms for each "group".