Overview of ATLAS Metadata Tools

David Malon <malon@anl.gov>

Argonne National Laboratory

Argonne ATLAS Analysis Jamboree

Chicago, Illinois

22 May 2009

This morning’s session

- David Malon: overview and selected tools
- Jack Cranshaw: finding data
  - Discovering available data and its characteristics
  - Selecting data at the dataset level
- Elizabeth Gallas: metadata at the run- and lumi-block-level
  - *Most* metadata needed to analyze event data are maintained at run- or luminosity-block-level granularity (trigger menu and prescales, detector status, quality, trigger counts, …)
  - Some end-user tools you should know
- Qizhi Zhang: event-level metadata (TAGs)
  - Now that you know which runs/datasets you want, how do select the events you want?
- David Malon: What to expect on Day 1

“One man’s data is another man’s metadata.” *Cicero (attrib.)*

- Metadata (data about data) has too many meanings, even in ATLAS, to address comprehensively here

- From Cicero to metadata:
  - Latin: Mors tua, vita mea. (What is death to you is life to me.)
  - English: One man’s meat is another man’s poison.*
  - ATLAS: One man’s metadata is …

But wait: If data is meat, then metadata … or is it the other way around?

And what’s with that meat/meta anagram, anyway?

- Emphasis here will be principally upon
  - metadata and tools to help you discover what data are available and what data you should use
  - Metadata and tools to help you understand and analyze your sample

*and French *a la* David: Ton poison, mon poisson. (One man’s fish is another man’s poison.)

Finding/selecting data: What's in a name?

- Production datasets conform to naming conventions.
  - And there is indeed a nomenclature document
- Names contain quite a bit of metadata, both explicit and encoded
  - Jack will explore this further
- Most people begin searching for datasets by pattern matching on dataset names, and for many purposes, this suffices
  - Which means they often can and do search data management catalogs directly, rather than using the AMI metadata repository
  - Though AMI, in principle and often in practice, contains much more information that could influence one's selection or one's understanding thereof

What data should you use?

- Imagine that after listening to Jack you have learned the following:
  - Cosmic ray commissioning datasets taken last fall begin with “data08”
  - “ESD” appears somewhere in the names of ESD datasets
  - If the magnet was on, “cosmag” appears somewhere in the name
    - *Exercise: which magnet?*
  - The Spring re-reprocessing of cosmics used the o4_r653 configuration tag
    - *And how might you have learned this?*
- Now you do pattern matching on data08*cosmag*ESD*r653*
  - You probably also chose a stream
    - *Exercise: What were/are the cosmics streams?*
- Of the returned datasets, which should you use?
  - Why, the Good Runs, of course
  - (a bit bogus here, since, presumably, ATLAS did not process the Bad Runs three times, but it leads us to …)

Good Run Lists

- Runs may be good for some purposes, inappropriate for others
  - "Good enough for government work"?
- An elaborate plan for Good Run List management has been proposed
- Based on Data Quality (DQ) flags
- Idea: if we have enough flags in our conditions database, then data "Good for Purpose X" should be definable as all events from runs and lumi blocks for which DQ Flag1 is Green and DQFlag2 is not worse than Yellow and DQFlag3 is Green and so on

Definition of DQ flag status (from Max Baak)

- Detector DQ flag status explained:
  - *Black: disabled Subdetector is disabled*
  - *Grey: undefined Very short runs, or problems DQ monitoring. Decided upon later.*
  - *Red: bad*
  - *Yellow: flawed Use with caution. Decided upon later.*
  - *Green: good*

⬛	Disabled
▩	Undefined
🟥	Bad
🟨	Flawed
🟩	Good

- Order: black < grey < red < yellow < green

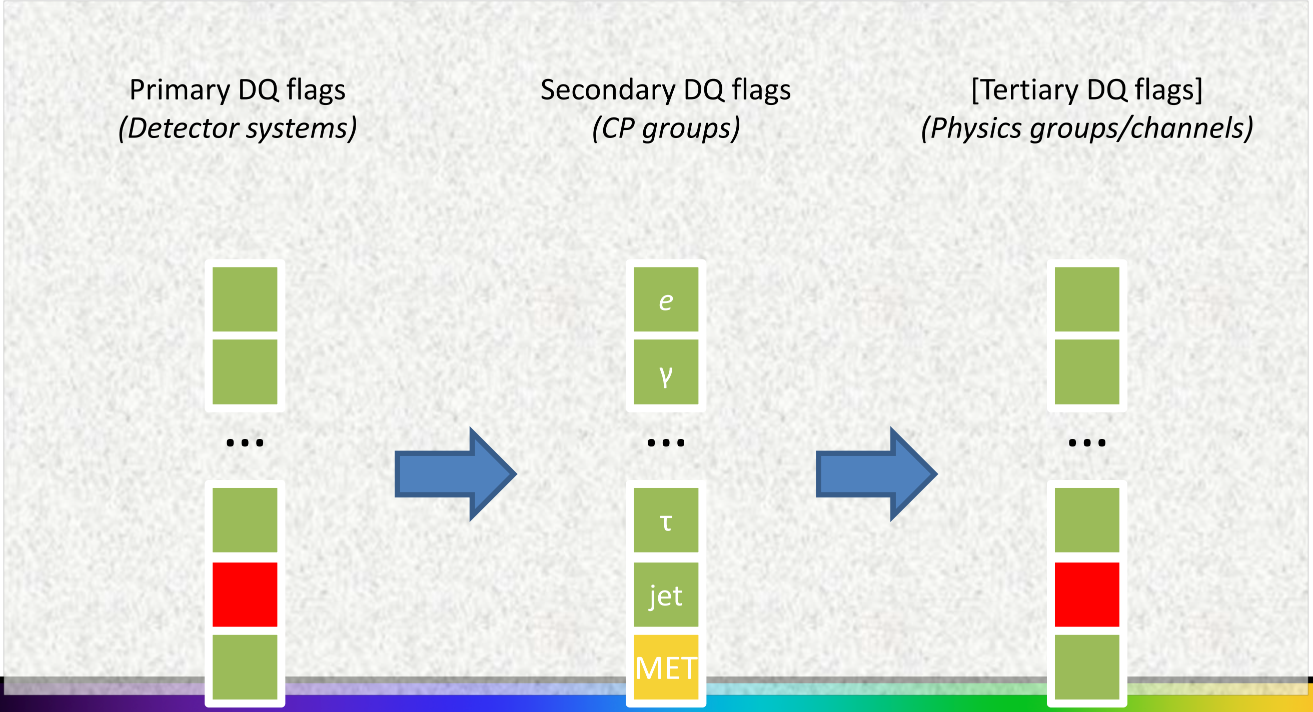
- Physics end-user does not have to deal with DQ flags.
  (Only in indirect way … more later)

Scheme for good runs/LBs list (adapted from M. Baak)

- Proposed scheme consists of three steps:

1. Lists of valid runs based on primary DQ flags set by DQ group.
2. Performance groups establish well-defined combinations of flags, secondary DQ flags, to declare data good or bad for given physics objects (e.g., Photon)
3. Depending on the final states are relevant for each analysis, Physics groups (optionally) define tertiary DQ flags accordingly

➔ define consistent list of GRLBs.



Good Run List status

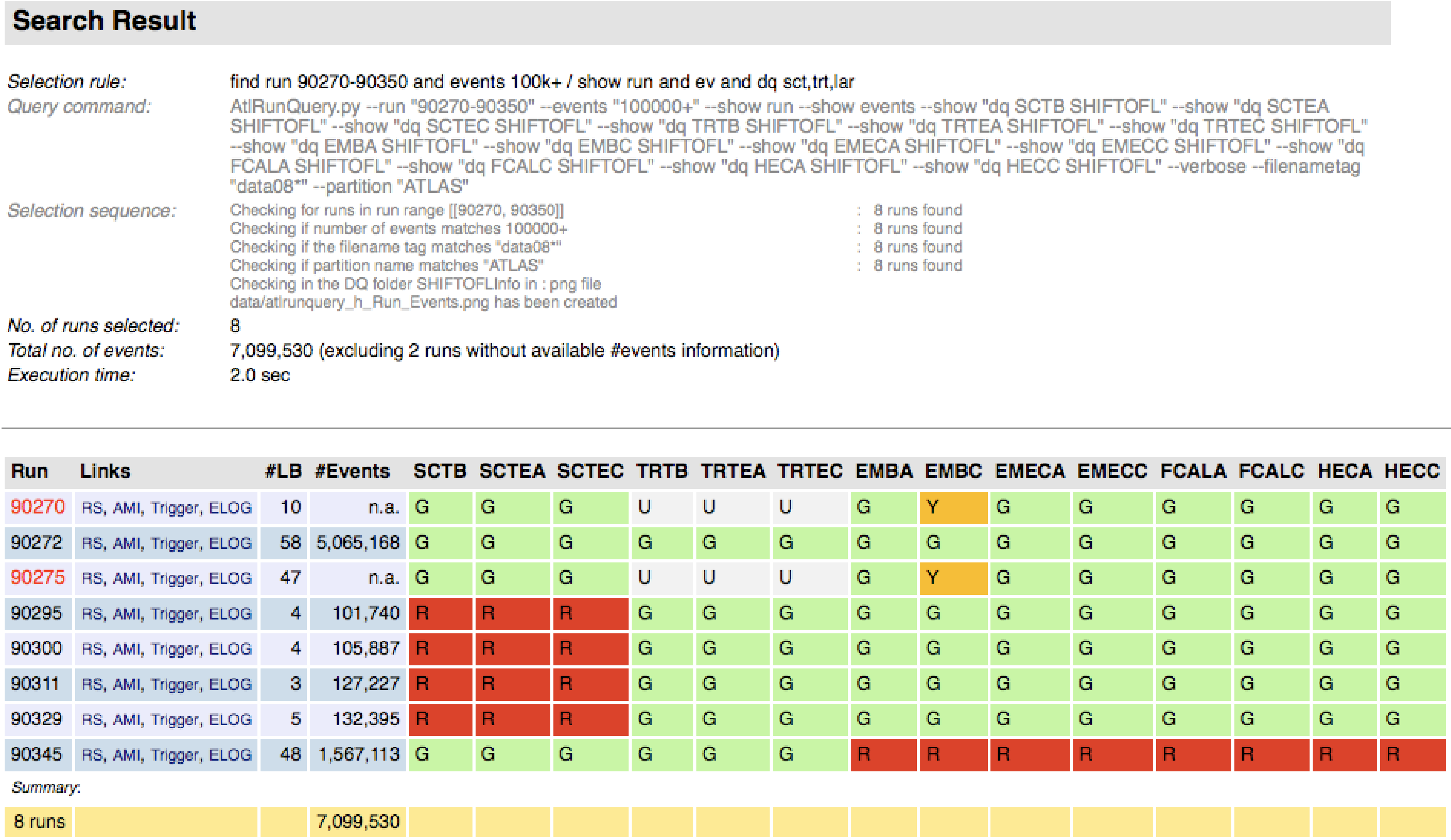
- Lowest-level quality flags exist (detector status)
  - Second- and third-tier flags are largely hypothetical at this point
- Volunteer to help determine how the second-tier flags are defined and set for your combined performance group!
- An XML exchange format (DTD) has been defined so that tools can share a representation of {run, lumi block} lists for a variety of purposes
  - Good Run Lists prominent among them
- GoodRunsList package now in CVS
  - Look for functionality in Release 15.3.0

De gustibus…

- What if your selection is no one's standard Good Run List?
- Imagine that you want to query detector status flags and run information directly to build your own list of runs
  - At least 100,000 events, (any) tile and (any) SCT active, …, toroid on, … and so on
- There are a variety of integrated tools in the works (and one could query the COOL folders that contain these data more or less directly)
- The most popular current tool is a web interface
  - See http://atlas.run_query.cern.ch

A tool you should try: http://atlas-runquery.cern.ch/

- Example: Run search with DQ flags



Search Result

Selection rule: find run 90270-90350 and events 100k+ / show run and ev and dq sct,trt,lar

Query command: AtlRunQuery.py --run "90270-90350" --events "100000+" --show run --show events --show "dq SCTB SHIFTOFL" --show "dq SCTEA SHIFTOFL" --show "dq SCTEC SHIFTOFL" --show "dq TRTB SHIFTOFL" --show "dq TRTEA SHIFTOFL" --show "dq TRTEC SHIFTOFL" --show "dq EMBA SHIFTOFL" --show "dq EMBC SHIFTOFL" --show "dq EMECA SHIFTOFL" --show "dq EMECC SHIFTOFL" --show "dq FCALA SHIFTOFL" --show "dq FCALC SHIFTOFL" --show "dq HECA SHIFTOFL" --show "dq HECC SHIFTOFL" --verbose --filenametag "data08*" --partition "ATLAS"

Selection sequence:

```
Checking for runs in run range [[90270, 90350]]        :  8 runs found
Checking if number of events matches 100000+           :  8 runs found
Checking if the filename tag matches "data08*"         :  8 runs found
Checking if partition name matches "ATLAS"             :  8 runs found
Checking in the DQ folder SHIFTOFLInfo in : png file
data/atlrunquery_h_Run_Events.png has been created
```

No. of runs selected: 8

Total no. of events: 7,099,530 (excluding 2 runs without available #events information)

Execution time: 2.0 sec

Run	Links	#LB	#Events	SCTB	SCTEA	SCTEC	TRTB	TRTEA	TRTEC	EMBA	EMBC	EMECA	EMECC	FCALA	FCALC	HECA	HECC
90270	RS, AMI, Trigger, ELOG	10	n.a.	G	G	G	U	U	U	G	Y	G	G	G	G	G	G
90272	RS, AMI, Trigger, ELOG	58	5,065,168	G	G	G	G	G	G	G	G	G	G	G	G	G	G
90275	RS, AMI, Trigger, ELOG	47	n.a.	G	G	G	U	U	U	G	Y	G	G	G	G	G	G
90295	RS, AMI, Trigger, ELOG	4	101,740	R	R	R	G	G	G	G	G	G	G	G	G	G	G
90300	RS, AMI, Trigger, ELOG	4	105,887	R	R	R	G	G	G	G	G	G	G	G	G	G	G
90311	RS, AMI, Trigger, ELOG	3	127,227	R	R	R	G	G	G	G	G	G	G	G	G	G	G
90329	RS, AMI, Trigger, ELOG	5	132,395	R	R	R	G	G	G	G	G	G	G	G	G	G	G
90345	RS, AMI, Trigger, ELOG	48	1,567,113	G	G	G	G	G	G	R	R	R	R	R	R	R	R
Summary:																	
8 runs			7,099,530														

Understanding the data you have

- You grabbed one file (dq2-get ..) of a dataset that you plan to analyze
  - For testing your code, getting your job options right, …
- What can you discover about the file, its contents, its provenance, …?
- Note that for most kinds of metadata, the answers for this file will be the same as for every other file in this dataset

- Desiderata for metadata and supporting infrastructure:
  - If you *have* a file, you shouldn’t need to consult a remote database to figure out what’s in it
  - If you *do not have* a file, a database should be able to help you determine what you are thereby missing

In-file metadata

- Event files contain (useful) information in addition to the events themselves
- In-file metadata is used for several purposes, including
  - Provenance tracking for eventual cross-section calculation
  - Cache for frequently-accessed non-event data
  - Sufficient information about processing and content to allow data-driven configuration of reader jobs
    - *Job options settings less error prone*
- Some command-line metadata utilities:
  - dumpFileMetadata.py: lists (some of) the in-file metadata
  - dumpVersionTags.py: lists conditions/calibrations/alignment/geometry tags used in producing this sample
  - checkFile.py: lists file contents by data type, size information, etc.

Example: dumpFileMetadata.py

```
[lxplus211] /user/m/malond>dumpFileMetaData.py -f
    /afs/cern.ch/user/i/ivukotic/public/theoutput.AOD.pool.root
No key given: dumping all file meta data
open file /afs/cern.ch/user/i/ivukotic/public/theoutput.AOD.pool.root
Warning: Cannot find transient class for EventStreamInfo_p2
List of meta data objects in file: (type : key )
  ==> To dump any single container: use '-k <key>'
  ==> To dump all: use '-d'
Type: LumiBlockCollection key: IncompleteLumiBlocks
Type: IOVMetaDataContainer key: _Digitization_Parameters
Type: IOVMetaDataContainer key: _GLOBAL_DETSTATUS_LBSUMM
Type: IOVMetaDataContainer key: _Simulation_Parameters
Type: IOVMetaDataContainer key: _TRIGGER_HLT_HltConfigKeys
Type: IOVMetaDataContainer key: _TRIGGER_HLT_Menu
Type: IOVMetaDataContainer key: _TRIGGER_LVL1_Lvl1ConfigKey
Type: IOVMetaDataContainer key: _TRIGGER_LVL1_Menu
Type: IOVMetaDataContainer key: _TRIGGER_LVL1_Prescales
Type: IOVMetaDataContainer key: _TagInfo
```

Example: dumpVersionTags.py

```
dumpVersionTags.py -f /afs/cern.ch/user/i/ivukotic/public/theoutput.AOD.pool.root
open file /afs/cern.ch/user/i/ivukotic/public/theoutput.AOD.pool.root
Warning: Cannot find transient class for EventStreamInfo_p2
---------------------------------------------
IOV Range: since [r,l]: [5200,0] until [r,l]: [5201,0]

tags: (tag name - value):
---------------------------------------------

/CALO/CaloSwClusterCorrections/calhits          'CaloSwClusterCorrections.calhits-v5'
/CALO/CaloSwClusterCorrections/etamod            'CaloSwClusterCorrections.etamod-v4'
/CALO/CaloSwClusterCorrections/etaoff           'CaloSwClusterCorrections.etaoff-v4_1'
/CALO/CaloSwClusterCorrections/gap              'CaloSwClusterCorrections.gap-v4'
...
/CALO/HadCalibration/CaloOutOfCluster            'CaloHadOOCCorr-CSC05-BERT'
/CALO/HadCalibration/CaloOutOfClusterPi0          'CaloHadOOCCorrPi0-CSC05-BERT'
/CALO/HadCalibration/H1ClusterCellWeights         'CaloH1CellWeights-CSC05-BERT'
...
/GLOBAL/BField/Map                          'BFieldMap-000'
...
/GLOBAL/BTagCalib/JetFitter                   'BTagCalib-03-00'
/GLOBAL/BTagCalib/JetProb                      'BTagCalib-03-00'
…
```

Example: dumpVersionTags.py (continued)

```
...
/LAR/ElecCalibMC/Pedestal                 'LARElecCalibMC-CSC02-J-QGSP_BERT'
/LAR/ElecCalibMC/Ramp                     'LARElecCalibMC-CSC02-J-QGSP_BERT'
/LAR/ElecCalibMC/Shape                    'LARElecCalibMC-CSC02-J-QGSP_BERT'
...
/TRIGGER/LVL1/Menu                        'HEAD'
/TRIGGER/LVL1/Prescales                    'HEAD'
/TRIGGER/LVL1/Thresholds                    'HEAD'
...
AtlasRelease                         'AtlasOffline-rel_0'
GeoAtlas                            'ATLAS-GEO-02-01-00'
IOVDbGlobalTag                          'OFLCOND-SIM-00-00-00'
MDT_support                          'MDT Big Wheel'
TGC_support                          'TGC Big Wheel'
```

Example: checkFile.py

```
Size:     6431.460 kb
Nbr Events: 12
================================================================================
    Mem Size       Disk Size        Size/Evt      MissZip/Mem  items  (X) Container Name (X=Tree|Branch)
================================================================================
    283.717 kb      14.049 kb       1.171 kb        0.000       12  (T) DataHeader
--------------------------------------------------------------------------------
      3.589 kb       0.244 kb       0.020 kb       0.345      12  (B) CaloCompactCellContainer_HLT_TrigT2CaloEgammaCells
      2.459 kb       0.333 kb       0.028 kb       0.446      12  (B) MissingET_p2_MET_Muon
      6.730 kb       0.390 kb       0.032 kb       0.381       1  (B) LumiBlockCollection_p1_IncompleteLumiBlocks
      9.137 kb       0.662 kb       0.055 kb       0.353      12  (B) egDetailContainer_p2_AtlfastIsoPhoShowerContainer
      9.194 kb       0.676 kb       0.056 kb       0.351      12  (B) egDetailContainer_p2_AtlfastIsoEleShowerContainer
      3.272 kb       0.719 kb       0.060 kb       0.359      12  (B) MuonSpShowerContainer_p1_MuonSpShowers
     10.793 kb       0.728 kb       0.061 kb       0.351      12  (B) RingerRingsContainer_tlp1_HLT_TrigCaloRinger
     48.794 kb       3.410 kb       0.284 kb       0.311      12  (B) CombinedMuonFeatureContainer_tlp1_HLT
     59.794 kb       3.460 kb       0.288 kb       0.591       1  (B) IOVMetaDataContainer_p1__TRIGGER_HLT_HltConfigKeys
    598.009 kb     108.280 kb       9.023 kb       0.057      12  (B) CaloClusterContainer_p6_HLT_TrigCaloClusterMaker
    368.636 kb     112.332 kb       9.361 kb       0.006      12  (B) HLT::HLTResult_p1_HLTResult_EF
   1433.260 kb     260.232 kb      21.686 kb       0.085      12  (B) Trk::TrackCollection_tlp3_Tracks
    508.399 kb     262.991 kb      21.916 kb       0.068      12  (B) CaloClusterContainer_p6_CaloCalTopoCluster
   1227.476 kb     393.227 kb      32.769 kb       0.072      12  (B) Rec::TrackParticleContainer_tlp1_TrackParticleCandidate
   1521.469 kb     550.463 kb      45.872 kb       0.016      12  (B) McEventCollection_p4_GEN_AOD
================================================================================
  33113.798 kb    4913.501 kb     409.458 kb       0.000      12  TOTAL (POOL containers)
================================================================================
```

New in 15.2.0: dump-athfile.py (see more in output file)

```
Py:AthFile          INFO ::::: summary :::::
 - file name      : /afs/cern.ch/user/i/ivukotic/public/theoutput.AOD.pool.root
 - file type      : pool
 - nentries       : 12
 - run number     : [5200L]
 - run type       : ['N/A']
 - evt number     : [30002L]
 - evt type       : ('IS_SIMULATION', 'IS_ATLAS', 'IS_PHYSICS')
 - lumi block     : [0L]
 - beam energy    : ['N/A']
 - beam type      : ['N/A']
 - stream tags    : [{'obeys_lbk': True, 'stream_type': 'calibration', 'stream_name': 'IDTracks'}, {'obeys_lbk': True, 'stream_type':
     'calibration', 'stream_name': 'LAr'}, {'obeys_lbk': True, 'stream_type': 'physics', 'stream_name': 'egamma'}, {'obeys_lbk': True,
     'stream_type': 'physics', 'stream_name': 'jetTauEtmiss'}, {'obeys_lbk': True, 'stream_type': 'physics', 'stream_name':
     'minbias'}]
 - stream names   : ['StreamAOD']
 - geometry       : ATLAS-GEO-02-01-00
 - conditions tag : OFLCOND-SIM-00-00-00
 - meta data      : ['/TRIGGER/LVL1/Lvl1ConfigKey', '/Digitization/Parameters', '/TagInfo', '/TRIGGER/LVL1/Prescales',
     '/GLOBAL/DETSTATUS/LBSUMM', '/TRIGGER/HLT/HltConfigKeys', '/TRIGGER/HLT/Menu', '/TRIGGER/LVL1/Menu
     '/Simulation/Parameters']
Py:AthFile          INFO :::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
Py:AthFile          INFO saving report into [athfile-infos.ascii]...
```

LumiCalc.py (using a run from FDR2c)

```
lxplus246] /user/m/malond>LumiCalc.py  --trigger=EF_e25_tight --r=52280  --mc
Data source lookup using /afs/cern.ch/atlas/software/builds/AtlasCore/15.2.0/InstallArea/XML/AtlasAuthentication/dblookup.xml file
Beginning calculation for Run 52280 LB [0-4294967294]
LumiB  L1-Acc  L2-Acc  L3-Acc   L1-pre   L2-pre   L3-pre LiveTime  IntL/nb-1
Rng-T 1345102   54991   23087                                3600.00      360.0
>== Trigger  : EF_e25_tight
IntL (nb^-1) :       360.00
L1/2/3 accept:      1345102    54991    23087
Livetime     :    3599.9999
Good LBs     :          120
BadStatus LBs:            0
```

… or use your file instead of specifying a run number, and see what happens(!)

“Where ignorance is bliss…

- You don’t care about every setting of the reconstruction job that created your sample
  - Unless you find a problem, or something you do not understand
- When you process a 200-file dataset and aggregate the output, you don’t care about the specifics of input file 73
  - Unless that job dies
- There are many kinds of metadata out there that you will not care about
  - Unless …
  - …but you should hope that *someone* is maintaining the metadata, someone who does not believe that “…’tis folly to be wise.”

Concluding musings

Where is the wisdom we have lost in knowledge?
Where is the knowledge we have lost in information? (T.S. Eliot)

Where is the information we have lost in data?
Where is the data we have lost? (D. M. Malon)

Perhaps good metadata will prevent data from “disappearing,” i.e., from being unusable for analysis purposes,

- or at least help us account for any losses correctly in our analyses.