ATLAS Metadata Status
(and example of metadata use)

- *Metadata:*
  - *Data which are not part of the event data but are needed in the analysis (e.g. detector status data measured asynchronously, trigger level 1 luminosity counters, ...)*
  - *Data which are part of the event data but are duplicated to a place where they are accessible quickly (e.g. event TAG database)*

- *Metadata Task Force had been established to assess available and required metadata sources - report available since a while*

- *Implementation work ongoing*
  - *Storage of online metadata sources from TDAQ*
  - *Event TAGs - database and/or file based*
  - *Luminosity data*
  - *Detector status & other Data Quality data*
  - *Other metadata (AMI)*
  - *COOL data preprocessing (between end of run and start of reconstruction)*

Metadata granularity
(validity time spans, units of access)

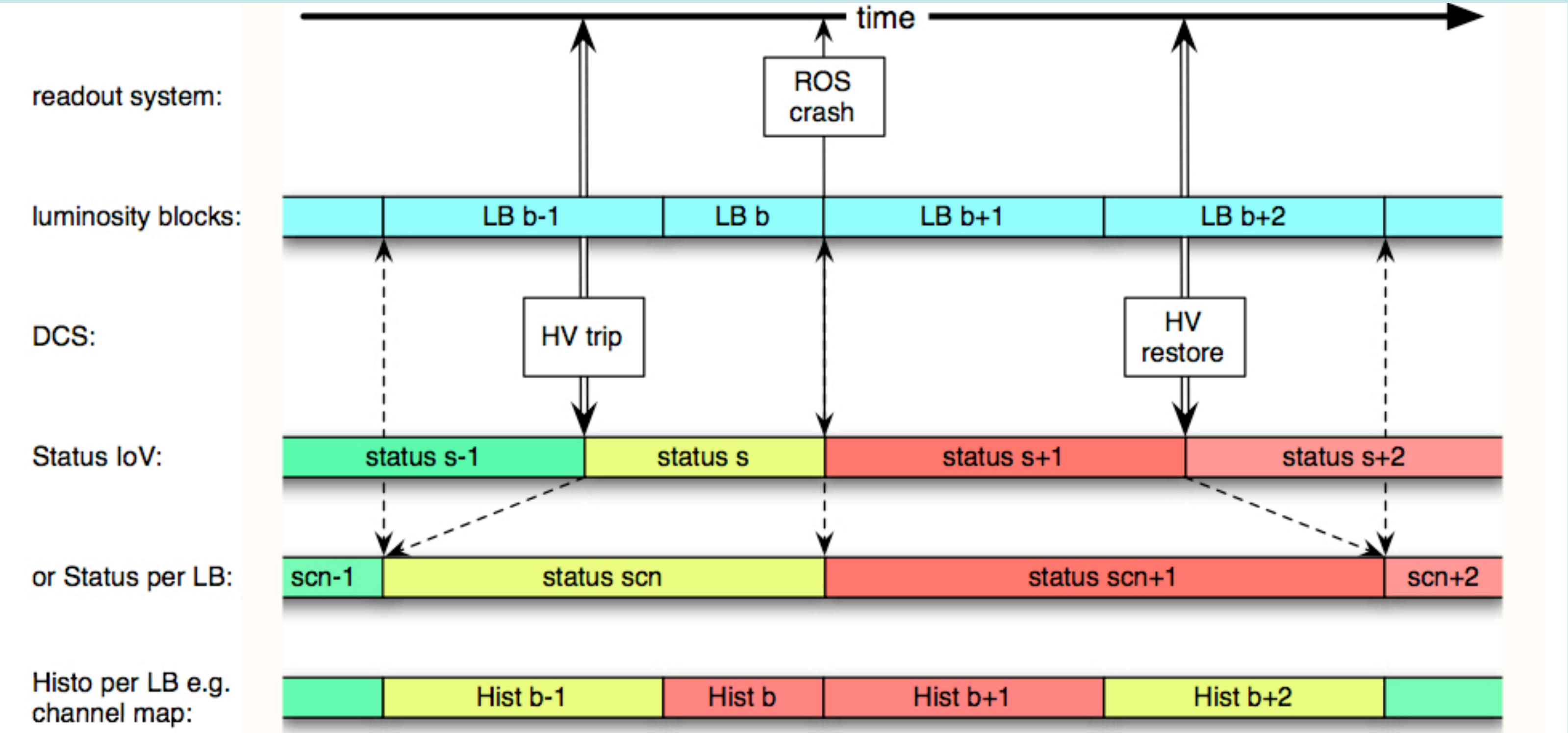
- *Event level*
  - *normally a small amount of classification information derived from the event data itself, such as trigger decision. Provides fast event selection without the need to access the event data files.*
- *Luminosity Block (LB) level*
  - *typically duration of O(min), but also represent an interval of validity for certain types of run status information which can be assessed synchronously at the event level during TDAQ readout.*
- *Interval of Stability (IOS) level*
  - *can be as short as a LB, but hopefully span many LBs. Reflect a stable set of conditions, like detector status, which influence data quality (DQ) and hence the ability to use the data for certain analyses.*
- *Run level*
  - *these include the composition of the TDAQ run partition, start/stop time...*
- *Fill level*
  - *LHC fill data (beam momenta etc), plus can provide quality data such as defects in bunch population.*
- *File level*
  - *a RAW file typically spans a LB, but there are several raw files per LB according to the number of Sub Farm Outputs (SFO) Files later in the reconstruction chain contain several LBs.*
  - *Important bookkeeping task - e.g. lumi assessment despite file loss (recommendations by Lumi TF)*
- *Stream level*
  - *(online) list of files, #events, trigger menus associated with data stream, LB bound or not.*
- *Dataset level*
  - *datasets are the unit of shipment of data in the Distributed Data Management (DDM).*

Primary sources of metadata (from Metadata TF Report)

- Online sources
  - Run Control
  - Run number server
  - Input from shift operator
  - DAQ configuration DB (OKS)
  - Trigger configuration DB (L1 & HLT)
  - Central Trigger Processor (CTP)
  - Event counters in the Higher Level Triggers (L2, EF) and Subfarm Inputs and Outputs (SFI, SFO)
  - ATLAS Detector Control System (DCS) data
  - LHC machine data - mostly via the DCS
    - Extra data from beam pickups close (~40m) to ATLAS, using different acquisition path - per bunch (1/min): phase, intensity, width, plus (1/run) complete pickup trace of one LHC revolution
    - LHC fill number to be stored alongside beam momenta, etc.
  - Monitoring applications
  - Event Filter farm Output (SFO) applications

Primary sources of metadata (2)

- Preprocessing of DCS and conditions data
  - before 1st pass reconstruction at Tier0



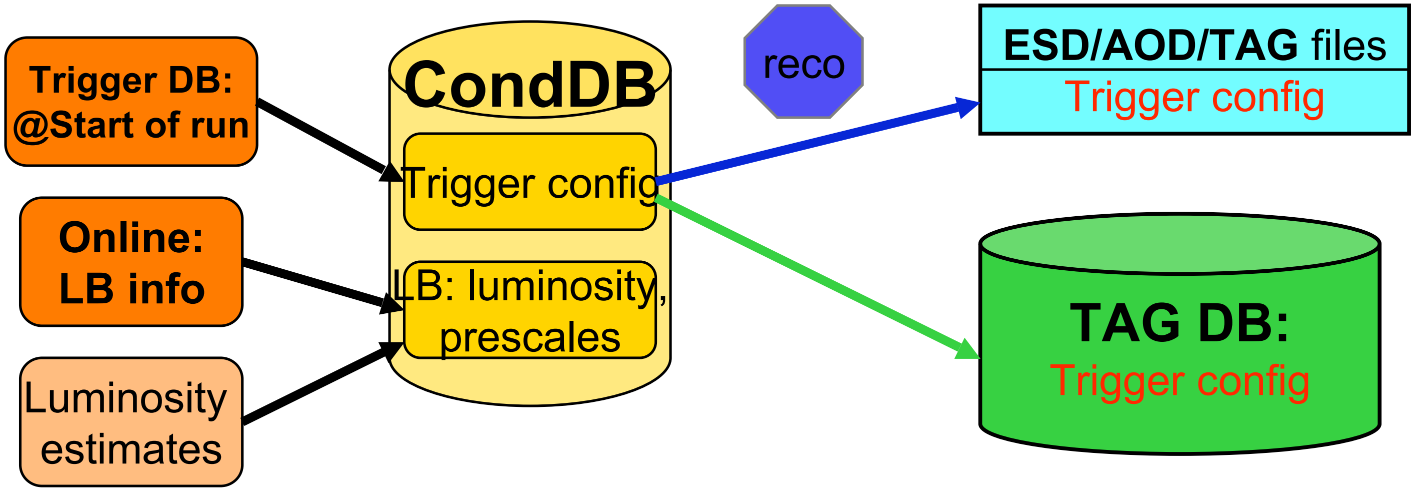
- Offline sources
  - Production database
  - DDM
  - Dataset Manager
  - Input from offline shift operator/user

Integrated luminosity calculation basics
(Richard Hawkings)

- For calculating the integrated luminosity in a physics data sample, base on concept of luminosity blocks (LB)
  - Always process every event from each LB which passes selection cuts
    - Keep track of **all** LBs processed, even if no events pass selection cuts
  - Calculate the integrated luminosity for each LB:

    $\int L=(L.\Delta t.f\ )\ /\ (p_1 p_2 p_3)$ … and sum over all LB
    - L is the instantaneous luminosity estimate for the LB (from whatever source)
    - Δt is the duration of the LB and f the livetime fraction
    - p_1, p_2 and p_3 are the trigger prescale factors at level 1,2 and EF
  - Note the integrated luminsoity estimate is a function of the trigger chain used
    - Both livetime fraction f and the prescales depend on the choice of trigger chain
- Ingredients:
  - Per-run information: trigger configuration (including HLT prescales)
  - Per-LB information: instantaneous L, LB duration, livetime frac, LVL1 prescales
- To also calculate ‘raw’ trigger cross-sections: σ=N/L
  - .. need counts of number of L1,L2, L3 accepts in each LB

Proposed dataflow for trigger config/luminosity


Trigger DB: @Start of run

Online: LB info

Luminosity estimates

CondDB

Trigger config

LB: luminosity, prescales

reco

ESD/AOD/TAG files

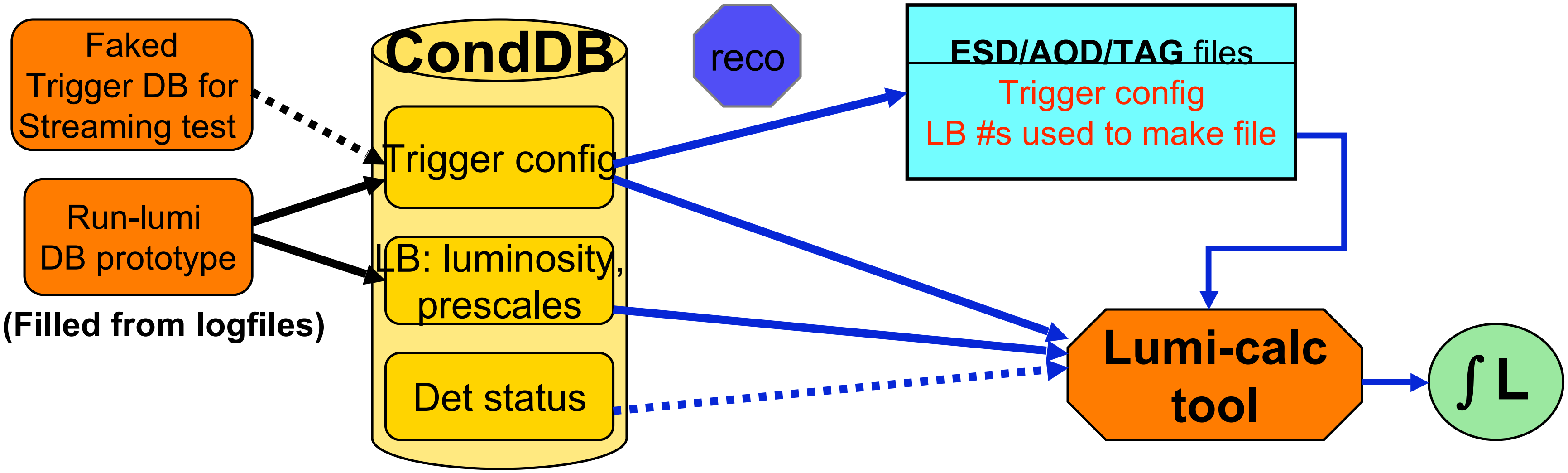
Trigger config

TAG DB:

Trigger config

- Trigger info needed: ‘accept’ bits in event data + mapping of bit assignment to names
  - Latter stored in Trigger configuration database, copied to COOL at start of each run, copied into ESD/AOD/TAG files during reconstruction, copied into relational TAGs during upload
  - Storage in event files will file-level metadata (one copy per file, not per event)
- Both event files and TAG DB will have enough information to decode trigger decision without use of trigger or conditions DB
  - Users work in terms of ‘e25i_v14’ rather than ‘bit 76’
- Luminosity block information (L1 prescales, LB start/end, trigger chain livetime) written by online into COOL each LB
  - Could also be copied to TAG if needed (not strictly necessary for TAG queries?)

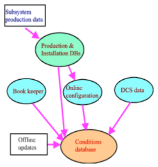
Streaming test data flow



- Fill COOL using trigger DB tools from streaming test trigger configuration
  - **Done**, using information from prototype run-lumi DB and 'home-made' loading code
  - Easier than the 'official' path from XML -> TriggerConfDB -> COOL (release versions…)
- Fill COOL using run and LB-level information from streamtest run-lumi database
  - Run-LB prototype database is a temporary 'holding place' for this data
  - Done, with some caveats to be understood (assignment of level 1 trigger bits to be redone)
- Fill COOL with some detector status 'by hand'
  - e.g. bad event filter status where we have 'lost' raw data files
  - **Done,** but not yet used by lumi calculation tool

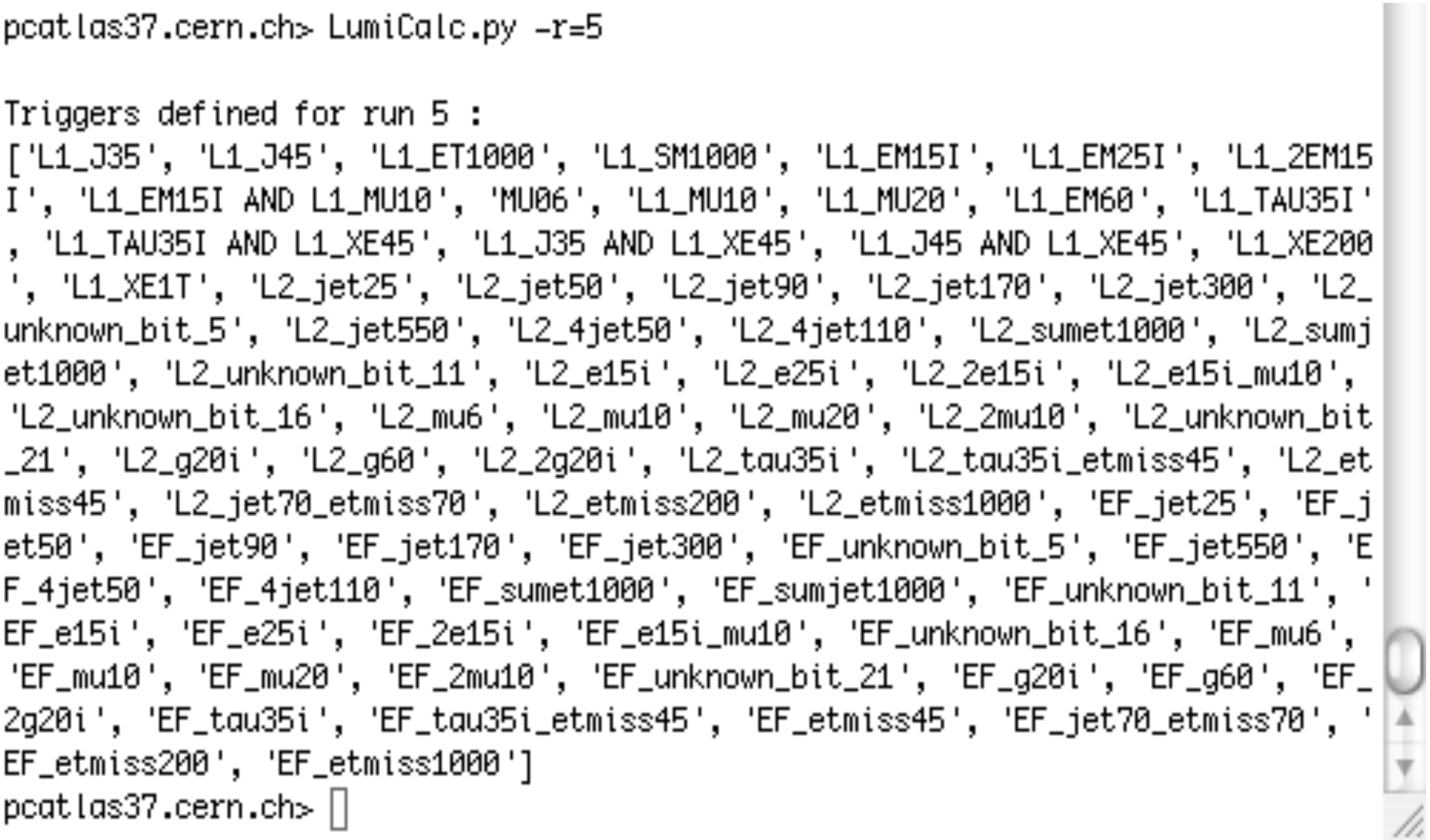
Data structures in the conditions database COOL

- Data uploaded to special database instance in production Oracle server
  - ATLAS_COOL_TRIGGER schema, STRMP200 database instance
    - Detector status information on ATLAS_COOL_GLOBAL, STRMP200 instance
- COOL folder structures:
  - /TRIGGER/LVl1/* and /TRIGGER/HLT/*: folders created by trigger DB tools, filled 'by hand' with special tool (would normally be filled online for each run/LB).
    - Trigger configuration: config keys, menus and relations, prescale factors
    - Some 'funnies' in the streaming test configuration: LVL2 triggers based on OR of two LVl1 bits (not allowed by the trigger hardware)
  - /TRIGGER/LUMI/* folders: LB information (start/end, luminosity, livetime) and folders to contain LVL1 and HLT trigger counters per luminosity block
    - Format was defined by RH, after some discussion with LVL1 and HLT people about what is expected/realistic
    - Consider it a 'prototype' for what we might expect online to deliver
- Command-line query tool LumiCalc.py written to interrogate COOL structures
  - Dump information on trigger configuration, calculate integrated luminosity
  - Examples follow of use of this tool on the STRMP200 database …



Luminosity query - example 1

- List all triggers defined in a particular run:



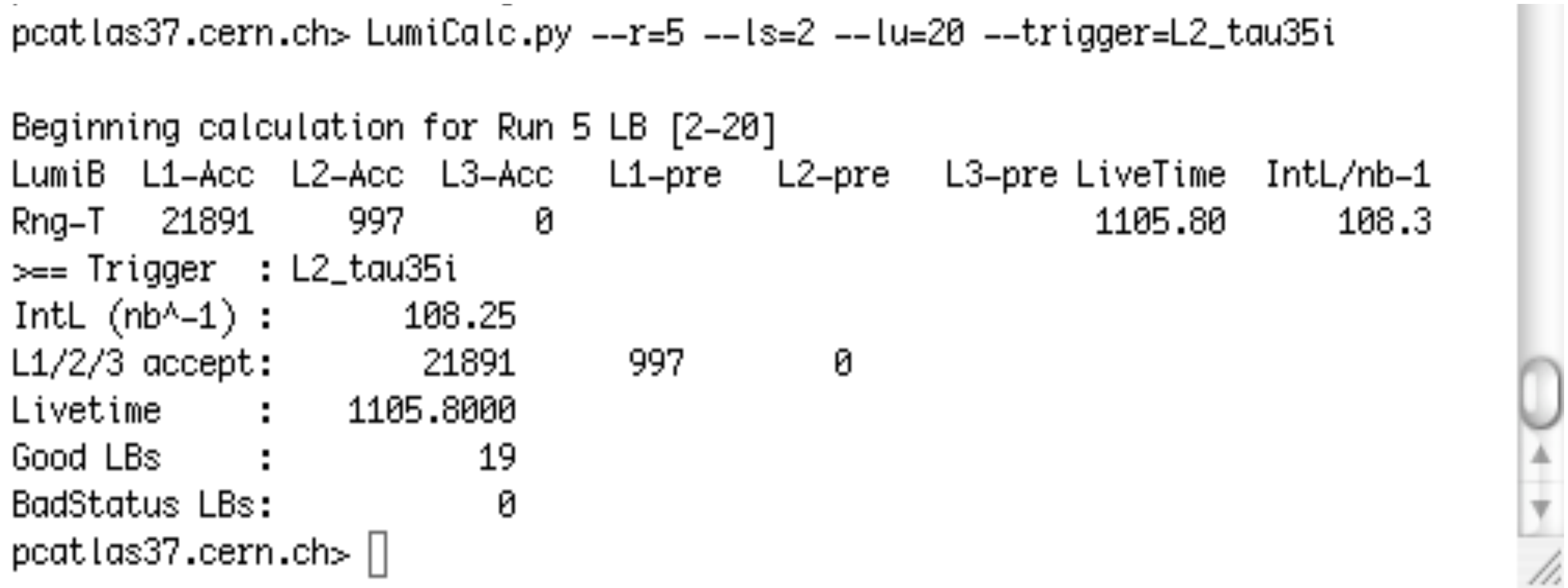
```
pcatlas37.cern.ch> LumiCalc.py -r=5

Triggers defined for run 5 :
['L1_J35', 'L1_J45', 'L1_ET1000', 'L1_SM1000', 'L1_EM15I', 'L1_EM25I', 'L1_2EM15
I', 'L1_EM15I AND L1_MU10', 'MU06', 'L1_MU10', 'L1_MU20', 'L1_EM60', 'L1_TAU35I'
, 'L1_TAU35I AND L1_XE45', 'L1_J35 AND L1_XE45', 'L1_J45 AND L1_XE45', 'L1_XE200
', 'L1_XE1T', 'L2_jet25', 'L2_jet50', 'L2_jet90', 'L2_jet170', 'L2_jet300', 'L2_
unknown_bit_5', 'L2_jet550', 'L2_4jet50', 'L2_4jet110', 'L2_sumet1000', 'L2_sumj
et1000', 'L2_unknown_bit_11', 'L2_e15i', 'L2_e25i', 'L2_2e15i', 'L2_e15i_mu10',
 'L2_unknown_bit_16', 'L2_mu6', 'L2_mu10', 'L2_mu20', 'L2_2mu10', 'L2_unknown_bit
_21', 'L2_g20i', 'L2_g60', 'L2_2g20i', 'L2_tau35i', 'L2_tau35i_etmiss45', 'L2_et
miss45', 'L2_jet70_etmiss70', 'L2_etmiss200', 'L2_etmiss1000', 'EF_jet25', 'EF_j
et50', 'EF_jet90', 'EF_jet170', 'EF_jet300', 'EF_unknown_bit_5', 'EF_jet550', 'E
F_4jet50', 'EF_4jet110', 'EF_sumet1000', 'EF_sumjet1000', 'EF_unknown_bit_11', '
EF_e15i', 'EF_e25i', 'EF_2e15i', 'EF_e15i_mu10', 'EF_unknown_bit_16', 'EF_mu6',
 'EF_mu10', 'EF_mu20', 'EF_2mu10', 'EF_unknown_bit_21', 'EF_g20i', 'EF_g60', 'EF_
2g20i', 'EF_tau35i', 'EF_tau35i_etmiss45', 'EF_etmiss45', 'EF_jet70_etmiss70', '
EF_etmiss200', 'EF_etmiss1000']
pcatlas37.cern.ch> 
```

  - Can also use --level=<n> to restrict query to triggers at a particular level

Luminosity query - example 2

- Calculate integrated luminosity seen by a particular trigger in a particular LB range

Here, trigger L2_tau35i, LBs 2-20 in run 5



```
pcatlas37.cern.ch> LumiCalc.py --r=5 --ls=2 --lu=20 --trigger=L2_tau35i

Beginning calculation for Run 5 LB [2-20]
LumiB  L1-Acc  L2-Acc  L3-Acc   L1-pre   L2-pre   L3-pre LiveTime  IntL/nb-1
Rng-T   21891     997       0                                1105.80     108.3
>== Trigger   : L2_tau35i
IntL (nb^-1) :       108.25
L1/2/3 accept:        21891      997        0
Livetime     :    1105.8000
Good LBs     :           19
BadStatus LBs:            0
pcatlas37.cern.ch> 
```

- Takes into account livetime, prescale factors (and ‘is trigger configured’?)
- Does not yet take into account detector status

Event TAGs
(David Malon et al.)

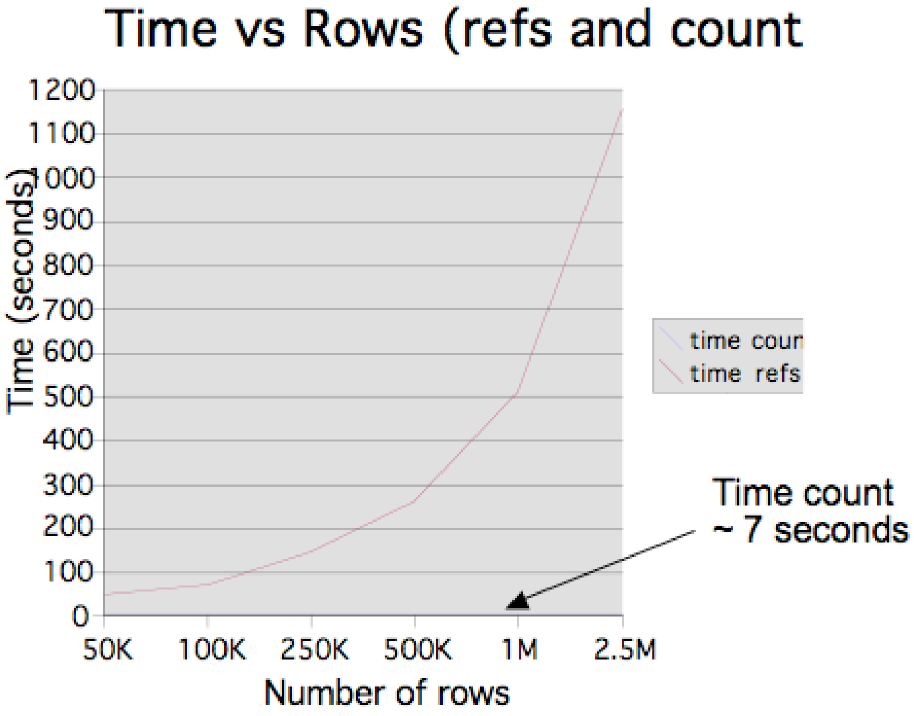
- *Small set of valiables to characterize an event for fast selection*
  - *~200 indexed variables; ~1 kB/event; ~1 TB/year*
- *TAGs produced as files during 1st-pass (and re-) reconstruction*
  - *Will be uploaded to a relational TAG database hosted at Tier0*
  - *Also uploaded to 1-few (volunteering) Tier1s*

    (data shipped there as TAG files, not via Oracle Streams)
  - *DB-based queries allow e.g. sorting of events by p_T etc.*
  - *TAGs could also be made available as files for ntuple-style usage*

    (PetaCache fast in-memory file system at SLAC as candidate)
- *Starting to gain experience now with TAGs produced in the steaming tests*
  - *~3 million evts used but only 300k trigger and go into TAG DB*
- *1 TB of faked TAGs loaded to DB for performance tests*
  - *Promising with one DB server instance, queries already optimized, and parallel querying to multiple servers would be possible*

Oracle scalability tests: preliminary indications

- Statistical queries (how many events satisfy this cut?) are fast: interactive response times
- Event selections show linear scalability
  - Not surprising in theory
  - "Horizontal" partitioning into blocks of runs, for example, allows parallelism
- Careful tuning is needed, but with such tuning, Oracle seems up to the challenge



- See talk of Luc Goossens et al. during March SW week (LMU Munich)

Metadata discussed at Physics Analysis Tools (PAT) Workshop at Bergen (last April)

- *Luminosity-block (LB) level metadata infrastructure*
- *Data Quality (DQ) information, metadata in the conditions database COOL*
- *Characterise metadata by few criteria and decide about locality - hopefully covering all use cases*
  - *Mutability, ubiquity, frequency*
- *In-file metadata infrastructure*
  - *Utilising multiple StoreGate stores - per-event, per-job, plus per-file, ...*
  - *New in release 13, including underlying machinery to support propagation of metadata from input to output*
- *Trigger-related metadata for analysis*
  - *Significant work underway to support decoding of trigger decisions offline from time-varying trigger configurations*
- *{Run, LB} provenance tracking*
  - *First use of in-file metadata infrastructure*

List of topmost metadata issues

- Analysis use cases
  - Anything fundamental overlooked?
- Complete the set of metadata sources
  - Anything fundamentally missing?
- Definition of complete set of metadata destinations
  - Databases: COOL, TAG, AMI, DDM (more required?)
  - Files: TAG files (technology?), replication of some non-event metadata in event files (data quality; what else?)
  - Guidelines to decide what metadata hosted at which level
- Cross-DB queries
  - E.g. Dataset related (AMI) together with Quality data (COOL) during dataset production
  - Navigation between metadata of different granularity
  - How to implement?
  - How to upgrade (new needs, schema changes)

Some references

- *Metadata Task Force Report - major comments received and changes made* https://edms.cern.ch/cedar/plsql/doc.info?cookie=6290895&document_id=833723
- *Also see Giovanna Lehmann's talk from last ATLAS overview week* http://indico.cern.ch/materialDisplay.py?contribId=2&amp;sessionId=20&amp;materialId=slides&amp;confId=11266
- *And the TWiki* https://twiki.cern.ch/twiki/bin/view/Atlas/MetaDataImplementation
- *Formal dictionary of metadata entities being set up in AMI* https://atlastagcollector.in2p3.fr:8443/AMI/servlet/net.hep.atlas.Database.Bookkeeping.AMI.Servlet.Command?linkId=349