Use of MySQL in the ALICE data-acquisition system

sylvain.chapeland@cern.ch PH/AID/DA

Database Futures Workshop
6-7 June 2011
CERN

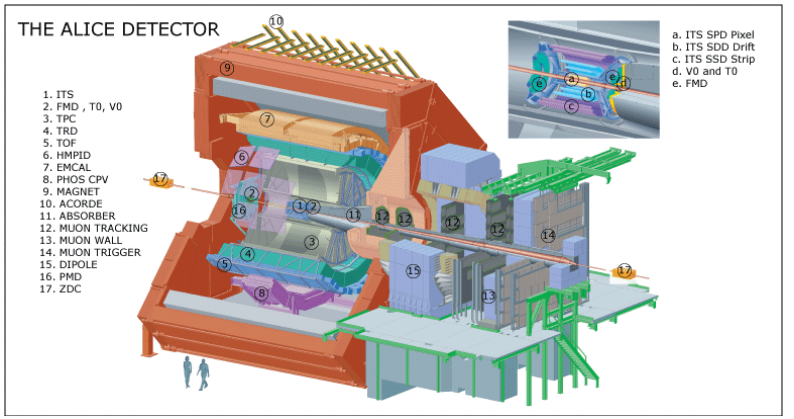
Outline

- Introduction
- The DAQ databases: description and access pattern
- Hardware and benchmarks
- Software: interfaces, features
- Conclusion

Introduction

- ALICE
  - 18 sub-detectors
  - 5 online systems

    ECS, DAQ, DCS, TRI, HLT



- DAQ in numbers

Readout bandwidth: 115 GB/s, plan up to 50GB/s used with HLT filter

Event Building bandwidth: 8 GB/s

In 2010: 1.2 GB/s in p-p up to 2.5 GB/s in Pb-Pb

Storage bandwidth performance:

4.5 GB/s writing, 2.5 GB/s reading and archiving to CASTOR

Amount of data recorded in 2010

Physics : 1.6 PB 1.8 x 10^9 events 1000 hours data taking

All : 4.7 PB 23 x 10^9 events 11000 hours data taking

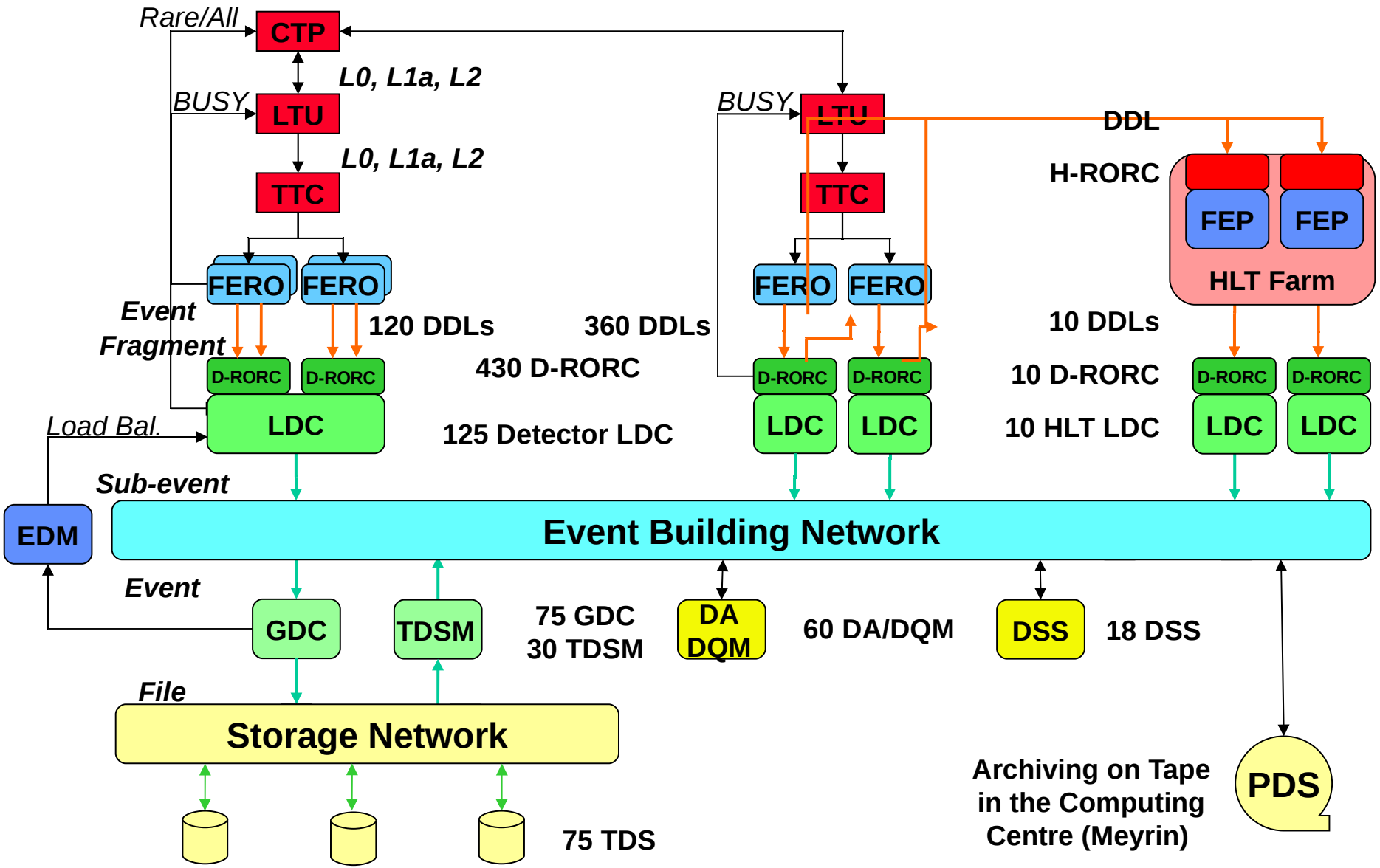
Facilities

460 Detector Data Links (DDL) in, 360 out (to HLT)

400 PCs

180 TB transient storage (soon 400TB)

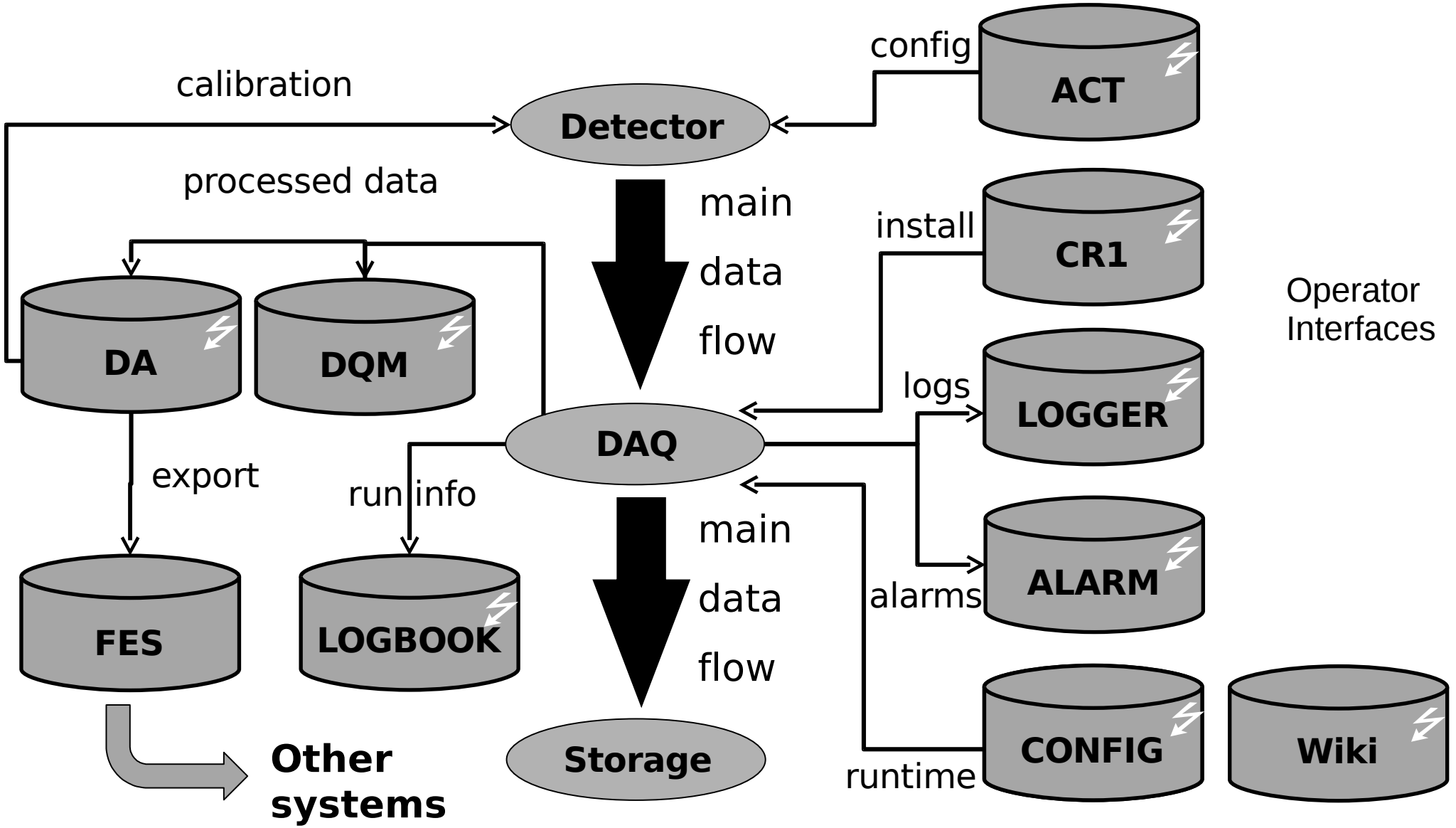
DAQ components



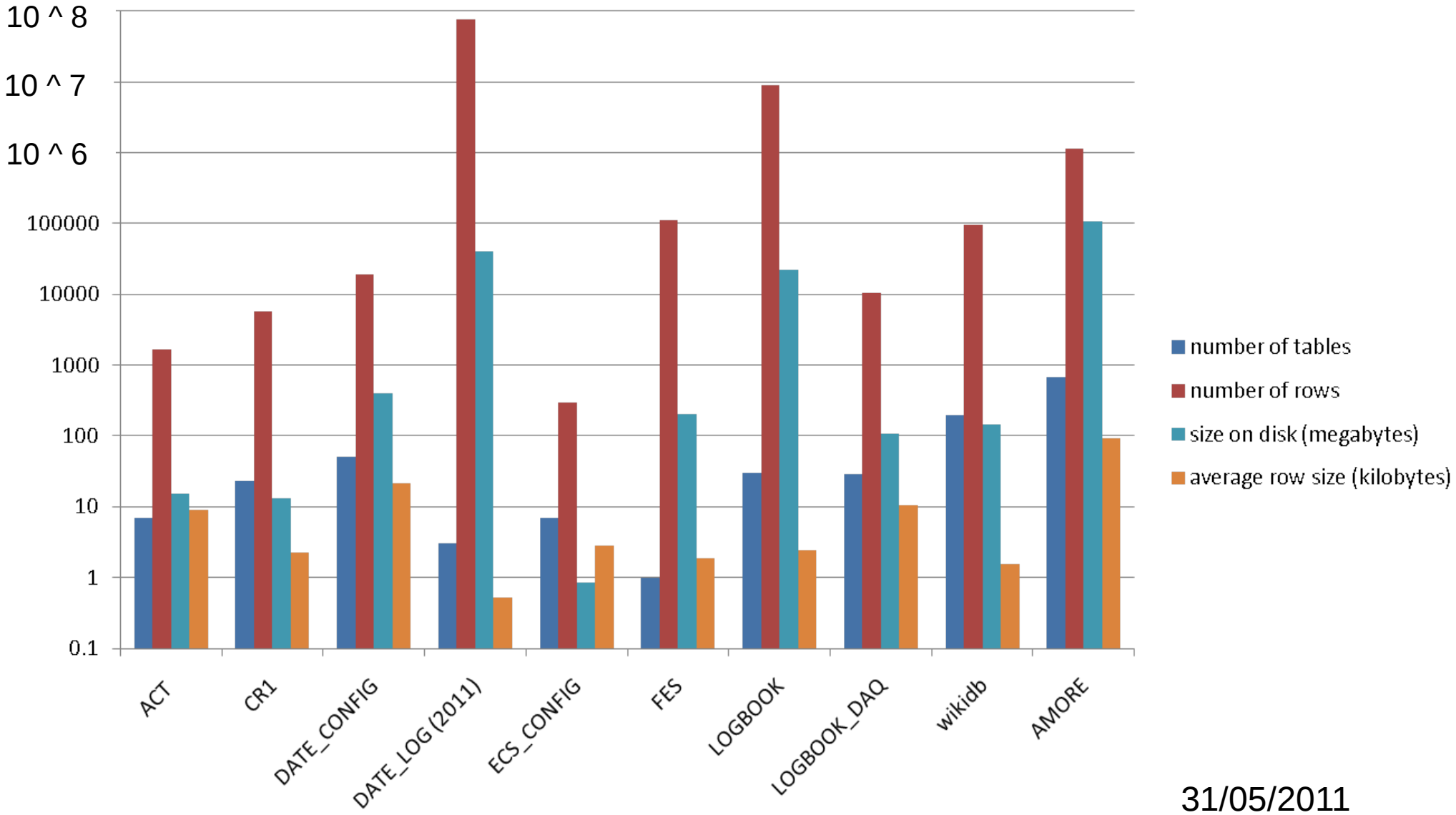
DAQ DB history

- Introduction of databases in 2004 to replace text-based configuration files
- MySQL selected based on
  - Performance
  - Lightweight installation for our multiple sites
  - Ease of use and maintenance
  - Know-how
- Extensive use in many other DAQ components
  - DB now indispensable to data taking
- Deployment
  - P2, lab, ~10 developers sites, ~20 user sites (development + production test beams)

DAQ databases overview



DB content



31/05/2011

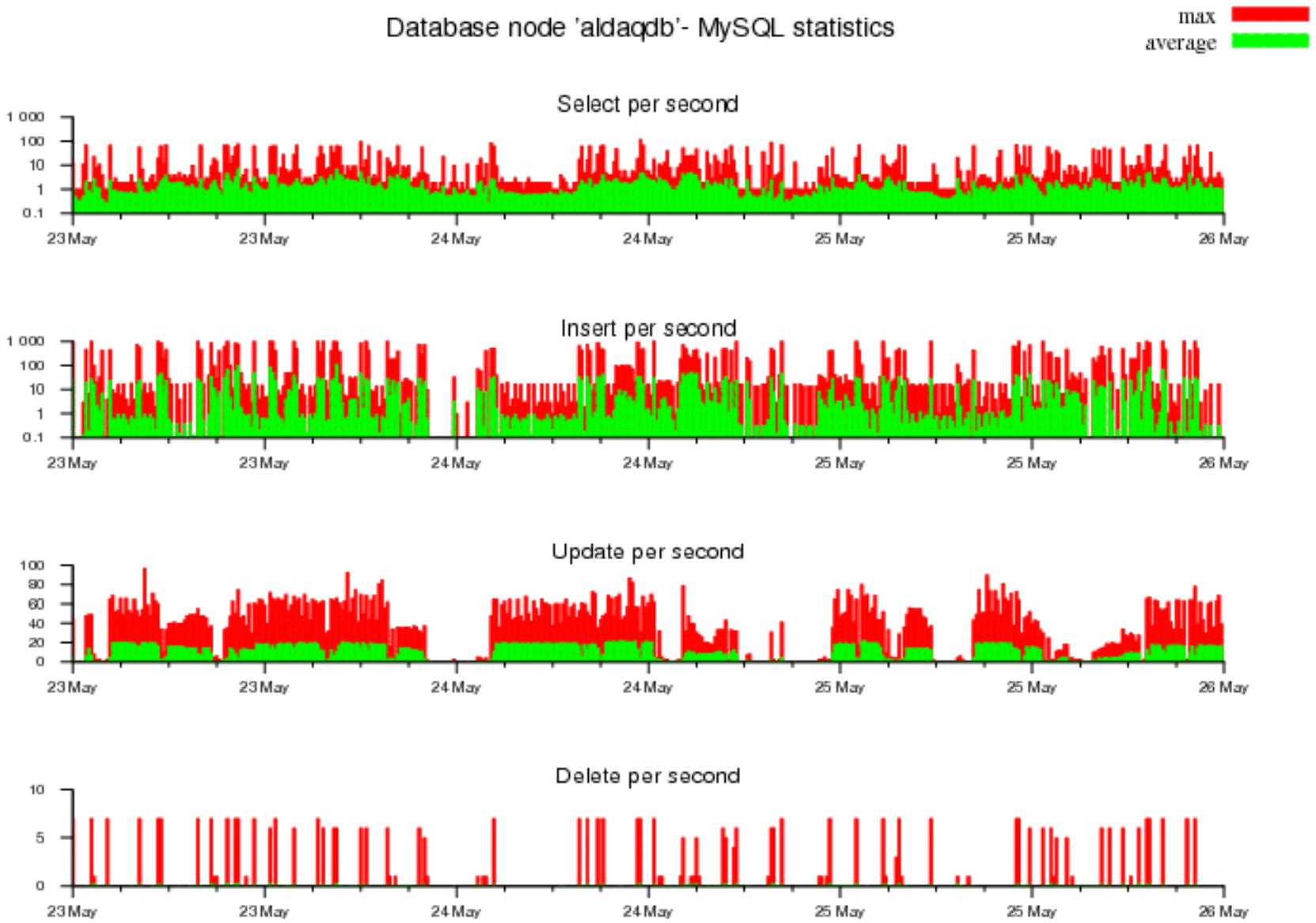
DB usage profiles

- CONFIG, CR1, ACT
  - Low data volumes, total data size is few MB, 20000 rows (with some rare larger entries)
  - Read peaks (installation, Start Of Run), 100s of concurrent clients
- LOGGER
  - Write intensive, single insert client
  - Peaks (Start/End Of Run), ~10000 inserts in few seconds
  - Full indexing for field search by interactive clients.
  - Millions of rows per week, ~300MB/day, archiving
  - Large query results
- DQM
  - Large data volume (objects in MegaBytes), total data > 100GB
  - Concurrent read/write
- LOGBOOK
  - Complex queries
  - Distributed insert/replace
  - Increasing size (now 22GB)

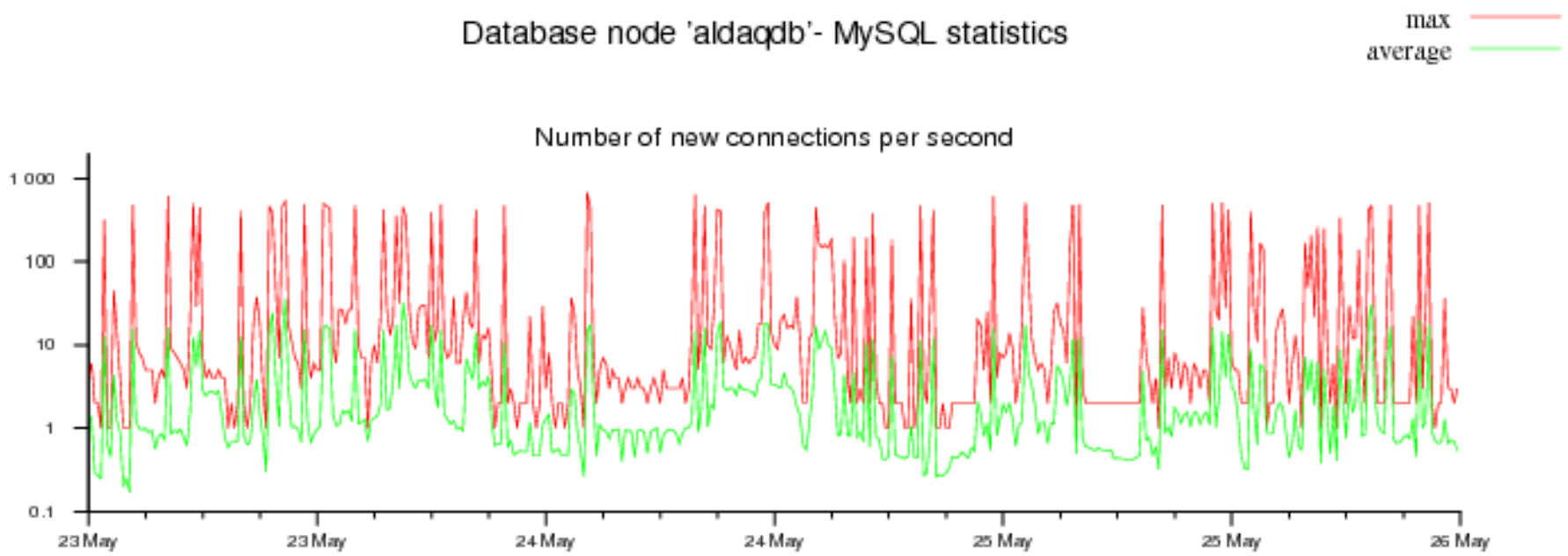
Server monitoring

- 1 server (aldaqdb) hosting all databases but DQM
- 1 server hosting only DQM
- Values presented:
  - Server variables sample time = 10s
  - Peak: max value per sample period
  - Average: average on 10 minutes

Queries

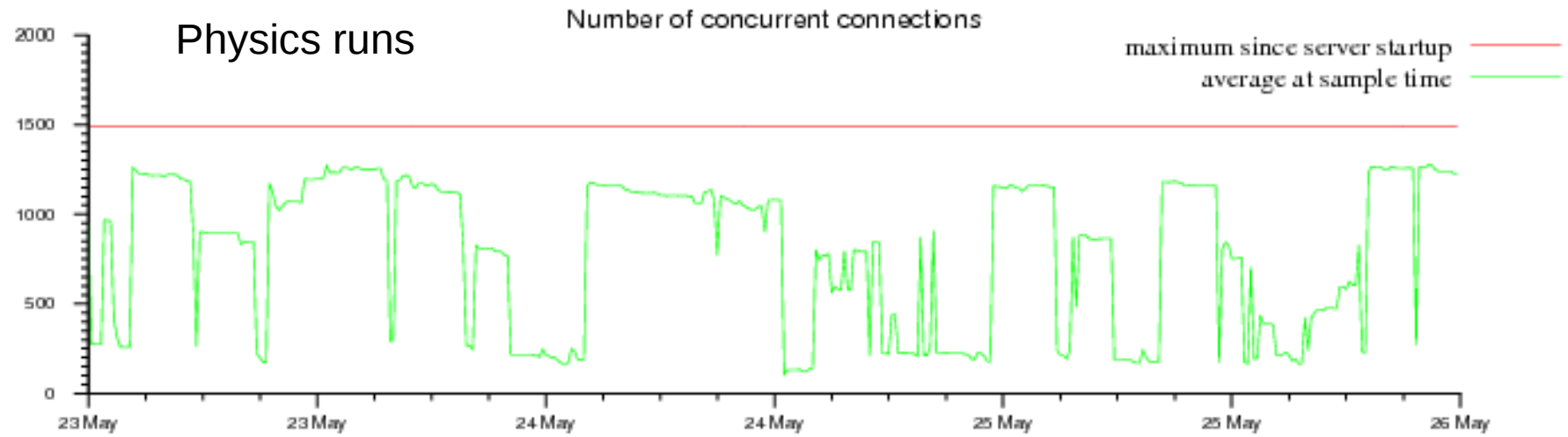

Database node 'aldaqdb'- MySQL statistics
max
average
Select per second
1 000
100
10
1
0.1
23 May
23 May
24 May
24 May
25 May
25 May
26 May
Insert per second
1 000
100
10
1
0.1
23 May
23 May
24 May
24 May
25 May
25 May
26 May
Update per second
100
80
60
40
20
0
23 May
23 May
24 May
24 May
25 May
25 May
26 May
Delete per second
10
5
0
23 May
23 May
24 May
24 May
25 May
25 May
26 May

Connections to server

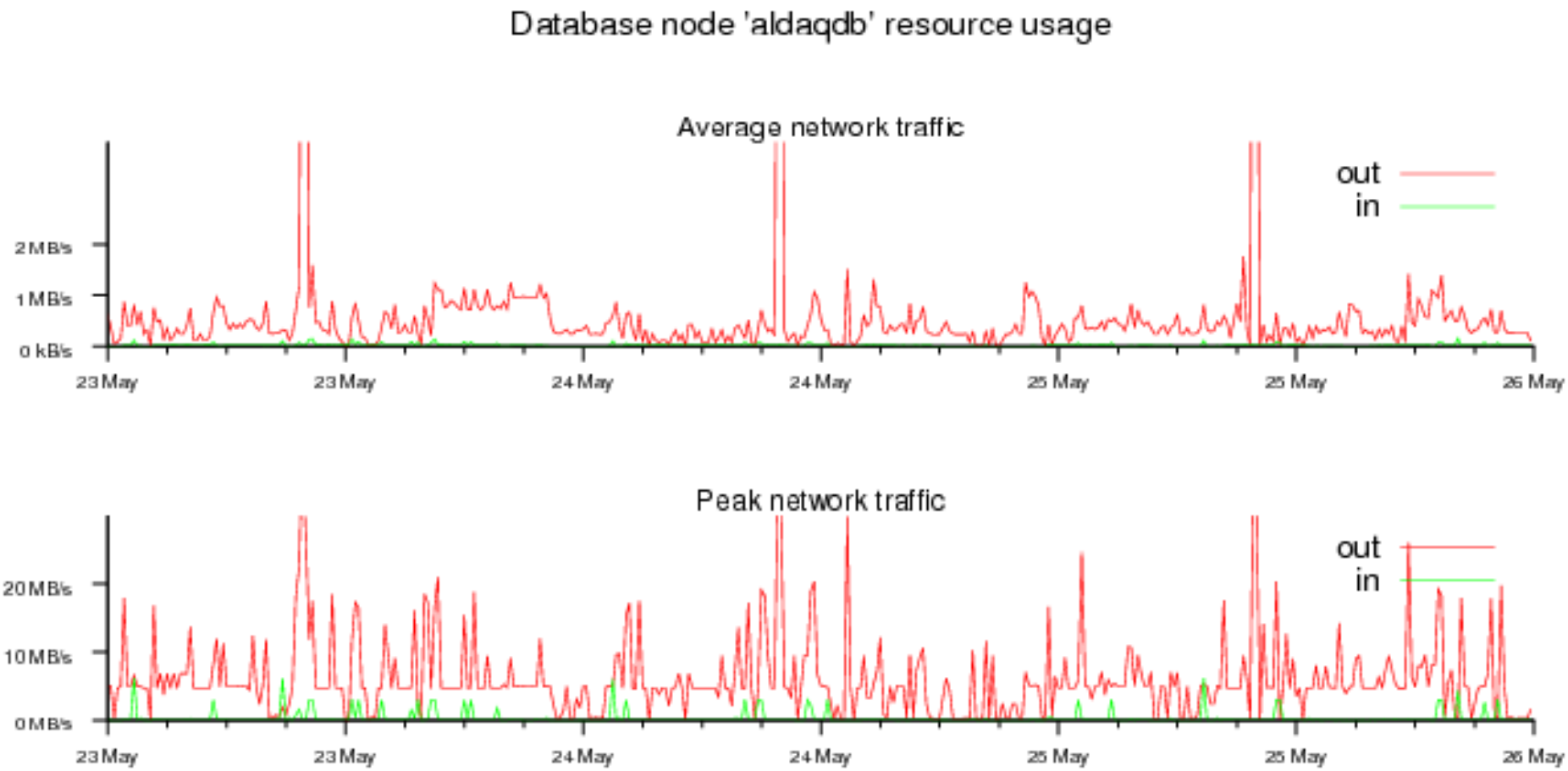


Start of run

Physics runs

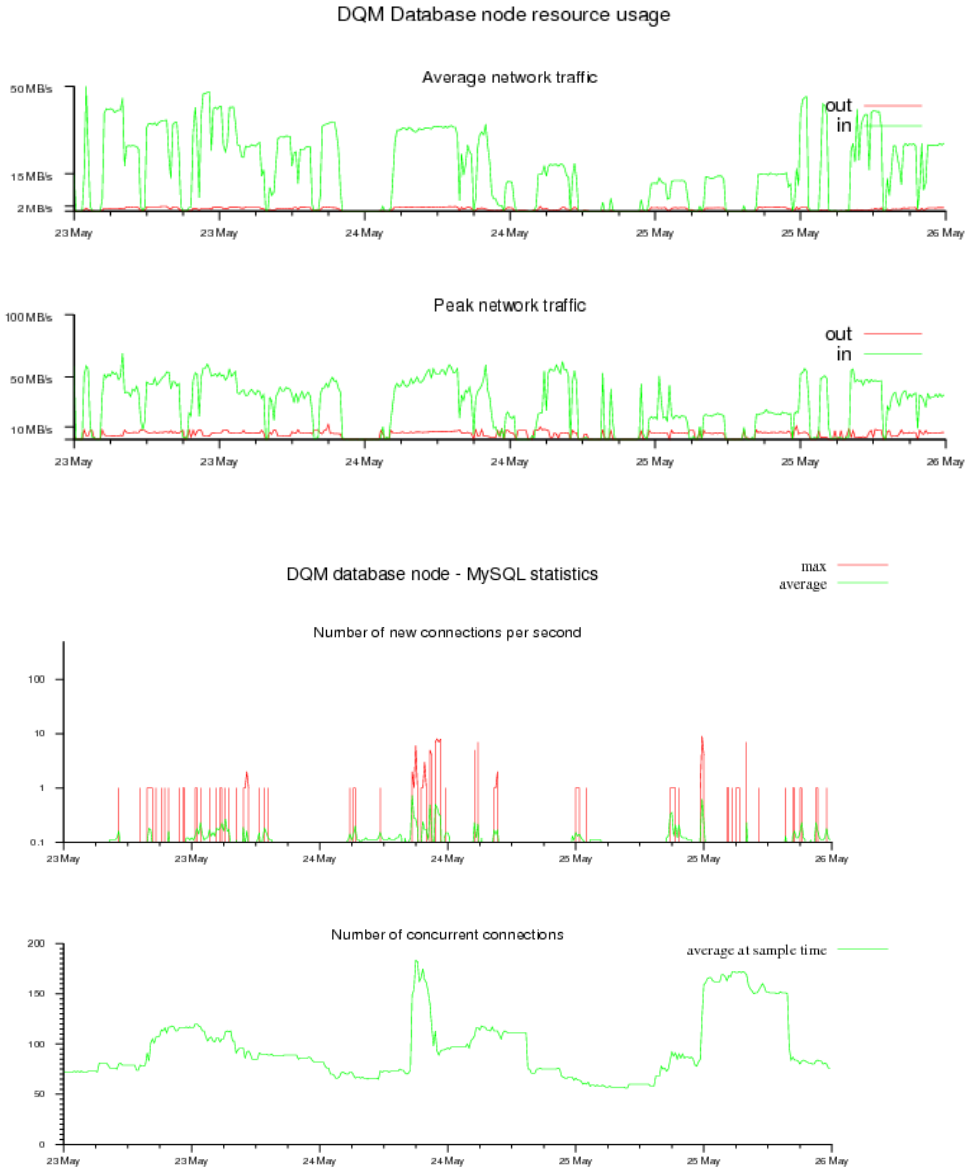


Network traffic



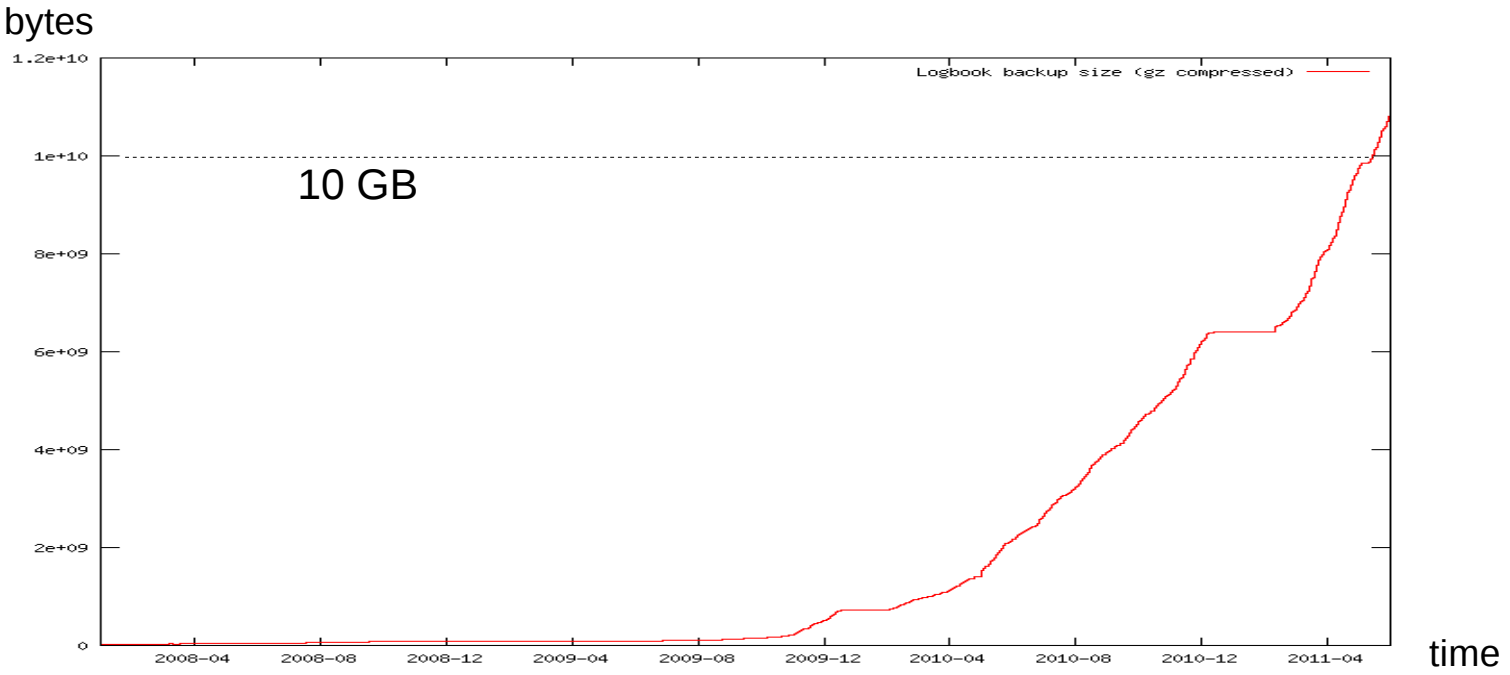
Daily backup

DQM DB



Different pattern:
more I/O, less clients

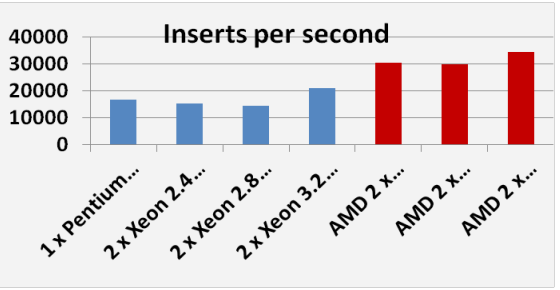
Data keeps growing



- LOGBOOK data increase with number of runs
- Size on disk = 2x compressed backup, 22GB at the moment, doubled in past 6 months (physics data taking + new features)
- Some large objects will be removed and stored as normal files
- Daily backup was heavy in the end, with previous hardware
- Replication was fine as replacement
- Various strategies to reduce size / availability, e.g. separation of online data and split history

Hardware

Extensive benchmarks performed to select production hardware



2006 (2007 in production)

- 2x dual-core AMD Opteron 275 @ 2.2 GHz
- FiberChannel RAID6 disk array 500GB
- SLC4-64

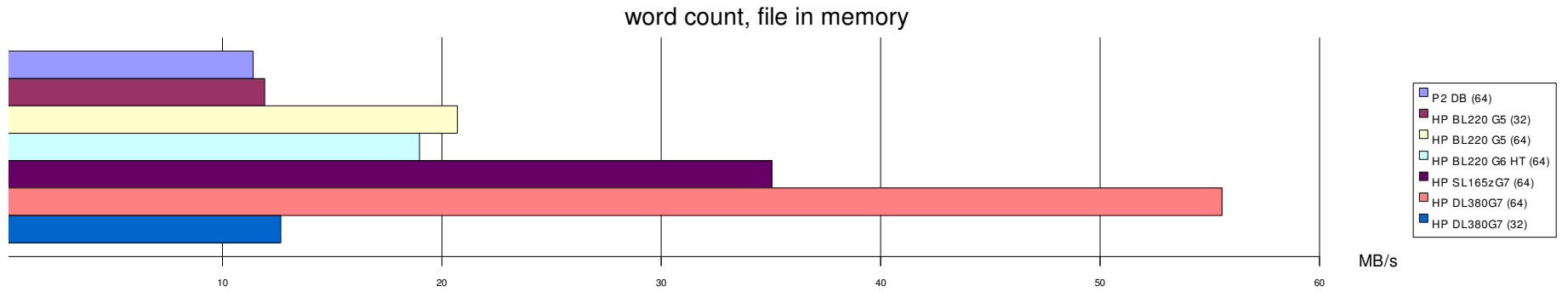
2010 (2011 in production) – HP DL380G7

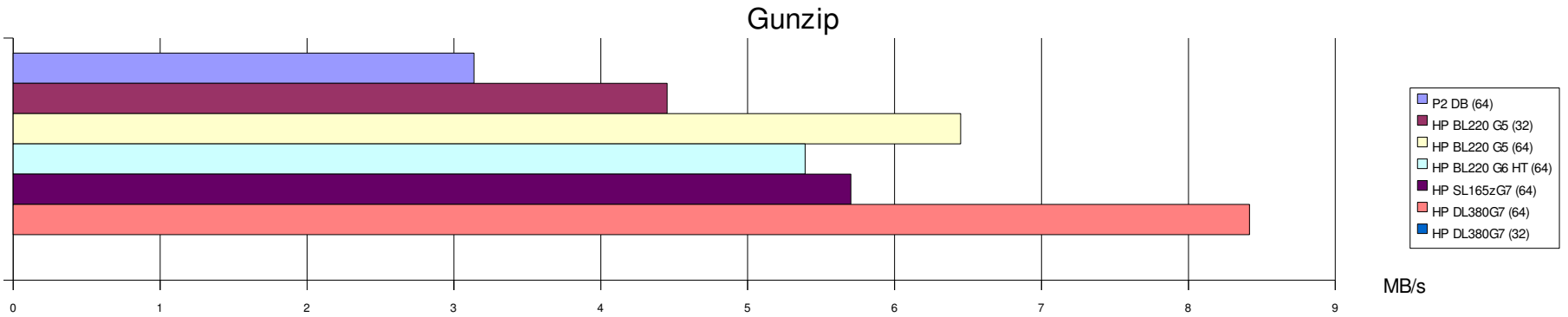
- 2x quad-core Intel Xeon X5677 @ 3.5 GHz
- internal RAID 0 + 1 SAS 15k 200GB
- Disk controller with flash write cache backup
- SLC5-64

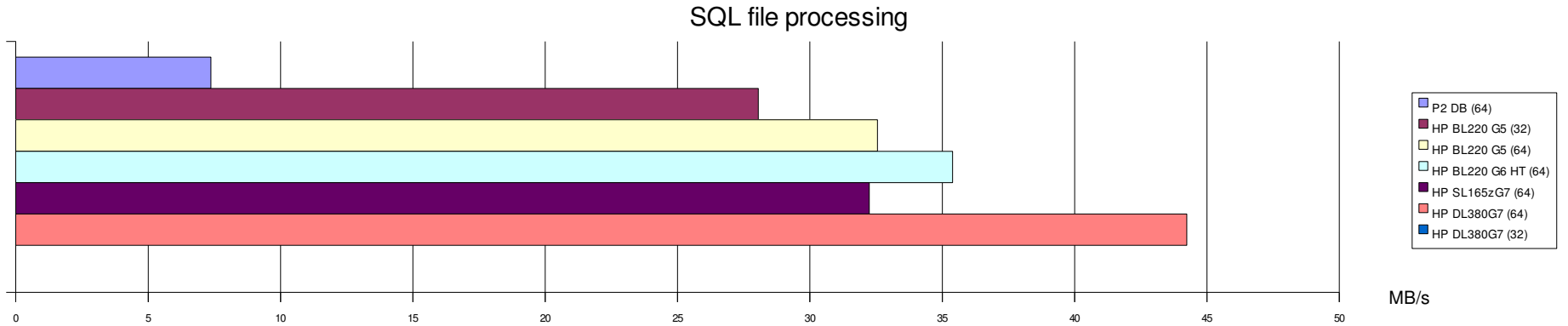
2010 benchmarks - machines tested

Type	Description	CPU	Cores (physical - logical)	Memory	OS
P2 DB (64)	P2 DB (2007)	2x 2 cores AMD Opteron 275 @ 1.00 Ghz	4 – 4	6G	SLC4 64
HP BL220 G5 (32)	blade G5	2x 4 cores E5450 @ 3.00GHz	8 – 8	16G	SLC4 32
HP BL220 G5 (64)	blade G5	2x 4 cores Intel E5450 @ 3.00GHz	8 – 8	16G	SLC5 64
HP BL220 G6 HT (64)	blade G5	2x 4 cores HT Intel E5530 @ 2.40GHz HyperThreading ON	8 – 16	16G	SLC5 64
HP SL165zG7 (64)	AMD G7	2x 12 cores AMD Opteron 6174 @ 2.20 Ghz	24 -24	32G	ubuntu 64
HP DL380G7 (64)	Intel g7	2x 4 cores E5640 @ 2.67GHz	8 – 8	24G	ubuntu 64
HP DL380G7 (32)	Intel g7	2x 4 cores E5640 @ 2.67GHz	8 – 8	24G	SLC4 32

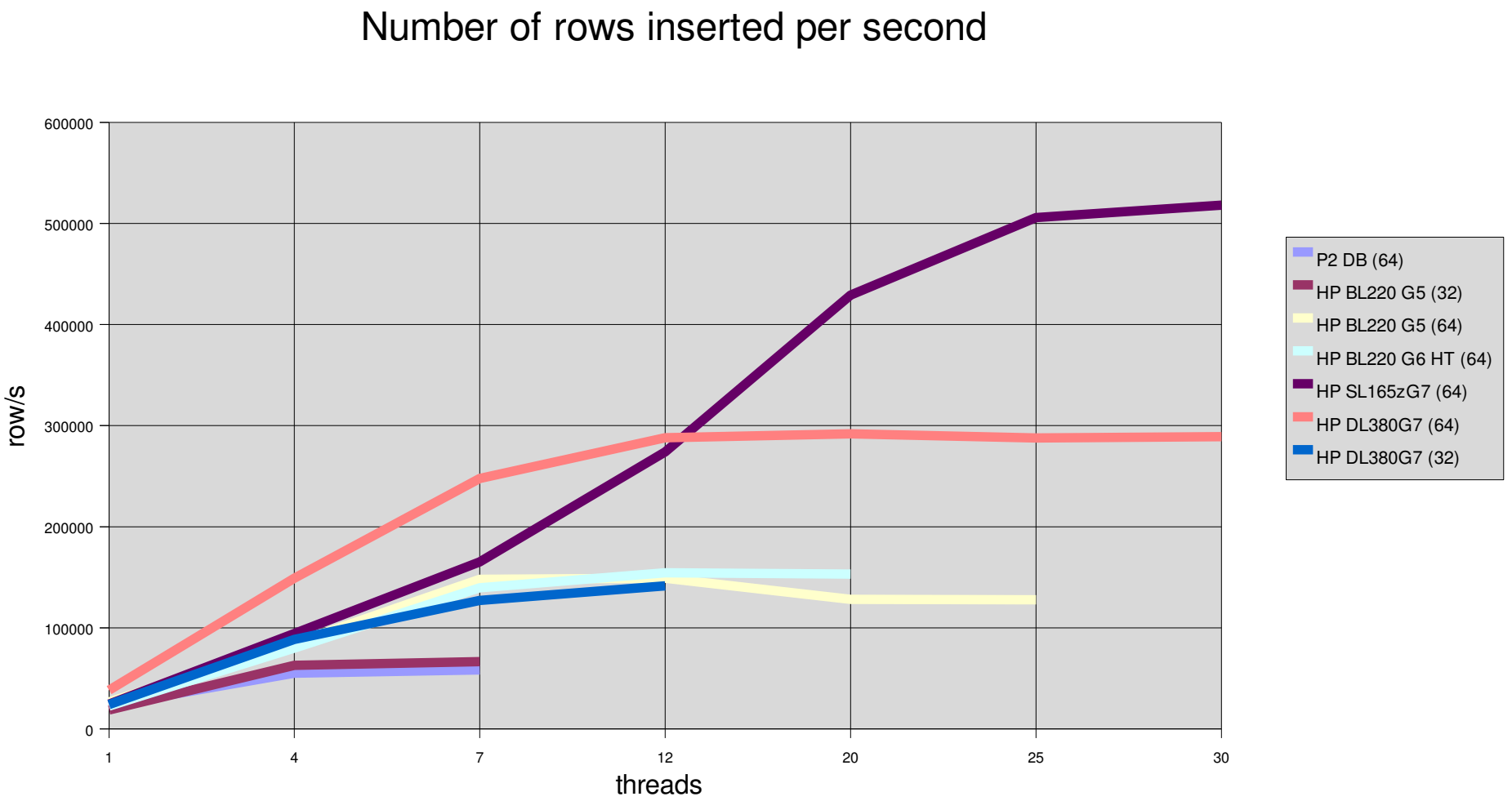
Benchmark results – single threaded applications







Benchmark results – multi threaded applications



Benchmark conclusions

(quote from report 05/2010)

*"As expected, machines must run with **64 bit OS** to get the best performance, as shown by the results of same hardware with different OS installed.*
Hyper-threading does not seem to improve nor reduce performance at higher thread count.
*The **G7 Intel CPU** offers by far the best per-core performance.*
The high-density G7 AMD CPU performs decently compared to previous generation CPU of higher CPU frequency, but not as well as the Intel G7. However, it scales very well with the number of threads, and finally largely outperforms the Intel model when all cores are active.

One can expect a best-case improvement of a factor 6 to 10 with the latest CPUs compared to the hardware in production at the moment. However, these numbers can be reached only in optimal parallelism situations. A factor 2-4 looks more realistic. It will depend very much on the type of applications.
*Peak power / minimal response time would probably be achieved with the top-end **high-frequency** Intel models (not tested here) and to a lesser extent with the E5640, whereas overall performance in serving a large number of clients would fit best to the AMD 6174."*

MySQL software

- Packaging
  - We take binary RPMs from the MySQL site
  - Use of the latest 'production' version, now 5.5
  - SLC repositories usually behind (SLC 4 mysql 3, SLC5 mysql 4, now 5.0)
- Support
  - Extensive documentation and knowledge base available
  - No problem so far
  - Reported 1 minor bug (change in packaging). Serious and rapid follow-up.
- Free version
  - 'community' version, no support / tools
- No (or little) tuning needed for performance
  - e.g. number of connections, maximum packet size, cache size, ...
- Lightweight maintenance (time spent close to zero)

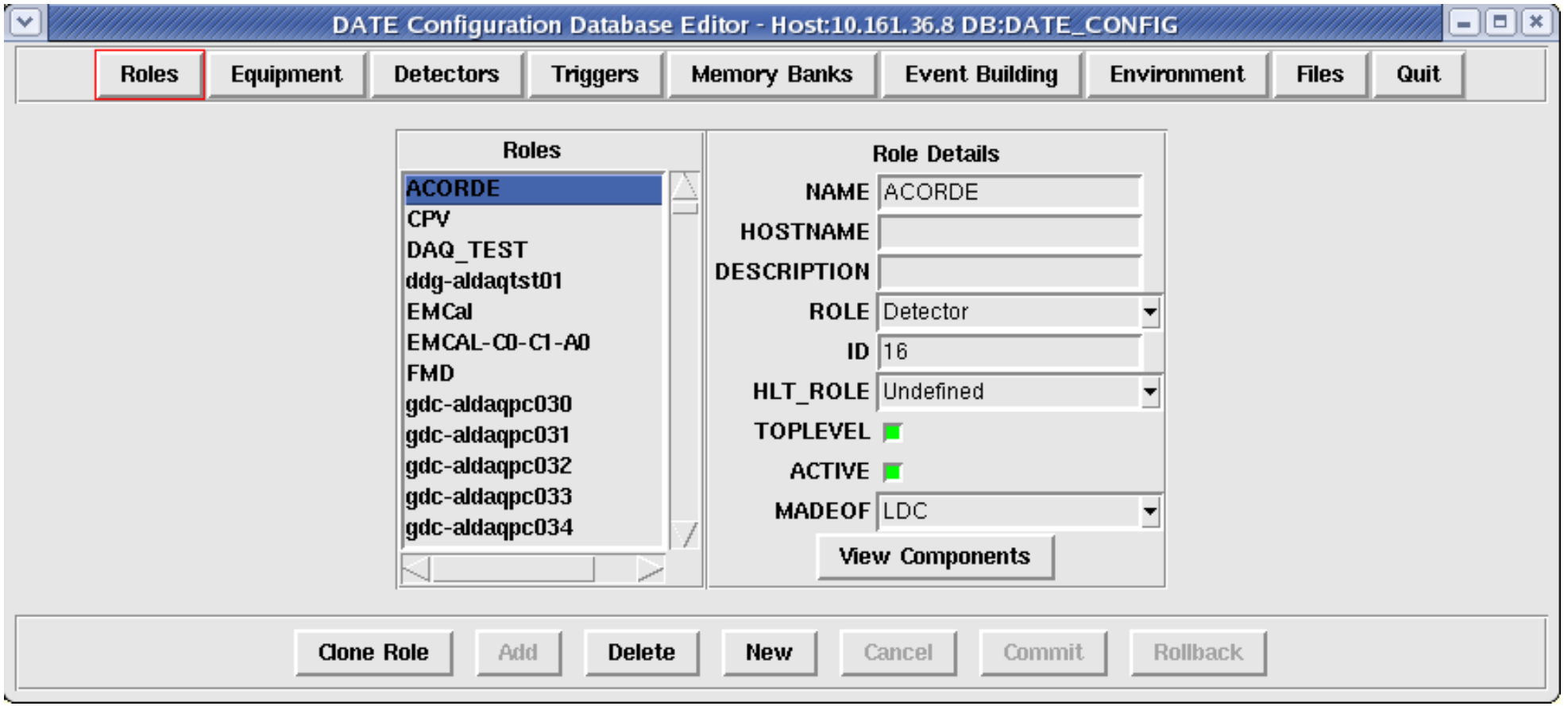
What about ...

- Security
  - Application specific credentials
  - No encryption (private network, no sensitive data)
- Data safety
  - Daily backups
  - RAID 0+1 and cold disk spare
  - Single case of severe crash last year after power cut / disk array file system corruption, where storage engine could not resume from log, fully recovered from replicated DB
- Availability
  - Several machines, services can be moved
  - Restored from backup or empty DBs

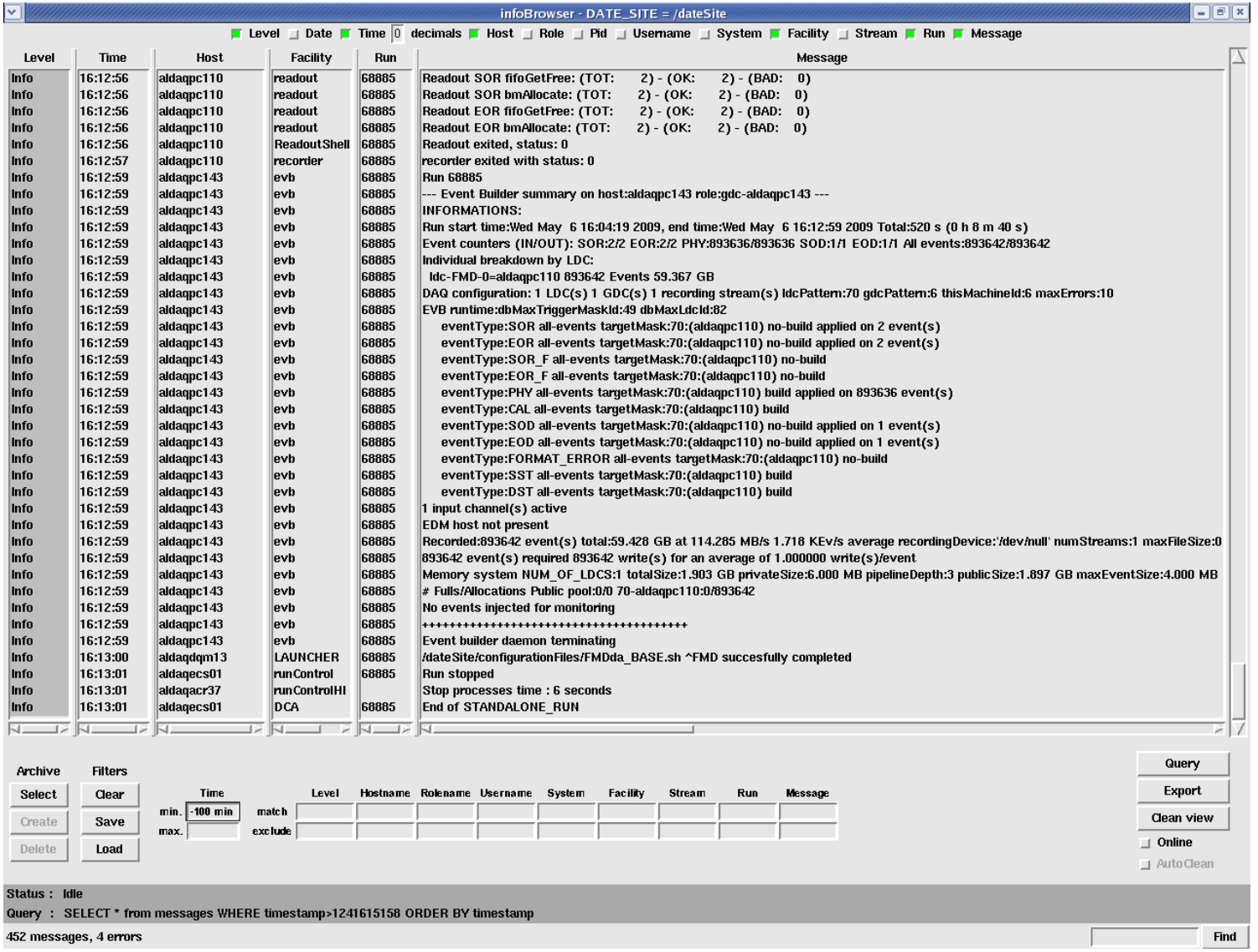
Data interfaces

- APIs
  - C for main applications
    - connect/disconnect, query, prepared statements
  - Tcl (in particular from UI / runControl)
    - Mysqltcl (we package the SLC5 rpm)
  - Direct SQL through command line client
    - From shell scripts
    - For interactive (debug/expert) queries
- GUIs
  - Tcl / Tk
  - PHP / HTML
  - MySQL QueryBrowser / Administrator (now called 'workbench')

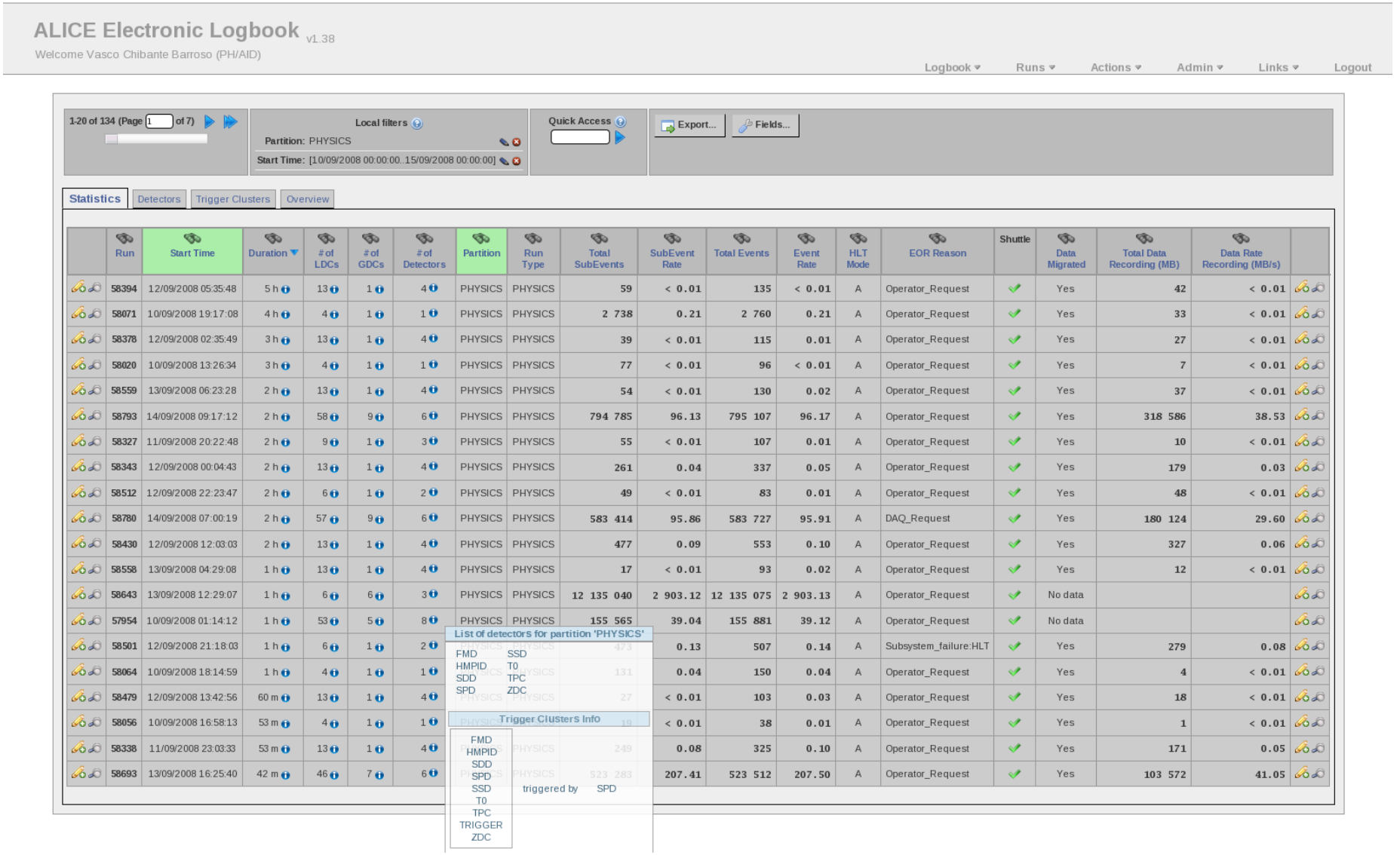
GUI snapshots



GUI snapshots



GUI snapshots



Some features we use

- Contraints & foreign keys
  - Configuration data integrity
- Transactions
  - Mainly from interactive clients (lock & rollback)
  - In APIs: updates of shared counters, multiple steps operations
- Partitioning
  - Automatic split of tables on a variable (e.g. log timestamp)
- Indexing
  - Needed for fast response time on queries, seen little effect on insert (but quite heavy on size)
- Triggers, Events (c.f. cron), stored procedures
  - e.g. to update global counters or lists, or for shared logic between different APIs
- Storage engine types
  - innoDb (constraints, transactions) MySQL (raw performance) RAM (fast transient data)

Some features we use

- Replication (1 master, several slaves)
  - Easy to configure
  - Good as backup replacement or hot spare setup
  - Remove query load from main server
  - NB: Cluster feature ('shared-nothing' redundancy) of MySQL seems nice, but not tried / needed for our system
- Backup
  - Crontab: dump database to SQL file and archive (RAID6 + tape)
  - Easy to reload, may take time (indexes)
  - Careful definition of mysqldump options
- Monitoring
  - Easy access to key server metrics (inserts, connects, etc)
- Extended server log
  - Punctual enabling of full query logs and analysis allowed to spot several client implementation issues

Conclusion and perspectives

- We are happy MySQL users
  - Performance and features right “out of the box”
  - Fits our (simple?) needs for a large range of data patterns
  - Heterogeneous DBs and usages demand careful planning and testing
  - After more than 3 years in production, excellent feedback on stability, reliability, performance

- No big change expected on requirements / needs
  - More DAQ components will use databases
  - Close look on existing DB growth over time
  - What about a SQL DB with data subscribe / notification interface ?