AliEn File Catalogue status and alternatives

Miguel Martinez Pedreira

A Large Ion Collider Experiment

European Organisation for Nuclear Research

Status

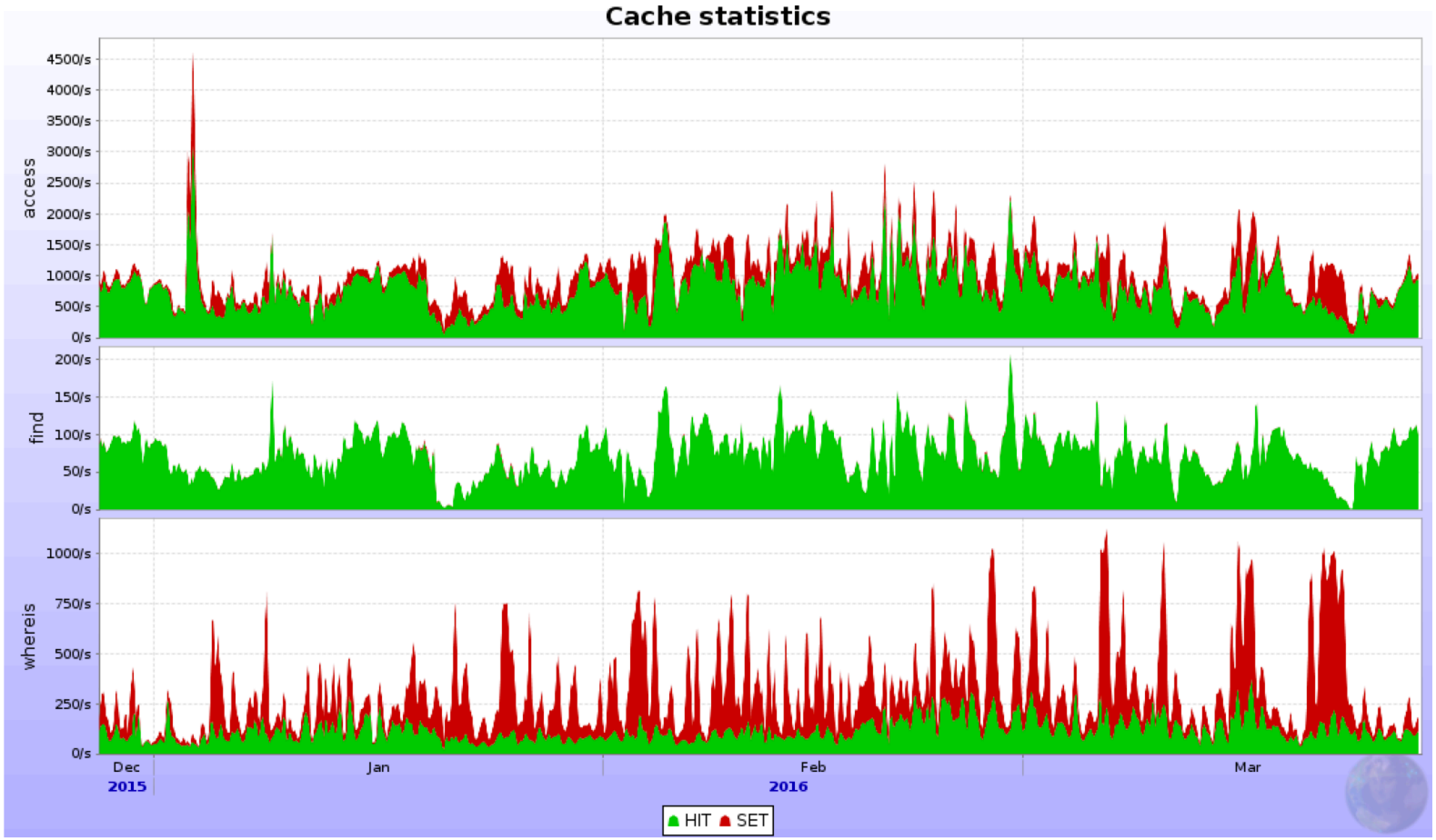
- Powerful server
  - ProLiant DL380 Gen9
  - Xeon E5-2687W v3 3.10GHz
    - 40 cpus
  - 755 GB RAM
  - SDD-based
  - MySQL 5.6 on latest Ubuntu
  - ~2 TB on disk – 2.7B PFNs, 2.1B LFNs – AVG load 4.5
    - Growing several M entries/day
      - Reached rates of 10-15M / day (many files per subjob)
    - Restoring takes more than a day, *optimize table* could save up to half a TB

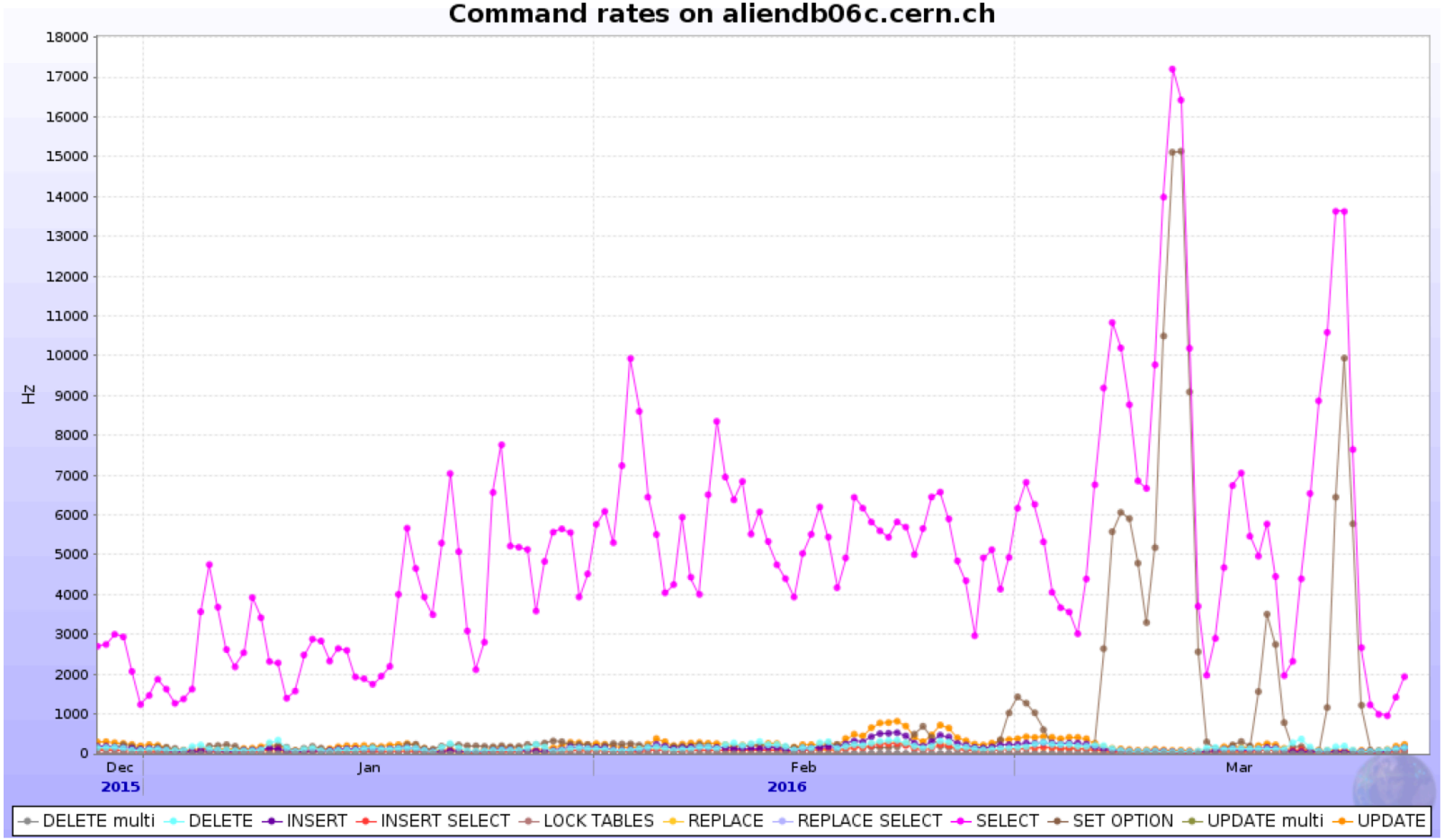
Machines status

db06c

	Machine status			Machine type					Disk	CPU utilisation (%)								Memory utilisation					Swap	
Machine	Online	Uptime	Load	Kernel	Machine model	CPU	CPUs	MHz	Space	usr	sys	iow	int	sint	steal	nice	idle	Total	Used	Buffers	Cached	Free	Used	Free
1. db6c		284d 1:31	3.85	3.19.0-21...	ProLiant DL380 Gen9	Xeon E5-2687W v3 3.10GHz	40	1200		7.538	0.936	0.093	0	0.629	0	0	90.8	755.8 GB	298.4 GB	197.4 MB	454.4 GB	2.867 GB	0	0
Total							**40**											**755.8 GB**	**298.4 GB**	**197.4 MB**	**454.4 GB**	**2.867 GB**	**0**	**0**
Average		**284d 1:31**	**3.85**							**7.538**	**0.936**	**0.093**	**0**	**0.629**	**0**	**0**	**90.8**	**755.8 GB**	**298.4 GB**	**197.4 MB**	**454.4 GB**		**0**	**0**

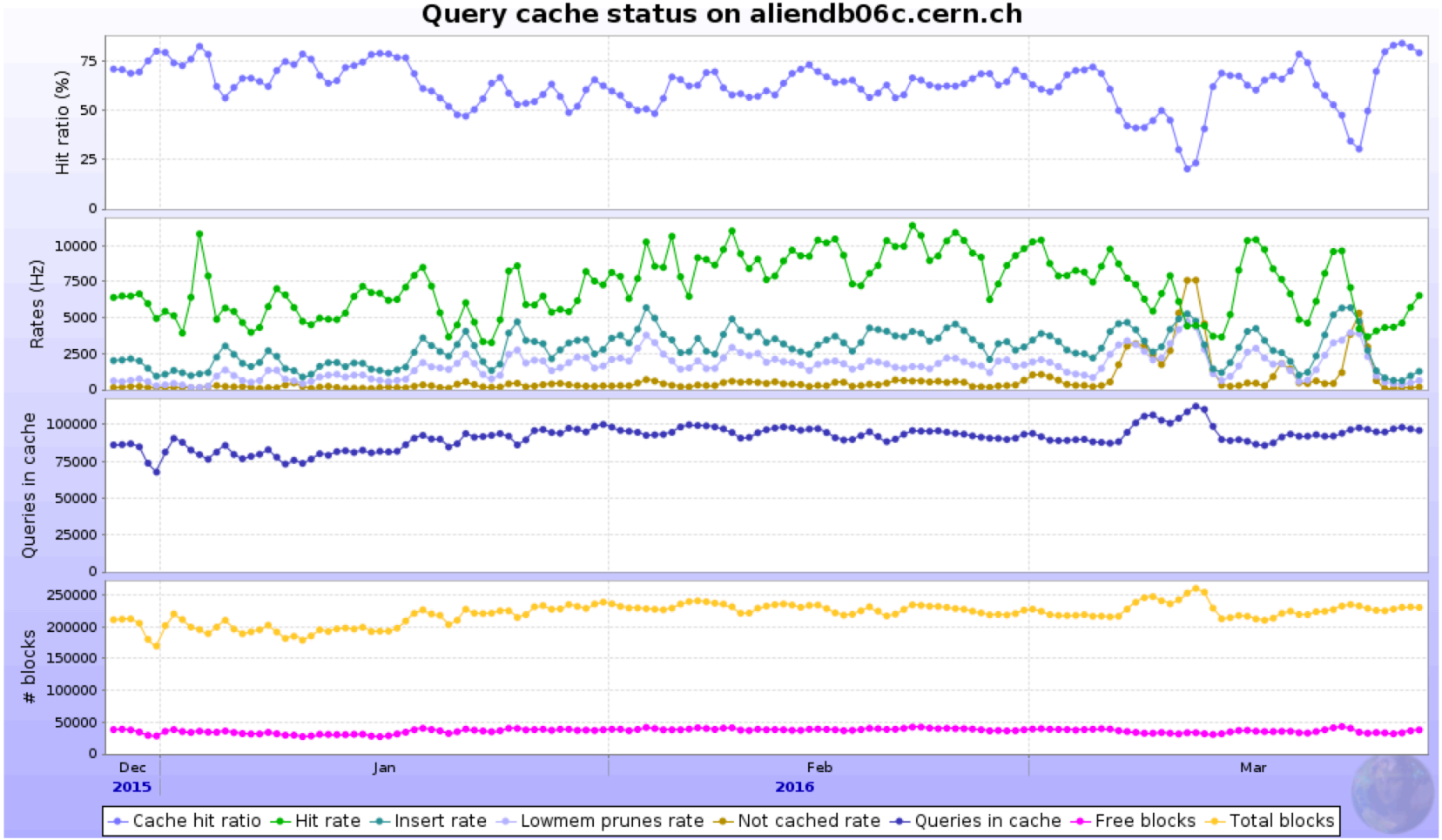
Cache


Cache statistics
access
4500/s
4000/s
3500/s
3000/s
2500/s
2000/s
1500/s
1000/s
500/s
0/s
find
200/s
150/s
100/s
50/s
0/s
whereis
1000/s
750/s
500/s
250/s
0/s
Dec
2015
Jan
Feb
2016
Mar
HIT
SET



Command rates on aliendb06c.cern.ch

	Series	Last value	Min	Avg	Max	Total
1.	DELETE multi	0	0	0	0	0
2.	DELETE	155.2	0	147.6	4344	1148498564
3.	INSERT	151.5	0	139.6	2051	1086667046
4.	INSERT SELECT	87.47	0	76.66	1189	596562145
5.	LOCK TABLES	0.082	0	0.058	2.492	454111
6.	REPLACE	79.42	0	78.47	956.7	610595643
7.	REPLACE SELECT	0	0	0	0	0
8.	SELECT	1937	0	5010	23333	38987115405
9.	SET OPTION	158.6	0	961.7	21260	7483238637
10.	UPDATE multi	0	0	0	0	0
11.	UPDATE	231.8	0	223	2339	1735538108
	Total	**2801**		**6637**		**51648669663**

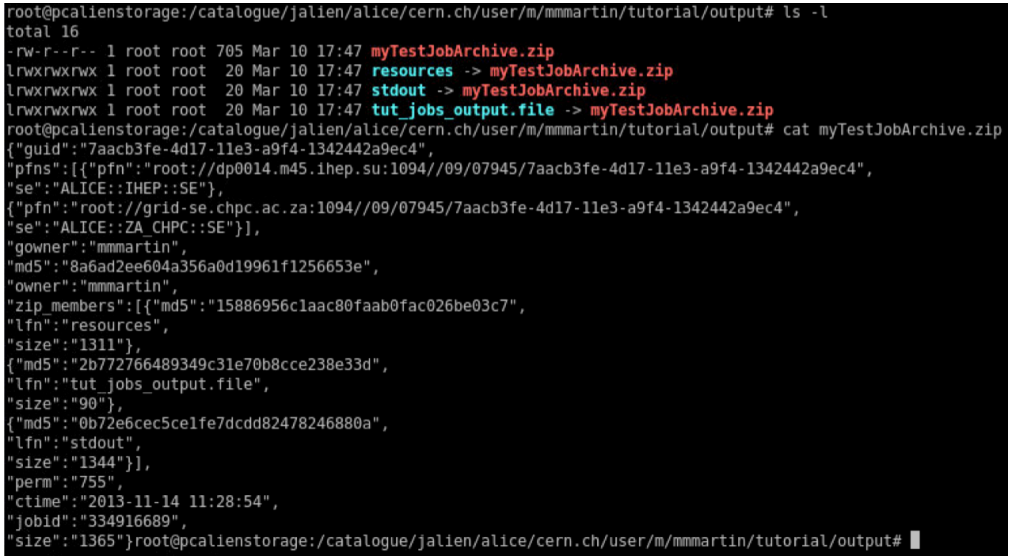

Query cache status on aliendb06c.cern.ch
Hit ratio (%)
Rates (Hz)
Queries in cache
blocks
Dec
2015
Jan
Feb
2016
Mar
Cache hit ratio
Hit rate
Insert rate
Lowmem prunes rate
Not cached rate
Queries in cache
Free blocks
Total blocks

Alternatives summary

- Mapping to FS
  - Btrfs or similar
  - CVMFS + Key-Value
- Simplification (guidless) + partitioning of current schema

Mapping to FS - Tool

- Created a dumper tool
  - Based on jAliEn
    - Few things missing added
    - Nicely manage threads with executors
  - Creates the hierarchy
    - JSON files
  - Archives
    - Content in archive file
    - Members are symbolic links
  - Logs files/folders/collections
    - Discovered significant amount of orphans or missing pfns



btrfs

- First try in standard ext4 in Ubuntu
  - Ran out of inodes after some million entries, as expected
- BTRFS (Binary trees)
  - Max number of files: 2^64
  - Max volume size: 16 EiB
  - Space-efficient packing of small files
  - Space-efficient indexed directories
  - Dynamic inode allocation
  - Writable snapshots, read-only snapshots (backups!)
  - Compression (zlib and LZO)
  - SSD (Flash storage) awareness
  - More: https://btrfs.wiki.kernel.org/index.php/Main_Page

```
root@pcalienstorage:/catalogue/jalien/json_out# df -h .
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda5       7.7T  971G  6.7T  13% /catalogue
root@pcalienstorage:/catalogue/jalien/json_out# df -i .
Filesystem     Inodes IUsed IFree IUse% Mounted on
/dev/sda5           0     0     0     - /catalogue
```

Mapping to FS - Timing

- About creation
  - Rate depends on folder and usage at the moment…
  - Done with master and slave (mostly master because of backups)
  - 50 threads = +~5 load
  - Optimize
    - Use N slaves
    - Load G space in memory
      - Servers with decent RAM…
- Examples (disk)
  - /alice/data/2010/ - 57.2M entries – 7h – 15 threads
  - /alice/data/2012/ - 141.6M entries – 37h – 20 threads
  - /alice/cern.ch/user/c/ - 3.5M entries – 40 min – 15 threads
  - /alice/sim/2014/ - 215M entries – 27h – 12 threads
- Examples (ssd)
  - /alice/data/2012/ - 120M entries – 21h - 15 threads
  - /alice/data/2013/ - 69M entries - 13h – 10 threads
  - /alice/data/2014/ - 362K – 40m – 15 threads

Initial benchmark and issues

- What about (both for CVMFS or btrfs FC):
  - Quotas, SE lookups, booking tables+locking…
- Benchmarking tool similar to the dump
  - Goes over base directory
  - Times the command/lfn
  - Tried with `ls` over each directory: FS saturated with just 10 threads, >1 s/ls

```
03/24/16 17:17:15
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           9.84    0.00    6.60   13.13    0.00   70.43

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda               0.00     0.40 1921.40    1.60 29697.60    12.80    30.90     4.92    2.56    2.56    0.00   0.52 100.08
```

  - Preparing yet more dumps on SSD based btrfs
    - Provided SSD iops are greatly bigger
    - FS on disk discarded…

Type	Threads	Folders	Ms / ls
Local SSD	1	10K	3 hot 13 cold
Local SSD	10	1M	2-3
Local SSD	50	1M	13
Local SSD	100	1M	26
NFS SSD	1	10K	114 ?
NFS SSD	10	1M	39 ?
DB (test02)	1	10K	5-7
DB	10	1M	7
DB	50	1M	23

```
30.03.2016 17:00:20
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          46.26    1.24   35.84    2.99    0.00   13.67

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda             665.60  7494.20  231.00 1244.60  4411.20 34972.80    53.38     0.47    0.32    0.25    0.34   0.11  16.16
sdb             138.20     0.00 3745.40    0.00 62200.00     0.00    33.21     0.51    0.14    0.14    0.00   0.08  31.28
```

100 threads local ssd

```
30.03.2016 16:11:06
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          47.85    1.16   35.05    3.79    0.00   12.14

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda             618.40    12.00  278.80   53.40  4977.60   263.20    31.55     0.09    0.26    0.28    0.19   0.13   4.32
sdb              97.60     0.00 3040.40    0.00 50278.40     0.00    33.07     0.42    0.14    0.14    0.00   0.09  28.00
```

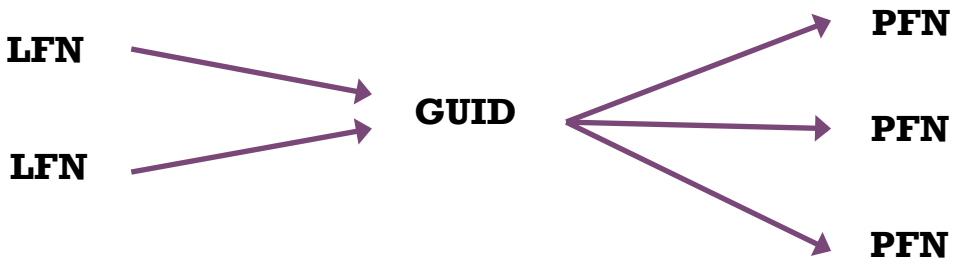
10 threads local ssd

CVMFS + Key-value approach

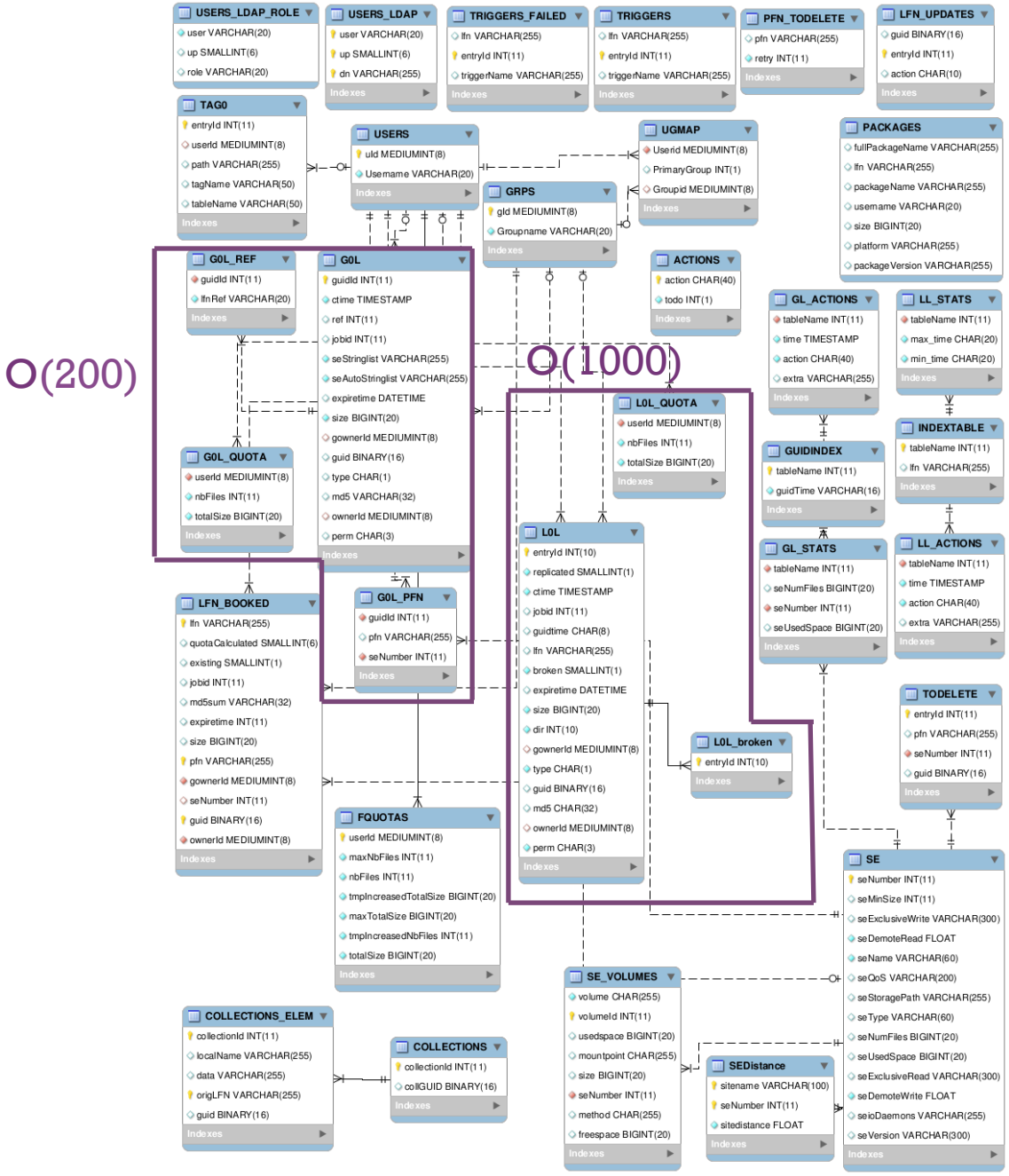
- CVMFS uses a similar data-representation as the AliEn FC
  - Main difference: the structure (SQLite based) is sent to the clients and DBs splitted per directories
    - The hard work relies now on the client!
    - We could use CVMFS as logical namespace
      - Can we create a namespace tool that everybody can use in the same way ?
      - And is known and trustable
    - We hold the metadata in attributes
    - Complemented by a 'service' that takes care of authorization
      - Namespace would be public but access to files
    - PFNs on key-value store using a key, like the LFN e.g.
  - What do we do about files that are written and just after read ?
    - The delay is on the best case in the order of some minutes (Jakob)
    - Merging of jobs, user space…
  - Which other features need to be addressed?
    - SE and User quotas: column-based could be tried instead ?
    - Booking table
    - Collections
    - Metadata (tags)
    - Security

GUIDless catalogue

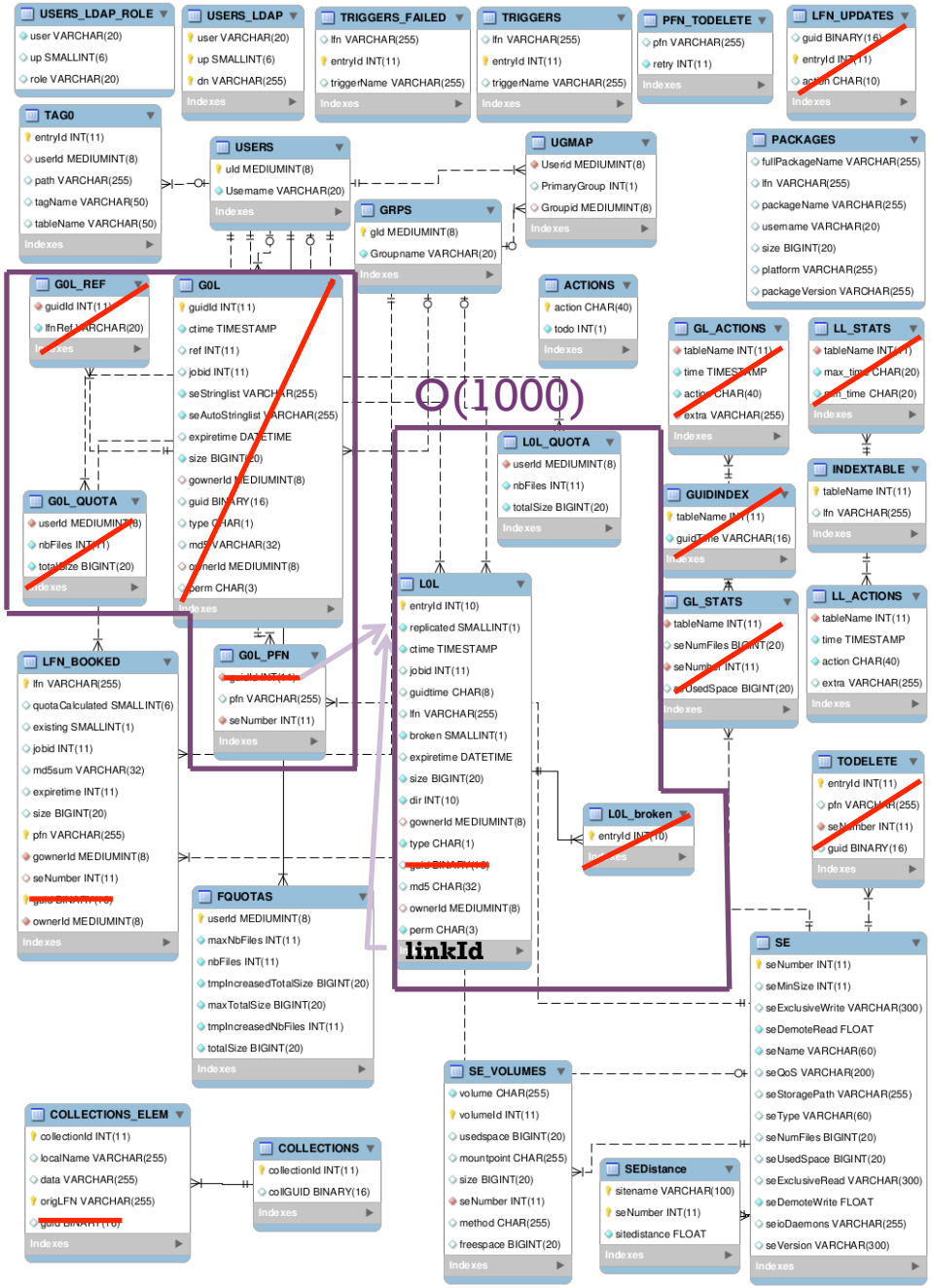
- GUIDs provide us with extra flexibility in the catalogue
  - But in practice is not used (perms for links, mirrors, lfn name changes)
  - Extra lookups + space



- We still lack having more integrity and optimizing the catalogue
  - FK + surrogated keys
  - Query minimization
  - Using InnoDB everywhere
    - Row-level locks, and more
  - Indexes
  - Table cleanups and split, collections, tags…


O(200)
O(1000)
USERS_LDAP_ROLE
user VARCHAR(20)
up SMALLINT(6)
role VARCHAR(20)
USERS_LDAP
user VARCHAR(20)
up SMALLINT(6)
dn VARCHAR(255)
TRIGGERS_FAILED
lfn VARCHAR(255)
entryId INT(11)
triggerName VARCHAR(255)
TRIGGERS
lfn VARCHAR(255)
entryId INT(11)
triggerName VARCHAR(255)
PFN_TODELETE
pfn VARCHAR(255)
retry INT(11)
LFN_UPDATES
guid BINARY(16)
entryId INT(11)
action CHAR(10)
TAG0
entryId INT(11)
userId MEDIUMINT(8)
path VARCHAR(255)
tagName VARCHAR(50)
tableName VARCHAR(50)
USERS
uId MEDIUMINT(8)
Username VARCHAR(20)
GRPS
gId MEDIUMINT(8)
Groupname VARCHAR(20)
UGMAP
Userid MEDIUMINT(8)
PrimaryGroup INT(1)
Groupid MEDIUMINT(8)
PACKAGES
fullPackageName VARCHAR(255)
lfn VARCHAR(255)
packageName VARCHAR(255)
username VARCHAR(20)
size BIGINT(20)
platform VARCHAR(255)
packageVersion VARCHAR(255)
G0L_REF
guidId INT(11)
lfnRef VARCHAR(20)
G0L
guidId INT(11)
ctime TIMESTAMP
ref INT(11)
jobid INT(11)
seStringlist VARCHAR(255)
seAutoStringlist VARCHAR(255)
expiretime DATETIME
size BIGINT(20)
gownerId MEDIUMINT(8)
guid BINARY(16)
type CHAR(1)
md5 VARCHAR(32)
ownerId MEDIUMINT(8)
perm CHAR(3)
ACTIONS
action CHAR(40)
todo INT(1)
GL_ACTIONS
tableName INT(11)
time TIMESTAMP
action CHAR(40)
extra VARCHAR(255)
LL_STATS
tableName INT(11)
max_time CHAR(20)
min_time CHAR(20)
G0L_QUOTA
userId MEDIUMINT(8)
nbFiles INT(11)
totalSize BIGINT(20)
L0L_QUOTA
userId MEDIUMINT(8)
nbFiles INT(11)
totalSize BIGINT(20)
GUIDINDEX
tableName INT(11)
guidTime VARCHAR(16)
INDEXTABLE
tableName INT(11)
lfn VARCHAR(255)
GL_STATS
tableName INT(11)
seNumFiles BIGINT(20)
seNumber INT(11)
seUsedSpace BIGINT(20)
LL_ACTIONS
tableName INT(11)
time TIMESTAMP
action CHAR(40)
extra VARCHAR(255)
LFN_BOOKED
lfn VARCHAR(255)
quotaCalculated SMALLINT(6)
existing SMALLINT(1)
jobid INT(11)
md5sum VARCHAR(32)
expiretime INT(11)
size BIGINT(20)
pfn VARCHAR(255)
gownerId MEDIUMINT(8)
seNumber INT(11)
guid BINARY(16)
ownerId MEDIUMINT(8)
G0L_PFN
guidId INT(11)
pfn VARCHAR(255)
seNumber INT(11)
L0L
entryId INT(10)
replicated SMALLINT(1)
ctime TIMESTAMP
jobid INT(11)
guidtime CHAR(8)
lfn VARCHAR(255)
broken SMALLINT(1)
expiretime DATETIME
size BIGINT(20)
dir INT(10)
gownerId MEDIUMINT(8)
type CHAR(1)
guid BINARY(16)
md5 CHAR(32)
ownerId MEDIUMINT(8)
perm CHAR(3)
L0L_broken
entryId INT(10)
TODELETE
entryId INT(11)
pfn VARCHAR(255)
seNumber INT(11)
guid BINARY(16)
FQUOTAS
userId MEDIUMINT(8)
maxNbFiles INT(11)
nbFiles INT(11)
tmpIncreasedTotalSize BIGINT(20)
maxTotalSize BIGINT(20)
tmpIncreasedNbFiles INT(11)
totalSize BIGINT(20)
SE
seNumber INT(11)
seMinSize INT(11)
seExclusiveWrite VARCHAR(300)
seDemoteRead FLOAT
seName VARCHAR(60)
seQoS VARCHAR(200)
seStoragePath VARCHAR(255)
seType VARCHAR(60)
seNumFiles BIGINT(20)
seUsedSpace BIGINT(20)
seExclusiveRead VARCHAR(300)
seDemoteWrite FLOAT
seioDaemons VARCHAR(255)
seVersion VARCHAR(300)
SE_VOLUMES
volume CHAR(255)
volumeId INT(11)
usedspace BIGINT(20)
mountpoint CHAR(255)
size BIGINT(20)
seNumber INT(11)
method CHAR(255)
freespace BIGINT(20)
SEDistance
sitename VARCHAR(100)
seNumber INT(11)
sitedistance FLOAT
COLLECTIONS_ELEM
collectionId INT(11)
localName VARCHAR(255)
data VARCHAR(255)
origLFN VARCHAR(255)
guid BINARY(16)
COLLECTIONS
collectionId INT(11)
collGUID BINARY(16)

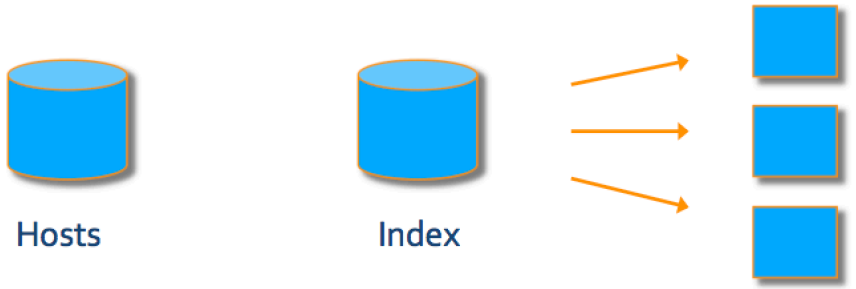
O(200)


USERS_LDAP_ROLE
user VARCHAR(20)
up SMALLINT(6)
role VARCHAR(20)
USERS_LDAP
user VARCHAR(20)
up SMALLINT(6)
dn VARCHAR(255)
TRIGGERS_FAILED
lfn VARCHAR(255)
entryId INT(11)
triggerName VARCHAR(255)
TRIGGERS
lfn VARCHAR(255)
entryId INT(11)
triggerName VARCHAR(255)
PFN_TODELETE
pfn VARCHAR(255)
retry INT(11)
LFN_UPDATES
guid BINARY(16)
entryId INT(11)
action CHAR(10)
TAG0
entryId INT(11)
userId MEDIUMINT(8)
path VARCHAR(255)
tagName VARCHAR(50)
tableName VARCHAR(50)
USERS
uid MEDIUMINT(8)
Username VARCHAR(20)
UGMAP
Userid MEDIUMINT(8)
PrimaryGroup INT(1)
Groupid MEDIUMINT(8)
PACKAGES
fullPackageName VARCHAR(255)
lfn VARCHAR(255)
packageName VARCHAR(255)
username VARCHAR(20)
size BIGINT(20)
platform VARCHAR(255)
packageVersion VARCHAR(255)
GRPS
gid MEDIUMINT(8)
Groupname VARCHAR(20)
G0L_REF
guidId INT(11)
lfnRef VARCHAR(20)
G0L
guidId INT(11)
ctime TIMESTAMP
ref INT(11)
jobid INT(11)
seStringlist VARCHAR(255)
seAutoStringlist VARCHAR(255)
expiretime DATETIME
size BIGINT(20)
gownerId MEDIUMINT(8)
guid BINARY(16)
type CHAR(1)
md5 VARCHAR(32)
ownerId MEDIUMINT(8)
perm CHAR(3)
ACTIONS
action CHAR(40)
todo INT(1)
GL_ACTIONS
tableName INT(11)
time TIMESTAMP
action CHAR(40)
extra VARCHAR(255)
LL_STATS
tableName INT(11)
max_time CHAR(20)
min_time CHAR(20)
G0L_QUOTA
userId MEDIUMINT(8)
nbFiles INT(11)
totalSize BIGINT(20)
L0L_QUOTA
userId MEDIUMINT(8)
nbFiles INT(11)
totalSize BIGINT(20)
GUIDINDEX
tableName INT(11)
guidTime VARCHAR(16)
INDEXTABLE
tableName INT(11)
lfn VARCHAR(255)
GL_STATS
tableName INT(11)
seNumFiles BIGINT(20)
seNumber INT(11)
seUsedSpace BIGINT(20)
LL_ACTIONS
tableName INT(11)
time TIMESTAMP
action CHAR(40)
extra VARCHAR(255)
L0L
entryId INT(10)
replicated SMALLINT(1)
ctime TIMESTAMP
jobid INT(11)
guidtime CHAR(8)
lfn VARCHAR(255)
broken SMALLINT(1)
expiretime DATETIME
size BIGINT(20)
dir INT(10)
gownerId MEDIUMINT(8)
type CHAR(1)
guid BINARY(16)
md5 CHAR(32)
ownerId MEDIUMINT(8)
perm CHAR(3)
linkId
L0L_broken
entryId INT(10)
TODELETE
entryId INT(11)
pfn VARCHAR(255)
seNumber INT(11)
guid BINARY(16)
LFN_BOOKED
lfn VARCHAR(255)
quotaCalculated SMALLINT(6)
existing SMALLINT(1)
jobid INT(11)
md5sum VARCHAR(32)
expiretime INT(11)
size BIGINT(20)
pfn VARCHAR(255)
gownerId MEDIUMINT(8)
seNumber INT(11)
guid BINARY(16)
ownerId MEDIUMINT(8)
G0L_PFN
guidId INT(11)
pfn VARCHAR(255)
seNumber INT(11)
FQUOTAS
userId MEDIUMINT(8)
maxNbFiles INT(11)
nbFiles INT(11)
tmpIncreasedTotalSize BIGINT(20)
maxTotalSize BIGINT(20)
tmpIncreasedNbFiles INT(11)
totalSize BIGINT(20)
SE
seNumber INT(11)
seMinSize INT(11)
seExclusiveWrite VARCHAR(300)
seDemoteRead FLOAT
seName VARCHAR(60)
seQoS VARCHAR(200)
seStoragePath VARCHAR(255)
seType VARCHAR(60)
seNumFiles BIGINT(20)
seUsedSpace BIGINT(20)
seExclusiveRead VARCHAR(300)
seDemoteWrite FLOAT
seioDaemons VARCHAR(255)
seVersion VARCHAR(300)
SE_VOLUMES
volume CHAR(255)
volumeId INT(11)
usedspace BIGINT(20)
mountpoint CHAR(255)
size BIGINT(20)
seNumber INT(11)
method CHAR(255)
freespace BIGINT(20)
SEDistance
sitename VARCHAR(100)
seNumber INT(11)
sitedistance FLOAT
COLLECTIONS_ELEM
collectionId INT(11)
localName VARCHAR(255)
data VARCHAR(255)
origLFN VARCHAR(255)
guid BINARY(16)
COLLECTIONS
collectionId INT(11)
collGUID BINARY(16)

O(1000)

GUIDless catalogue

- Use LFN+timestamp for uniqueness
  - We could keep pfn naming as in old version
  - Though LFN is more human readable
- What do we get:
  - ~40% of the catalogue goes away
    - From GUID tables and PFN links
  - All the LFN-PFN lookups also gone
  - We can still keep links in the way are used in ALICE (previous slide)
  - Simplicity!
- We can also partition
  - Code and DB are ready for this
  - Data and Users
  - This can be done in N servers…



Final comments

- Any solution will require significant amount of work ☺
  - And time to move from the current system, given the size of the catalogue and amount of features
- We have a very relational model
  - that needs the structure/fields to be addressed individually in some cases
- Good news having several possible alternatives ensuring scalability
- Some interesting links…
  - DBs used by biggest sites on Internet
  - DB engines ranking
  - Facebook running 1800 MySQL servers with 2 DBAs before NoSQL
- Discussion / questions ?