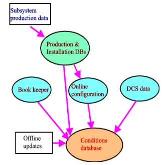
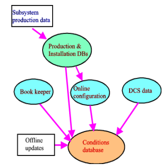


ATLAS SW week - Nov/Dec 2010

Database developments/optimizations in ATLAS

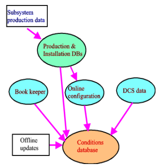
Gancho Dimitrov (DESY)



Outline

- Evolution of the PANDA database
- Problems with queries that ‘search’ into data of the most recent hours/ days
- The lifecycle of the PVSS data
- New organization of the DQ2 traces data
- Conclusions

Issues with the PANDA live and archive data



- The PANDA system is the ATLAS workload management system for the production and user analysis jobs
- All information relevant to a single job is stored in 4 basic tables. The tables are 4 because the most important stats are kept separately from the other space consuming attributes like job parameters, files details, inputs, outputs .. etc.
- The 'live' data is kept in a separate schema that keeps jobs of the most recent 3 days. Jobs that get status 'finished' or 'failed' are moved to the archive PANDA schema.

  ATLAS_PANDA => ATLAS_PANDAARCH

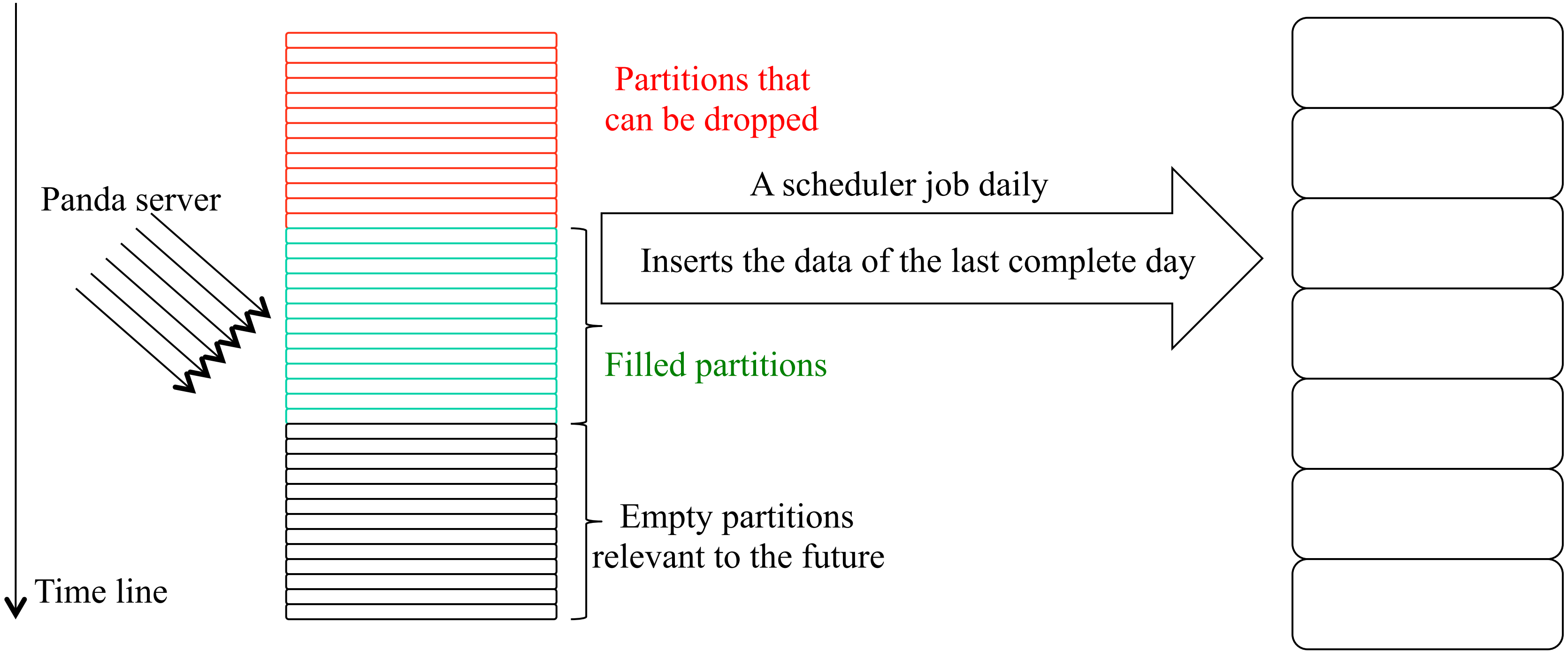
  Before the last reprocessing campaign, the data move was done from cron jobs. For each job that is marked for archiving, all relevant rows are inserted into the archive tables.

  The problem is that on average it takes a second for a single job to be moved from the PANDA to the PANDAARCH tables and an additional problem is that on the archive tables we have many indexes of type BITMAP, and thus launching more than a single data moving process causes row lock contention on the BITMAP index segments (NOT a scalable approach)

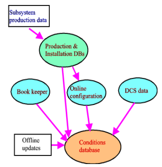
New data segments organization

ATLAS_PANDA.JOBSARCHIVED4 table:
partitioned on a 'modificationtime' column.
Each partition covers a time range of a day
(the same to be applied for other tables as well)

ATLAS_PANDAARCH.JOBSARCHIVED
partitioned on a 'modificationtime' column.
Each partition covers a time range of 3 days
as of 10th Nov 2010 (before was 30 days)



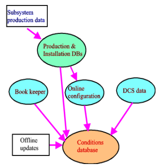
Attention: for setting that model in place, the PANDA server would need to be stopped for ~ 2 hours (most of the time would be spend on building indexes). A day from 6th- 9th Dec. has to be chosen



What can be expected from the change

- High scalability: the Panda archived jobs insertion and deletion will be done on table partition level <u>instead</u> of row level as before.
- Removing the already copied data will NOT be IO demanding (very little redo plus will NOT produce undo) as that will be a simple Oracle operation over a table segment (alter table … drop partition …) and its relevant index segments
- As all the indexes will be locally partitioned, for some SQL statement this will require few more Oracle block reads, but that would be a negligible impact.

Issues with the PANDA live and archive data (2)



- High rate of repeatable polling queries on the PANDA 'live' jobs(2-3000 times/hour) caused high CPU usage on the server side.

  To address the issue: Use of a Squid for caching the result of such queries. Even an often refresh (say minutely) would relieve the Oracle server to great extent.

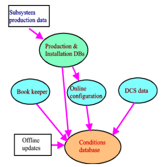
- Oracle stats on the large PANDAARCH tables often do NOT reflect the reality (stale stats) and that leads to inefficient data access paths. Effectively Oracle puts preference on choosing the time based index which leads to performance degradation for most of the queries.

  To address the issue:

  The hint NO_INDEX(…) was placed in the panda monitoring queries plus on 10th Nov I changed the partitioning boundary from 30 to 3 days (= smaller data segments)

  Or explore the approach of updating the Oracle table columns stats automatically on regular basis with our own custom values!

Issues with the PANDA live and archive data (3)



- With the separation of the PANDA and PANDAARCH data into different DB schemas (respectively, most recent 3 days and all the rest) the client code expected that this will be always true. However in the last reprocessing campaign that rule broke. (daily ~5-700 000 jobs)

  To address the issue:

  - I put new bulk copying procedure that works much faster than the conventional insertion is in place

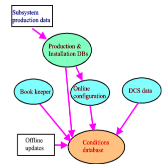
  - Oracle views (stored selects) will be put for comprising the data from both sources, the PANDA and PANDAARCH data (similar to the PVSS approach)

- Quite of a burden for the client code is setting the correct hints in the SELECTs

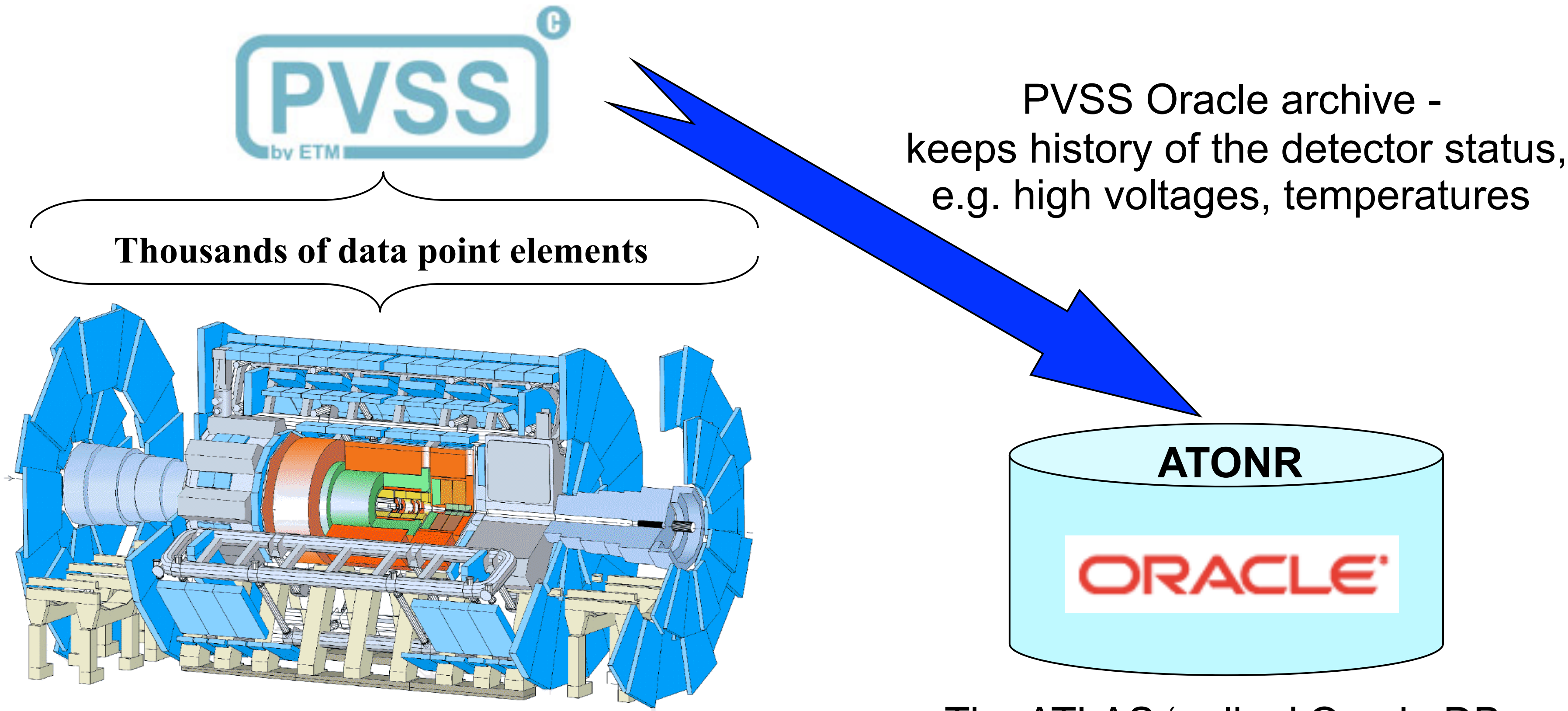
  To address the issue:

  The hints can be incorporated in the views themselves giving us (the DBAs) a freedom for changing them when we consider that action as beneficial.

Introduction to the PVSS system and its use in ATLAS

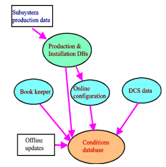


PVSS (Prozessvisualisierungs und Steuerungssystem) is a control and data acquisition system being in use in the LHC experiments since year 2000.



The ATLAS detector

The ATLAS 'online' Oracle DB



The ATLAS PVSS DB accounts and table desc.

- A database schema per subdetector (as total 14)

ATLAS_PVSSCSC
ATLAS_PVSSCSC_W
ATLAS_PVSSDCS
ATLAS_PVSSDCS_W
ATLAS_PVSSDSS
ATLAS_PVSSDSS_W
ATLAS_PVSSIDE
ATLAS_PVSSIDE_W
ATLAS_PVSSLAR
ATLAS_PVSSLAR_W
ATLAS_PVSSLUC
ATLAS_PVSSLUC_W
ATLAS_PVSSMDT
ATLAS_PVSSMDT_W
ATLAS_PVSSPIX
ATLAS_PVSSPIX_W
ATLAS_PVSSRPC
ATLAS_PVSSRPC_W
ATLAS_PVSSSCT
ATLAS_PVSSSCT_W
ATLAS_PVSSTDQ
ATLAS_PVSSTDQ_W
ATLAS_PVSSTGC
ATLAS_PVSSTGC_W
ATLAS_PVSSTIL
ATLAS_PVSSTIL_W
ATLAS_PVSSTRT
ATLAS_PVSSTRT_W

EVENTHISTORY_00000002
EVENTHISTORY_00000003
EVENTHISTORY_00000004
EVENTHISTORY_00000005
EVENTHISTORY_00000006
EVENTHISTORY_00000007
EVENTHISTORY_00000008
EVENTHISTORY_00000009
EVENTHISTORY_00000010
EVENTHISTORY_00000011
EVENTHISTORY_00000012
ELEMENT_ID
TS
VALUE_NUMBER
STATUS
MANAGER
TYPE_
USER_
SYS_ID
BASE
TEXT
VALUE_STRING
VALUE_TIMESTAMP
CORRVALUE_STRING
CORRVALUE_NUMBER
CORRVALUE_TIMESTAMP
OLVALUE_STRING
OLVALUE_NUMBER
OLVALUE_TIMESTAMP

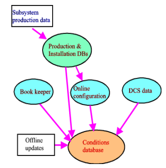
Table is 'switched' when it reaches a certain size and a view is updated to keep them together for the application to access the data (the EVENTHISTORY view)

Data point elements, in the LAR case are about 4500

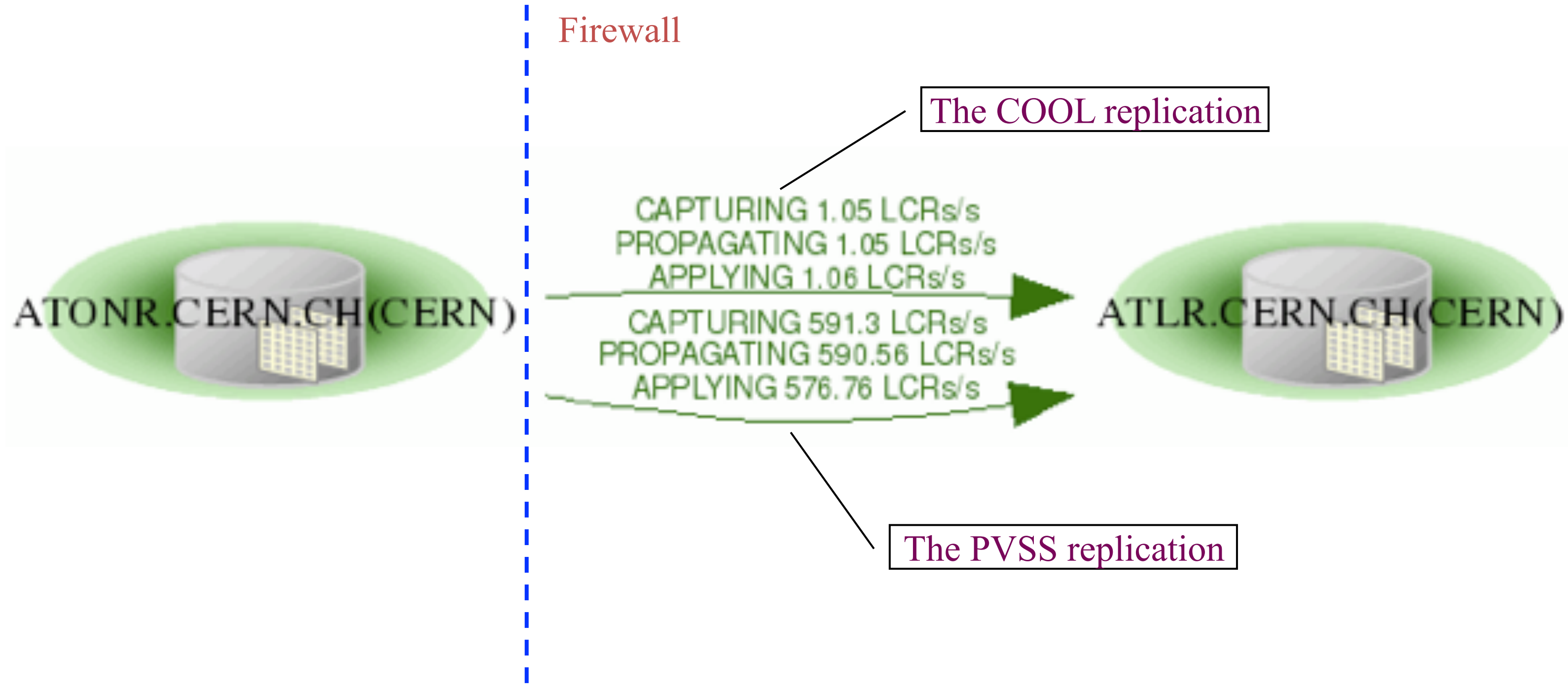
Not used from ATLAS, get NULL values, thus do not take occupy space

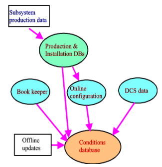
The row length is in the range 55-60 bytes

The need of having PVSS data replication from ATONR to ATLR (‘online’ => ‘offline’)



- In order to have the PVSS data accessible for the sub-detector expert analysis from the CERN public network and even from outside CERN a need for its replication showed up.





Sliding window for the PVSS Archive on the ATONR

- An idea of keeping only the data of the most recent 12 months on the ATONR (sliding window) popped up naturally.

  The reasons are:

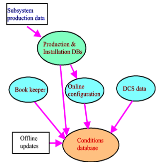
  - the operators in the ATLAS control room do NOT need to look furhter than 12 months in the past.

  - the complete archive is already on the ATLAS 'offline'

  - the 'online' DB is vital for the datataking and is wise to be kept smaller in case of a need of recovery operation.

  Currently the PVSS data (all tables and index segments) of the last 12 months occupies ~ 2.5 TB

Sliding window for the PVSS Archive on the ATONR (2)



1) That approach implies a move from the current « tablespace size threshold » to a « time interval » one – promising results from the tests

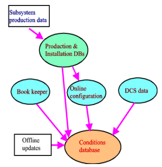
2) As each PVSS table resides into its own tablespace, for ATLAS that would mean ~ 100 tablespaces / year. Producing so many tablespaces (files) on the 'offline' side is not acceptable from administration POV.
To address the last, I introduced a special code in the Streams Apply handler which combines the PVSS tables of each sub-detector and an year in a common tablespace.

3) An important is to prevent table dropping on the source DB from being propagated on the destination DB.

A double protection is foreseen – a tagged session on the source DB and special code in the APPLY handler on the destination DB that discards any dropping table messages.

The tests so far are very positive. The move towards of putting the changes on production is to be agreed … Naturally this would be when there is a LHC technical stop

A generic problem with the Oracle statistics gathering approach



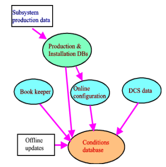
- For queries that are interested in data of the most recent hours, often get non-optimal execution plan and thus consume a lot of resources.

  e.g. For the ‘WHERE modiftime > SYSDATE - 1/2’ the Optimizer considers that there are only few rows relevant to that condition even if the statistics are very recent (computed from the last night). In reality, for a ½ day in several different schemas we could get tens or hundreds of thousands rows. With the wrong statistics Oracle produces non-optimal execution plans.

  A real case is where more then two indexes exist and Oracle decides for the inappropriate one or when a join of two tables is needed, Oracle chooses NESTED LOOPs within a index range scan is taking place instead of HASH JOIN. That leads to much more buffer reads (respectively IO and CPU)

  To address that problem, I had to ‘strength’ the queries with a lot of hints for instructing the Optimizer (e.g. INDEX_RS_ASC, NO_INDEX, CARDINALITY, USE_HASH ...etc)

Distributed Data Management System (DDM) – a move to a new organization of the traces data



- Each operation on ATLAS dataset level on the grid get registered on the DDM database (hosted on the ATLAS 'offline' database)
- So far the data was kept in a range partitioned table (an Oracle partition per month). Each partition having more than 100 mln rows and is expected to be more and more in the future.
- The table has an index on a column of timestamp type. This index often becomes a hot spot as contention is caused on high concurrent inserts.

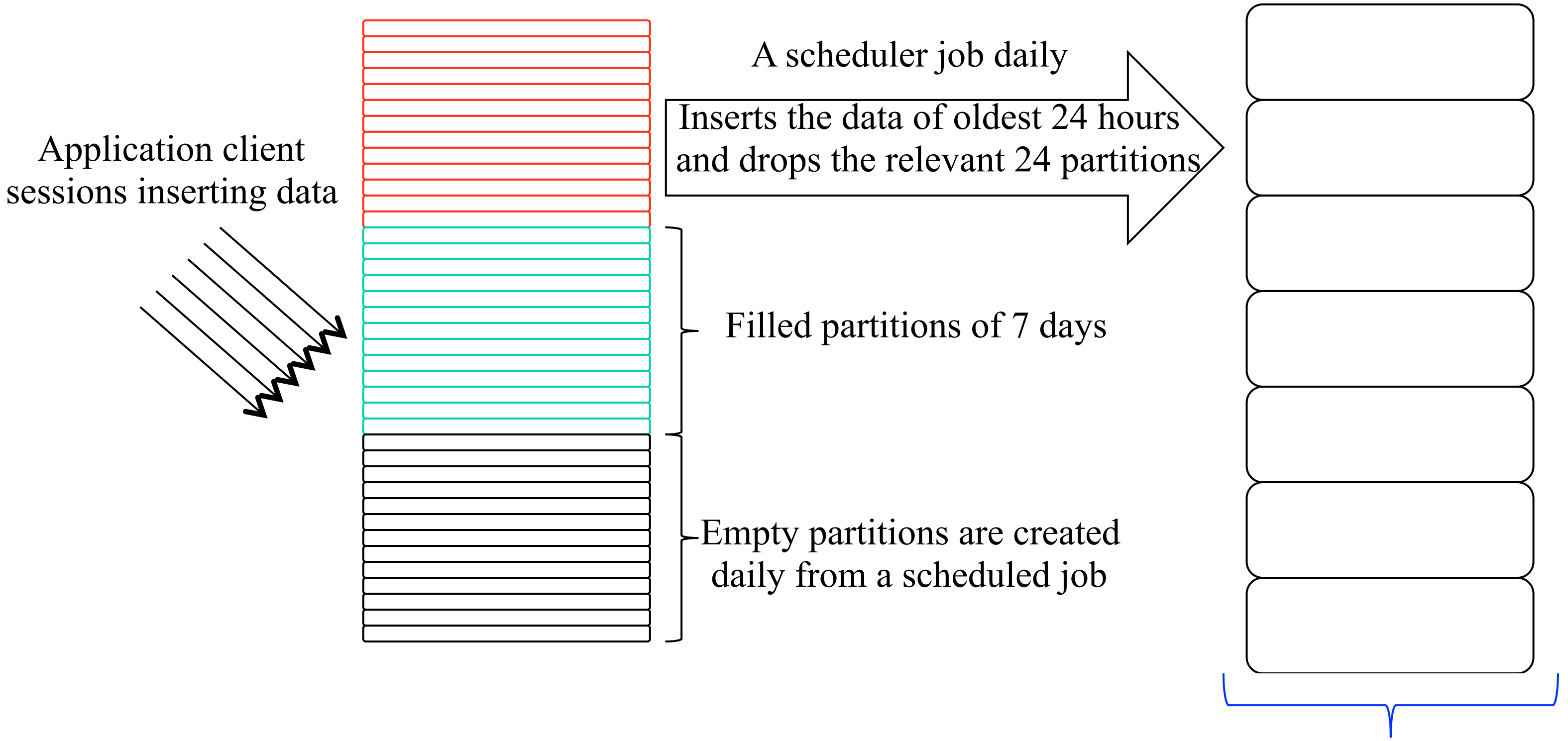
- To address the above, I designed a different organization.

  The idea is NOT to rely on any indexes, but rather have the data 'chopped' on pieces appropriate for the queries plus apply data compression as second step.

Schema of the new traces data organization

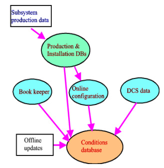
T_TRACES –
partitioned on a ‘time’ column.
Each partition covers a time range of an hour

T_TRACESARCH (with compression) –
partitioned on a ‘time’ column.
Each partition covers a time range of 7 days



1) New partitions are created from a scheduler job weekly
2) The compressed data segments occupy
three times less space in comparison with the
non-compressed T_TRACES ones

Conclusions



- With the current successful year of datataking the data volumes on the ATLAS databases grown progressively.

  The challenge is to keep the DB applications that rely on the Oracle databases well tuned and perform as the user expects.

- To fulfill the above new design and tuning techniques were (or planned to be) put in place (some of them presented into these slides)