STAR DBs

Near And Long Term Plans

Run 4 Online Stats

- Data Taken

  300 million events taken (this includes in addition to physics, fast detector, test runs, bad runs, etc.)

- Space ~63 gig

  + nightly backups and log files

- MySql Health – No problems
  - In fact – one table had > 9 million rows

Plans for Run 5

- Hardware Upgrades (onlsun1)
- Production Machines should be isolated
- Restructure of the eventTag db (DAQ ?)
  - 50 gig
- Further integrate bufferbox
- Fold in old – evp (another E450)
  - No single point of failure
  - Quasi –fault tolerance
  - Additional backup/ redundancy
  - Gives an addition platform for “some” development

Long Term Planning

- Although, hardware upgrades are sometimes effective… they are seldom the best answer for long term planning
  - Caveat – STAR DB servers stay somewhat current with available technology.
- Scalability is essential – so we need a programmatic and algorithmic changes
  - DAQ1000 max data taking @ 1kHz, this is still under investigation

ONLINE - Long Term – So Far

- Repository for Discarded Suns (E450s)
  - I like this machine, very robust and stable
- More sophisticated “cluster”
- Incremental minor upgrades.
  - This can still be expensive and we could eventually go the way of DAQ and get some cheap Linux boxes
- Make use of our online Linux “cluster”

OFFLINE – Configuration

- Master/Slave Replication
  - Master (all writes (inserts and updates))
    - robinson.star.bnl.gov
  - Slaves (all reads (selects…))
    - Two DNS Round-Robins
      - db.star.bnl.gov – 4 machines – used primarily for production
      - dbx.star.bnl.gov – 2 machines – used primarily for analysis
      - Six offsite slaves

What's New With MySQL

- Major upgrade to 4.0.20
  - Worked well
  - Added some new administrative features
- 4.1.4 is now beta – will be production by '05 so I've begun to test it.
- New table type - "cluster".

OFFLINE - Issues

- Performance, Performance, Performance

The Usual Suspects Are:

Database Configuration

Usage (queries – and programmatic algos)

Database design

Hardware

OFFLINE – Near Term (done)

- HARDWARE Upgrade

  - For dbx.star.bnl.gov swapped out one
    2-P3 1.4 / 1 gig – memory
    for
    two 2-3.06 Zeon HT / 3 gig - memory

  it was time and it HELPED!

Configuration

- We are presently optimized as far as how out dbs are set up
  - Indexes, buffers etc…
  - Actual queries are optimal
- With the upgrade came the query cache
  - Remembers an executed query and caches its results
  - Next time the server sees ‘exact’ same query, query is not executed, data is returned from the cache.

Query 1 - Tables

- When run directly on the server these queries run at 100hz or better.
- The cost of the API and network reduces this to ~58hz

  which means ~30 seconds to complete this pass

- Although 30 seconds is not a large amount of time, the information returned very rarely changes. Repeated queries of unchanged data, is always a “red flag” pointing to potential areas to optimize.

Query 2 - data

- Real Completion Time (seconds)
  CPU Completion Time (milliseconds)

  - SVT 57 (average of ten runs) 5 (average of ten runs)
    - Huge configuration – thousands of empty tables to fill
  - EMC 97 17.510
    - Large amount of data stored as blobs
  - TOF 8 0.80
  - TRG 13 .010
  - TRACKER 16 .840

Long Term

- Not a lot of low hanging fruit
- There are two passes
  - First rarely changes
    - heap table
    - cache
- Rethink our design
  - less blobs
  - smaller trees
- New Server Technology
  - distributed computing
  - MySql Cluster

Service Task Complete
THANK YOU!

Dmitry Arkhipkin and

Julia Zoulkarneeva

- Database Query Tool
- Will be available off the Database Page

FAQs

- In doubt please ask ?
- Where do I connect for analysis?
  - Check your configuration file
    - dbServers.xml in your home directory
  - dbx.star.bnl.gov
  - DO NOT connect to robinson (Please)
- TIME STAMPS
  - http://www.star.bnl.gov/STAR/comp/db/timestamp.ht

Timestamp Quick Tutorial 1

- Three Timestamps
  - beginTime
    - DAQ Time
  - entryTime
    - Time value is put into the database
  - Deactive
    - Time value is ‘turned off’

Timestamp Quick Tutorial 2

- Insert a record bt1 = daqTime = et1
- Insert second record bt2>bt1
- Insert third record bt3<bt1

Three validity windows

6) Valid for bt1 ->bt2
7) Valid for bt2 -> infinity
8) Valid to bt3 -> bt1

Timestamp Quick Tutorial 3

- A production time (pt1) was tagged after bt1 but before the entry of bt2.
- Later, after entries of bt2 and bt3 we want to reproduce data from pt1. Pt1 gets passed (dbv switch) to StDbLib and based on et1 ONLY gets data from bt1.

Timestamp Quick Tutorial 4

- Three entries are in and pt is done, time for a new production…BUT… a value must be changed….we DEACTIVATE.
- We don’t delete, or over write the value because we may still want to reproduce that “exact” original production.
- SQL has a where clause that enforces pt2>dt1 || dt = 0. A query can still be run that allows pt1<dt1 which means the first entry was valid at the time of pt1, therefore, it is used.