Using the LSST Catalog Database: Present & Future

Jacek Becla
Daniel Wang
Douglas Smith
Kian-Tat Lim

LSST All Hands Meeting
August 13 - 17, 2012
Outline

- The LSST Catalog Database Introduction
- Qserv introduction
- Queries supported / not supported
  - Feedback needed!!!
- SciSQL
- User interfaces and restrictions
- Qserv demo
- Using qserv
- Installing qserv
## Database Catalogs: Summer 2012 vs Baseline

**Summer 2012 [row count]**
- GoodSeeingCoadd [200]
- GoodSeeingCoadd_Metadata [100K]
- GoodSeeingSource [0.7M]
- Source [0.033B]
- GoodSeeingForcedSource [0.25B]
- RefObject [0.8M]
- RefObjMatch [1M]
- RefSrcMatch [34M]
- Science_Ccd_Exposure [0.1M]
- Science_Ccd_Exposure_Extra [0.1M]

**Baseline [row count: DR1, DR11]**
- N/A
- Science_Ccd_Exposure
- Object [13B, 38B]
- Source [247B, 4930B]
- ForcedSource [566B, 32100B]
- "external catalog(s)" [tbd]
- RefObjMatch [tbd]
- N/A
- Science_Ccd_Exposure
- Science_Ccd_Exposure
- Science_Ccd_Exposure
Database Catalogs: Summer 2012 vs Baseline

Size-wise

- <<1% of DR1
- independent database scaling tests done with 2B objects and 55B sources last year, ~10x planned for this year

Content-wise

- Substantially different
- Lots of experimental columns, columns supporting algorithm development
- Example: ~45 now vs 7 anticipated in ForcedSource

See schema browser at: http://lsst1.ncsa.uiuc.edu/schema for details
Database Configuration & Performance

Current configuration

- Single instance MySQL server
- Fairly powerful hardware
- Server and data moderately tuned

Performance

- Highly query-dependent. Seconds to few hours range seen
- 6-table join, ¼ billion row table, ~ million random rows fetched
- ... don’t expect miracles

Baseline

- Scalable, distributed, multi-server system (qserv)
- ~100 servers
- Come see Breakout 3.1 (Thu 4:00pm) for more details

Performance

- Few sec, few hours, few days depending on query/catalog
- ~ predictable (share scan speed)
Typical Database Query Load

Summer 2012

- PipeQA
  - lots of queries
- Several scientists running SDQA and science queries
  - <~2 simultaneous queries at any given time

Baseline

- ½ million queries per day:
- ~50 quick + ~20 complex queries at any given time

Common queries:
http://dev.lsstcorp.org/trac/wiki/dbQueries
Qserv – A Parallel and Distributed Database

- Chunking data, distributing across many machines
- Spatially contiguous pieces, with overlaps, no inter-node communication
- Hard challenges: data scale, I/O optimizations, efficient parallelism, operating under failure
- Built using existing parts (xrootd, mysql).
  - Like MapReduce, but providing a SQL interface and tuned for LSST needs (e.g., neighbor correlations)
- In R&D phase
  - Have working prototype
  - Scaling tests up to 150 nodes, on 10% DR1 data set size (30+ billion row table)
Qserv - Architecture
Caveats

- Partitioned tables processed in parallel if multi-node
- Spatial restriction optimizations
- Single-table queries
  - Accelerated for partitioned tables (Object, Source, ForcedSource):
    - Spatial restrictions
    - objectId indexing
- Join queries
  - Object-Object accelerated: $O(kn)$ instead of $O(N^2)$
  - Object-*Source accelerated: $O(kn)$ instead of $O(NM)$
  - Joins with other tables not accelerated, but these tables are small
Things That Don’t Work (yet)

- Queries other than SELECT
  - Update, drop, create, delete, load data, select into outfile...
- Sub-queries
- Functions, stored procedures must be loaded by admin
- Variables
  - Eg., SET
- Permissions
- Data loading
- Databases not named
- Databases with different partitioning schemes
Good-performance, Difficult & Impossible Queries

See http://dev.lsstcorp.org/trac/wiki/dbQueryDifficulty
<table>
<thead>
<tr>
<th>Schema</th>
<th>Scale (final DR)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object:</td>
<td>~500 columns</td>
</tr>
<tr>
<td>Source:</td>
<td>~100 columns</td>
</tr>
<tr>
<td>ForcedSource:</td>
<td>~10 columns</td>
</tr>
</tbody>
</table>
LSST Query Types

- Small
- Medium
- Large
- Super-large
- Impossible (technology)
- Impossible (resources)
- Single Object
- May include Sources or ForcedSources

```
SELECT *
FROM Object
JOIN Source USING (objectId)
WHERE objectId = ...
```
Medium

- Objects in small area (few arcmin)
- May include neighborhoods

```
SELECT *
FROM Object
WHERE qserv_areaSpec...
```
− All Object scan

SELECT ...
FROM Object
WHERE ...
Super-Large

- All Object/Object, small-radius spatial join
- All Object/Source scan
- All Object/ForcedSource scan
Super-Large Examples

SELECT A.objectId, B.objectId
FROM Object AS A, Object AS B
WHERE scisql_angSep(A.ra_SG, A.decl_SG, B.ra_SG, B.decl_SG) < 2.0/60
    AND ...

SELECT O.objectId, myFunction(S.taiMidPoint, S.psfFlux)
FROM Object AS O
    JOIN Source AS S USING (objectId)
WHERE ...
GROUP BY O.objectId
Impossible (Resources)

- Large intermediates
- Large computations
Impossible (Resources) Examples

```
SELECT A.objectId, b.ObjectId
FROM Object AS A, Object AS B
WHERE ...
LIMIT 100
```

```
SELECT *
FROM Object
ORDER BY rGaussianFlux DESC
```
Your Feedback Is CRITICAL

What queries? Performance expectations? Priority?

- If you won’t tell us what types of queries you will run, it is possible these queries will be unbearably slow, or simply rejected by the system.
SciSQL

What is it?

- Science-specific tools and extensions for MySQL: user defined functions and stored procedures

Contents

- Spherical geometry (HTM-based)
- Photometry
- Scalable statistics

Docs at: http://lsst1.ncsa.uiuc.edu/schema/sciSQL_0.3/

Code at: https://launchpad.net/scisql
Qserv – User Interfaces

– Standard MySQL today

– http://dev.lsstcorp.org/trac/wiki/dbScalableArchSQL

– http://dev.lsstcorp.org/trac/wiki/dbQservLimitations

– Certain things will get simpler – hiding queries like: http://dev.lsstcorp.org/trac/wiki/MySQLhints
Qserv Demo
Using Qserv

Current setup
- Lsst-db2 server at slac, old hardware
- Summer 2012 data
- Sandbox to play with. Non-production environment
- Expecting very few selected DM users to try “now”

Near future (~December 2012)
- On LSST10, beta
- For tests by a small group of users
- Pre-alpha “early adopter” distribution for other projects

Longer term (mid next year?)
- No special restrictions – comparable to access to mysql on lsst10 now
Qserv Roadmap

2011
- Focus on demonstrating scalability

2012
- Focus on usability, stability and manageability

2013
- Focus on speed and performance, in particular optimizing concurrent load (shared scans)

2014 and beyond
- Advanced features: user tables, metadata, cross-match, query fault recovery, support for updates, partition management, resource management, authentication and authorization, and more
Qserv Installation

For brave and seriously interested...

http://www.slac.stanford.edu/exp/lsst/qserv/download/

- This provides all needed packages for setup of a Qserv system
- Quick install notes are in Install.txt, start with a RHEL6/SL6 system, run qserv-install.
- Server management is handled with qserv-admin, to take things up and down, and partition and load data.
- Works with pt11 data loading now, bugs and fixes found with supporting the Summer 2012 data loading, these will go into the next release (in a week or so).
- Qserv setup and install has been tested at CC-IN2P3, and JHU and it is working there, along with SLAC. More installs will help work out bugs, and make things easier and better.
Discussion