

Organizing the Extremely Large LSST Database for Real-Time Astronomical Processing

ADASS

London, UK

September 23-26, 2007

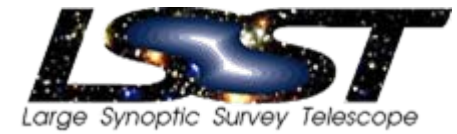
*Jacek Becla¹, Kian-Tat Lim¹, Serge Monkewitz²,
Maria Nieto-Santisteban³, Ani Thakar³*

¹ Stanford Linear Accelerator Center

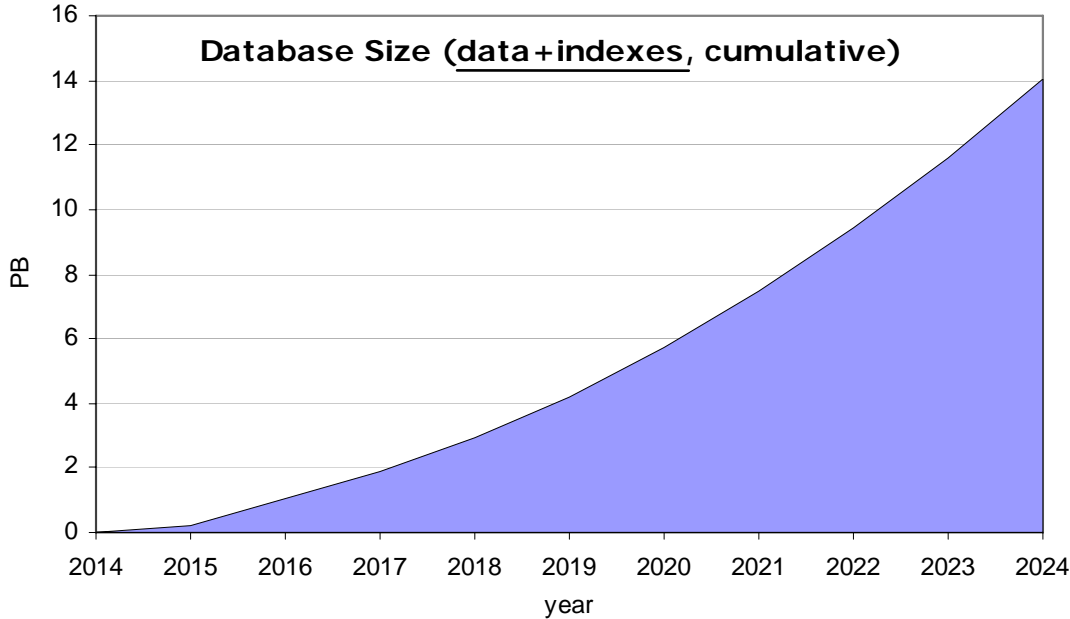
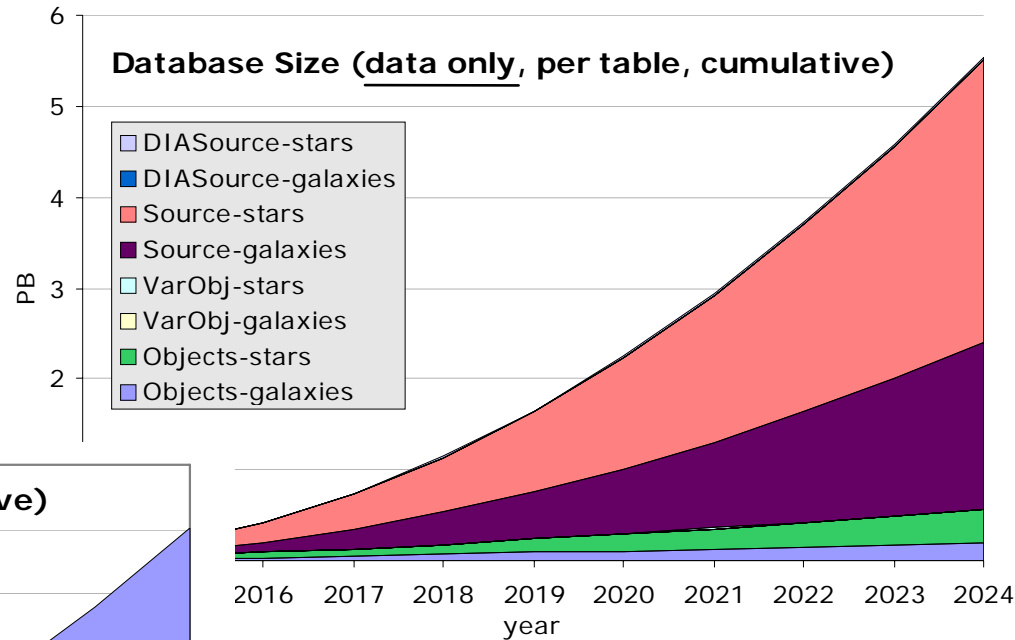
² California Institute Of Technology

³ Johns Hopkins University

Expected LSST Database Volumes



- 16×10^{12} rows
- 6 petabytes (data only)
- 14 petabytes (data and indexes)



SDSS

- 68×10^9 rows
- 0.02 petabyte

**LSST 700x larger,
but much later**

Real-Time Alert Generation

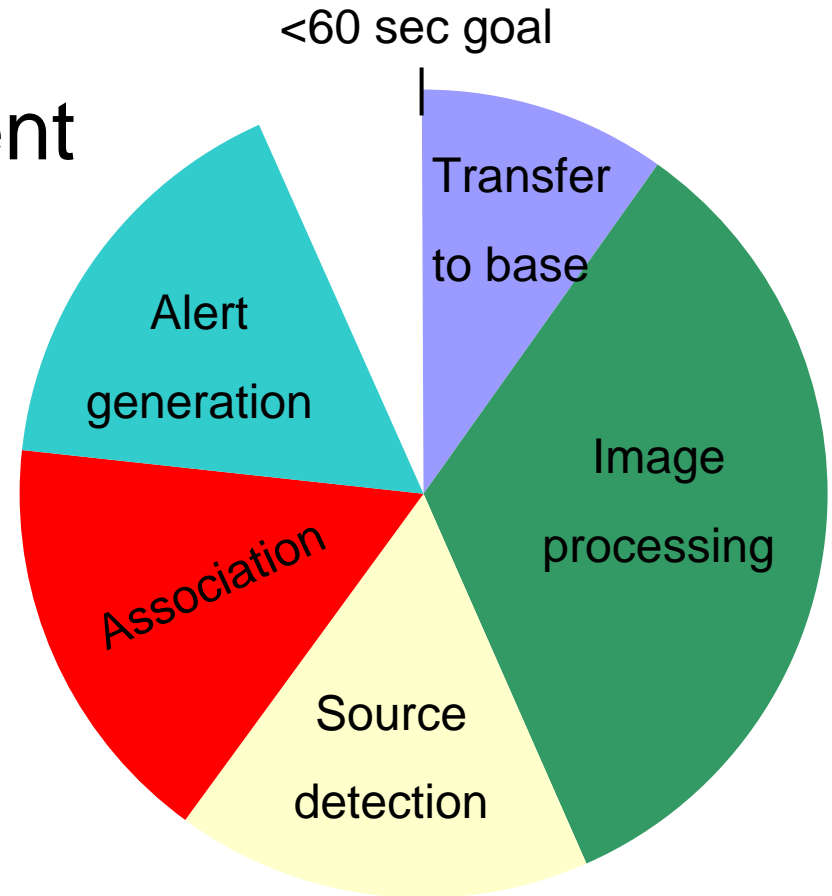
- Goal: generate transient alerts in under 60 sec

- Association <10 sec

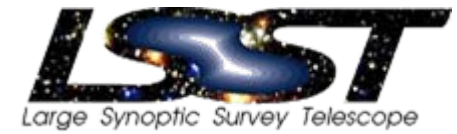
- Approach:

- Minimize I/O

- Maximize computational efficiency



Association Steps



Difference Source
Collection

*new detections,
aver. 40K, max 100K
per "visit"*

Object
Catalog

*historical data,
~50x10⁹*

MOPS
Catalog

*known moving
objects
w/predicted
positions*

Updates must be available when subsequent observations are processed

← cross-correlate →

mark matched sources

update (~40K)

← cross-correlate →

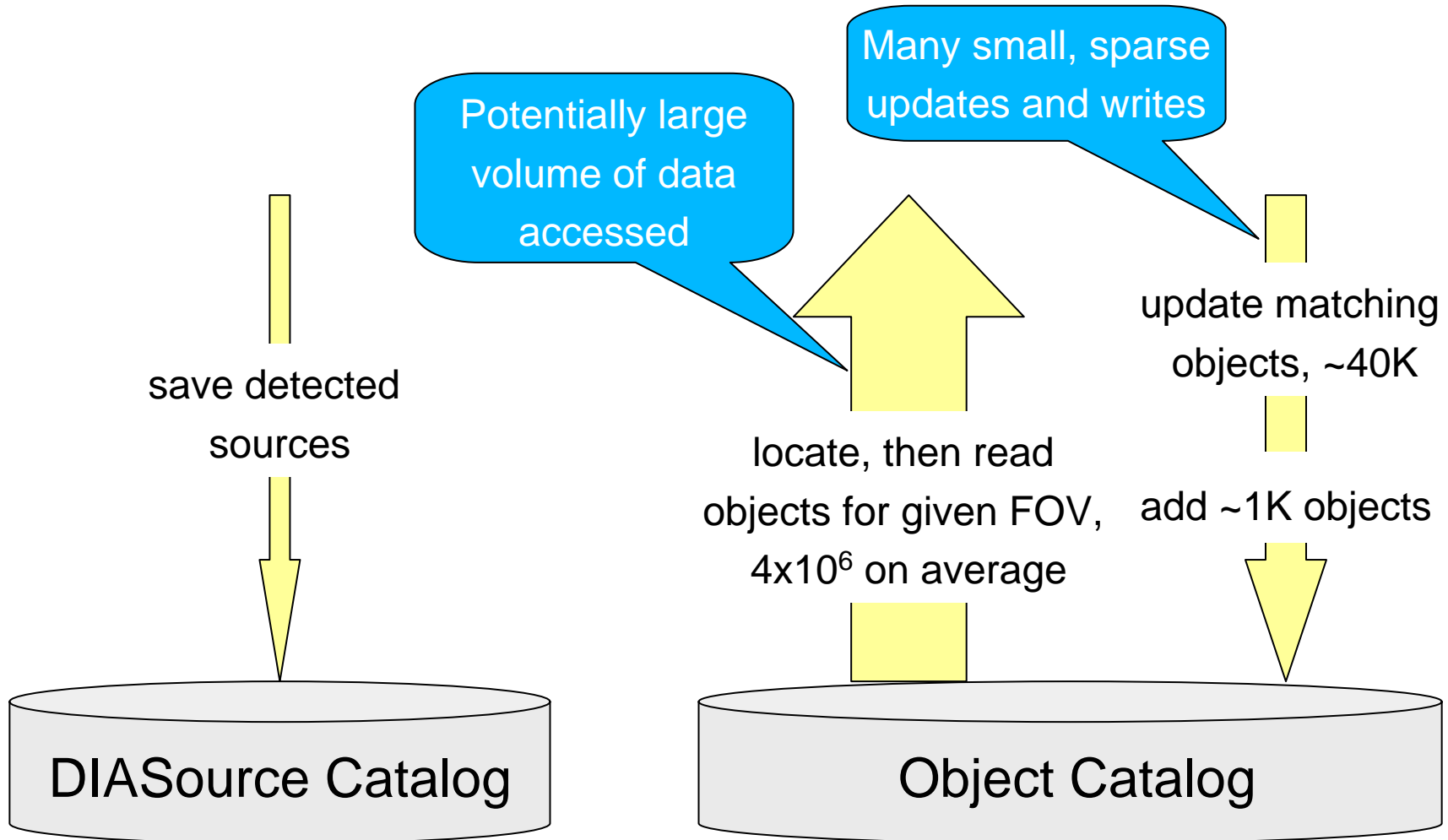
mark matched sources

add new objects for unmatched sources (~1K)

persist new sources

persist new objects and updates

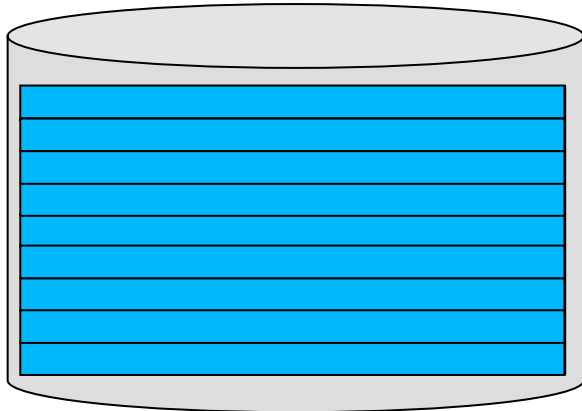
I/O: Large Volumes, Sparse Updates



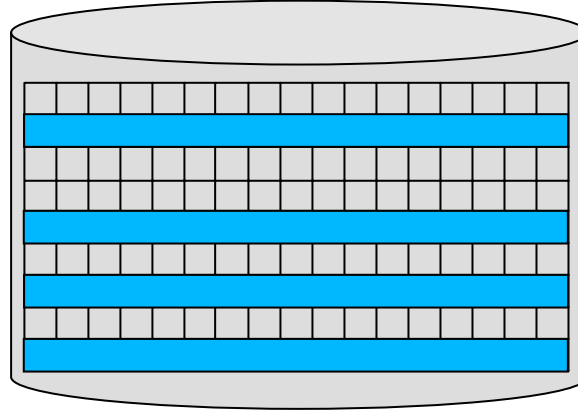
Minimizing and Sequentializing I/O

- **Match only against objects in or near processed FOV**
 - reduces problem from $100K \times 50 \times 10^9$ to $100K \times 4 \times 10^6$
- **Fetch only columns used during cross-match**
 - reduces problem from $4 \times 10^6 \times 1,658$ to $4 \times 10^6 \times 24 = \sim 100MB$
- **Cluster together data accessed together (spatially)**
- **Maintain specialized copy of data used by cross-match**
 - RA, Decl, objectId for objects
- **Partition data, round-robin across several disks**
- **Serve data from memory during time-critical period**
 - removes dependency on slow disks

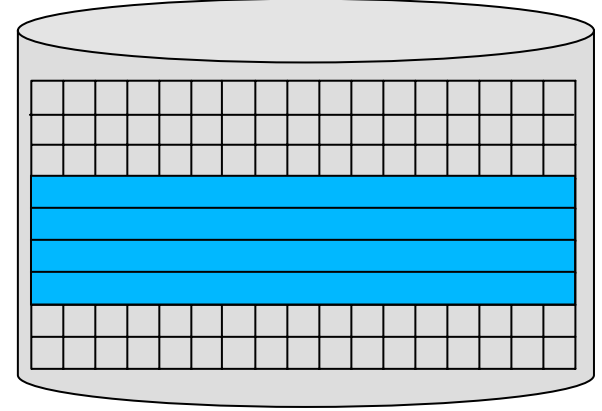
Minimizing and Sequentializing I/O



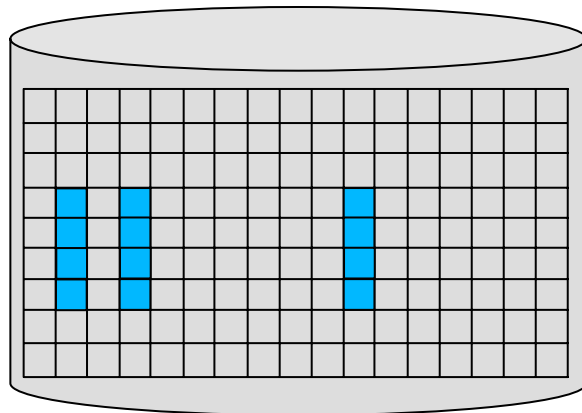
Scan whole catalog - naive



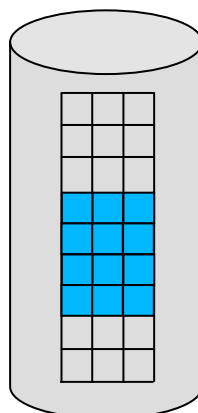
Match against processed FOV



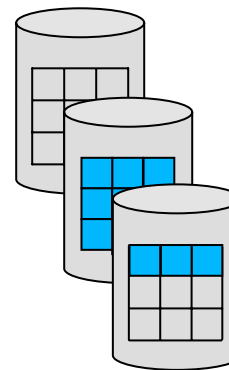
Cluster data spatially



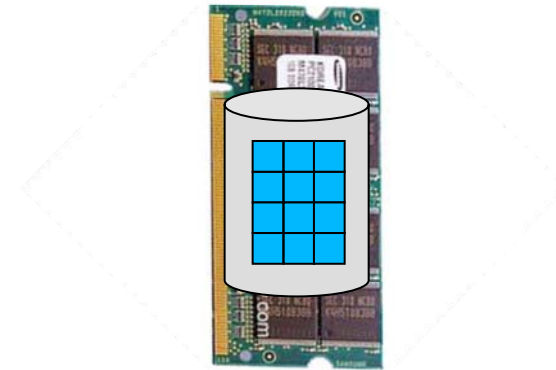
Fetch used columns only



Keep specialized copy



Partition data



Serve from memory

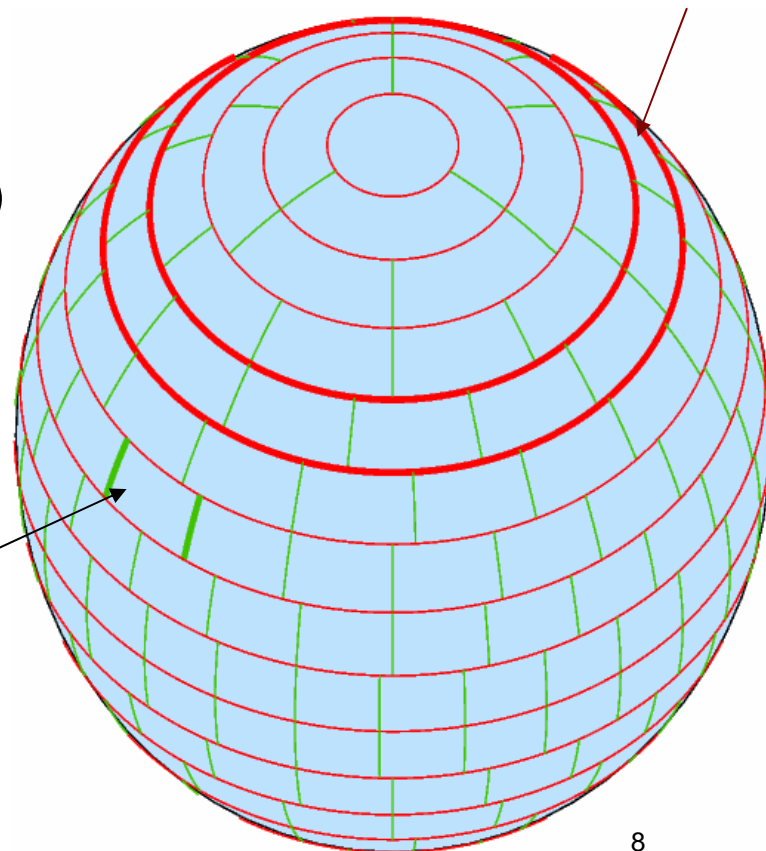
Partitioning Data

- **Divide difference sources and objects into declination stripes and physically partition into chunks**

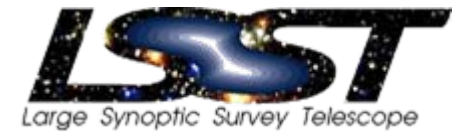
– optimal #partitions: $O(300,000)$

Stripes are a fraction of a field-of-view high (less than one to minimize wasted I/O around the circular FOV)

Chunks are one stripe height wide



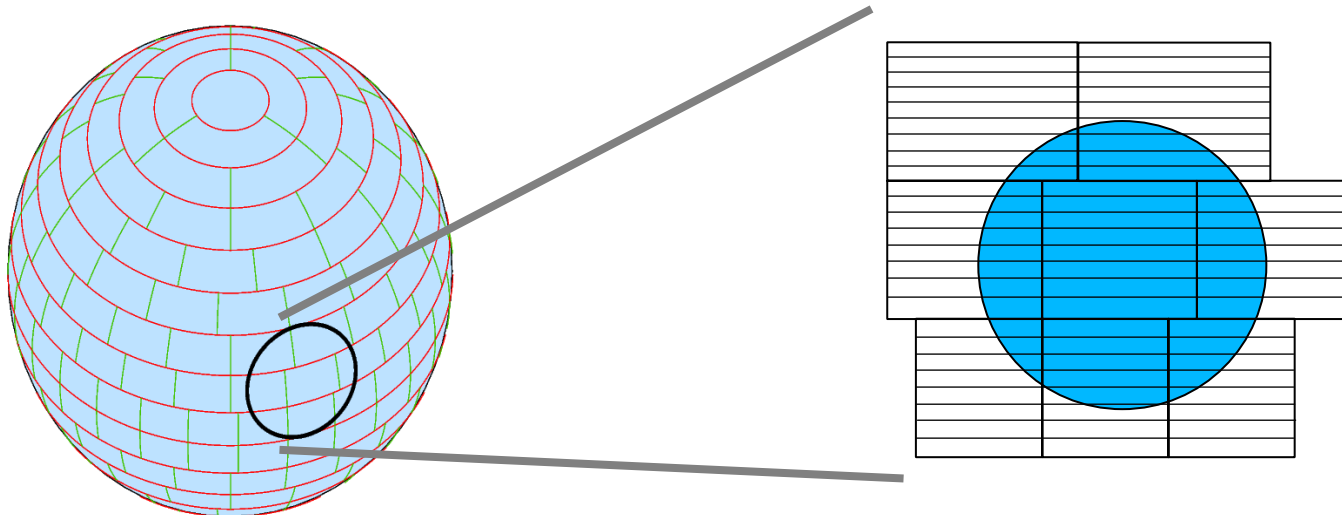
Serving Data From Memory



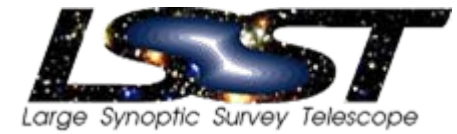
- **Pre-fetch variable objects to memory prior to processing when FOV known**
- **Time-critical period:**
 - lookup newly variable objects on disk (<1,000)
 - all other reads/writes from RAM (98+% of I/O)
- **Flush updates to disk after processing**
- **Get back fault tolerance by mirroring**

Maximizing Computational Efficiency: Algorithm

- **Generate declination zones dynamically in-memory**
 - **Zones are several search radii high**
 - more than one to minimize index overhead
- **Sort data within zones by RA on the fly**
- **Match difference sources and objects within zones**



Maximizing Computational Efficiency: Implementation



- **Push computation from database to application**
 - specialized algorithm faster than generic database approach
 - moving data to computation, not computation to data
 - works because we significantly reduced accessed data volume
 - result: significant speed up
- **Parallelize computation**
 - by zone

Results

- **Average real-LSST scale test**
 - ~40K sources vs ~4 million objects
- **Using a single 2002-era server**
 - SunFire V240 UltraSparc IIIi 1.5GHz
- **Results**
 - read and index object data: 9.4 sec
 - read and index source data: 0.91 sec
 - perform cross-match: 0.14 sec

Required: <30 sec

Required: <10 sec

- **LSST Real-Time spatial cross-match**
 - **large scale**
 - **stringent timing requirements**
 - **still easily doable on a single server**
with appropriate setup and data organization