Qserv: A Distributed Petascale Database for the LSST Catalogs

Fritz Mueller,[1] Igor Gaponenko,[1] John Gates,[1] Andrew Hanushevsky,[1] Fabrice Jammes,[2] Kian-Tat Lim,[1] Andrei Salnikov,[1] and Colin T. Slater[3]

[1]*SLAC National Accelerator Laboratory, 2575 Sand Hill Rd., Menlo Park, CA 94025, USA*

[2]*Université Clermont Auvergne, CNRS/IN2P3, Laboratoire de Physique de Clermont, F-63000 Clermont-Ferrand, France*

[3]*University of Washington, Dept. of Astronomy, Box 351580, Seattle, WA 98195, USA*

Abstract. Qserv is a distributed, shared-nothing, SQL database system being developed by the Vera Rubin Observatory to host the multi-petabyte astronomical catalogs that will be produced by the LSST survey. Here we sketch the basic design and operating principles of Qserv, and provide some updates on recent developments.

1. Introduction

The Legacy Survey of Space and Time (Ivezić et al. 2019) is a "deep fast wide" optical and near-IR survey of half the sky in *ugrizy* bands to *r* 27.5 (36 nJy) based on 825 visits over a 10-year period, to be carried out by the Vera C. Rubin Observatory in Chile.

The astronomical catalogs to be produced by the survey are notionally described in Jurić et al. (2021), and the corresponding database schema is described in Slater et al. (2019). By the 10th year of the survey, the catalog database is expected to run to approximately 60 trillion rows, requiring more than 10 petabytes of storage before considerations for replication, indices, or other overheads.

At the outset of construction, the project could not identify a mature database product capable of operating at these scales while also meeting project requirements for robust spherical geometry, predictable query response under heavy concurrent load, platform/hardware affordability, and open source licensing. The Rubin Observatory thus embarked on the design and implementation of Qserv.

2. The Qserv Approach

The evolution and progress of Qserv has been previously reported in Becla et al. (2006), Wang et al. (2011), and Becla et al. (2017). Qserv's principal design elements are as follows:

Distributed and Parallel. Qserv is designed as a parallel database, where catalog data is spatially partitioned and distributed uniformly over a collection of shared-nothing

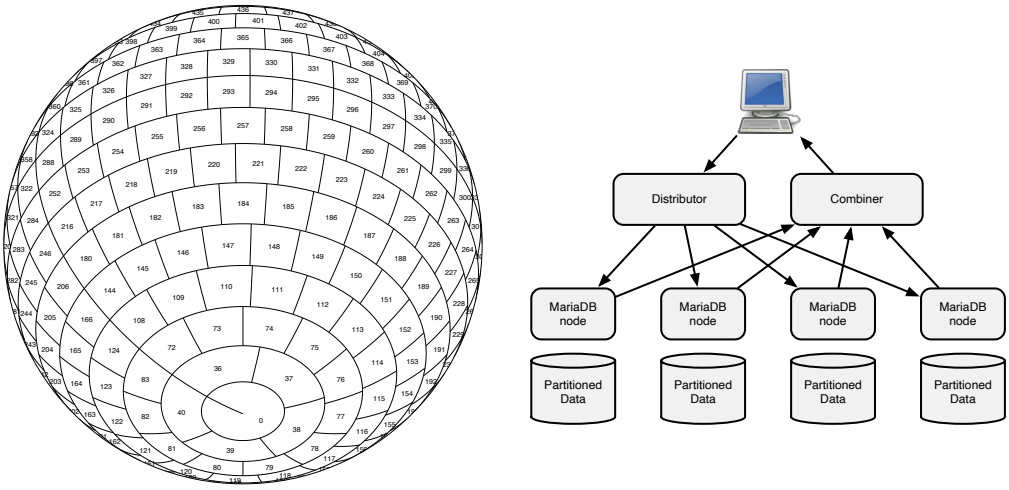

Figure 1. [Left] Qserv partitioning scheme: rings of declination plus polar caps; roughly equi-area. [Right] Qserv "map/reduce" query processing. Data is stationary on back-end nodes.

back-end nodes. Queries are handled in a "map/reduce" paradigm: after arriving at a SQL proxy, queries are analyzed for spatial coverage and rewritten into a collection of supporting per-partition queries which are then distributed to involved back-end nodes for execution ("map" phase). The results of these per-partition queries are subsequently returned to the proxy node where they are aggregated and any necessary summary statistics computed ("reduce" phase). Results are then returned to the user.

Partitioning and Replication. Qserv divides data into spatial partitions of roughly equal area. Since partitions are small with respect to higher-density areas of the survey and spread over back-end nodes using a non-area-based scheme, density-differential-induced skew is averaged out among the nodes.

Qserv additionally stores a precomputed amount of overlapping data from spatially adjacent partitions alongside each partition. Using this data, spatial joins can be computed correctly within a preset distance without needing data from other partitions that may be located on other nodes.

Data partitions are dynamically replicated across back-end nodes for resilience and high-availability. If a node becomes unavailable, Qserv is capable of transparently re-dispatching in-flight per-partition queries involving partitions on that node toward other available partition replicas. Partitions that become under-represented are re-replicated and the cluster is periodically re-balanced in the background. This same facility enables the addition of nodes to a Qserv cluster or the draining of nodes for maintenance as online operations.

Shared Scans. Shared scans are a key component of the Qserv architecture and enable predictable query throughput on fixed hardware under heavily concurrent loads. As per-partition queries arrive at worker nodes, they are held and grouped by involved partition(s). Each node scans its local partitions in a cyclic and coordinated fashion. When a scan visits a partition its table backing files are locked into memory and all

pending batched queries involving that partition are executed; these batched queries thus share I/O, thereby reducing the total I/O load. The tables are then released and the scan proceeds to the next local partition.

Multiple scan queues per node are supported to accommodate scans that execute at significantly different speeds due to different amounts of I/O required. In this way, time-consuming I/O intensive scans need not hold up scans of other tables that could otherwise proceed more quickly. Some capacity at each node is also reserved to be able to quickly answer ad-hoc, non-scan queries to partitions on that node.

Per-partition queries may join a scan queue in any phase as they arrive at a node, and remain on the scan until the involved partition has been visited.

Designed for Purpose. Qserv, while broadly general purpose, has been specifically designed to meet the LSST catalog use-case. This has allowed leverage of some significant simplifications to keep complexity low and to keep the project feasible for a small development team. For example: the LSST catalogs are to be released annually and are a read-only product; thus Qserv does not need to address SQL UPDATE semantics nor does it need to encompass a full transactional model.

Astronomical use demands robust spherical geometry within the database; Qserv has addressed this by providing a library of precise spherical geometry UDFs which are made available for use of per-partition queries on the back-end worker nodes.

Qserv has leveraged mature, open source, off-the-shelf components wherever possible; these include MariaDB, the XRootD distributed file system, and Google protobufs.

Qserv is fully containerized and has a Kubernetes operator to facilitate deployment on Kubernetes, either on-prem or in the commodity cloud. We have found about a 15% performance impact when running with network storage on the Google cloud compared to running with locally attached storage on an on-prem cluster.

3. Performance Targets and Sizing

Performance targets for Qserv in the context of LSST have been established as follows. Under a combined, sustained load of 100 concurrent small ad-hoc queries and 50 concurrent full scans:

Retrieve common attributes of an object, or all objects within a small area of the sky	< 10 seconds
Scan entire sky (billions of objects)	≈ 1 hour
Deeper analysis (incl. object extended attribute BLOBs)	≈ 8 hours
Detection or forced photometry full scans	≈ 12 hours

Since Qserv is principally I/O limited, clusters are sized according to target full-scan times using the following simple relations:

$$\frac{\text{table size(s)}}{\text{sustained I/O rate} * \text{workers}} = \text{target scan time}$$

$$\frac{(\text{table size(s)} + \text{overlap} + \text{local indices}) * \text{replication}}{\text{workers}} = \text{local worker storage}$$

With consideration of the latest LSST data sizing models (O'Mullane et al. 2021), this works out to a cluster of approximately 100 worker nodes for year 1 of the survey, peaking at approximately 450 worker nodes in year 6, with approximately 45 TiB of local storage attached to each node.

4. Conclusions

Qserv's horizontal scaling capabilities have been demonstrated out to several hundreds of nodes, and the system has been successful to date in meeting the construction and commissioning needs of the Rubin Observatory serving pre-cursor datasets from clusters located at the National Center for Supercomputing Applications and in the Google cloud. Most recently this has included support for a community of several hundred users as an integrated component of the Rubin Science Platform for Rubin Observatory's Data Preview 0.2 (O'Mullane et al. 2021; O'Mullane et al. 2022). Additional clusters are now being commissioned at each of the LSST international data centers (USA, France, UK). We hope the astronomical community may also find use for Qserv and its design ideas beyond the LSST survey.

Acknowledgments. This material or work is supported in part by the National Science Foundation through Cooperative Agreement AST-1258333 and Cooperative Support Agreement AST1836783 managed by the Association of Universities for Research in Astronomy (AURA), and the Department of Energy under Contract No. DE-AC02-76SF00515 with the SLAC National Accelerator Laboratory managed by Stanford University.

References

Becla, J., Hanushevsky, A., Nikolaev, S., Abdulla, G., Szalay, A., Nieto-Santisteban, M., Thakar, A., & Gray, J. 2006, in Observatory Operations: Strategies, Processes, and Systems, edited by D. R. Silva, & R. E. Doxsey, vol. 6270 of Proc. SPIE, 62700R. arXiv:cs/0604112

Becla, J., et al. 2017, LSST Data Management Database Design. URL https://LDM-135.lsst.io/

Ivezić, Ž., et al. 2019, ApJ, 873, 111. arXiv:0805.2366

Jurić, M., et al. 2021, Data Products Definition Document. URL https://lse-163.lsst.io/

O'Mullane, W., Economou, F., Huang, F., Speck, D., Chiang, H.-F., Graham, M. L., Allbery, R., Banek, C., Sick, J., Thornton, A. J., Masciarelli, J., Lim, K.-T., Mueller, F., Padolski, S., Jenness, T., Krughoff, K. S., Gower, M., Guy, L. P., & Dubois-Felsmann, G. P. 2021, arXiv e-prints, arXiv:2111.15030. arXiv:2111.15030

O'Mullane, W., et al. 2021, DM sizing model and cost plan for construction and operations. URL https://DMTN-135.lsst.io/

— 2022, Data Preview 0.2 and Operations rehearsal for DRP. URL https://RTN-041.lsst.io/

Slater, C., et al. 2019, LSST Database Baseline Schema. URL https://LDM-148.lsst.io/

Wang, D. L., Monkewitz, S. M., Lim, K.-T., & Becla, J. 2011, in State of the Practice Reports (New York, NY, USA: ACM), SC '11, 12:1. URL http://doi.acm.org/10.1145/2063348.2063364