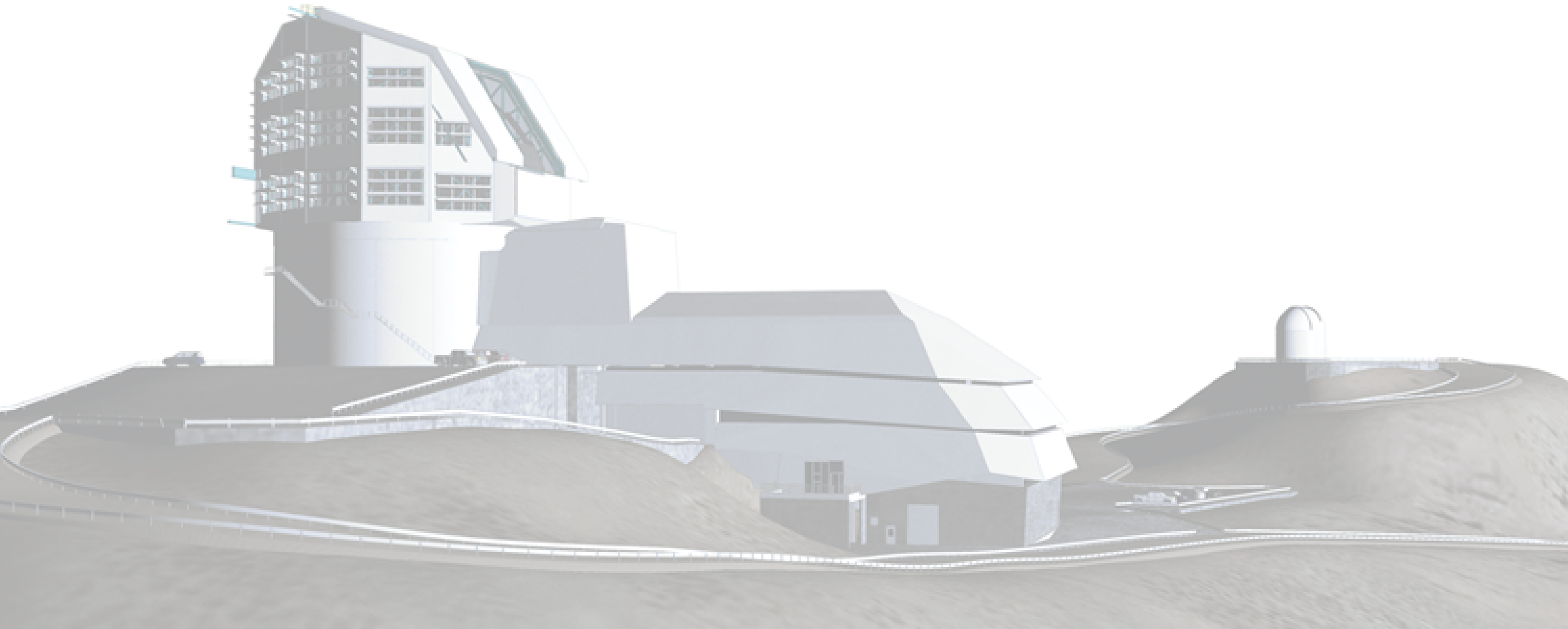
Vera C. Rubin Observatory
Rubin Observatory Operations

Data Preview 0.2 and Operations rehearsal for DRP.

William O'Mullane, Yusra Alsayyad, Hsin-Fang Chiang, Frossie Economou, Huan Lin, Fritz Mueller, Tim Jenness

RTN-041

Latest Revision: 2022-09-16



Abstract

DM delivered software to operations to perform processing of the DESC DC2 data as well as enhancements to the portal and Qserv for interaction with the results. The release of this was called Data Preview 0.2 and the production of the data products and publication of them were carried out in an operational manner. This provides valuable insights for operational data releases.

Change Record

Version	Date	Description	Owner name
1	2022-08-02	Unreleased.	William O'Mullane

Document source location: `https://github.com/lsst/rtn-041`

Contents

Data Preview 0.2 and Operations rehearsal for DRP.

1 Introduction

Between December 2021 and May 2022 the DESC DC2 (LSST Dark Energy Science Collaboration (LSST DESC) et al., 2021) was reprocessed with Rubin Science pipelines V23 [DMTR-351].[1] Between May and June the catalogs were ingested to Qserv, tutorials and documentation were updated and the Data Preview 0.2 data release was made on time at the end of June. A number of procedures were developed and practiced to achieve this. Planning for DP0.1 and DP0.2 are in RTN-001. We shall discuss the process in the following sections:

- Management and communication is discussed in Section 2
- An overview of the processing is given in Section 3
- Quality assurance and feedback to processing is discussed in Section 4
- Community engagement, tutorials and documentation are discussed in Section 6

2 Management and communication

Here we cover the management structures in place for DP0.2 this includes the groups and meetings like the change control for the pipeline version.

2.1 Oversight

The Data Production Leadership Team (DPLT) consisted of representatives from all the teams involved in the data preview process as well as the data facilities. The DPLT met fortnightly to discuss any issues and minutes were recorded on Confluence.

The membership of the DPLT was:

- William O'Mullane

[1] `https://pipelines.lsst.io/v/v23_0_2/index.html`

- Bob Blum
- Leanne Guy
- Frossie Economou
- Tim Jenness
- Yusra AlSayyad
- Hsin-Fang Chiang
- Michelle Butler (NCSA)
- Richard Dubois (USDF)
- George Beckett (UKDF)
- Fabio Hernandez (FrDF)

This encompasses practically the entire operations DPLT as depicted in Figure 1

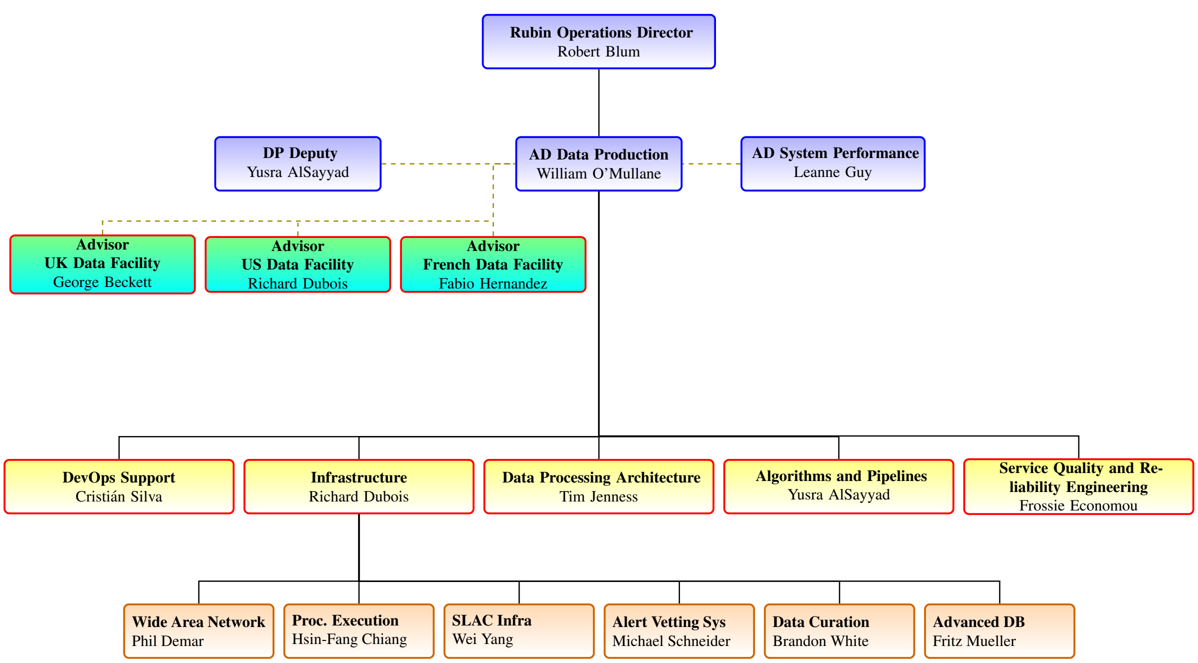


FIGURE 1: Data Production Leadership in operations - most of which was involved in DP0.2.

2.2 Coordination

During the data preview 0.2 process there was a regular coordination meeting every two weeks (out of phase with the DPLT meeting) with minutes recorded on Confluence. This meeting was attended by all the people directly involved in the data preview process: management, the processing infrastructure team, the science platform team, the execution team, the pipelines team, the verification and validation team, and the community engagement team.

This allowed the different teams to report on status and bring up any issues that needed to be addressed and made everyone aware of progress. Data Preview 0.1 had been released during this period and this allowed us to also include feedback from the Community Engagement Team as they interacted with the existing delegates and prepared updated tutorials for DP0.2.

Data Facility representatives were present at this meeting even though all the processing was being done on the Interim Data Facility at Google (O'Mullane et al., 2021).

2.3 Work Management

We used Jira to track work related to the Data Preview. Epics and milestones were created in the `PREOPS` Jira Project. Story tickets were then attached to each epic but in order to properly integrate the work into existing Data Management processes, any tickets that would result in code changes in pipelines software or middleware packages were created in the Data Management Jira project.

The status of the epics and how they related to the relevant milestones was monitored as part of the weekly coordination or DPLT meetings.

2.4 Change Control

Data Preview 0.2 used v23.0.x of the LSST Science Pipelines Software and that was derived from a weekly release from September 2021 (`w.2022.40`). We decided to group processing into distinct "steps" that allowed updates to the software used in later stages of processing to be worked on whilst earlier steps were executing.

We continued to want to use the v23 release for all data processing and that required that we had a process to determine which patches would need to be back-ported to the release branch as needed before each step could begin.

The Data Management Change Control Board (DMCCB) and DPLT delegated authority to a new Data Release Steering Committee that had the following membership:

- Yusra AlSayyad, representing the pipelines team.
- Leanne Guy and Colin Slater, representing the verification and validation team.
- Hsin-Fang Chiang, representing the execution team.
- Tim Jenness, representing the data processing architecture team.

The Board met weekly on Tuesday at 8:30am Pacific Time and also had a Slack channel to discuss any issues that would come up between meetings. Minutes for the meetings were recorded on Confluence.

The process for deciding on a back-port is as follows:

1. A request is made that a ticket should be applied to the release branch by applying a `backport-v23` tag to the Jira ticket.
2. The board would then discuss the relative merits of the back-port and if approved a `backport-approved` label would be added.
3. The work on the back-port would then be scheduled by the relevant T/CAM following instructions in the developer guide.[2]
4. Once the code is on the `v23.0.x` branch a `backport-done` label would be applied.

A Jira query was constructed to find all the tickets and track their porting status. There were 61 tickets approved for back-porting as part of the Data Preview 0.2 and version 23 release process. If a ticket was rejected its label was removed, making it hard to determine counts for the number of tickets in that category. Three tickets were left in the requesting state in case

[2] `https://developer.lsst.io/work/backports.html`

they were needed, one is for a clean-up to the database schema that was discovered after we had finalized the processing; another was for an improvement to the graph-building efficiency but would have involved a very difficult back-port because there had been a package reorganization since the release branch had been created; and the final ticket was an improvement to the matched catalog filtering.

Once all the necessary back-porting has been completed for a specific step, the release manager would be instructed to start the process of creating a new patch release of the Science Pipelines. During DP0.2 we made two additional formal releases of the version 23 software: v23.0.1 and v23.0.2. This allowed us to state which release was used for each step, although we ensured that changes in later patch releases would not affect the processing from steps that were already completed using older patch releases.

3 DP0.2 processing on Google

The data processing was done using the Production and Distributed Analysis System (PanDA; DMTN-168) on the Interim Data Facility (IDF). The PanDA system handled workflow orchestration and job retries. Based on pre-production testing, six PanDA queues (`moderatemem`, `highmem`, `highmem-non-preempt`, `extra-highmem`, `extra-highmem-non-preempt`, `merge`) were deployed with the Google Kubernetes Engine (GKE) clusters of different configurations such as the memory-to-CPU ratio to accommodate different types of jobs. A separate `merge` queue was deployed to limit database connection. The clusters automatically scaled their sizes based on the demand of the workload. Besides infrastructure logs, pipeline job logs were streamed into Google Cloud Logging for real-time searching and troubleshooting. A daily error log summaries page was also set up.

The production processing was organized into seven logical "steps". The high level workflow of workflows and the step organization is described in RTN-001 Sect. 2.1.4. Pipelines, V&V, and Processing teams all focused on one step at a time. Before the production processing started in each step, "pilot runs" with candidate software were carried out and signed off by V&V and Pipelines teams (Sect 2).

The workflow generation and submission were done via the PanDA BPS plugin from a notebook of the integration instance of the RSP. The BPS YAML configurations can be found at a GitHub repo (https://github.com/lsst-dm/dp02-processing). Workflow progress was tracked via PanDA's iDDS monitoring page deployed at CERN and the JIRA-based campaign tooling (RTN-023). On many occasions rescue workflows skipping successful jobs were run.

The processing took place between Dec 18, 2021 and May 16, 2022 and the total cpu usage over the course of DP0.2 was approximately 2.5M core-hour; the compute resource usage is summarized in RTN-039.

Notable issues which came up during the production were summarized as follows.

- step1
  - The RSP's "large" memory option has 12 GB and imposed a limit on the size (number of quanta) of the workflows that could be submitted, as otherwise quantum graph generation failed due to an out-of-memory error. A "huge" memory option became available in late January 2022, doubling the memory to 24 GB and enabling larger workflows to be submitted for subsequent processing steps.
  - The quantum graph build time increased from about 1.5 hours for submissions at the beginning of step1 processing to about 8 hours at the end. This was due to the use of a single output chained collection, which needed to be examined in quantum graph generation, but which grew ever larger as production proceeded, thus increasing the graph generation time. To avoid this issue, starting in step3, we used separate output collections for the different workflow submissions in a given processing step, and then joined the separate collections together once all workflows were completed.
  - To improve efficiency, we clustered together 315 quanta (consisting of different step1 tasks for multiple detectors in the same exposure) into a single job on the PanDA system. However, if one quantum failed among these 315 quanta, the whole job stopped and led to unattempted quanta. The intent for this behavior was so that downstream jobs would know about upstream failures, but in this case, the detectors were independent of each other, and there were no downstream jobs in our step1 workflows. To resolve this issue of unattempted quanta, we ran rescue workflows where quanta for different detectors were no longer clustered with each other. Subsequently the default behavior was changed so that an individual failure among clustered quanta would not stop the remaining quanta from being attempted.
- step2

  - We expected that step2 would need a total of 3 workflows (≈ 6600 visits each), based on quantum graph generation tests, but execution butler creation became the bottleneck instead, and we instead required 14 smaller workflows (≈ 1500 visits each). For example, for the first of these smaller step2 workflows, quantum graph generation took 23 min, but execution butler creation took 7 hours, while job compute time was 2.5 hours (wall clock). Before step3 processing started, this issue was resolved in DM-33345, and execution butler generation time was dramatically improved, e.g., from 7 hours down to 20 minutes in this step2 example, or from 1 hour down to 6.5 minutes for a 1-tract step3 workflow.

- step3
  - Large numbers of simultaneous assembleCoadd (= image coaddition) jobs led to "too many request" errors reading from the object store (PREOPS-1034). Algorithmically, the rate spike was because the coaddition was done in small chunks and data reading happened for each chunk and each input warp. This could be alleviated by caching input files instead of requesting them from the object store each time. The caching configuration in the butler repo was changed so not to expire anything in coaddition. The first 7 step3 workflows needed to be redone due to this problem. This also led to the investigation of the "hidden" errors in the coadds, where coadd images were reported ok but actually had problems (DM-33786). More details are discussed in Sect. 4). Later we found out that the initial fix of the caching was too aggressive and led to out-of-disk-space errors for a couple of large faro tasks (matchCatalogsPatchMultiBand and matchCatalogsTract), but subsequently caching was better optimized to be sufficient for coadds, but not so much to cause disk space problems for the faro tasks.
  - Long-running forcedPhotCoadd jobs were failing/re-attempting at a much higher rate than before, caused by a change in the PanDA pilot version, which reset the job heartbeat timeout limit back to its default of 2 hours, whereas it had been set to 20 hours previously. This issue also uncovered the need for more frequent heartbeat message logging for long-running forcedPhotCoadd (DM-33854) and faro tasks (DM-33820). The timeout limit was set again to 20 hours to allow step3 production to continue efficiently.
  - About once every two workflows or so, a deblend job or two failed due to out-of-memory error (DM-33690), because of very bright/large objects on the coadd image. These failures were fixed in rescue workflows, but required long run times (>10 hrs) and large memory (≈ 40 GB) using the `extra-highmem` queue, though ulti-

mately with deblending results that were likely unreliable. The most extreme example (tract 4648, patch 29) took 12 unsuccessful attempts before finally succeeding in the `extra-highmem-non-preempt` queue (DM-33947), after 2 days 22.7 hours run time, 190 GB memory, and extra efforts from the PanDA team to bypass heartbeat logging issues. This was the last image processing job to be completed in step3. DM-33690 implemented deblending configuration changes to skip these problematic deblends, so this should not be an issue subsequent to DP0.2.

  - The large faro tasks matchCatalogsPatchMultiBand and especially matchCatalogsTract took long run times and large memory ($\approx$ 130 GB for the latter). They needed to be run on the `extra-highmem` queue and were also prone to preemption and multiple re-attempts due to heartbeat logging issues (DM-33820).
  - Some delays arose when jobs became stuck as they were unexpectedly scheduled to a non-IDF queue, which had its "region" set erroneously to "LSST", same as for the IDF queues. This issue was corrected by the PanDA support team.
  - There were also some delays due to unexpected consequences from PanDA server and client updates, again resolved by the PanDA support team.

- step4
  - An attempt was made to use the Google Artifact Registry (GAR) instead of Docker Hub to download the LSST processing stack. However, this resulted in intermittent authentication problems that led to 20-50% job failures in three step4 workflows. Despite an attempt to reconfigure the IDF computing cluster involved, the authentication problems continued, and we reverted to using Docker Hub and reran the affected workflows.
  - We also saw a drop in the maximum number of running jobs in the IDF queue involved, from nearly 4000 down to as few as about 2000, possibly related to the above authentication issues. PanDA support increased the maximum number of new worker nodes created from 50 to 80 to help stabilize the number of running jobs back to nearly 4000.

- step5
  - We encountered a very slow start to jobs running on the PanDA system due to the large number of jobs per task (>100,000) in the first step5 workflow. This was solved by reducing the maximum number of jobs per task from the default value of 70,000 down to 30,000, so that a single large task was divided into smaller "chunks" (each

with <30,000 jobs) that ran much more efficiently in the PanDA system. We do need an iDDS server with more resources.

- step6, step7
  - Minor or no issues.

4 Data Product Quality Assurance

Description on the checks run, perhaps a few plots . Anything to do differently next time ?

5 Science Platform, user front end for DP0.2

DP0.1 upgrade image services any new issues but mainly deployment and control, issue handling

5.1 Qserv

Qserv is a horizontally scalable parallel SQL database system, developed by the Rubin Observatory specifically to support the LSST production catalog use case. Qserv is further described in LDM-135.

The DP0.2 simulated survey covers about 1/60th of the anticipated on-sky area of the full survey (roughly 300/18,000 sq. degrees), at about half the anticipated observational depth of the full survey. The generated catalogs contain ~136 billion rows in total. On disk, including necessary indexes, this amounts to ~30 TiB of database storage. A Qserv cluster with one head node and five worker nodes was comissioned in the IDF cloud computing environment to host and serve this data.

5.1.1 Enhancements delivered during DP0.2

- **Ingest systems:** At these scales, a trouble-free database ingest from DRP pipeline outputs to a complete online database takes several days. All catalog data must be format converted from the parquet outputs as delivered by DRP to CSV files needed for efficient load at the Qserv worker nodes. Data after conversion must be sharded, distributed, loaded at worker nodes, and appropriate indices compiled. A significant fraction of the Qserv codebase is concerned only with the management, tracking, and automation of these ingest activities.

Enhancement of these Qserv subsystems continued during the course of the DP0.2 exercise, and improvements were deployed continuously into production. These improvments were aimed at streamlining the ongoing ingest campaigns, and included such features as provision of fully asynchronous task APIs to upstream ingest tooling, refinements to the ingest state machine for improved observability of ongoing ingest activities, and finer-grained control over publication/retraction/replacement life-cycle of individual tables.

- **Diagnostic monitoring:** Qserv provides its own in-built administration dashboard, which has proved to be an indispensible tool for monitoring and troublshooting operating instances. This subsystem also saw lots of improvement during the course of DP0.2, including instrumentation of many additional aspects of Qserv related to ingest and query processing activities, an enhanced query monitor / query history, and overall performance improvments to the administration dashboard itself.
- **Optimized point-in-polygon spherical geometric queries:** Rather late in timeline for DP0.2 a decision was taken to host IVOA ObsCore observational metadata within Qserv, since doing so would enable a much richer interaction model with image data and image services being offered for the first time in DP0.2. This interaction model required support for point-in-polygon spherical geometry queries, and needed spatial indexing features beyond the previously established scope of Qserv for DP0.2 in order to perform acceptably. A workable strategy for efficiently processing these sorts of queries based on integration of several bits of existing functionality was developed and successfully delivered into production.
- **Photometry UDFs:** Some photometry-related UDFs were developed and contributed upstream to the SciSQL project (https://github.com/smonkewitz/scisql), along with general modernization and infrastructural improvements of SciSQL. These UDFs can simplify construction of queries against the LSST data model, which to date is committed to provision of linear flux measurements rather than logarithmic magnitudes (DOCUMENT-27758).

5.1.2 Some lessons learned

DP0.2 proved to be a very valuable exercise in shaking down Qserv and surrounding tooling and operating processes. Some take-aways related providing catalog database service for DP0.2:

- **Timely schema metadata is hard to get.** Rubin's middleware has an extremely flexible data model. While this provides a lot of agility in pipeline development, when it comes to provision of catalog services at some point before database ingestion a concrete schema commitment must be made.

  While Rubin does have an accepted format for such schema descriptions and a sufficient source control and deployment practice around this, the schema descriptions themselves run to the many thousands of lines and nobody chomps at the bit to make sure these are maintained, current, complete, and error free. Otherwise ready-to-go catalogs have on more than one occasion been held up in release by the need for last-minute revisions to the schema descriptions.

  Remediation: much has been done already to drive schema consumers toward a single source of truth and to streamline the deployment and update of schemas, once updates are committed to source control. Since schema description deliverables have consistently been seen to lag, it remains for management to appropriately pre-load the demand for these.

- **Data curation – try (even) harder.** Often times, catalog data curation issues are not easily detected until the "end of the line", when a completed data release is presented for database ingestion. Representation issues, key constraints, and referential integrity issues may not be apparent until a full set of tables is available. Some curation issues may not even be detected until *after* database ingestion, when the catalogs can first conveniently be queried in the large.

  For DP0.2, there was a plan to mitigate turbulence at the end of the line by obtainin and ingesting a representationally complete subset of catalog products early in the release cycle. In practice, this had only limited success. Inevitable delays on all fronts compressed the time between availability of the subset products and the full release products. Additionally, sequencing of DRP activity meant while some subset tables were available usefully early in the release cycle, others were not available until much later, closer to the full release. Predictably, curation issues *did* exist that thus were not identified until fairly late in the release cycle, and release of several tables in the data products was delayed while these tables were re-processed upstream and then re-ingested into the database.

Remediation: work with DRP to see if a more complete set of tables can feasibly be obtained earlier in the release cycle. Schedule subset production and ingest even more conservatively.

- **Watch out for automated test lacunae.** Qserv contains some distributed reference match machinery to facilitate performant joins between catalogs. It turns out that DP0.2 includes a truth table and associated match table which depend upon this functionality. Though this feature is not otherwise commonly in use, the Qserv development team maintained a high level of confidence that it was ready to go, because it had known coverage in the automated integration test suites, and "if it was broken, we'd know about it."

  As it turned out, the relevant integration tests had been commented out of the test suite sometime in the previous year as a temporary measure while working through some containerization/deployment issues, test coverage had never been subsequently restored, and the uncovered feature, when put into use, was found to have some minor issues.

  Remediation: a thorough survey of the test suites was conducted, and commented/disabled tests were updated as necessary and returned to active service.

6 Community engagement

Organisation and execution of the tutorials, docs, assemblies ..

A References

[LDM-135], Becla, J., Wang, D., Monkewitz, S., et al., 2017, *Data Management Database Design*, LDM-135, URL `https://ldm-135.lsst.io/`,
Vera C. Rubin Observatory Data Management Controlled Document

[DMTR-351], Carlin, J., 2022, *Characterization Metric Report: Science Pipelines Version 23.0.0*, DMTR-351, URL `https://dmtr-351.lsst.io/`,
Vera C. Rubin Observatory Data Management Test Report

[DOCUMENT-27758], Ivezić, Ž., the LSST Project Science Team, 2018, *On the Choice of LSST Flux Units*, DOCUMENT-27758, URL `https://pstn-001.lsst.io/`,
Vera C. Rubin Observatory

LSST Dark Energy Science Collaboration (LSST DESC), Abolfathi, B., Alonso, D., et al., 2021, ApJS, 253, 31 (`arXiv:2010.05926`), doi:10.3847/1538-4365/abd62c, ADS Link

[RTN-001], O'Mullane, W., 2021, *Data Preview 0: Definition and planning.*, RTN-001, URL `https://rtn-001.lsst.io/`,
Vera C. Rubin Observatory Technical Note

O'Mullane, W., Economou, F., Huang, F., et al., 2021, arXiv e-prints, arXiv:2111.15030 (`arXiv:2111.15030`), ADS Link

[DMTN-168], Padolski, S., Ye, S., 2022, *Running Science Pipelines using PanDA*, DMTN-168, URL `https://dmtn-168.lsst.io/`,
Vera C. Rubin Observatory Data Management Technical Note

[RTN-039], Yanny, B., 2022, *Compute Resource Usage of DP0.2 production run*, RTN-039, URL `https://rtn-039.lsst.io/`,
Vera C. Rubin Observatory Technical Note

[RTN-023], Yanny, B., Slater, C., Padolski, S., et al., 2021, *Campaign Tooling – tools for generating, monitoring and tracking data processing campaigns*, RTN-023, URL `https://rtn-023.lsst.io/`,
Vera C. Rubin Observatory Technical Note

B Acronyms

Acronym	Description
CAM	CAMera
CSV	Comma Separated Values
DC2	Data Challenge 2 (DESC)
DESC	Dark Energy Science Collaboration
DM	Data Management

DMCCB	DM Change Control Board
DMTR	DM Test Report
DP0	Data Preview 0
DPLT	DP Leadership Team
DRP	Data Release Production
FrDF	French Data Facility
IDF	Interim Data Facility
IVOA	International Virtual-Observatory Alliance
LDM	LSST Data Management (Document Handle)
LSST	Legacy Survey of Space and Time (formerly Large Synoptic Survey Telescope)
NCSA	National Center for Supercomputing Applications
OPS	Operations
RTN	Rubin Technical Note
SQL	Structured Query Language
T/CAM	Technical/Control (or Cost) Account Manager
UKDF	United Kingdom Data Facility
USDF	United States Data Facility