

UC Berkeley

UC Berkeley Previously Published Works

Title

From Raw to Curated Data: A Lakehouse Approach for Scientific Workflows

Permalink

<https://escholarship.org/uc/item/7049c4mp>

Authors

Berket, Karlo

Huck, Patrick

Mathis, Tyler

et al.

Publication Date

2026-03-18

Peer reviewed



From Raw to Curated Data

A Lakehouse Approach for Scientific Workflows

March 2026

Abstract

This report provides a technical overview of how to go from raw to curated data in three stages using a lakehouse approach. We focus on the application of open source tools in scientific use cases (while noting parallels to enterprise and commercial alternatives). Our goal is to provide scientific data managers and infrastructure providers with a common frame of reference for understanding and applying modern lakehouse technologies and approaches.

Technical report LBNL-2001748

<https://escholarship.org/uc/item/7049c4mp>

Report Authors

Karlo Berket, Patrick Huck, Tyler Mathis (The Materials Project), Dan Gunter, Shane Canon, Drew Paine

The authors fully rewrote this document to ensure quality and clarity after leveraging ChatGPT for an initial outline.

HPDF Project Team

Thomas Jefferson National Accelerator Facility (TJNAF): Graham Hayes, Ilya Baldin, Ellen Kriskovich, Amitoj Singh

Lawrence Berkeley National Laboratory (LBNL): Ana Kupresanin, Lavanya Ramakrishnan (retired), Richard Shane Canon, Becci Totzke, Drew Paine, Karlo Berket, Johanna Cohoon, Ben Maxwell, Daniel Gunter, Sufi Kaur, Shreyas Cholia, Patrick Huck, Valerie Hendrix

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research under contracts DE-AC05-06OR23177 (TJNAF) and DE-AC02-05CH11231 (LBNL).

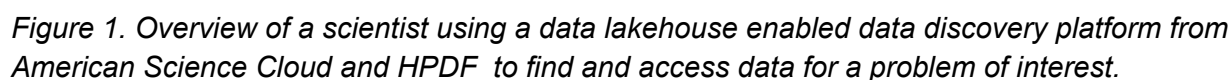
This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

This report reflects the analysis and opinions of the authors. Insights and recommendations will evolve as available information changes and details are further clarified.

Table of Contents

Table of Contents.....	3
Introduction.....	4
Lakehouse Fundamentals and the Apache Iceberg Ecosystem.....	5
Comparison with other ecosystems.....	5
Three Zones of Data: Raw, Cleaned, and Curated.....	5
Zone One — Raw Data.....	6
Zone Two — Cleaned Data.....	6
Zone Three — Curated Data.....	7
The value of the zoned approach.....	7
Applying the Raw–Cleaned–Curated Pipeline to Scientific Data.....	8
Wrangling Varied Datasets.....	8
The Materials Project — A Case Study in Scientific Lakehouse Implementation.....	9
Overview — Zone Mapping and Technology Choices.....	9
The Migration Journey – From Legacy Infrastructure to a Cloud-Native Lakehouse.....	10
Zone Two: The Most Challenging Transition.....	10
Cloud-Native Compute on Zone Two.....	11
Empowering Users Through the Data Lake.....	11
Conclusion.....	12
References.....	13

Scientific experiments and simulations often produce large volumes of raw data that are messy, unstructured, and heterogeneous. Converting raw experimental or simulation data into AI-ready (curated) datasets is a fundamental challenge in scientific data engineering [9]. Researchers consequently need to transform this raw input into clean, structured, and curated datasets suitable for analysis, machine learning, or agentic AI uses. A modern solution to manage this process is the data lakehouse architecture, which combines the flexibility of data lakes (for storing raw data) with the robust management and performance features of data warehouses [16]. Early data lakes provided cheap, scalable storage for raw data but often lacked governance and structure, leading to disorganized, messy data. The lakehouse architecture addresses this by layering data zones (or stages) that progressively provide improved data quality and structure, all within a unified system that supports robust querying and governance — for example like the case in Figure 1 where a scientist is querying for relevant data using a discovery agent. This document provides an introduction to a data lakehouse approach, how to apply it to scientific use cases, and a real-world example from The Materials Project.



Lakehouse Fundamentals and the Apache Iceberg Ecosystem

A data lakehouse is a modern data architecture that enables analytics on both raw and structured data in one platform. Figure 1 illustrates how this architecture is incorporated into the American Science Cloud [3] platform in support of the Genesis Mission [13]. It combines the scalable storage and format variety of a data lake with the transactional integrity and performance optimizations typical of a data warehouse [19]. In practice, this means scientists can store all their data (raw instrument logs, simulation outputs, images, etc.) in a central repository (e.g. cloud object storage) and still query and refine it using SQL engines or AI tools. Crucially, lakehouses make use of additional metadata in the form of table formats like Iceberg [6], Delta Lake [10] or Hudi [5]. These metadata layers offer useful abstraction and management of raw files to end users.

Apache Iceberg is a leading open-source table format that underpins many lakehouse implementations. Iceberg introduces a transaction-aware metadata layer on top of data lake storage. In essence, instead of dealing with individual files, one can treat a collection of files as a table with full ACID (Atomicity, Consistency, Isolation, and Durability) [1] transactions and schema management. Iceberg supports features like schema evolution (adding or changing columns without breaking queries), time-travel queries (querying past snapshots of data), hidden partitioning (improving query speed without manual file path hacks), and atomic commits for consistent writes. This brings database-like reliability to data lakes. For example, Iceberg can track versioned schema changes and let you rollback or audit data easily via time-travel features. It works with various query processing engines (Spark [7], Trino [11], Flink [4], etc.), making it flexible in scientific environments where different tools might be used. Iceberg's strong support for schema evolution and multi-engine interoperability makes it especially well-suited to incremental, multi-stage data workflows.

Comparison with other ecosystems

Databricks pioneered this "lakehouse" concept with their medallion architecture (Bronze–Silver–Gold layers) [10] utilizing the Delta Lake table format. Apache Hudi is an open data lakehouse platform, built on the Hudi open table format, with a focus on an incremental processing framework [5]. Whether using Databricks, Hudi, or another platform built on open source formats, the fundamental idea is the same: use layered data refinement in a lakehouse to reliably go from raw data to high-quality data products. Here we focus on Apache Iceberg as the open, ecosystem-neutral option with strong vendor support [15], but the principles apply broadly. Iceberg can even be used within Databricks as an alternative table format [24].

Three Zones of Data: Raw, Cleaned, and Curated

A best practice for lakehouse data organization is to separate data into three zones: raw, cleaned, and curated. Each zone has a specific role in the pipeline from raw input to curated output (see Figure 2 for an illustration). Each zone builds upon the previous, incrementally improving data quality, structure, and context.

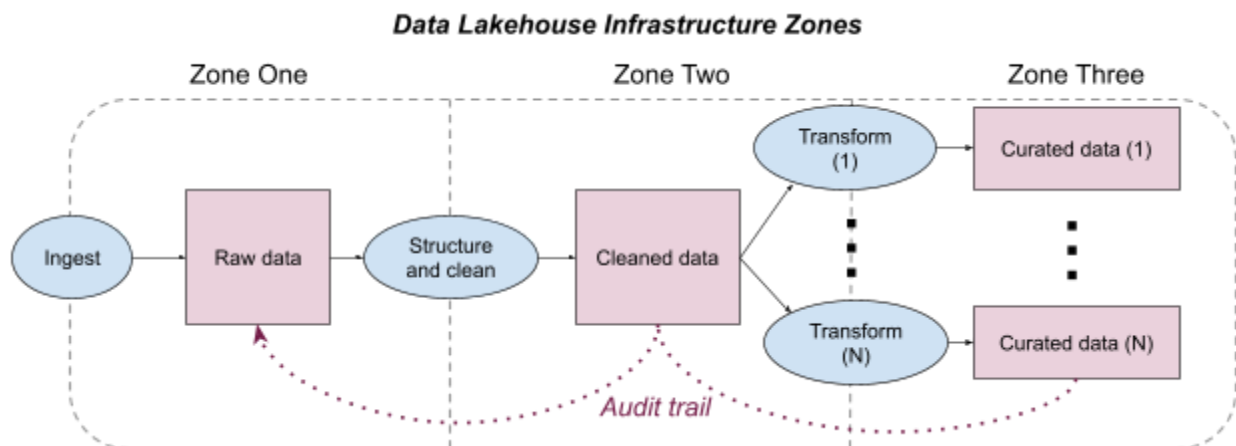


Figure 2. Conceptual diagram of a pipeline for transforming raw input to curated output via three zones.

Zone One — Raw Data

The raw zone ingests data in its original form with minimal or no transformation. This zone contains the original source-of-truth data exactly as it was produced by experiments or simulations — e.g. untidied instrument logs, sensor readings, simulation outputs. The goal is to preserve a historical archive of the original data (including any errors or quirks) for long-term traceability. Data here may be semi-structured or unstructured (e.g. nested JSON, CSV dumps, binary files) and might have evolving schemas or irregular formats. Access to the raw layer is often strictly limited to maintain integrity, since it can contain sensitive or messy data straight from the source. In a scientific context, examples range from raw telescope images, experimental CSV logs, or simulation results generated on an HPC system. The raw layer ensures nothing is lost.

To ensure rigorous provenance and auditability, raw data ingestion may leverage cloud object storage metadata. By appending immutable tags to S3 objects upon creation—such as the IAM user identity, instrument ID, pipeline run ID, and the git commit hash of the ingestion script—teams can maintain a strict audit trail. This guarantees that if a processing mistake occurs downstream, data stewards can not only revert to the original, unadulterated data, but also trace the artifact back to its exact origin and the user who generated it, ensuring absolute reproducibility and that no information is ever lost.

Zone Two — Cleaned Data

The cleaned zone takes raw data and applies deterministic transformations and cleaning steps to make the data more structured, queryable, and reliable. Typical operations include parsing and casting data types, normalizing formats and units, deduplicating records, handling missing values, and basic data quality validation (e.g. removing obviously corrupt entries) [9]. Light integrations or joins might also occur here, for example, linking an experiment ID to a lookup table of experiment metadata, or merging multiple raw files by time stamp. The result of the cleaned layer is that raw bytes are turned into query-ready tables with a consistent schema. In scientific workflows, this might involve steps like calibrating sensor readings, converting file

formats (e.g. from proprietary binary to standard Parquet or CSV), normalizing units across datasets, or filtering out experimental noise. The data is now cleaned and enriched enough that data scientists and engineers can begin to analyze it with common tools. This zone greatly improves data quality and consistency, making downstream work more reliable.

Zone Three — Curated Data

The curated zone produces the final refined datasets that are ready for analytics, machine learning, or specific domain use-cases. Data in this layer is highly structured, aggregated, and optimized for end-user consumption. While in a business context this might mean tables tailored for intelligence reports, in a scientific setting, it typically consists of feature matrices for AI models, statistical summary tables, or highly specialized domain-specific data products. Curated data is often the result of joining multiple cleaned (Zone Two) tables, performing advanced computations or statistical aggregations, and applying final transformations such as feature engineering. For example, from a validated dataset of simulation results, researchers might derive a curated table containing pre-computed features (e.g., averages, anomalies, or event flags) per simulation run. This curated table is then ready to feed into a machine learning model. Curated data provides business or application-centric views that are fully vetted and ready for consumption, whether by AI or other analysis tools. Downstream users (analysts, scientists, AI models) can trust that curated data is of high quality, governed, and analysis-ready. Because this tier represents the "gold standard" of a project's output, we recommend that scientific teams assign data steward(s) that make the decisions on what data belongs in this zone.

During curation, data may be enhanced with expert-applied labels or annotations for AI pipelines. It is crucial to distinguish these feature-level annotations from governance metadata. In the curated zone, "labeling" involves row-level or data-point-level attributes—such as flagging a specific sensor pulse as a 'failure' to serve as target variables (y) for machine learning. Conversely, governance metadata manages dataset-level attributes and discovery tags (e.g., "Contains PII" or "Project Ownership"), facilitating high-level compliance and searchability across the broader lakehouse ecosystem.

The value of the zoned approach

A zoned approach fundamentally enforces a separation of concerns for different stages of data. One team or process can ingest raw data while another cleans it, and yet another derives final insights, all in parallel once a strong pipeline is established. This model also supports better long-term governance and reproducibility. Given raw data is preserved, any issues found in the final curated layer can be traced backwards (with audits or time-travel on tables) to see what went wrong and the data can be reprocessed from raw or cleaned as needed. This three-stage pipeline provides a clear path from ingestion to insight with checkpoints to catch errors or drifts in data along the way. For scientific projects, this can be invaluable. It means you can always regenerate your curated dataset from the original experimental data if the cleaning logic needs to change or as new raw data arrives.

Applying the Raw–Cleaned–Curated Pipeline to Scientific Data

In scientific use cases, implementing these three zones helps tackle the unique challenges of experimental, observational, and simulation data. Raw scientific data often comes in diverse formats (images, time-series, proprietary binary files, etc.) and may lack uniform structure or standardization. By ingesting into a raw zone up front the team ensures all data is captured reliably first and can worry about structure later.

Wrangling Varied Datasets

The cleaned zone for scientific data involves heavy data wrangling and harmonization. Common pre-processing tasks include handling missing or corrupt readings, normalizing values (e.g. adjusting all temperatures to °C, or scaling data by a constant), managing categorical variables (like converting instrument codes to human-readable labels), and ensuring consistent units and formats across experiments [9]. Scientific data must often be enriched with metadata. For instance, attaching experimental conditions, calibration information, or simulation parameters to each data record. At this stage, a scientist or data steward might also filter the data to improve quality (e.g., remove measurements outside physical plausible ranges). The cleaned layer might convert proprietary or domain-specific formats into more analysis-friendly formats. For instance, scientists may transform raw NetCDF or GRIB files into standardized Parquet tables with one row per grid point or per time snapshot, adding uniform units and coordinate systems. If the data requires labeling (say, flagging events in sensor signals or annotating images), this might be done in the cleaned, or later in the curated zone, depending on the workflow. The outcome is that by the time data reaches the curated zone, it is uniform, cleaned, and enriched with all necessary context.

In the curated zone, scientific data is made ready for AI or other analysis workflows. This often means performing any remaining transformations needed for machine learning algorithms, such as generating feature vectors, aggregating data to meaningful statistical measures, or splitting data into training/validation sets. For example, data from a fusion energy experiment in the cleaned zone might be each sensor reading that has been cleaned and aligned in time. In contrast, the curated data could be a table of aggregated pulse characteristics (one row per experimental run with features like peak temperature, energy output, etc., plus a label indicating success or failure of the run). In the curated zone, data may also be sharded or partitioned in a way that is optimal for AI training. This could include splitting large arrays into smaller chunks or converting data into specialized formats like TFRecord or HDF5 if needed [9].

In a lakehouse context, curated data often remains in Parquet or Iceberg tables for easy query. It can also be exported to specialized formats for HPC or deep learning as an end step. The key is that the curated layer is ready to use for its intended purpose, whether that's feeding a deep learning model or answering a scientific question. A recent study of AI readiness in scientific HPC workflows emphasizes that achieving AI-ready data is labor-intensive because it requires all these intensive steps – cleaning, normalization, feature engineering, and format conversion

– especially given the scale of modern scientific datasets [9]. The structured pipelines easily achievable in a lakehouse helps manage this complexity in a repeatable way.

The Materials Project — A Case Study in Scientific Lakehouse Implementation

[The Materials Project](#) (MP) [14] provides a production-scale example of the zoned lakehouse approach applied to computational materials science. MP produces curated datasets of computed material properties, including electronic bandstructures and density of states, thermodynamic stability descriptors such as phase diagrams and energies above the convex hull, elastic tensors and derived mechanical properties (bulk modulus, shear modulus, elastic anisotropy), and dielectric and piezoelectric response functions. These data sets are served to a user community of over 730,000 researchers through RESTful API endpoints that expose structured, schema-validated representations of materials data. This data is created through a combination of HPC workflows and automated ETL pipelines running over large sets of input molecular structures, augmented by experimental data and user-contributed data, and is accessed through interactive web-based interfaces and Python APIs.

Over the past eighteen or so months, MP has implemented a medallion-style data lake architecture, migrating from a legacy on-premises infrastructure to a cloud-native stack that closely mirrors the three-zone model described above.

Overview — Zone Mapping and Technology Choices

MP has three data zones that map directly onto the raw–cleaned–curated framework. MP’s implementation adopts domain-specific terminology to reflect the materials science workflow:

Zone One (Raw) consists of semi-structured and unstructured text file outputs from density functional theory (DFT) simulation workflows (primarily VASP electronic structure calculations) executed on HPC systems. This zone is currently under active development. Pipelines exist for parsing Zone One artifacts into Zone Two data products, but the full access patterns and data architecture have not yet been finalized

Zone Two (Parsed) contains fully structured Parquet [22] representations of the simulation outputs from Zone One. This zone consists of multiple data products, some feeding directly into Zone Three pipelines and others serving as standalone queryable datasets, such as simulated electronic bandstructure objects [18].

Zone Three (Build) contains fully structured Parquet data products derived by processing Zone Two’s parsed data through ETL (Extract, Transform, Load) pipelines. These are the final, curated datasets served to users through The Materials Project’s API, website, and data lake.

MP uses Delta Lake [10] rather than Apache Iceberg [6] as its table format. Delta Lake provides the same core capabilities discussed elsewhere in this report.

The Migration Journey – From Legacy Infrastructure to a Cloud-Native Lakehouse

MP had been operational for over a decade with well-defined API endpoints and existing data processing pipelines. Although some aspects of the MP migration are specific to this starting point, four broadly applicable lessons emerged:

- When well-defined data products already exist, working backward from curated-zone specifications provides a concrete framework for the many discretionary decisions in data cleaning and schema design.
- Legacy scientific data collections with years of schema drift represent the bulk of the curation effort, and reconciling that drift is a harder problem than selecting a table format.
- Lightweight cloud-native tools such as DuckDB [12] and delta-rs [25] can deliver sufficient performance for scientific ETL workloads without the operational complexity of distributed compute frameworks.
- Making lakehouse data products publicly accessible in open formats creates a multiplier effect, enabling users to build analyses that go far beyond what any single API or development team can anticipate.

The transition to a lakehouse architecture involved migrating from an on-premises MongoDB-backed [20] infrastructure to a cloud-native stack built around Parquet and Delta Lake. Because MP had a large existing user base, the team began with a firm understanding of what all Zone Three data products needed to look like. The existing API contracts and user expectations served as the team's specification. Zone Three's requirements were fixed, and the architecture was built backward from there.

Zone Two: The Most Challenging Transition

The most difficult aspect of MP's migration was Zone Two. The primary source of parsed data was a MongoDB collection accumulated over more than ten years. During that time, simulation workflows, workflow management software, and parsing logic had evolved repeatedly. Many different users generated this litany of Zone One data. The result was significant schema inconsistency within what was nominally a single collection. This is a common outcome in long-lived scientific data systems where the "schema" is implicitly defined by whatever code happened to produce the data at a given point in time.

Migrating this collection into a clean, consistently-schemed Zone Two data product required extensive curation spanning several categories of transformation. Decisions about how to clean, restructure, and validate were directly influenced by downstream Zone Three pipeline requirements. Because the curated outputs were well specified the MP team was able to work backward to determine exactly what the parsed data needed to look like. This goal-driven curation strategy — defining the end state first and letting it dictate the cleaning logic — was, and continues to be, an effective organizing principle for what would otherwise have been an open-ended data wrangling effort. The full set of migration procedures has been documented and is publicly available as a reference for other groups facing similar legacy data challenges in MP's documentation [23].

Cloud-Native Compute on Zone Two

The Materials Project operates a cost-effective, cloud-based compute plane for producing Zone Three products thanks to Zone Two's clean Delta tables housed in cloud object storage. The ETL pipelines make direct use of delta-rs (a Rust-native Delta Lake implementation with Python bindings [25]) and DuckDB (an in-process analytical query engine [12]) to query Zone Two data products without requiring a heavyweight distributed compute framework like Spark [7]. This lightweight approach keeps infrastructure costs low while providing the query performance needed at MP's scale.

Empowering Users Through the Data Lake

Beyond MP's own internal pipelines, the lakehouse architecture has opened new possibilities for MP's user community — at no additional engineering cost beyond the work MP was already doing. The data products in MP's data lake are the same artifacts that MP's own ETL pipelines produce and consume internally. MP makes these Delta tables publicly accessible on cloud object storage (via the AWS Open Data program [21]). This effectively exposes its internal data infrastructure to users without building or maintaining any additional data-serving layer. There is no separate export step, no bespoke data packaging, and no additional API endpoints to engineer. The curated Delta tables that power MP's own internal build pipelines are simply made readable to the public.

The immediate benefits of this approach were realized within MP's own ecosystem. By integrating the data lake directly with the Materials Project API Python client, MP achieved massive operational performance gains. Rerouting bulk data retrieval to the AWS Open Data program enabled a seamless transition for users while achieving a 10x speedup for full collection downloads. Crucially, circumventing the legacy operational database for these heavy requests significantly reduced CPU pressure on MongoDB, stabilizing the core infrastructure while simultaneously improving the user experience.

This architecture represents a significant expansion of capability compared to what is possible through REST API alone. The entire ecosystem of Delta Lake compatible query engines becomes available to all interested users. A user can, for example, write a DuckDB SQL script that scans MP's parsed task data and electronic structure data directly from the public S3 bucket, performs a cross-table join filtered by specific physical criteria (such as crystal system and calculation type), and writes the results to a local Parquet file. This is all possible without writing any Python or making any API calls that would incur load or cost on MP's infrastructure. This kind of ad hoc, cross-product query would be difficult or impossible to express through a traditional REST API but becomes straightforward when users have direct access to the underlying data lake tables. Because Delta Lake is compatible with a broad ecosystem of query engines — including Spark [7], Trino [11], BigQuery [8], Athena [2], and others — users can integrate MP data into whatever analytical infrastructure they already operate [17].

In this way, the lakehouse architecture creates a multiplier effect: the engineering investment in structuring data for internal use automatically yields a highly performant public data platform

whose capabilities grow with the external tooling ecosystem rather than with MP's own development bandwidth.

Conclusion

Building a raw-to-curated data pipeline in a lakehouse environment is an effective strategy for scientific data engineering teams to handle the perpetual deluge of observational, experimental, and simulation data. By dividing the workflow into raw, cleaned, and curated zones, you introduce structure and quality checks at each step, turning chaotic raw inputs into trustworthy, AI-ready outputs. Apache Iceberg's lakehouse table format provides the technical foundation for this pipeline, ensuring that data in each zone is managed with ACID guarantees, traceable history, and flexible schemas. These are all crucial factors for stewarding scientific datasets that change over time and require high integrity. Using this approach, data engineers who are domain scientists can incrementally refine their data while maintaining reproducibility (you always have the raw baseline) and scalability (the architecture supports petabytes of data on cheap storage with cloud compute engines on top).

In summary, a lakehouse architecture with raw-cleaned-curated zones offers a clear path from ingestion to scientific knowledge by progressively improving data quality and value. Scientific use cases benefit greatly from this pattern since raw sensor or simulation data is preserved, intermediate cleansing and normalization is formalized, and final curated datasets are ready for advanced analytics or AI modeling. This introductory overview sets the stage for more detailed exploration into each layer (for example, diving into specific tools for ETL, or strategies for handling schema drift and metadata in scientific contexts). By adopting an open lakehouse ecosystem like Apache Iceberg, scientific organizations can build robust data pipelines that bridge the gap between raw data and actionable, curated knowledge for AI-enabled research.

References

- [1] ACID: 2025. <https://en.wikipedia.org/wiki/ACID>.
- [2] Amazon Athena: <https://aws.amazon.com/athena/>. Accessed: 2026-03-11.
- [3] American Science Cloud – Genesis Mission Genesis Mission is a national initiative to build the world’s most powerful scientific platform to accelerate discovery science, strengthen national security, and drive energy innovation: <https://amsc.energy.gov/>. Accessed: 2026-03-18.
- [4] Apache Flink®: <https://flink.apache.org/>. Accessed: 2026-03-06.
- [5] Apache Hudi: <https://hudi.apache.org/>. Accessed: 2026-03-06.
- [6] Apache Iceberg – Apache Iceberg™: <https://iceberg.apache.org/>. Accessed: 2026-03-06.
- [7] Apache Spark™ - Unified Engine for large-scale data analytics: <https://spark.apache.org/>. Accessed: 2026-03-06.
- [8] BigQuery – AI data platform: https://cloud.google.com/bigquery?utm_source=pmax&utm_medium=display&utm_campaign=n=Cloud-SS-DR-GCP-1713658-GCP-DR-NA-US-en-pmax-Display-pmax-All-BigQuery&utm_content=c--x--9008163-21713147502&qclsrc=aw.ds&qad_source=1&qad_campaignid=22037004910&qclid=CjwKCAjwpcTNBhA5EiwAdO1S9pIGPBEWCmbIVzwZktlYD0VgsOSoFKkMaubn-3pHWNmADcqonv2a7xoC3J8QAvD_BwE. Accessed: 2026-03-11.
- [9] Brewer, W. et al. 2025. Data Readiness for scientific AI at scale. *arXiv [cs.AI]*.
- [10] Delta Lake: <https://delta.io/>. Accessed: 2026-03-06.
- [11] Distributed SQL query engine for big data: <https://trino.io/>. Accessed: 2026-03-06.
- [12] DuckDB – An in-process SQL OLAP database management system: <https://duckdb.org/>. Accessed: 2026-03-11.
- [13] Genesis Mission: <https://genesis.energy.gov/>. Accessed: 2026-03-18.
- [14] Horton, M.K. et al. 2025. Accelerated data-driven materials science with the Materials Project. *Nature Materials*. 24, 10 (Oct. 2025), 1522–1532. DOI:<https://doi.org/10.1038/s41563-025-02272-0>.
- [15] Iceberg Vendors: <https://iceberg.apache.org/vendors/>. Accessed: 2026-03-11.
- [16] Kaplan, A. and Kara, A. 2026. *The data lakehouse for dummies®*, 2nd databricks special edition. John Wiley & Sons.
- [17] Leveraging External Query Engines: <https://docs.materialsproject.org/materials-project-data-lakehouse/leveraging-external-query-engines>. Accessed: 2026-03-11.
- [18] Materials Project OpenData - Electronic Structure: <https://materialsproject-parsed.s3.amazonaws.com/index.html#core/electronic-structure/>. Accessed: 2026-03-11.
- [19] Merced, A. 2024. Introduction to the Iceberg Data Lakehouse. *Dremio*.
- [20] MongoDB: The World’s Leading Modern Data Platform: <https://www.mongodb.com/>. Accessed: 2026-03-11.
- [21] Open Data on AWS: <https://aws.amazon.com/opendata/>. Accessed: 2026-03-11.
- [22] Parquet: <https://parquet.apache.org/>. Accessed: 2026-03-11.
- [23] Task Collection Migrations: <https://docs.materialsproject.org/data-production/task-collection-migrations>. Accessed: 2026-03-11.
- [24] What is Apache Iceberg in Databricks? <https://docs.databricks.com/aws/en/iceberg/>. Accessed: 2026-03-06.
- [25] *delta-rs: A native Rust library for Delta Lake, with bindings into Python*. Github.