



Metadata

March 10, 2026

in release

3 min read

Apache Iceberg Rust 0.9.0 Release

The Apache Iceberg community is pleased to announce version 0.9.0 of [iceberg-rust](#).

This release covers development work from early January through early March 2026 and is the result of merging **109 PRs** from **28 contributors**, including **8 new contributors**. See the [changelog](#) for the complete list of changes.

`iceberg-rust` is a native Rust implementation of the Apache Iceberg specification, providing high-performance libraries for reading, writing, and managing Iceberg tables in Rust applications and through Python bindings (`pyiceberg-core`).

Release Highlights

Trait-Based Storage Architecture

The most significant change in this release is a new trait-based storage architecture that decouples `iceberg-rust` from any single storage backend. The new [Storage trait](#) provides a clean abstraction for file I/O, with [integration into FileIO](#) to make it the primary extension point for adding new storage backends.

This release includes two native storage implementations built on this trait:

- LocalFsStorage**: A native local filesystem backend
- MemoryStorage**: An in-memory backend useful for testing

The existing OpenDAL-based storage was [moved to its own crate](#) (`iceberg-storage-opendal`), keeping the core library lightweight while preserving full OpenDAL support as an optional dependency. See the [Storage trait RFC](#) for the design rationale.

Enhanced DataFusion Integration

DataFusion integration continues to expand with DDL and query optimization improvements:

- CREATE TABLE**: [Native support](#) for creating Iceberg tables through DataFusion SQL, with [schema validation for partition projection](#)
- DROP TABLE**: [Support for dropping tables](#) through DataFusion SQL
- LIMIT pushdown**: [Pushes row limits](#) into Iceberg scans to avoid reading unnecessary data
- Predicate pushdown**: Added support for [Boolean](#), [IsNaN](#), [Timestamp](#), [Binary](#), and [LIKE/StartsWith](#) predicates
- INSERT INTO**: [Support for inserting data](#) into Iceberg tables through DataFusion SQL, with automatic sort-based clustering for partitioned writes

Reader Performance Improvements

The Arrow reader received several performance optimizations as part of an [ongoing effort](#) to reduce I/O overhead during reads:

- Byte range coalescing**: [Implemented `get\_byte\_ranges`](#) on `AsyncFileReader` to coalesce small reads into fewer I/O operations
- Fast-path single-threaded reads**: [Optimized `ArrowReader::read`](#) for single-concurrency scenarios by avoiding unnecessary async overhead
- Metadata size hints**: [Added Parquet metadata size hint option](#) to `ArrowReaderBuilder`, eliminating unnecessary `stat()` calls when the metadata size is known
- File size propagation**: [Pass data file and delete file sizes](#) to the reader to avoid redundant metadata lookups
- Reduced builder overhead**: [Avoided redundant `create\_parquet\_record\_batch\_stream\_builder\(\)` calls](#)
- Timestamp support**: [Added timestamp type support](#) in column creation

38-Digit Decimal Precision

The `rust_decimal` crate was [replaced with `fastnum`](#), enabling full 38-digit decimal precision as required by the Iceberg specification. This also includes a [fix for decimal byte representation](#) used in hashing.

Bug Fixes

- Table upgrades**: [Fixed v2 to v3 table upgrades](#) that previously failed
- Sort order validation**: [Reserved sort order IDs](#) can no longer contain fields
- Partition field IDs**: [Fixed reuse of partition field IDs](#) in `AddSpec`
- REST catalog security**: [Filtered sensitive headers](#) from error logs to prevent credential leakage
- SqlCatalog**: [Fixed `SqlCatalogBuilder`](#) not persisting the supplied catalog name
- S3Tables**: [Correctly interpret warehouse](#) as `table_location` for S3Tables

Breaking Changes

This release includes several breaking changes:

- Storage architecture**: The [Storage trait](#) is now integrated with FileIO, and [OpenDAL storage moved](#) to `iceberg-storage-opendal`
- MSRV**: [Bumped to Rust 1.92.0](#)
- Decimal**: [Replaced `rust\_decimal` with `fastnum`](#) for 38-digit precision support

Dependency Updates

- [Upgraded to DataFusion 52.2](#)
- Various updates to Arrow, Parquet, and other dependencies

Infrastructure Improvements

- Security scanning**: [Added CodeQL workflow](#) for GitHub Actions security analysis
- Workflow permissions**: [Added explicit least-privilege permissions](#) to workflows
- Faster tests**: [Adopted nextest](#) for faster test execution with [parallel integration tests](#)
- Python tooling**: [Switched to uv](#) for Python dependency management

What's Next

Work is already underway on several features for upcoming releases:

- Storage trait expansion**: With the new trait-based storage architecture landed in 0.9.0, the community is looking to [integrate additional storage backends](#), such as `object_store`, making it easy for users to opt into alternative storage implementations.
- Table encryption**: [AES-GCM encryption support](#) is in progress, bringing Iceberg's table-level encryption spec to Rust with Java-compatible ciphertext formats. See the [table encryption RFC](#) for the full design, including envelope encryption with a chained key hierarchy and KMS integration.

Getting Involved

The `iceberg-rust` project welcomes contributions. We use GitHub [issues](#) for tracking work and the [Apache Iceberg Community Slack](#) for discussions.

The easiest way to get started is to:

- Try `iceberg-rust` with your workloads and report any issues you encounter
- Review the [contributor guide](#)
- Look for [good first issues](#)

For more information, visit the [iceberg-rust repository](#).

Table of contents

Release Highlights

Trait-Based Storage Architecture

Enhanced DataFusion Integration

Reader Performance Improvements

38-Digit Decimal Precision

Bug Fixes

Breaking Changes

Dependency Updates

Infrastructure Improvements

What's Next

Getting Involved

Features

Schema Evolution
Hidden Partitioning
Partition Evolution
Serializable Isolation
Branching and Tagging
Optimistic Concurrency
Advanced Filtering
Compute Engine Integrations
REST Catalog
Multiple language APIs

Get Started

Spark Quickstart
Hive Quickstart
Open Table Spec
Docs
Blog
Talks

Community

Support
Mailing Lists
Iceberg Events
Issues
Contribute
Guidelines

ASF

Apache Software Foundation/
Thanks
Sponsorship
Security
License

