ProjectX

LAKEHOUSE · DATASHEET

Open Iceberg tables you can always take with you.

Tables in the open Apache Iceberg format on S3-compatible storage, with ACID writes, snapshots, time travel and schema evolution — catalogued, classified and governed by the same policy engine as everything else, and readable by any engine that speaks Iceberg.

Why teams choose it

01
Leaving is a copy, not a project
Tables are Apache Iceberg on S3-compatible storage behind a standard Iceberg REST catalog, so any engine that speaks the format can read them where they are.

02
Last Tuesday stays last Tuesday
Every write is a snapshot. Read a table as of a timestamp or a snapshot id, branch it to test a change, and roll back when a load goes wrong.

03
Change the schema without a rewrite
Add, drop, rename and reorder columns safely, and evolve partitioning without moving the data you already have.

04
Governed at the column
Columns carry their own classification and description, sensitive ones can be encrypted, and row and column rules are applied when a query is planned, for every engine.

In numbers

1
open table format, readable by any Iceberg engine

4
storage tiers, hot to archive, moved by policy

Any
past snapshot readable, by time or id

What it does

Iceberg tables with ACID writes

Create, alter and describe tables; append, overwrite, upsert, merge and delete as atomic commits.

Every write is one atomic commit that either lands whole or not at all, so a reader never sees half a load. Upserts and merges are first-class operations rather than a delete-and-rewrite you script yourself.

Snapshots, branches and time travel

List and tag snapshots, branch a table, roll back, and read as of any timestamp or snapshot id.

A snapshot is the table at one instant, and every write makes one. Tag the month-end close, branch to try a backfill, and roll back if it is wrong — the history is the table's, not a backup you hope exists.

Catalog, classification and statistics

Tenant-scoped namespaces, column descriptions and sensitivity, row counts and freshness, and data contracts producers commit to.

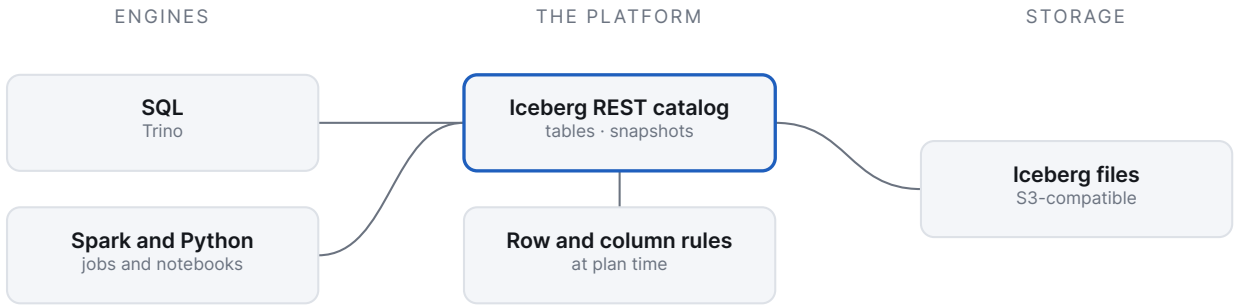
The catalog knows what each column means and how sensitive it is, how big and how fresh each table is, and what its producer has promised — so a consumer finds the right table and knows whether to trust it.

Maintenance and tiering handled

Compaction, clustering, snapshot expiry and orphan cleanup, with data moved between hot, warm, cold and archive by policy.

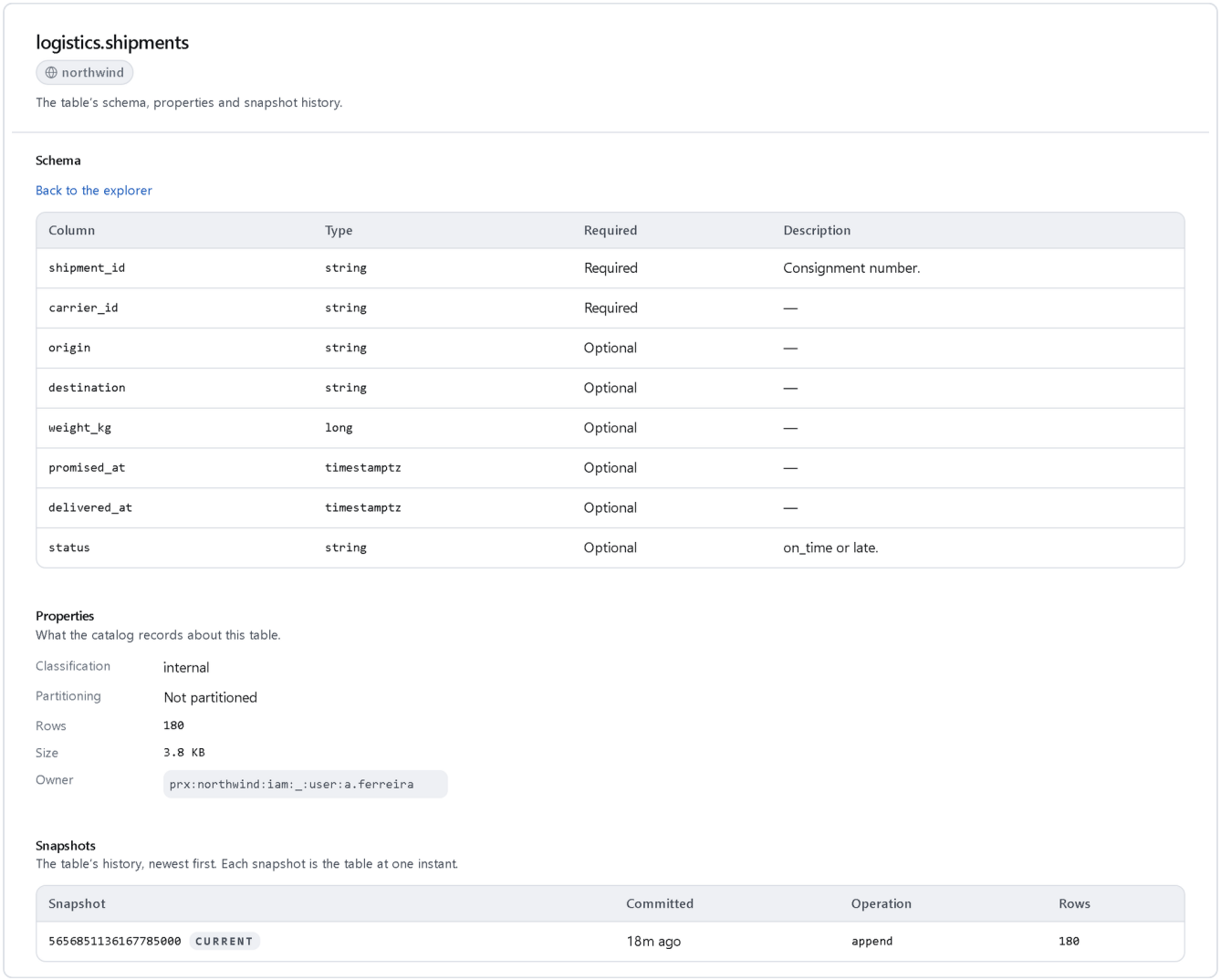
Small files are compacted, old snapshots expired and orphaned files removed on a schedule, and partitions move to cheaper tiers by policy — the housekeeping that otherwise becomes a cost and a performance problem.

How it fits together



Data lands as Iceberg files in S3-compatible storage. The REST catalog records each table and its snapshots, and every engine reads through it with the same row and column rules applied at plan time.

In the console



logistics.shipments

northwind

The table's schema, properties and snapshot history.

Schema

Back to the explorer

Column	Type	Required	Description
shipment_id	string	Required	Consignment number.
carrier_id	string	Required	—
origin	string	Optional	—
destination	string	Optional	—
weight_kg	long	Optional	—
promised_at	timestamptz	Optional	—
delivered_at	timestamptz	Optional	—
status	string	Optional	on_time or late.

Properties

What the catalog records about this table.

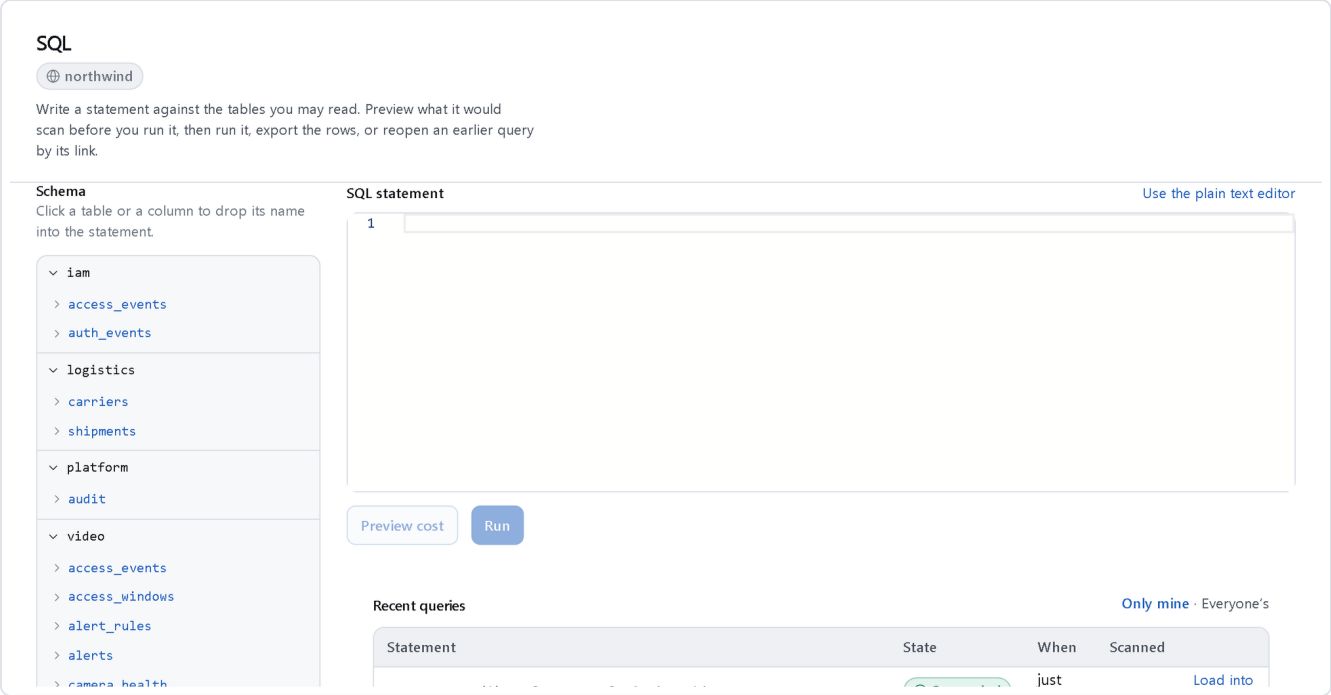
Classification	internal
Partitioning	Not partitioned
Rows	180
Size	3.8 KB
Owner	prx:northwind:iam:_:user:a.ferreira

Snapshots

The table's history, newest first. Each snapshot is the table at one instant.

Snapshot	Committed	Operation	Rows
5656851136167785000 CURRENT	18m ago	append	180

One Iceberg table: schema, properties and the snapshot history. Screens shown with demo data.



The SQL editor, with the namespaces and tables you may read beside it. Screens shown with demo data.

Key ideas

Snapshot

The table at one instant. Every write makes one, so a past state can be read, tagged, branched from or rolled back to.

Namespace

A tenant-scoped folder for tables, such as logistics or finance, that grants and placement attach to.

Data contract

What a table's producer commits to — its columns, types and rules — validated on every write and versioned when it changes.

In depth

Security model
Data is encrypted at rest with per-tenant keys, sensitive columns can be encrypted individually, and row and column rules are applied to the query plan before any engine reads a file.

Retention with legal hold
Retention policies are bound to catalog entries, so a table knows how long it is kept; a legal hold stops deletion until it is lifted.

Lineage as the work happens
Pipelines, syncs and model training record where data came from and where it went, so a figure traces back through every step to its source table.

Sharing as a grant
Share a table with another workspace by grant rather than by export, and revoke it immediately with the revocation on the audit record.

How it works

01 **Create a namespace and a table** · Console
Declare the columns, their types and their sensitivity; the catalog assigns field ids so a rename stays a rename.

02 **Write as atomic commits** · API
Append, upsert or merge rows; every write becomes a snapshot.

03 **Query with any engine** · SQL
SQL, Spark or any Iceberg reader, each held to the same row and column rules.

04 **Keep it healthy** · Policy
Compaction, expiry and tiering run by policy, and lineage records where every table came from.

Who it is for

Data platform teams
One copy of the data that every engine and team can read, without a proprietary format in the way.

Open Iceberg tables behind a standard REST catalog, with ACID writes and snapshots from day one.

Regulated companies
Sensitive columns, retention obligations and an answer to “who could read this?”.

Column classification, column-level encryption, retention bound to the table and rules enforced at plan time.

Teams leaving a warehouse
A move that does not trade one lock-in for another.

An open format and a standard catalog, with Delta Lake tables readable during the move.

Works with

Console · API · Iceberg REST catalog · SQL · Python SDK

Questions buyers ask

Can other engines read our tables?
Yes. Tables are Apache Iceberg on S3-compatible storage, served by a standards-compatible Iceberg REST catalog, so engines that speak Iceberg can read them in place.

What happens when a load goes wrong?
Every write is a snapshot. Roll the table back to the snapshot before the bad load, or read it as of an earlier time while you investigate.

How are sensitive columns handled?
Each column carries a classification, sensitive columns can be encrypted, and row and column rules are applied when a query is planned — the same for SQL, notebooks and the API.

Can we bring Delta Lake tables?
Delta Lake tables are readable, so you can query them while you move them to Iceberg.

How is it priced?
Storage by the byte-hour and compute by the second at published rates in Platform Credits, on the same invoice as everything else. The compute rates are on the pricing page.

Pricing

Current published rates, in your currency: https://adaydev.com/pricing/products