

# Exploring Apache Iceberg - A Modern Data Lake Stack

Engineering

Bloomberg

EuroPython 2024  
July 12, 2024

Gowthami Bhogireddy  
Enterprise Data Lake Engineering

TechAtBloomberg.com

# Who Am I?



Software Engineer at  
Bloomberg

**TechAtBloomberg.com**

© 2024 Bloomberg Finance L.P. All rights reserved.

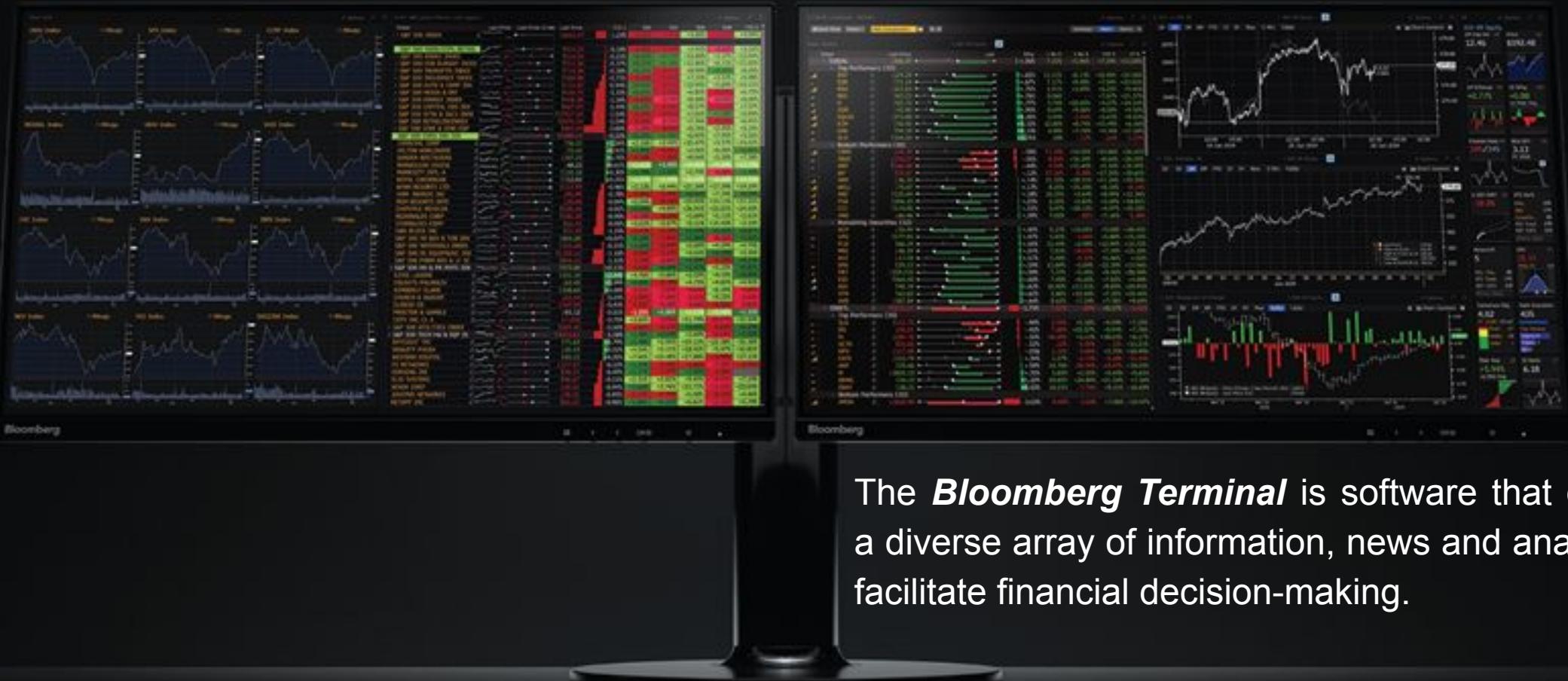
👉 <https://www.linkedin.com/in/gowthami-bhogireddy-7335427b>

/

👉 Pylceberg Open Source Contributor

👉 First time EuroPython speaker, yay!

**Bloomberg**  
Engineering

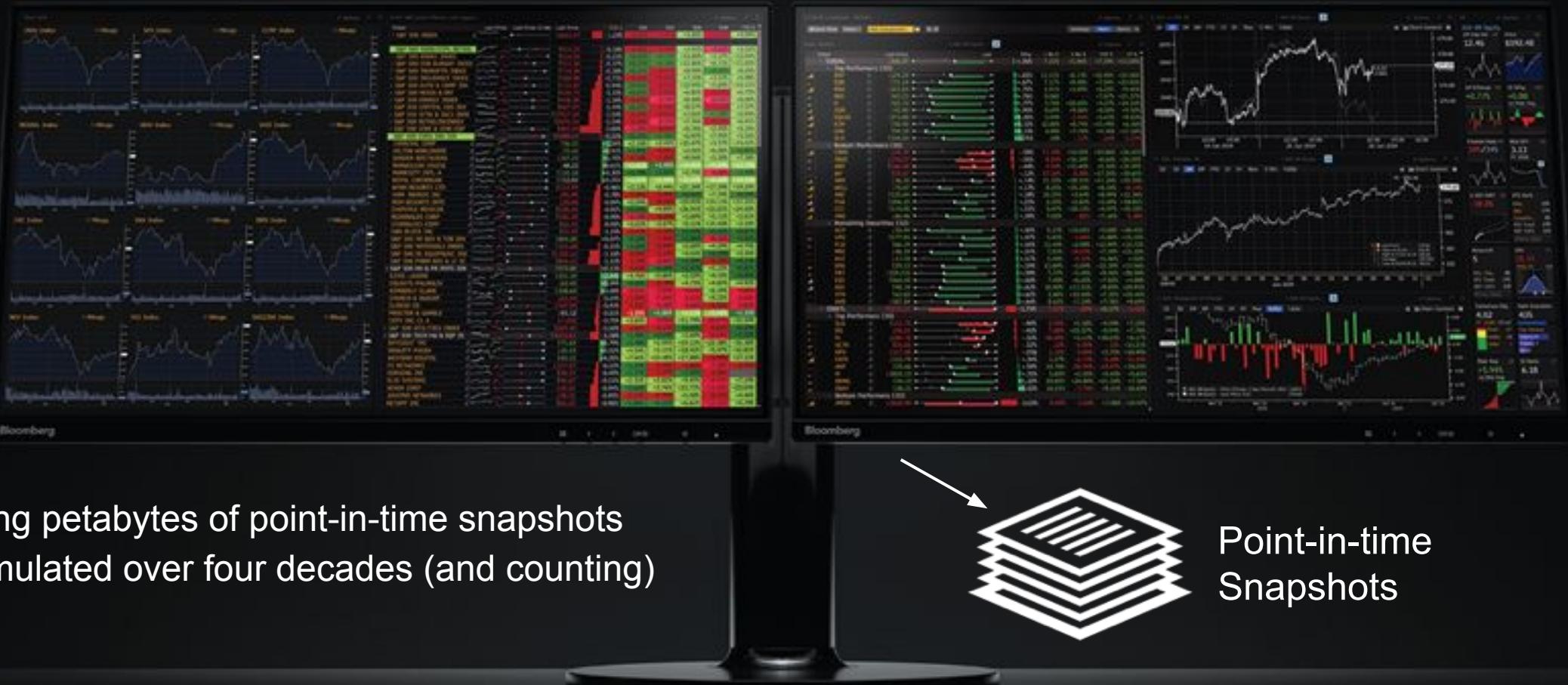


The **Bloomberg Terminal** is software that delivers a diverse array of information, news and analytics to facilitate financial decision-making.

[TechAtBloomberg.com](https://TechAtBloomberg.com)

© 2024 Bloomberg Finance L.P. All rights reserved.

**Bloomberg**  
Engineering



Using petabytes of point-in-time snapshots  
accumulated over four decades (and counting)

Point-in-time  
Snapshots

TechAtBloomberg.com

© 2024 Bloomberg Finance L.P. All rights reserved.

Bloomberg  
Engineering



**1,000,000,000,000,000**

**1,000,000,000,000,000**

**TechAtBloomberg.com**

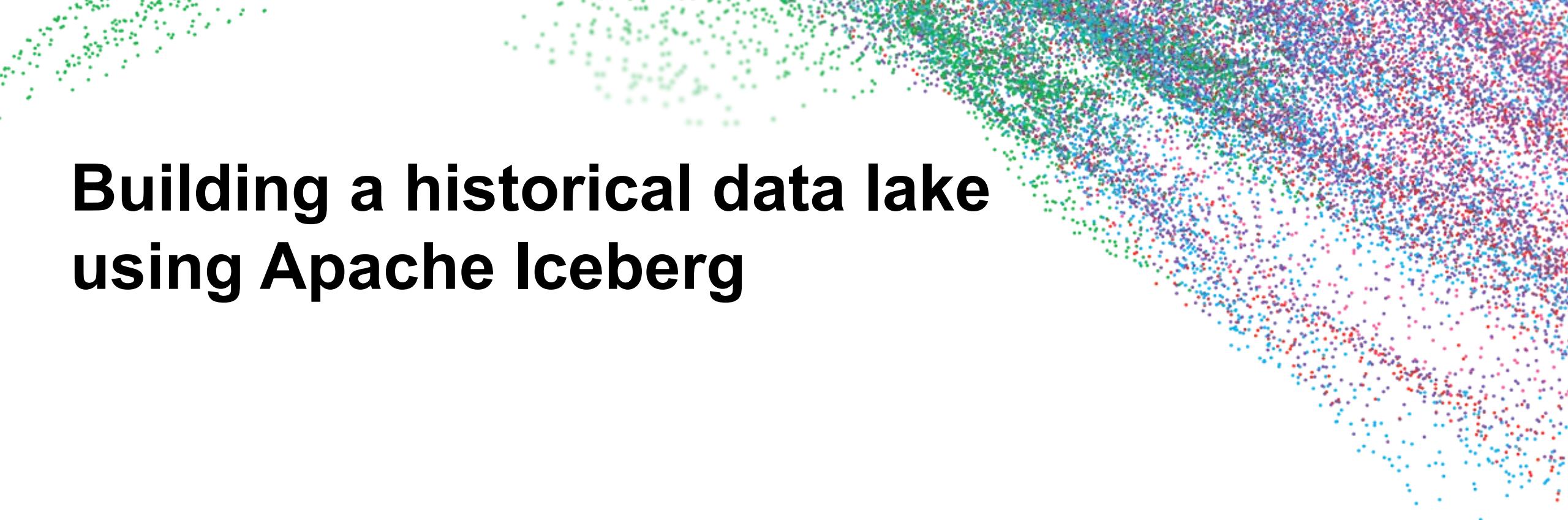
© 2024 Bloomberg Finance L.P. All rights reserved.

**Bloomberg**  
Engineering

# Agenda

1. Building a historical data lake using Apache Iceberg
2. Demo - Ways to interact with Apache Iceberg

# Building a historical data lake using Apache Iceberg



**TechAtBloomberg.com**

© 2024 Bloomberg Finance L.P. All rights reserved.

**Bloomberg**  
Engineering

# Example data

TIMESTARTED=Wed Feb 23 17:32:22 EST 2022

COMPANY\_ID

TICKER

PRICE

START-OF-DATA

AAPL US Equity|0|3|BBG000B9XRY4|AAPL|173.46|Common Stock

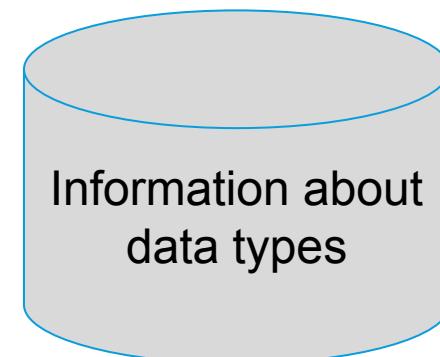
IBM US Equity|0|3|BBG000BLNNH6|IBM|129.43|Common Stock

...

END-OF-DATA

DATARECORDS=2

TIMEFINISHED=Wed Feb 23 17:41:32 EST 2022



COMPANY\_ID: string  
TICKER: string  
PRICE: double

# Summary of Problems

- **Data Format** - Text files
- **Data Volume** - PBs of data
- **Scale** - Tens of millions of files (and only growing)

# Data Lake Requirements

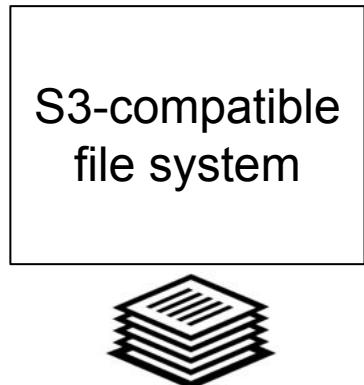
Golden copy of data

Audit trail

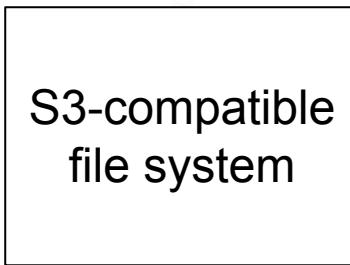
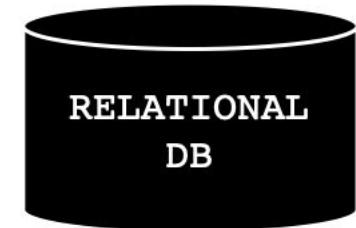
Schema  
Evolution

ACID guarantees

Data Analysis



Thousands of daily jobs  
using custom sources;  
async historical backfills



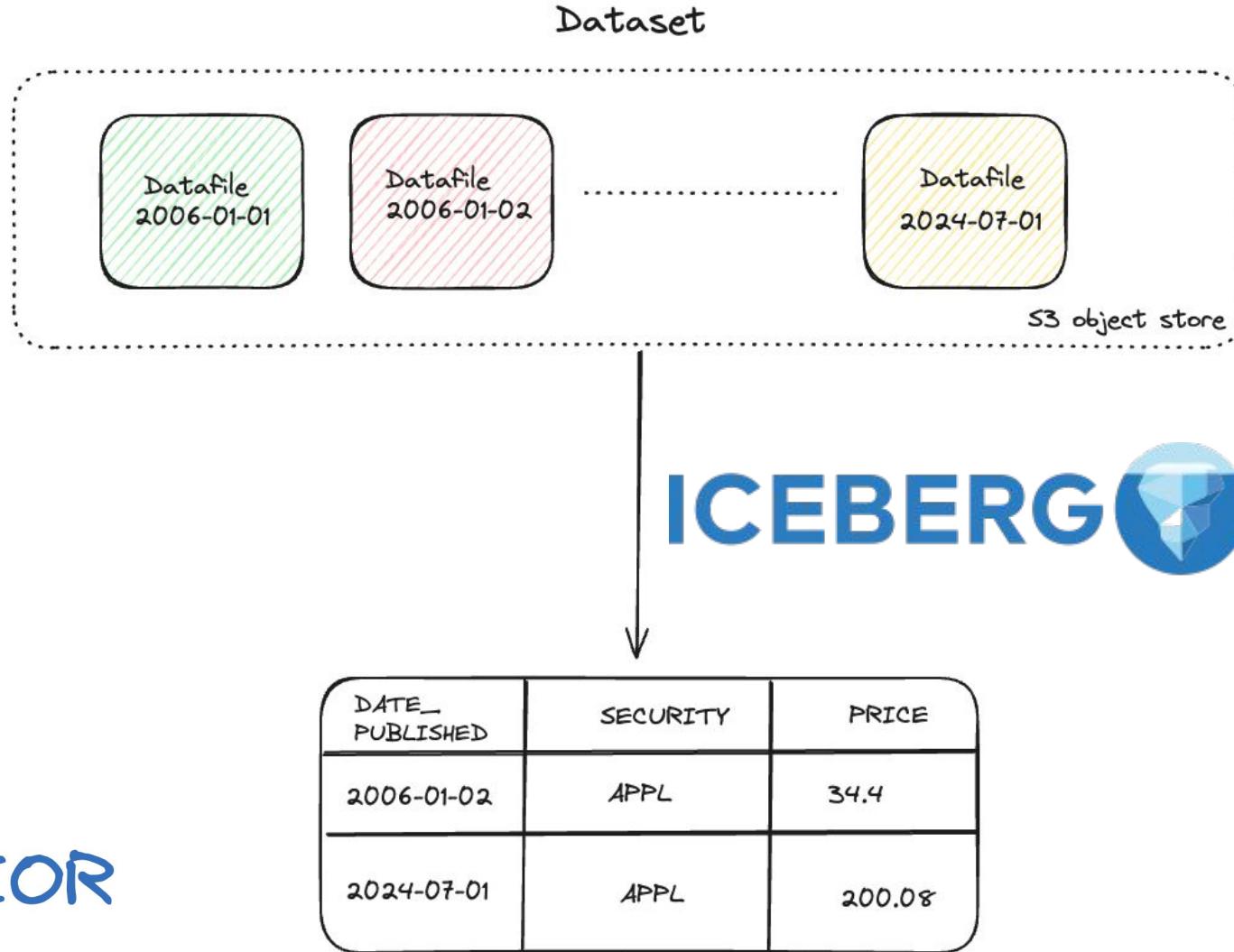
# What is Iceberg?

**TechAtBloomberg.com**

© 2024 Bloomberg Finance L.P. All rights reserved.

**Bloomberg**  
Engineering

SQL  
LIKE  
BEHAVIOR



 Open table format

 Storage layer

 Compute Engine

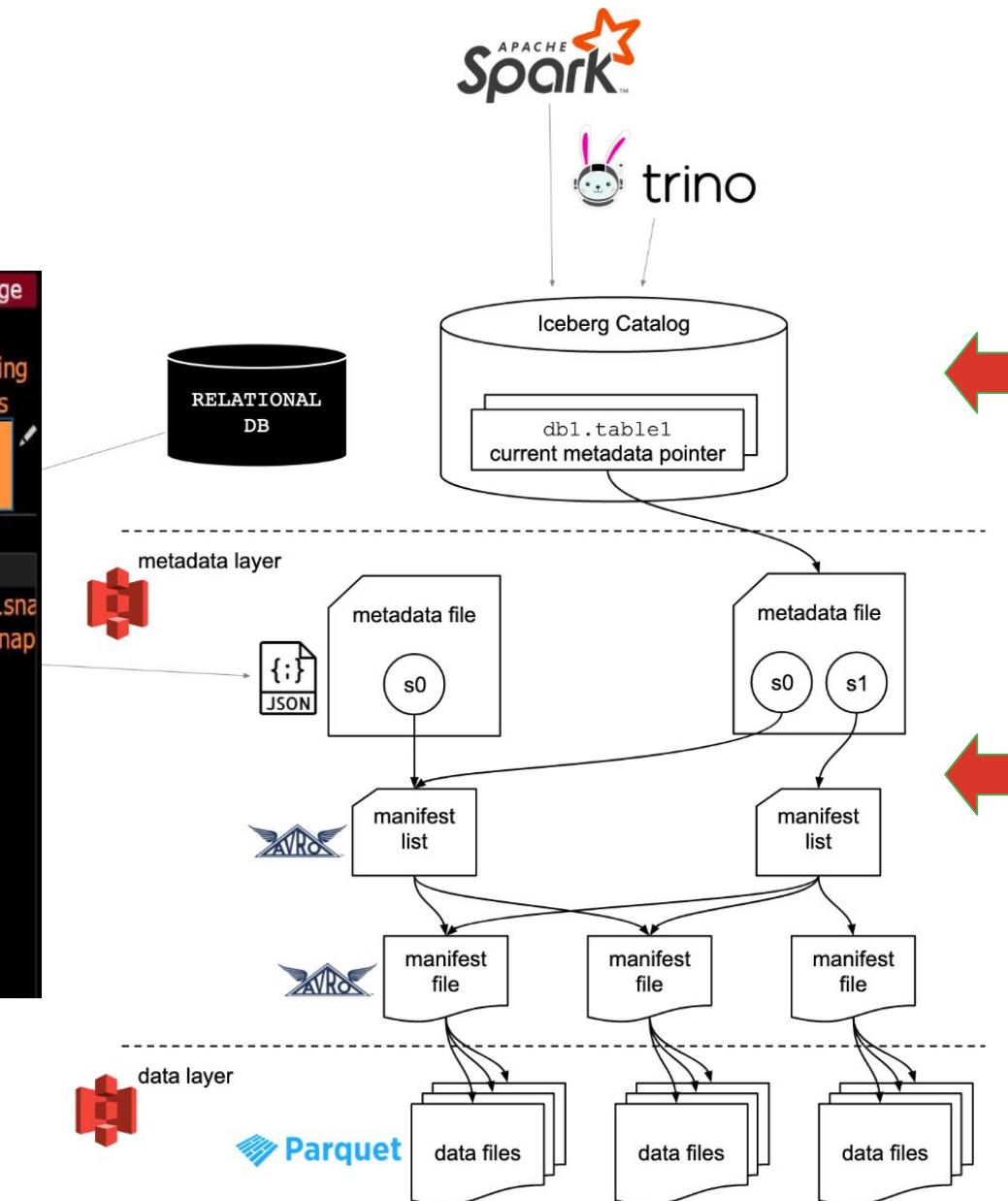
# Iceberg Internals

The screenshot shows the Bloomberg Database UI interface. At the top, there is a navigation bar with tabs: Run Query, Show Tables, Clear Query, History, C2DB, More, and DB Bridge. Below the navigation bar, there are dropdown menus for DB Type (comdb2), Name (iceberg), Env (dev), Precision, Time Zone, Column Width (200), Blob As String, Authenticate, Explain Only, and View Spaces. A query is entered in the text input field: `SELECT * FROM 'iceberg_tables'`. The results section displays the output of the query:

catalog_name	table_namespace	table_name	metadata_location
1. iceberg	demo	fake_snaps	s3a://warehouse/demo/fake_snaps
2. iceberg	test	fake_snaps	s3a://warehouse/test/fake_snaps

2 rows returned in 37 ms.

Bloomberg Database UI



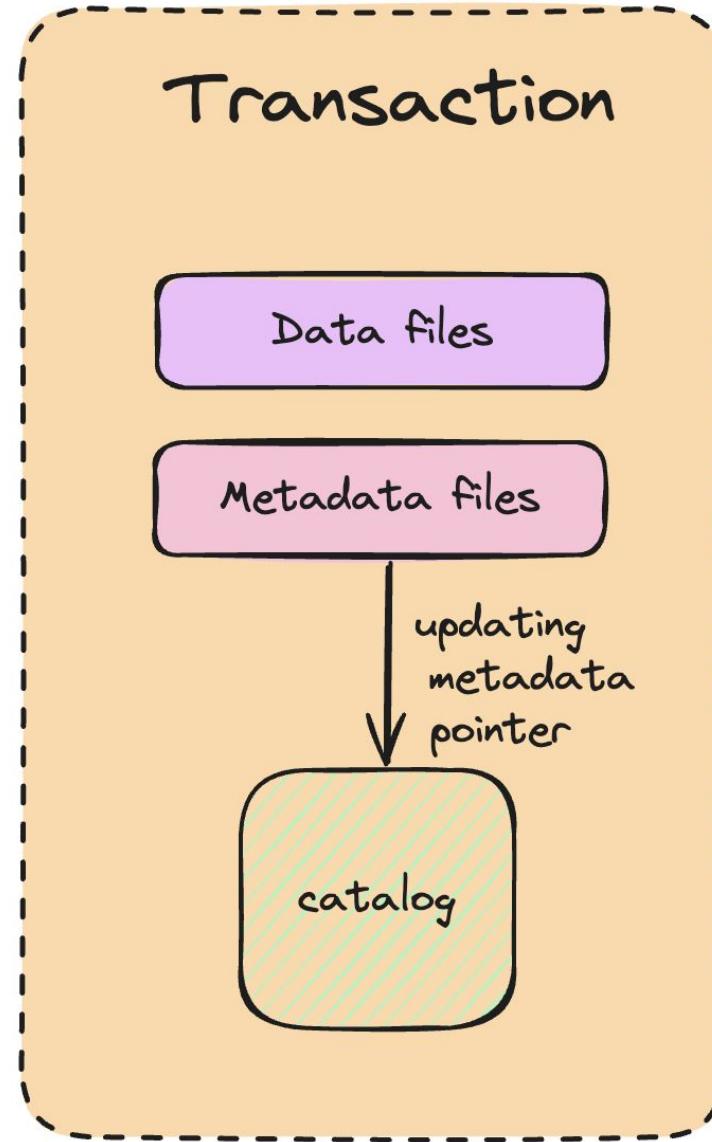
# Why Iceberg?

**TechAtBloomberg.com**

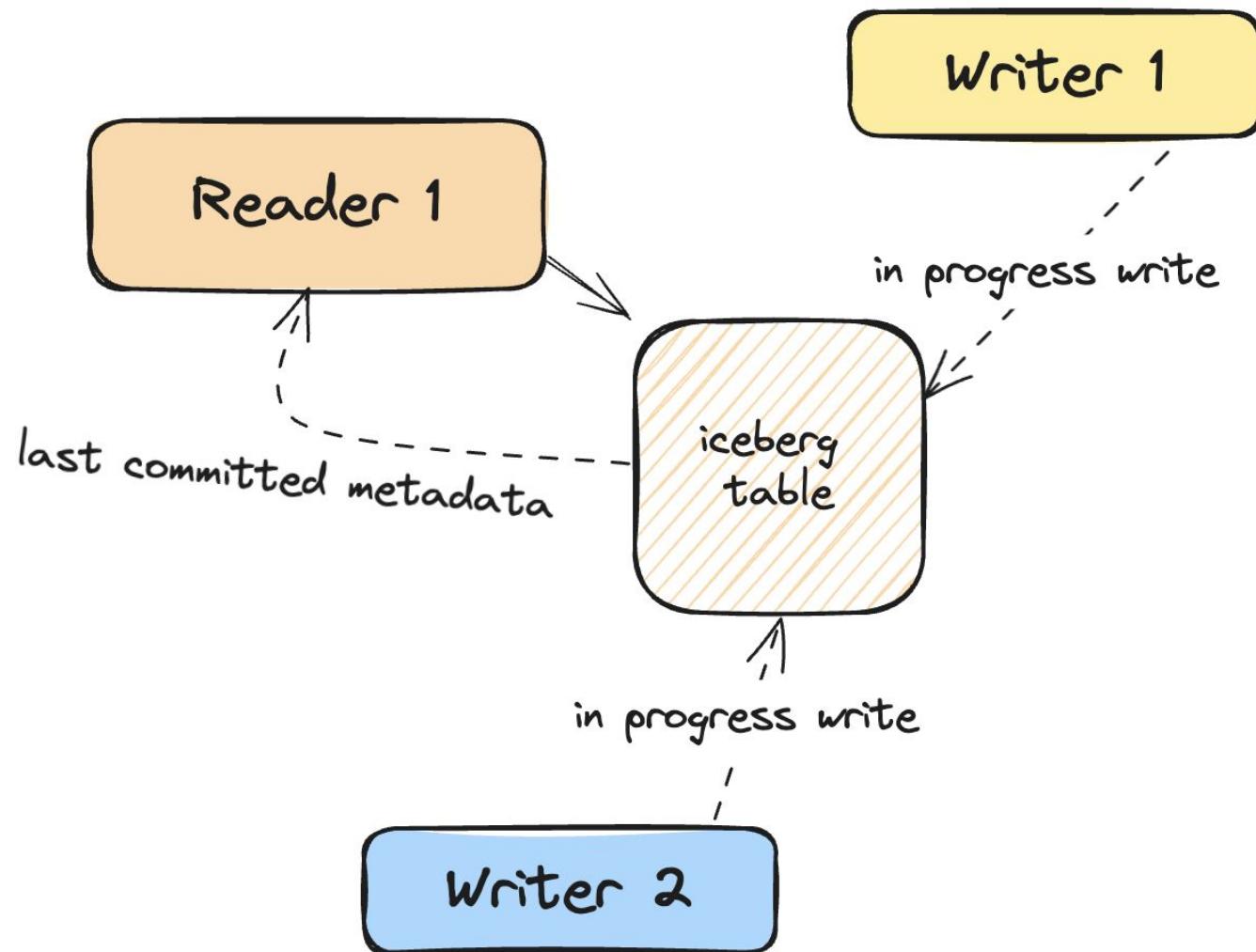
© 2024 Bloomberg Finance L.P. All rights reserved.

**Bloomberg**  
Engineering

# ALL OR NOTHING



# OPTIMISTIC CONCURRENCY ISOLATION



# Schema Evolution

**TechAtBloomberg.com**

© 2024 Bloomberg Finance L.P. All rights reserved.

**Bloomberg**  
Engineering

*Europython*

SESSION_ID
SESSION_NAME
SESSION_TYPE

+ SESSION\_DURATION



*Europython*

SESSION_ID
SESSION_NAME
SESSION_TYPE
SESSION_DURATION

**Metadata Only!!**

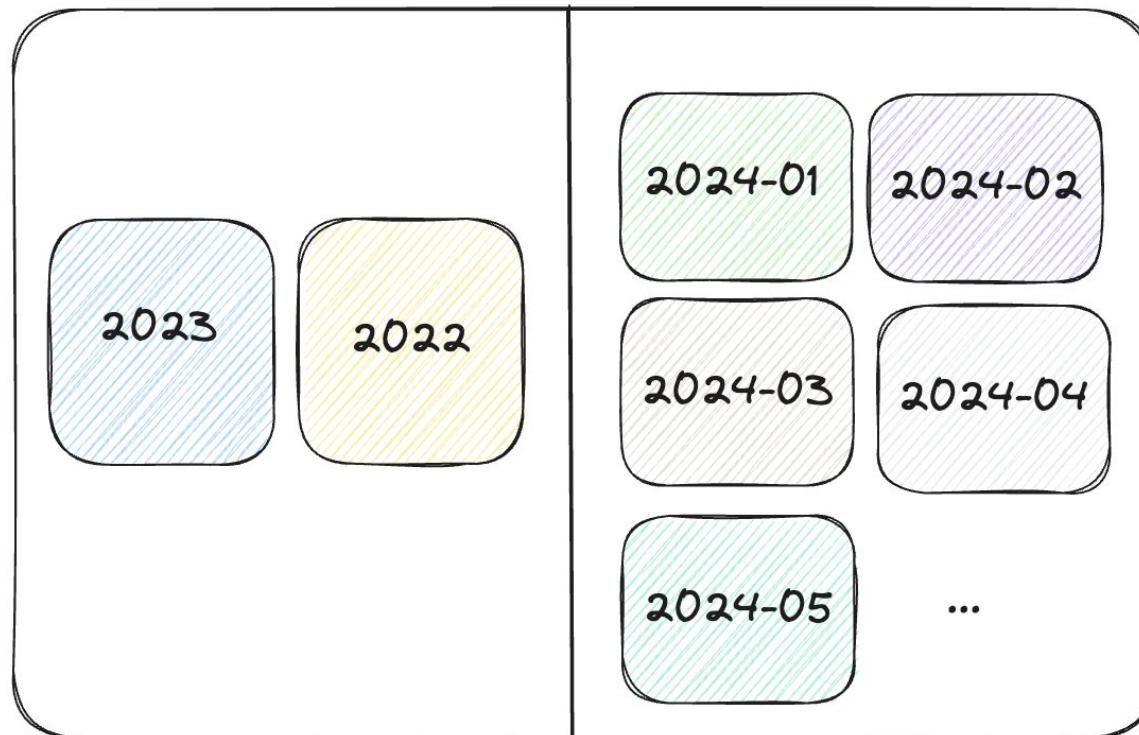
# Partition Evolution

**TechAtBloomberg.com**

© 2024 Bloomberg Finance L.P. All rights reserved.

**Bloomberg**  
Engineering

PARTITIONED BY YEAR      PARTITIONED BY MONTH

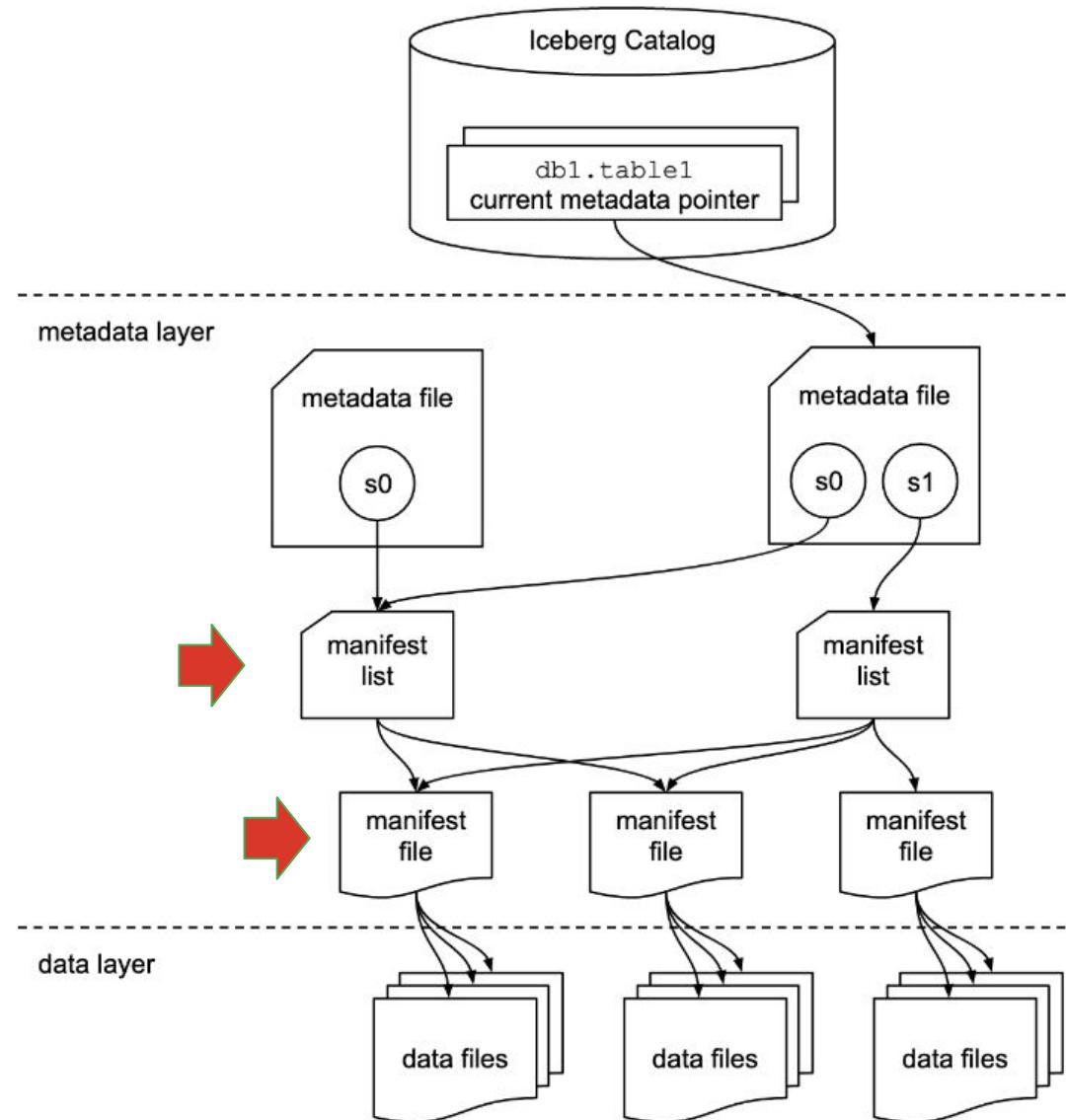


# Query Planning

**TechAtBloomberg.com**

© 2024 Bloomberg Finance L.P. All rights reserved.

**Bloomberg**  
Engineering

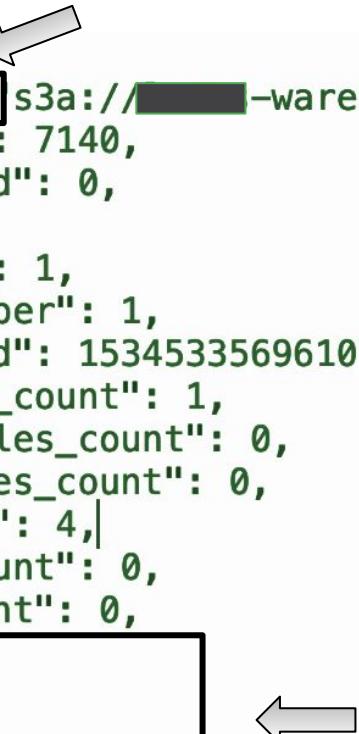


```
[{  
    "status": 1,  
    "snapshot_id": 1534533569610702800,  
    "data_file": {  
        "content": 0,  
        "file_path": "s3a://[REDACTED]-warehouse/test/europython/data/  
          SESSION_DATE_year=2024/00000-6-12d78028-5ec4-4080-8f1d-adace111c2a8-00001.parquet",  
        "file_format": "PARQUET",  
        "partition": {  
            "SESSION_DATE_year": 54  
        },  
        "record_count": 4,  
        "file_size_in_bytes": 1395,  
        "column_sizes": [...  
        ],  
        "value_counts": [...  
        ],  
        "null_value_counts": [...  
        ],  
        "nan_value_counts": [],  
        "lower_bounds": [...  
        ],  
        "upper_bounds": [...  
        ],  
        "split_offsets": [  
            4  
        ],  
        "sort_order_id": 0  
    }]  
}]
```

## MANIFEST FILE

## .AVRO

```
[{"manifest_path": "s3a://[REDACTED]-warehouse/test/europython/metadata/8261a1ce-3376-4126-a200-4dbd0e028690-m0.avro",  
 "manifest_length": 7140,  
 "partition_spec_id": 0,  
 "content": 0,  
 "sequence_number": 1,  
 "min_sequence_number": 1,  
 "added_snapshot_id": 1534533569610702800,  
 "added_data_files_count": 1,  
 "existing_data_files_count": 0,  
 "deleted_data_files_count": 0,  
 "added_rows_count": 4,  
 "existing_rows_count": 0,  
 "deleted_rows_count": 0,  
 "partitions": [  
     { ...  
     }  
 ]}]
```



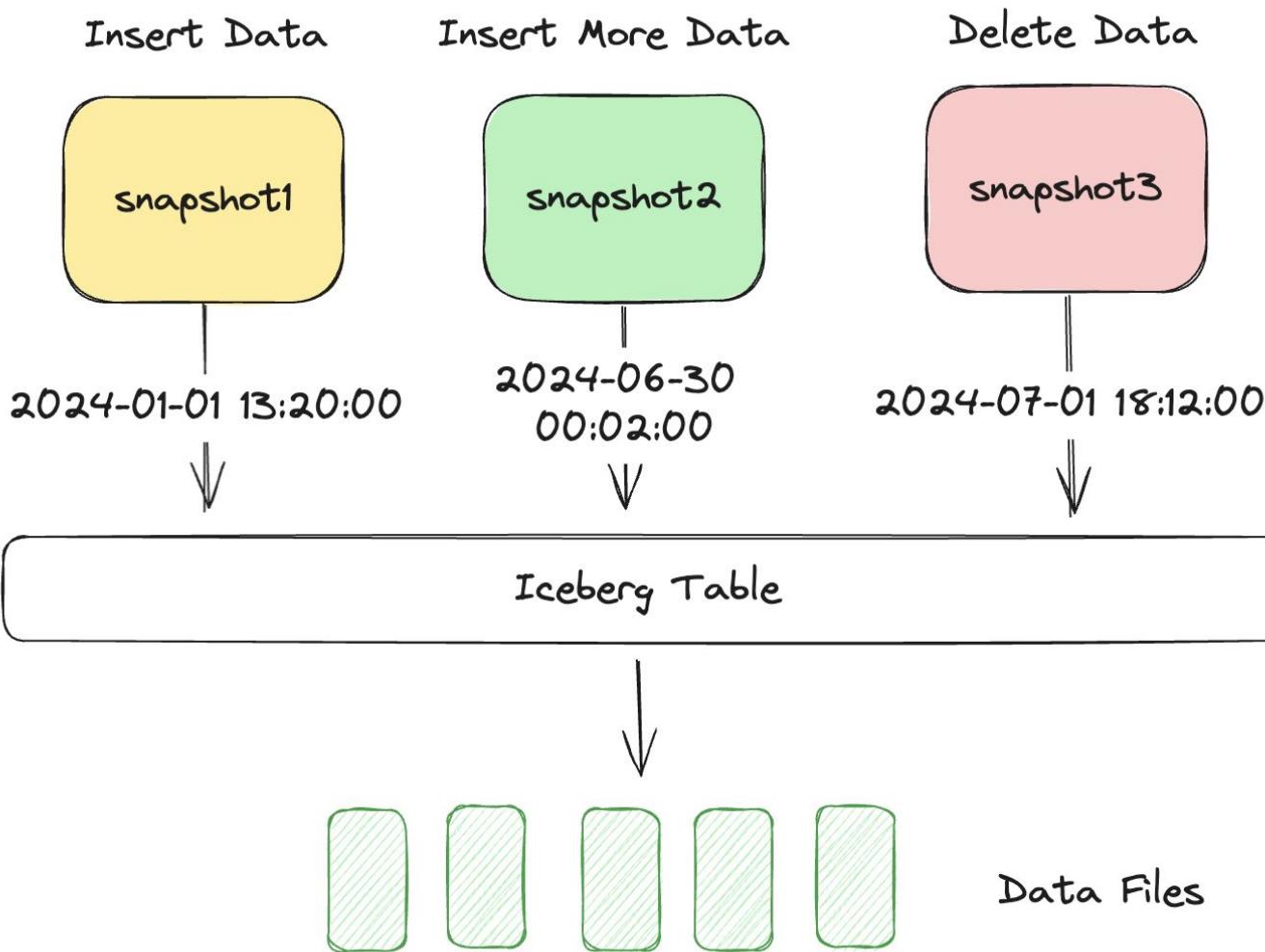
# MANIFEST LIST .AVRO

# Time Travel

**TechAtBloomberg.com**

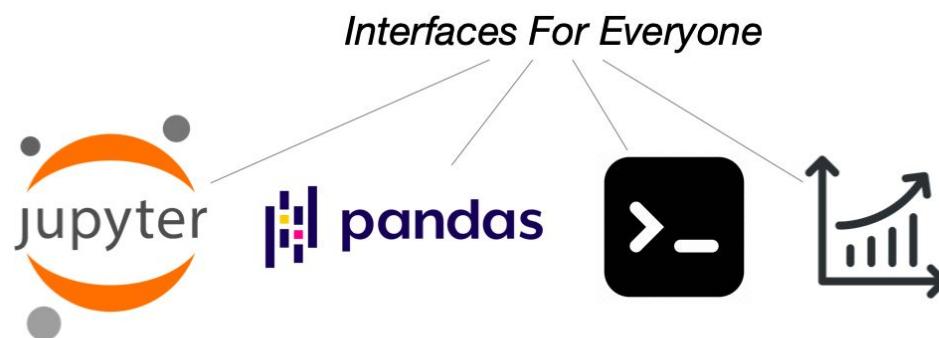
© 2024 Bloomberg Finance L.P. All rights reserved.

**Bloomberg**  
Engineering



```
SELECT * FROM TABLE  
WHERE VERSION AS OF  
SNAPSHOT1;
```

ANALYSIS



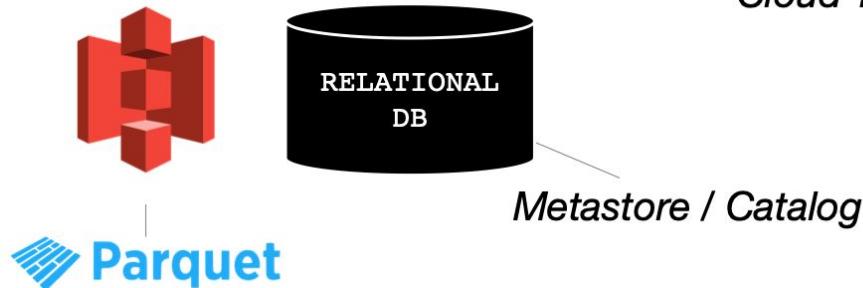
PROCESSING



FORMAT



STORAGE



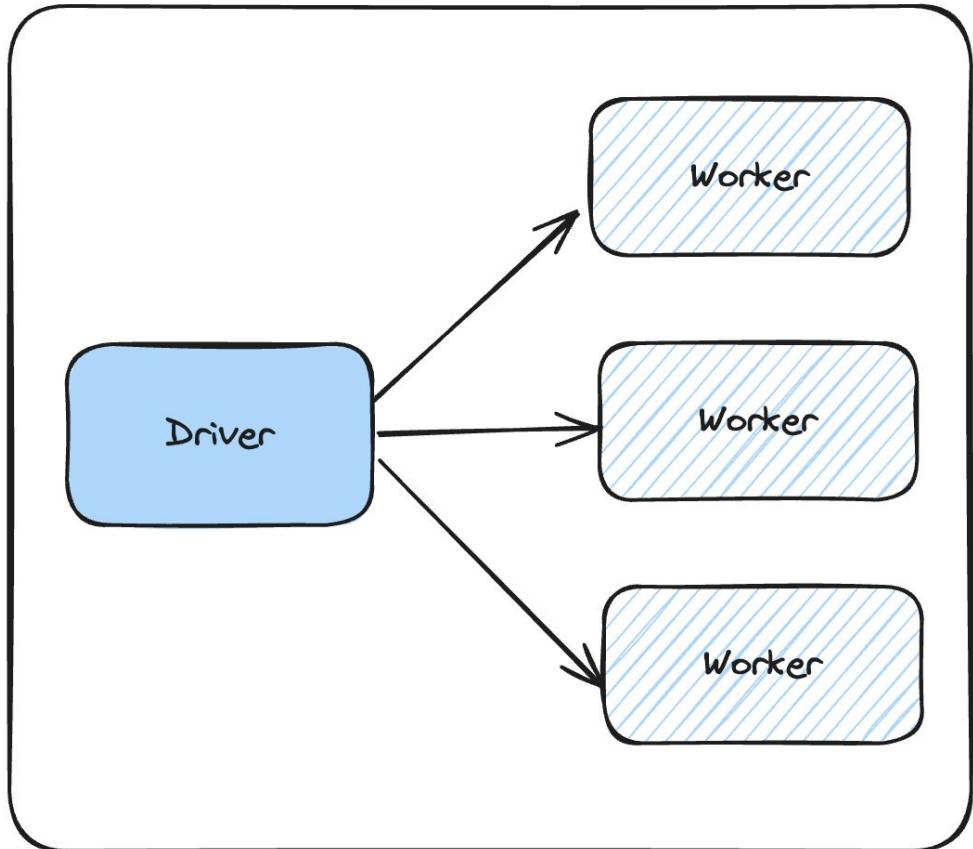
# Using Python with Apache Iceberg

**TechAtBloomberg.com**

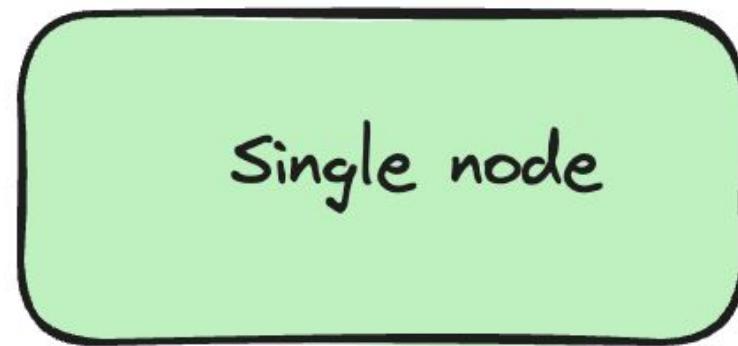
© 2024 Bloomberg Finance L.P. All rights reserved.

**Bloomberg**  
Engineering

## #1: PySpark



## #2: Pylceberg



# PySpark



**TechAtBloomberg.com**

© 2024 Bloomberg Finance L.P. All rights reserved.

**Bloomberg**  
Engineering

```
JARS = [  
    "https://xxxxx/root-repos/org/apache/iceberg/iceberg-spark-runtime-3.5_X.XX/X.X.X/iceberg-spark-runtime-3.5_X.XX-X.X.X.jar"] ←  
  
spark = (  
    SparkSession.builder.config(  
        "spark.sql.extensions",  
        "org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions",  
    )  
.config("spark.sql.catalog.XXX", "org.apache.iceberg.spark.SparkCatalog") ←  
.config("spark.sql.catalog.XXX.warehouse", "s3a://data-warehouse") ←  
.config("spark.sql.catalog.XXX.catalog-impl", "org.apache.iceberg.jdbc.JdbcCatalog") ←  
.config("spark.sql.catalog.XXX.uri", "jdbc://mydb://localhost:3306/iceberg",)  
.config("spark.jars", ",".join(JARS)))  
spark = SparkSession.builder.getOrCreate()
```

```
spark.sql("""
```

catalog namespace table



```
    CREATE TABLE IF NOT EXISTS XXX.test.europython USING iceberg  
    TBLPROPERTIES (  
        'format-version' = '2'  
    )  
    PARTITIONED BY (year(SESSION_DATE))  
    AS SELECT * FROM df
```

```
""")
```

```
df.show()
```

Last executed at 2024-06-26 15:02:29 in 227ms

SESSION_ID	SESSION_NAME	SESSION_TYPE	SESSION_DATE
1	ICEBERG WALKTHROUGH	TUTORIAL	2024-07-08
2	KEY NOTE	ANNOUNCEMENTS	2024-07-09
3	PYICEBERG TALK (LONG SESSION)		2024-07-08
4	PYTEST	TALK	2024-07-09

```
spark.sql("""select * from █████.test.europython""")
```

Last executed at 2024-06-27 16:58:01 in 4.53s

SESSION_ID	SESSION_NAME	SESSION_TYPE	SESSION_DATE
1	ICEBERG WALKTHROUGH	TUTORIAL	2024-07-08
2	KEY NOTE	ANNOUNCEMENTS	2024-07-09
3	PYICEBERG TALK (LONG SESSION)	(LONG SESSION)	2024-07-08
4	PYTEST	TALK	2024-07-09

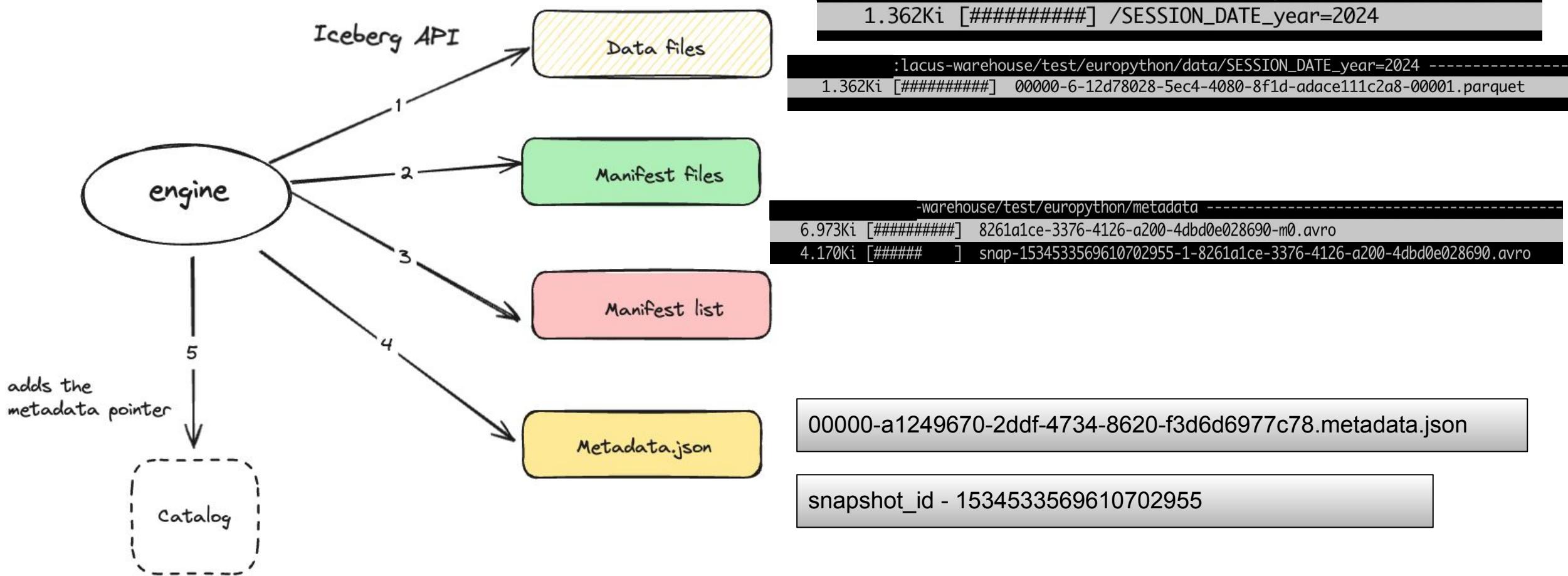
```
{"metadata-location": "s3a://XXX-warehouse/test/europython/metadata/00000-a1249670-2ddf-4734-8620-f3d6d6977c78.metadata.json", "metadata": {"location": "s3a://XXX-warehouse/test/europython", "table-uuid": "f0a46b3c-4be7-4394-891a-e131d919a249", "last-updated-ms": 1719428590109, "last-column-id": 4, "schemas": [{"type": "struct", "fields": [{"id": 1, "name": "SESSION_ID", "type": "int", "required": false}, {"id": 2, "name": "SESSION_NAME", "type": "string", "required": false}, {"id": 3, "name": "SESSION_TYPE", "type": "string", "required": false}, {"id": 4, "name": "SESSION_DATE", "type": "date", "required": false}], "schema-id": 0, "identifier-field-ids": []}], "current-schema-id": 0, "partition-specs": [{"spec-id": 0, "fields": [{"source-id": 4, "field-id": 1000, "transform": "year", "name": "SESSION_DATE_year"}]}, {"default-spec-id": 0, "last-partition-id": 1000, "properties": {"owner": "job", "write.metadata.delete-after-commit.enabled": "true", "commit.manifest.target-size-bytes": "8388608", "write.metadata.metrics.default": "full", "write.parquet.compression-codec": "zstd"}, "current-snapshot-id": 1534533569610702955, "snapshots": [{"snapshot-id": 1534533569610702955, "sequence-number": 1, "timestamp-ms": 1719428590109, "manifest-list": "s3a://XXX-warehouse/test/europython/metadata/snap-1534533569610702955-1-8261a1ce-3376-4126-a200-4dbd0e028690.avro", "summary": {"operation": "append", "spark.app.id": "spark-application-1719426812731", "added-data-files": "1", "added-records": "4", "added-files-size": "1395", "changed-partition-count": "1", "total-records": "4", "total-files-size": "1395", "total-data-files": "1", "total-delete-files": "0", "total-position-deletes": "0", "total-equality-deletes": "0"}, "schema-id": 0}], "snapshot-log": [{"snapshot-id": 1534533569610702955, "timestamp-ms": 1719428590109}], "metadata-log": [], "sort-orders": [{"order-id": 0, "fields": []}], "default-sort-order-id": 0, "refs": {"main": {"snapshot-id": 1534533569610702955, "type": "branch"}}, "format-version": 2, "last-sequence-number": 1}, "config": {}}
```

```
{  
    "metadata-location": "s3a://[REDACTED]-warehouse/test/europython/metadata/  
        00000-a1249670-2ddf-4734-8620-f3d6d6977c78.metadata.json",  
    "metadata": {  
        "location": "s3a://[REDACTED]-warehouse/test/europython",  
        "table-uuid": "f0a46b3c-4be7-4394-891a-e131d919a249",  
        "last-updated-ms": 1719428590109,  
        "last-column-id": 4,  
        "schemas": [...]  
    },  
    "current-schema-id": 0,  
    "partition-specs": [...]  
},  
    "default-spec-id": 0,  
    "last-partition-id": 1000,  
    "properties": {  
        "owner": "job",  
        "write.metadata.delete-after-commit.enabled": "true",  
        "commit.manifest.target-size-bytes": "8388608",  
        "write.metadata.metrics.default": "full",  
        "write.parquet.compression-codec": "zstd"  
    },  
    "current-snapshot-id": 1534533569610702800,  
    "snapshots": [...]  
},  
    "snapshot-log": [...]  
},  
    "metadata-log": [],  
    "sort-orders": [...]  
},  
    "default-sort-order-id": 0,  
    "refs": {...}  
},  
    "format-version": 2,  
    "last-sequence-number": 1  
},  
    "config": {}  
}
```

-warehouse/test/europython -----

20.435Ki [#####] /metadata

1.362Ki [# ] /data



```
-- status: integer (nullable = true)
-- snapshot_id: long (nullable = true)
-- sequence_number: long (nullable = true)
-- file_sequence_number: long (nullable = true)
-- data_file: struct (nullable = true)
  -- content: integer (nullable = true)
  -- file_path: string (nullable = true)
  -- file_format: string (nullable = true)
  -- partition: struct (nullable = true)
    -- SESSION_DATE_year: integer (nullable = true)
-- record_count: long (nullable = true)
-- file_size_in_bytes: long (nullable = true)
-- column_sizes: array (nullable = true)
  -- element: struct (containsNull = true)
    -- key: integer (nullable = true)
    -- value: long (nullable = true)
-- value_counts: array (nullable = true)
  -- element: struct (containsNull = true)
    -- key: integer (nullable = true)
    -- value: long (nullable = true)
-- null_value_counts: array (nullable = true)
  -- element: struct (containsNull = true)
    -- key: integer (nullable = true)
    -- value: long (nullable = true)
-- nan_value_counts: array (nullable = true)
  -- element: struct (containsNull = true)
    -- key: integer (nullable = true)
    -- value: long (nullable = true)
-- lower_bounds: array (nullable = true)
  -- element: struct (containsNull = true)
    -- key: integer (nullable = true)
    -- value: binary (nullable = true)
-- upper_bounds: array (nullable = true)
  -- element: struct (containsNull = true)
    -- key: integer (nullable = true)
    -- value: binary (nullable = true)
-- key_metadata: binary (nullable = true)
-- split_offsets: array (nullable = true)
  -- element: long (containsNull = true)
-- equality_ids: array (nullable = true)
  -- element: integer (containsNull = true)
-- sort_order_id: integer (nullable = true)
```

```
root
  -- manifest_path: string (nullable = true)
  -- manifest_length: long (nullable = true)
  -- partition_spec_id: integer (nullable = true)
  -- content: integer (nullable = true)
  -- sequence_number: long (nullable = true)
  -- min_sequence_number: long (nullable = true)
  -- added_snapshot_id: long (nullable = true)
  -- added_data_files_count: integer (nullable = true)
  -- existing_data_files_count: integer (nullable = true)
  -- deleted_data_files_count: integer (nullable = true)
  -- added_rows_count: long (nullable = true)
  -- existing_rows_count: long (nullable = true)
  -- deleted_rows_count: long (nullable = true)
  -- partitions: array (nullable = true)
    -- element: struct (containsNull = true)
      -- contains_null: boolean (nullable = true)
      -- contains_nan: boolean (nullable = true)
      -- lower_bound: binary (nullable = true)
      -- upper_bound: binary (nullable = true)
```

# Schema evolution - metadata only!!!

```
spark.sql("ALTER TABLE XXX.test.europython ADD COLUMNS (SESSION_DURATION_MINS int)")
```

```
spark.sql("""select * from [REDACTED] test.europython""")
```

Last executed at 2024-06-27 16:58:01 in 4.53s

SESSION_ID	SESSION_NAME	SESSION_TYPE	SESSION_DATE	SESSION_DURATION_MINS
1	ICEBERG WALKTHROUGH	TUTORIAL	2024-07-08	NULL
2	KEY NOTE	ANNOUNCEMENTS	2024-07-09	NULL
3	PYICEBERG TALK (LONG SESSION)	TALK	2024-07-08	NULL
4	PYTEST	TALK	2024-07-09	NULL



# Updated Schema in metadata.json

```
"schemas": [{"type": "struct", "fields": [{"id": 1, "name": "SESSION_ID", "type": "int", "required": false}, {"id": 2, "name": "SESSION_NAME", "type": "string", "required": false}, {"id": 3, "name": "SESSION_TYPE", "type": "string", "required": false}, {"id": 4, "name": "SESSION_DATE", "type": "date", "required": false}], "schema-id": 0, "identifier-field-ids": []},  
  
{"type": "struct", "fields": [{"id": 1, "name": "SESSION_ID", "type": "int", "required": false}, {"id": 2, "name": "SESSION_NAME", "type": "string", "required": false}, {"id": 3, "name": "SESSION_TYPE", "type": "string", "required": false}, {"id": 4, "name": "SESSION_DATE", "type": "date", "required": false}, {"id": 5, "name": "SESSION_DURATION_MINS", "type": "int", "required": false}], "schema-id": 1, "identifier-field-ids": []}],  
  
"current-schema-id": 1
```



```
"schemas": [  
  {  
    "type": "struct",  
    "fields": [  
      {  
        "id": 1,  
        "name": "SESSION_ID",  
        "type": "int",  
        "required": false  
      },  
      {  
        "id": 2,  
        "name": "SESSION_NAME",  
        "type": "string",  
        "required": false  
      },  
      {  
        "id": 3,  
        "name": "SESSION_TYPE",  
        "type": "string",  
        "required": false  
      },  
      {  
        "id": 4,  
        "name": "SESSION_DATE",  
        "type": "date",  
        "required": false  
      }  
    ]  
},
```

```
spark.sql("SELECT SESSION_ID, SESSION_NAME FROM  
XXX.TEST.EUROPYTHON;")
```

**Columns are selected from  
data files using field IDs**

# Pylceberg API

**TechAtBloomberg.com**

© 2024 Bloomberg Finance L.P. All rights reserved.

**Bloomberg**  
Engineering

```
>>> pip install "pyiceberg[sql-sqlite]"
```

```
1  from pyiceberg.catalog.sql import SqlCatalog
2
3  warehouse_path = "/tmp/warehouse"
4  catalog = SqlCatalog(
5      "default",
6      **{
7          "uri": f"sqlite:///{warehouse_path}/pyiceberg_catalog.db",
8          "warehouse": f"file://{warehouse_path}",
9      },
10 )
11 catalog.create_namespace("test")
```

```
>>> pip install pyarrow
```

```
import pyarrow.parquet as pq  
data = pq.read_table("s3://test/insert_europython.parquet")
```

---

```
pyarrow.Table
```

```
SESSION_ID: int64
```

```
SESSION_NAME: string
```

```
SESSION_TYPE: string
```

```
SESSION_DATE: string
```

```
DURATION_MINS: int64
```

```
----
```

```
SESSION_ID: [[5, 6]]
```

```
SESSION_NAME: [[ "Airflow and Python", "Python modules in Rust" ]]
```

```
SESSION_TYPE: [[ "TALK", "TALK" ]]
```

```
SESSION_DATE: [[ "2023-07-07", "2023-07-08" ]]
```

```
DURATION_MINS: [[30, 45]]
```

```
tbl = catalog.create_table("test.europython_pyiceberg", schema=data.schema)
tbl.append(data)
```

---

```
/tmp/warehouse/  
/tmp/warehouse//pyiceberg_catalog.db  
/tmp/warehouse//test.db  
/tmp/warehouse//test.db/europython_pyiceberg  
/tmp/warehouse//test.db/europython_pyiceberg/data  
/tmp/warehouse//test.db/europython_pyiceberg/data/00000-0-ee45bb2f-6c80-4272-a2e9-0dc45a826b  
1c.parquet  
/tmp/warehouse//test.db/europython_pyiceberg/metadata  
/tmp/warehouse//test.db/europython_pyiceberg/metadata/00001-384de88c-d052-4148-9e43-d77a4630  
3de1.metadata.json  
/tmp/warehouse//test.db/europython_pyiceberg/metadata/e95cd1f9-5fc6-4535-991c-ee90212ba3a2-m  
0.avro  
/tmp/warehouse//test.db/europython_pyiceberg/metadata/00000-356f91ad-6777-409f-a4fb-0f052f31  
fa1f.metadata.json  
/tmp/warehouse//test.db/europython_pyiceberg/metadata/snap-1181746681384757451-0-e95cd1f9-5f  
c6-4535-991c-ee90212ba3a2.avro
```

```
from pyiceberg.expressions import GreaterThanOrEqual

scan = tbl.scan(
    row_filter=GreaterThanOrEqual("DURATION_MINS", 35),
    selected_fields=("SESSION_NAME", "SESSION_DATE"),
    limit=10,
)

file_paths = [task.file.file_path for task in scan.plan_files()]

['file:///tmp/warehouse/test.db/europython_pyiceberg/data/00000-0-ee45bb
2f-6c80-4272-a2e9-0dc45a826b1c.parquet']
```

```
scan.to_arrow()  
scan.to_pandas()
```

pyarrow.Table

SESSION\_NAME: string

SESSION\_DATE: string

----

SESSION\_NAME: [[ "Python modules in Rust" ]]

SESSION\_DATE: [[ "2023-07-08" ]]

# Pylceberg 0.7.0

The main objective of 0.7.0 is to have partitioned writes (non-exhaustive list :)

- Support for merge-into / upsert: [Merge into / Upsert #402](#)
  - Support partitioned appends: [Support Appends with TimeTransform Partitions #784](#)
  - Support partial deletes: [Support partial deletes #569](#)
- Support for parallelizing writes: [Parallel Table.append #428](#) [Faster ingestion from Parquet #346](#)
  - Support parallelized writes: [Bin-pack Writes Operation into multiple parquet files, and parallelize writing WriteTask S #444](#)
- Support `table_exists` on catalog: [check if table exist #406](#) [Add `table\_exists` method to the Catalog #507](#), fixed in [Add `table\_exists` method to Catalog #512](#)
- Metadata tables:
  - Files assigned to [@Gowthami03B](#), PR in [Add Files metadata table #614](#)
  - Snapshots assigned to [@Fokko](#) in [Add Snapshots table metadata #524](#)
  - History assigned to [@ndrluis](#) in [Add history inspect table #828](#)
  - Metadata log entries [@kevinjqliu](#) (issue in [feat request\] Add `metadata\_log\_entries` metadata table #594](#)):
    - [Metadata Log Entries metadata table #667](#)
  - Manifests [@geruh](#): PR in [Add manifests metadata table #717](#)
  - Partitions assigned to [@syun64](#) (issue in [Support get partition table with filter #24](#)): [Add Partitions Metadata Table #603](#)
  - References assigned to [@geruh](#) in [Add Refs metadata table #602](#)
  - Entries assigned to [@Fokko](#) in [Add entries metadata table #551](#)
- Manifest read/write improvements:
  - Implement rolling writes: [Implement rolling manifest-writers #596](#) [feat: add RollingManifestWriter #650](#)
  - Caching of manifests: [Implement caching of manifest-files #595](#)
- Incremental append scan: [Incremental Append Scan #533](#)

Table of contents

Create a table

Load a table

Catalog table

Static table

Write support

Schema evolution

Union by Name

Add column

Rename column

Move column

Update column

Delete column

Table properties

Query the data

Apache Arrow

Pandas

DuckDB

Ray

Daft

# Contributions to the Pylceberg community

## Support partitioned writes #208

[Open](#) [4 tasks](#) Fokko opened this issue on Dec 12, 2023 · 26 comments

apache / iceberg-python

Code Issues Pull requests Actions Projects Security Insights

**add\_files support partitioned tables #531**

Merged Fokko merged 5 commits into apache:main from syun64:add-files-partitioned on Mar 21

Conversation 20 Commits 5 Checks 7 Files changed 6

syun64 commented on Mar 18 · edited

As a follow up to #506, this PR introduces the support for adding files as DataFiles to partitioned tables.

## allow override env-variables in load\_catalog #45

Merged Fokko merged 2 commits into apache:main from bdilday:bf-config-rhs-override on Oct 13, 2023

Conversation 3 Commits 2 Checks 6 Files changed 4

bdilday commented on Oct 6, 2023

## Partitioned Append on Identity Transform #555

Merged Fokko merged 20 commits into apache:main from jquin61:identity-partitioned-append on Apr 5

Conversation 42 Commits 20 Checks 7 Files changed 11

jquin61 commented on Mar 28 · edited

## Change Append/Overwrite API to accept snapshot properties #419

Merged Fokko merged 13 commits into apache:main from Gowthami03B:snapshot-properties on Mar 19

Conversation 15 Commits 13 Checks 7 Files changed 3

Gowthami03B commented on Feb 12

No description provided.

## Retry with new Access Token on 419 response #340

Merged Fokko merged 10 commits into apache:main from anupam-saini:as-token-refresh on Feb 20

Conversation 6 Commits 10 Checks 6 Files changed 4

anupam-saini commented on Jan 31

Closes #234

- Attempts to retry 2 more times if the initial call fails.
- The retry is only triggered for AuthorizationExpiredError response.

Bloomberg

Engineering

# Thank you!

We are hiring: [bloomberg.com/engineering](https://bloomberg.com/engineering)

Contact me: [gbhogireddy@bloomberg.net](mailto:gbhogireddy@bloomberg.net) 

TechAtBloomberg.com

# Bloomberg

## Engineering

# Q&A

[TechAtBloomberg.com](https://TechAtBloomberg.com)

© 2024 Bloomberg Finance L.P. All rights reserved.