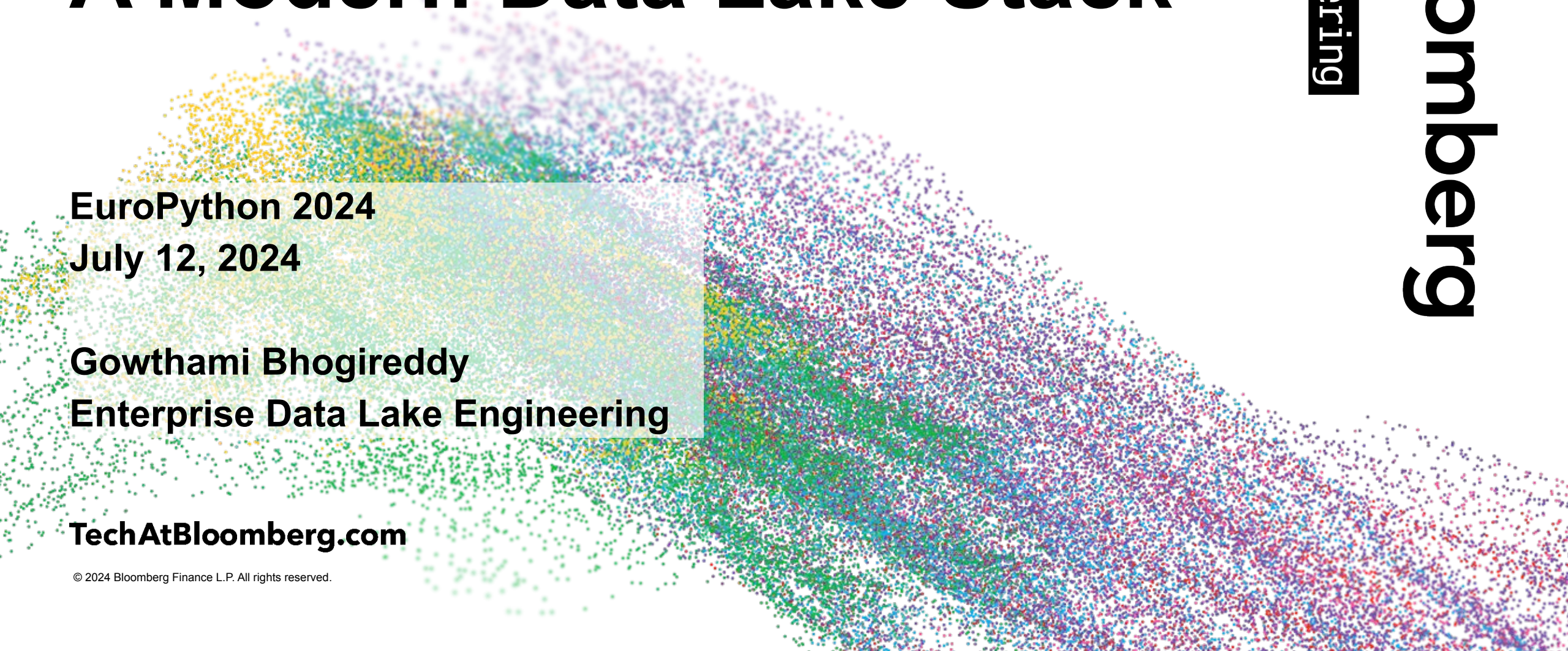
Exploring Apache Iceberg - A Modern Data Lake Stack

Bloomberg

Engineering



EuroPython 2024
July 12, 2024

Gowthami Bhogireddy
Enterprise Data Lake Engineering

TechAtBloomberg.com

Who Am I?

Software Engineer at Bloomberg

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

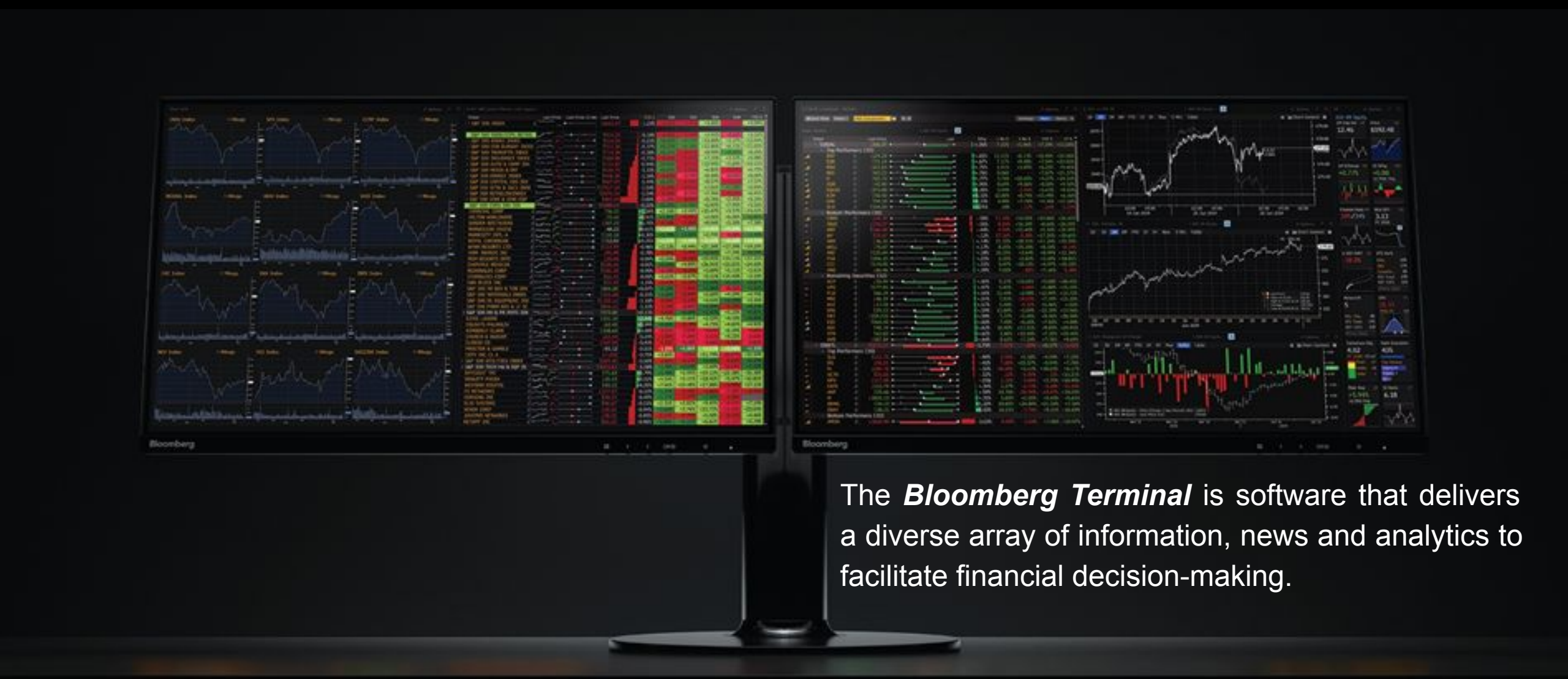
👉 PyIceberg Open Source Contributor

👉 First time EuroPython speaker, yay!

TechAtBloomberg.com

Bloomberg

Engineering

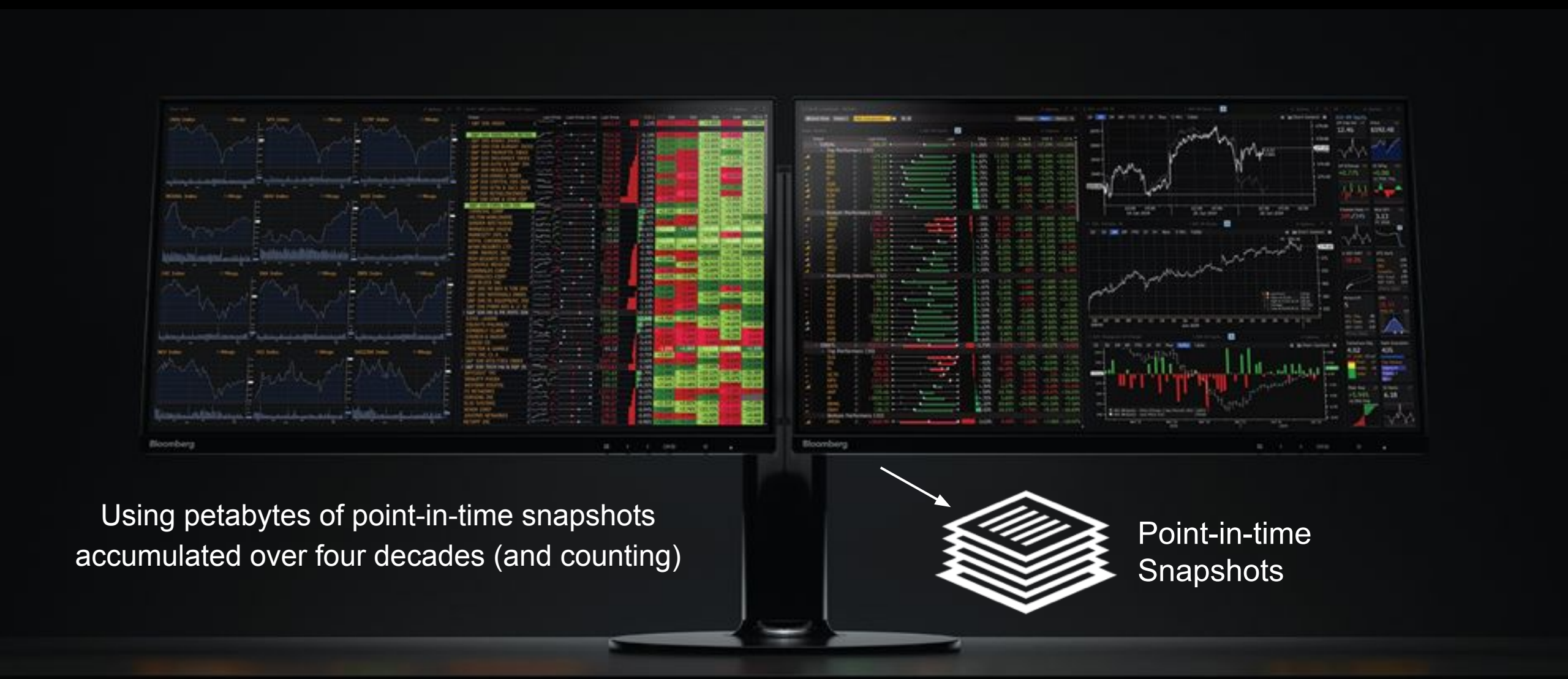


The ***Bloomberg Terminal*** is software that delivers a diverse array of information, news and analytics to facilitate financial decision-making.

TechAtBloomberg.com

Bloomberg

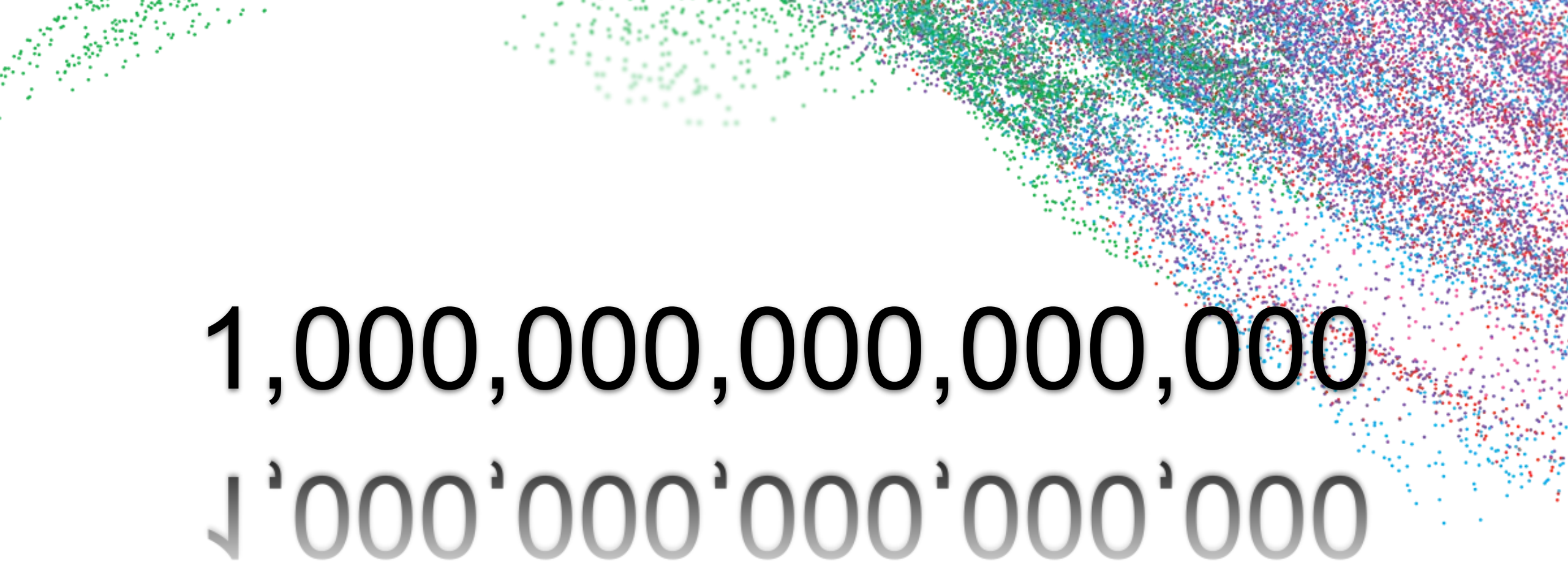
Engineering



TechAtBloomberg.com

Bloomberg

Engineering



TechAtBloomberg.com

Bloomberg

Engineering

Agenda

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

TechAtBloomberg.com

Bloomberg

Engineering

Building a historical data lake using Apache Iceberg

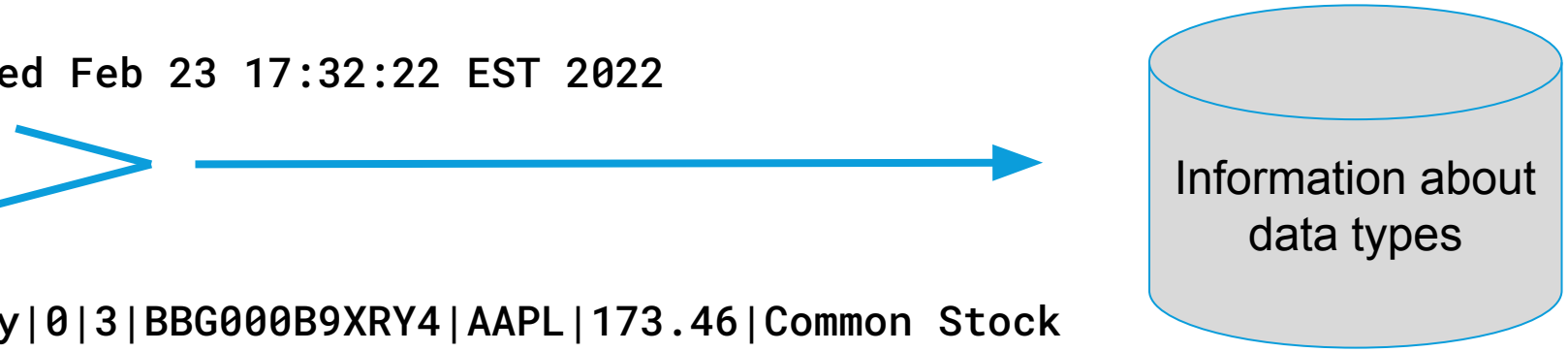
TechAtBloomberg.com

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
```

TechAtBloomberg.com

Bloomberg

Engineering

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

TechAtBloomberg.com

Bloomberg

Engineering

TechAtBloomberg.com

Bloomberg

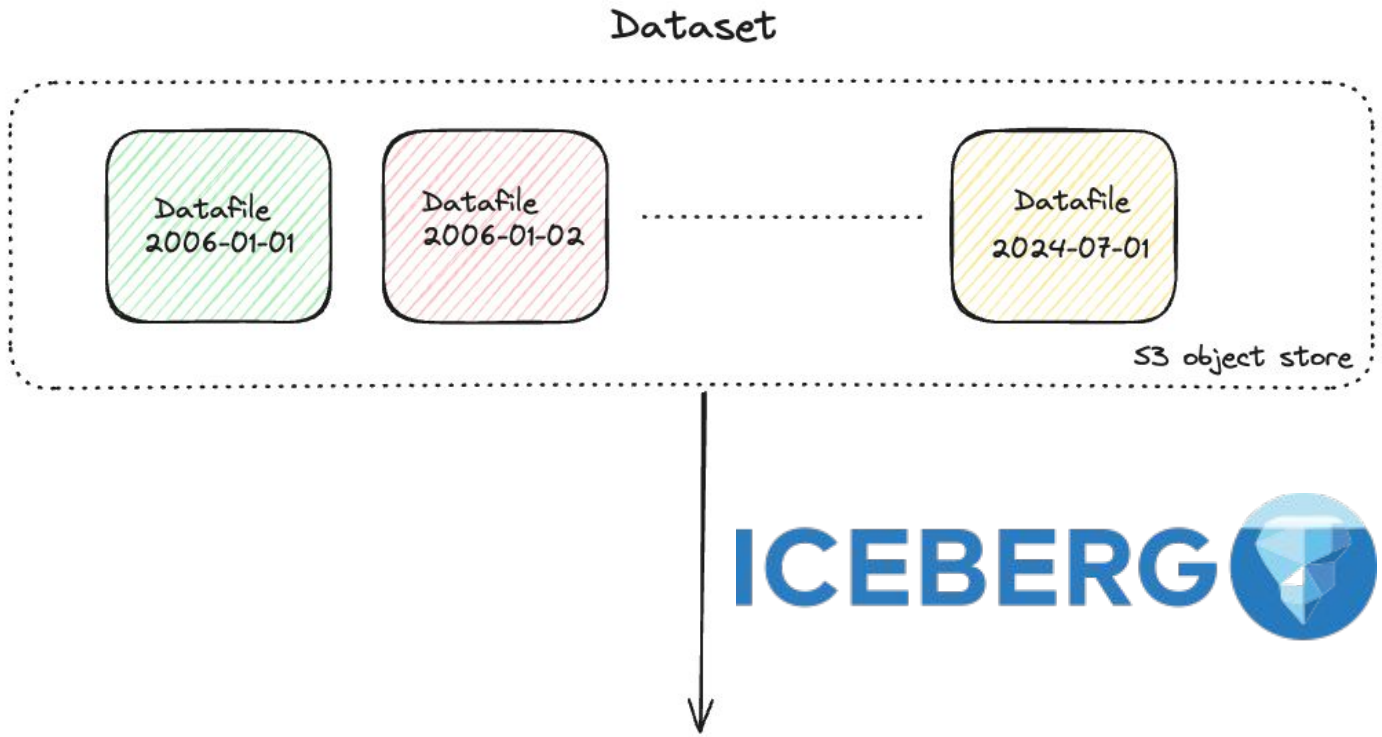
Engineering

What is Iceberg?

TechAtBloomberg.com

Bloomberg

Engineering



SQL
LIKE
BEHAVIOR

DATE_PUBLISHED	SECURITY	PRICE
2006-01-02	APPL	34.4
2024-07-01	APPL	200.08

TechAtBloomberg.com

Bloomberg

Engineering

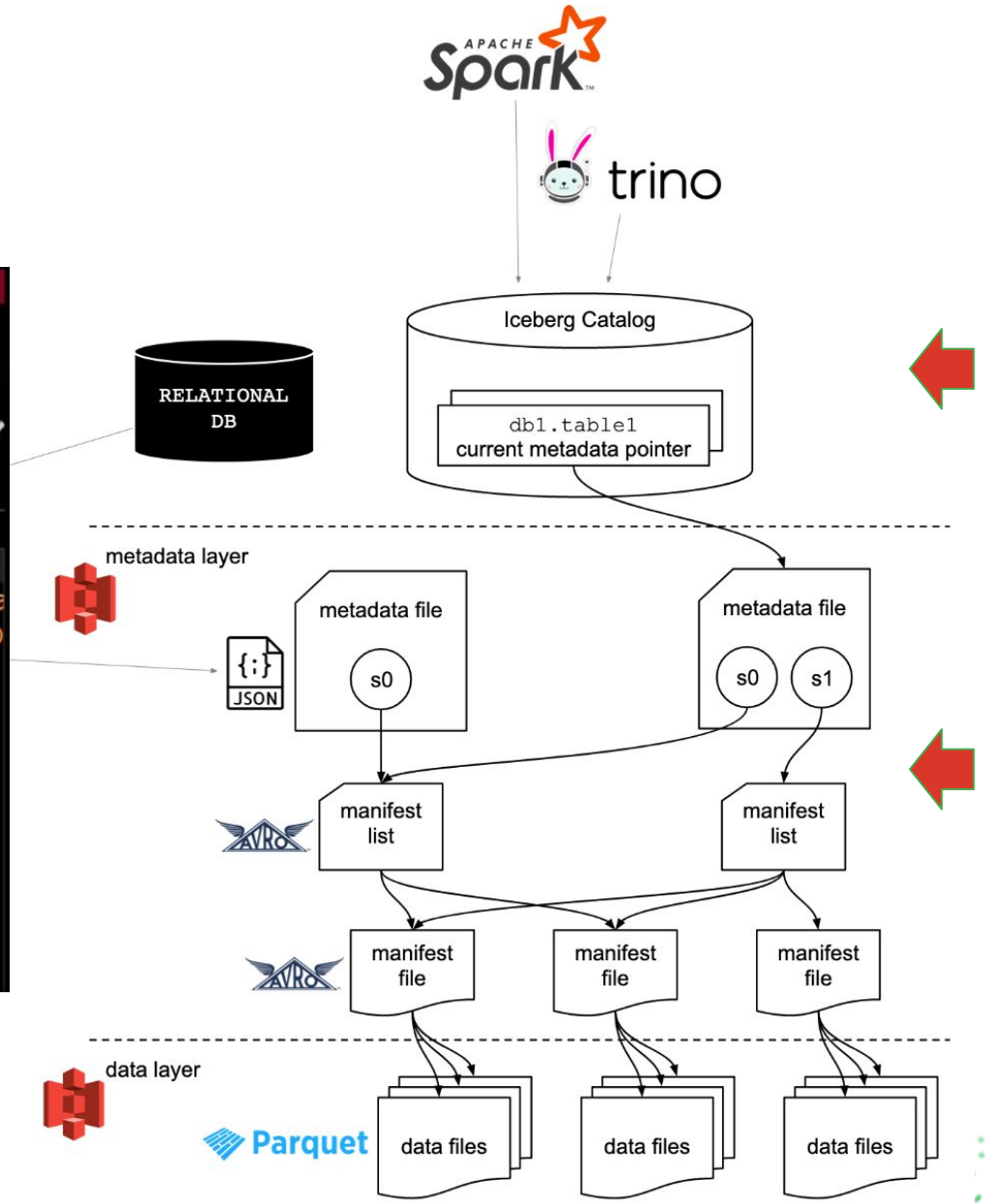
✅ Open table format

❌ Storage layer

❌ Compute Engine

Iceberg Internals

Bloomberg Database UI



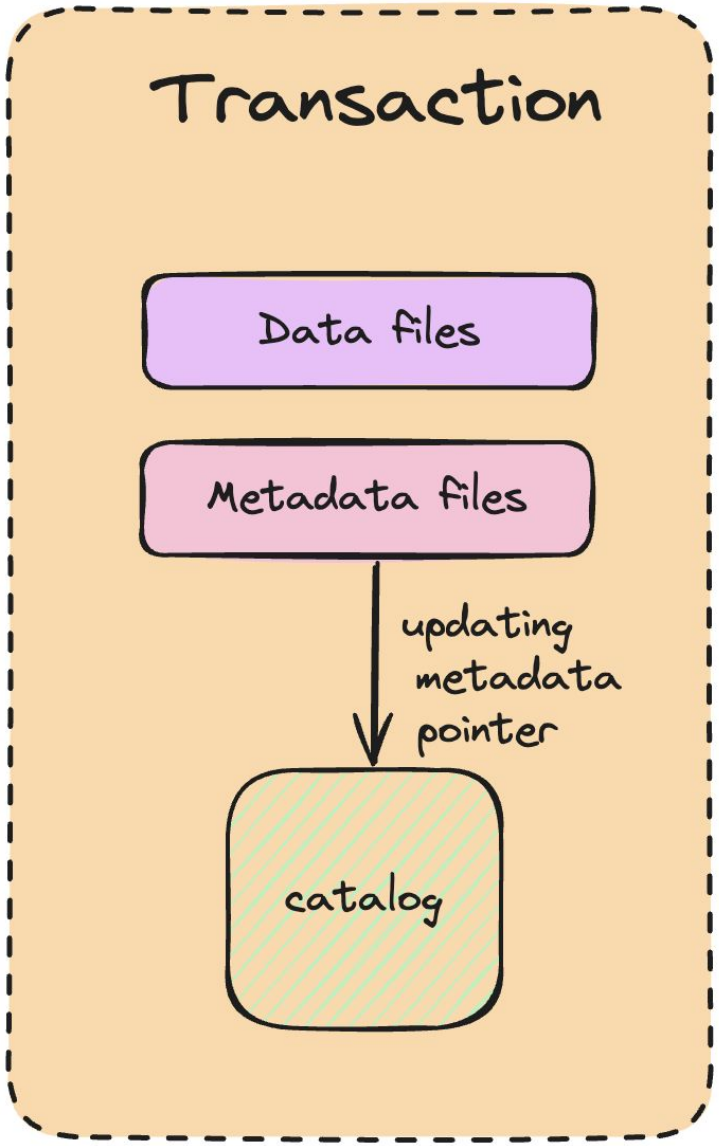
Why Iceberg?

TechAtBloomberg.com

Bloomberg

Engineering

ALL OR NOTHING



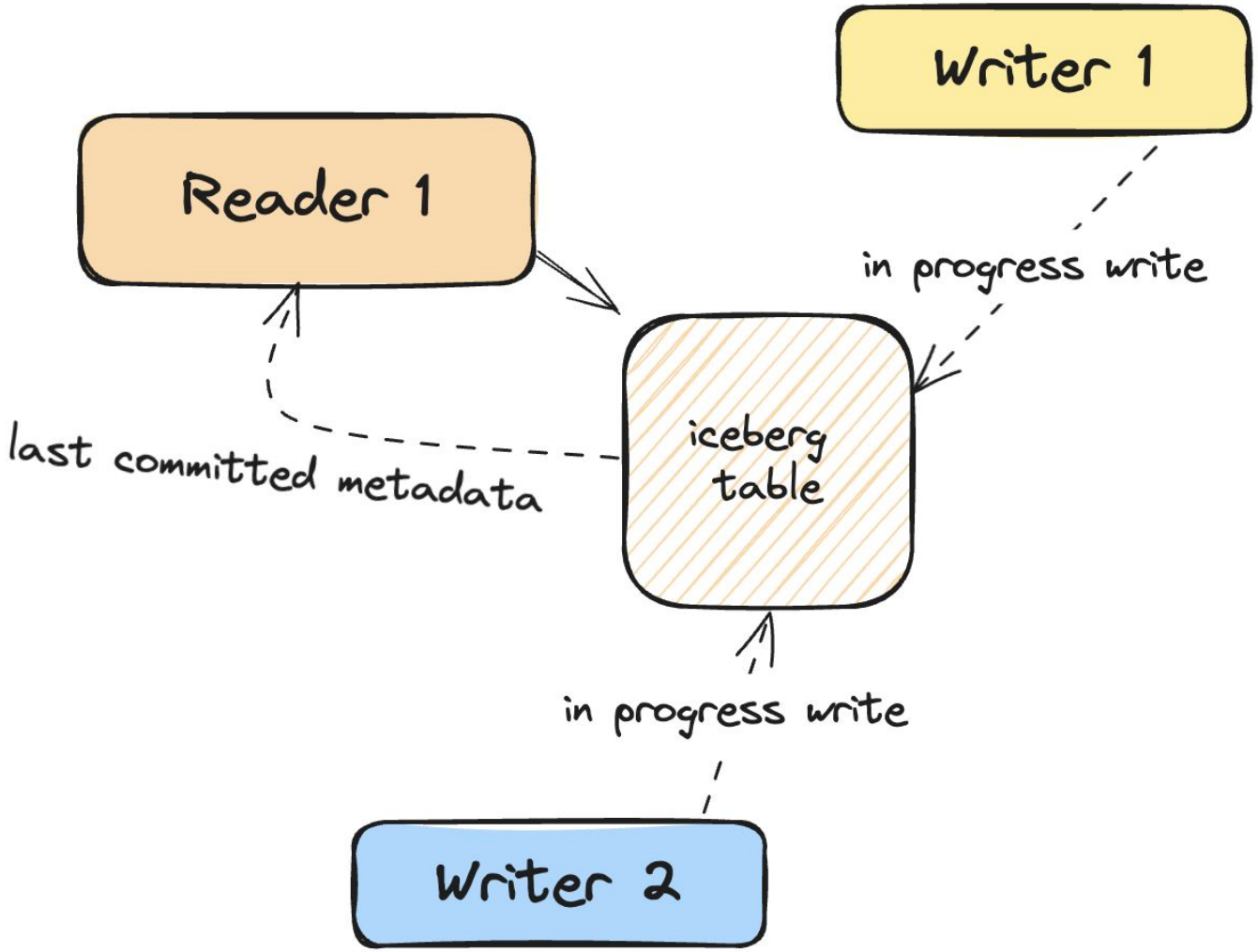
TechAtBloomberg.com

Bloomberg

Engineering

OPTIMISTIC CONCURRENCY

ISOLATION



TechAtBloomberg.com

Bloomberg

Engineering

Schema Evolution

TechAtBloomberg.com

Bloomberg

Engineering

Europython

SESSION_ID
SESSION_NAME
SESSION_TYPE

\+ SESSION_DURATION

Europython

SESSION_ID
SESSION_NAME
SESSION_TYPE
SESSION_DURATION

Metadata Only!!

TechAtBloomberg.com

Bloomberg

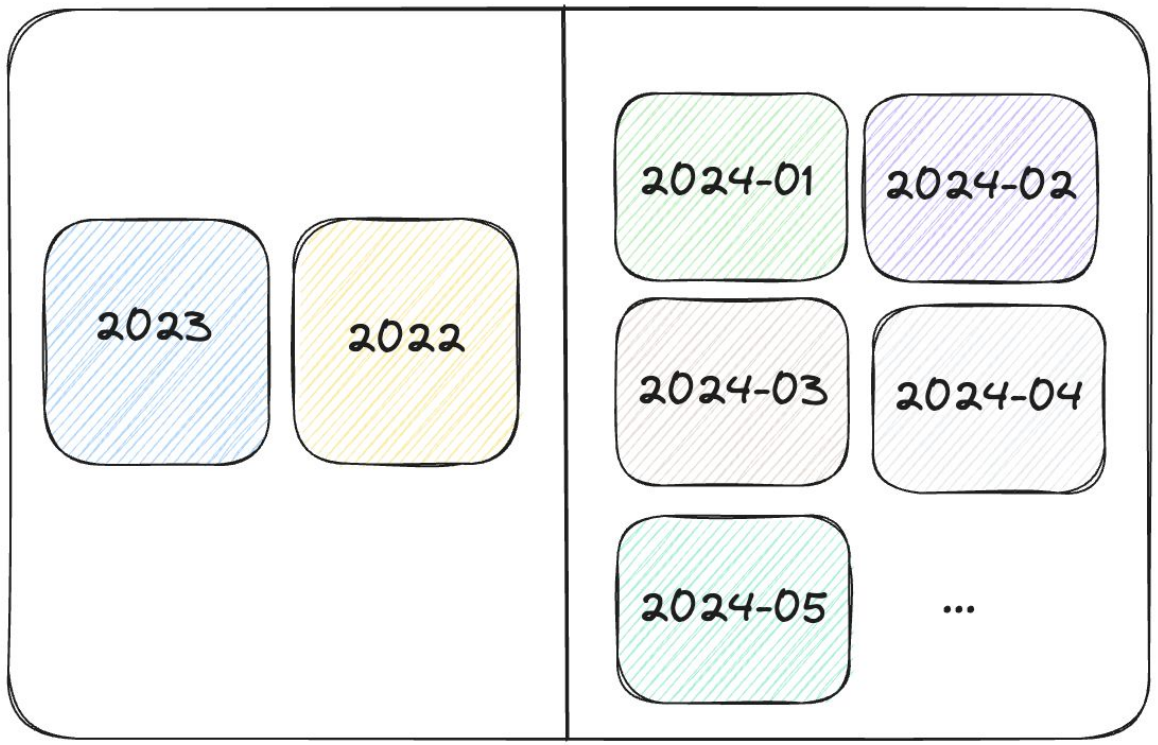
Engineering

Partition Evolution

TechAtBloomberg.com

Bloomberg

Engineering



TechAtBloomberg.com

Bloomberg

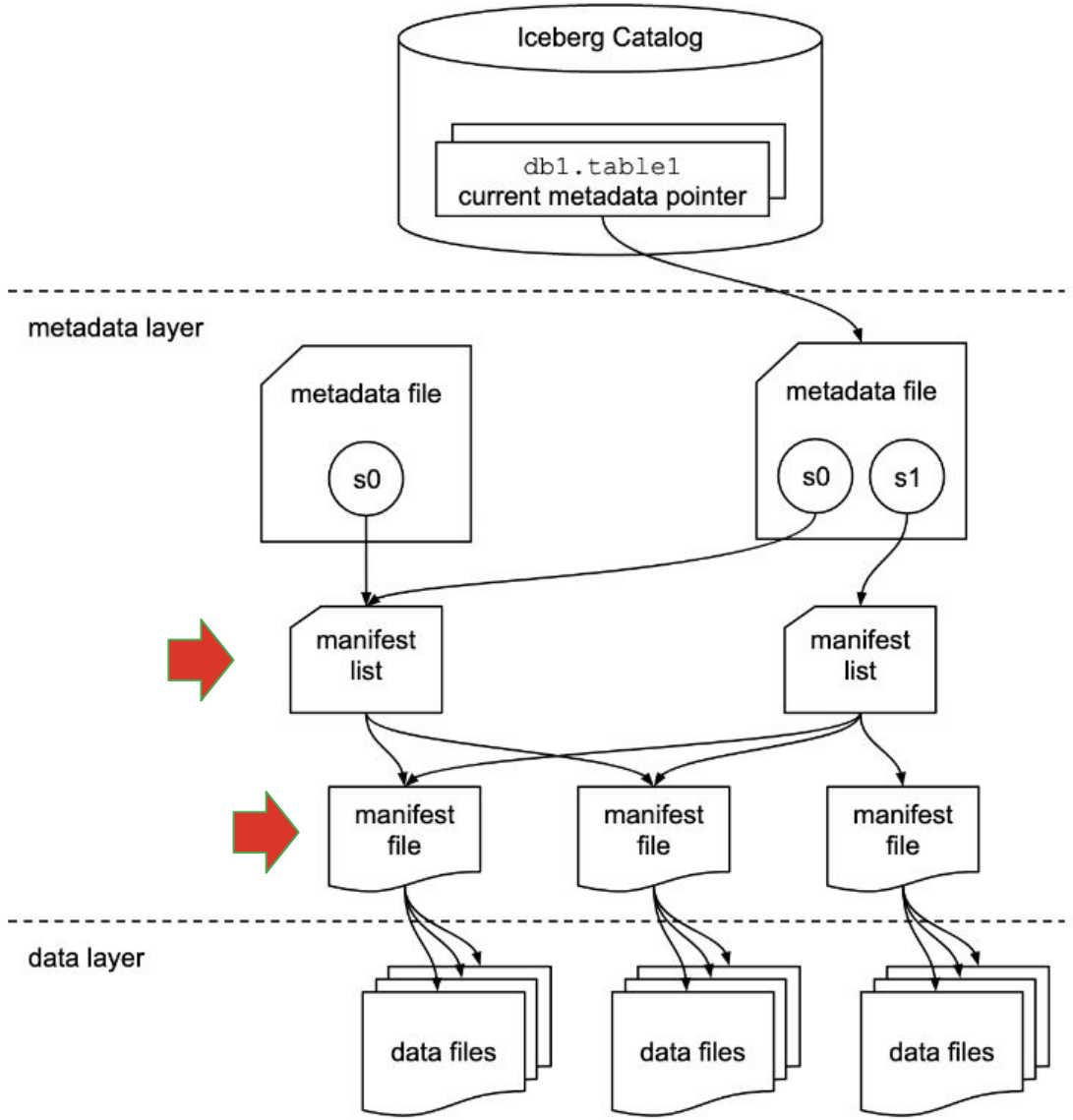
Engineering

Query Planning

TechAtBloomberg.com

Bloomberg

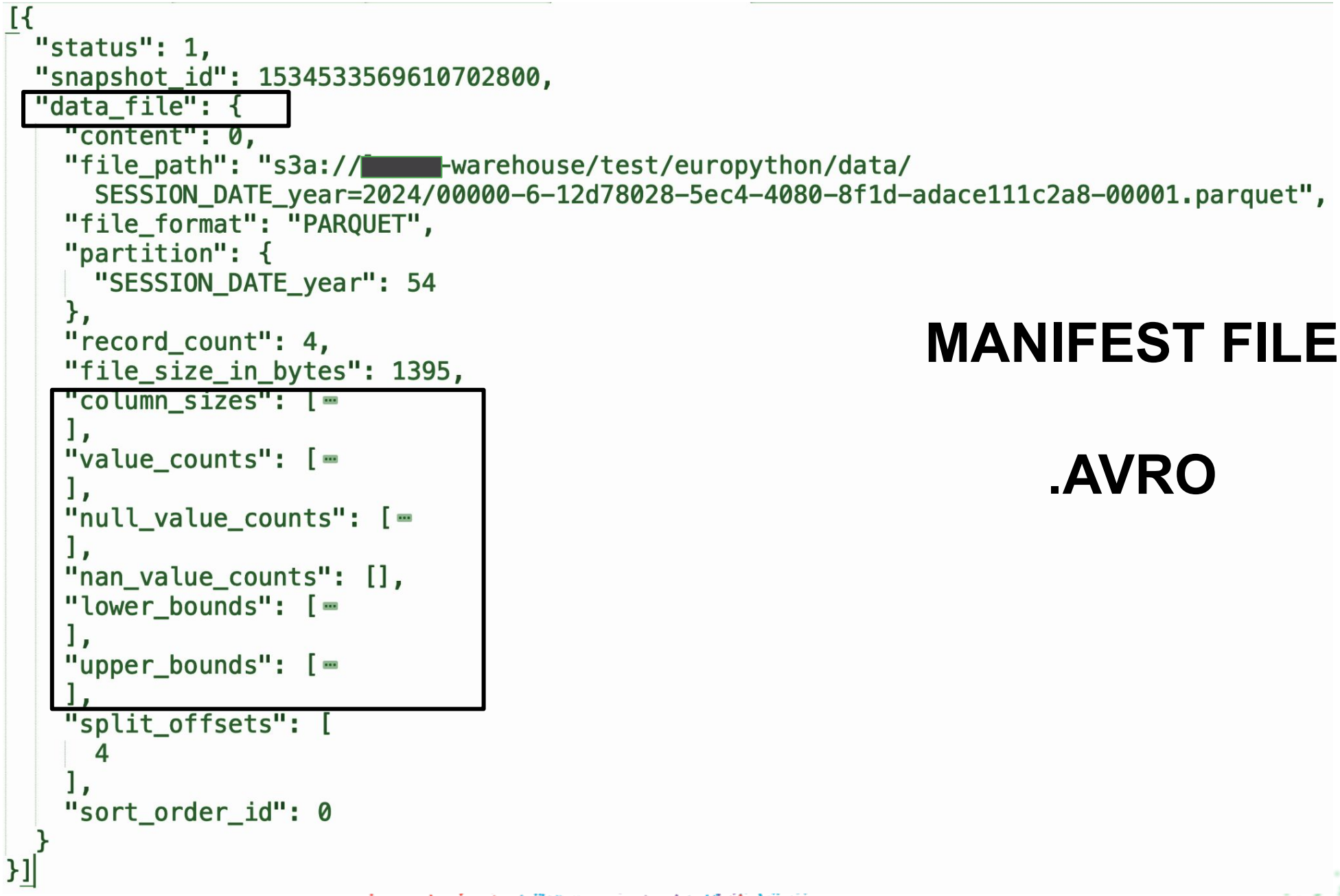
Engineering



TechAtBloomberg.com

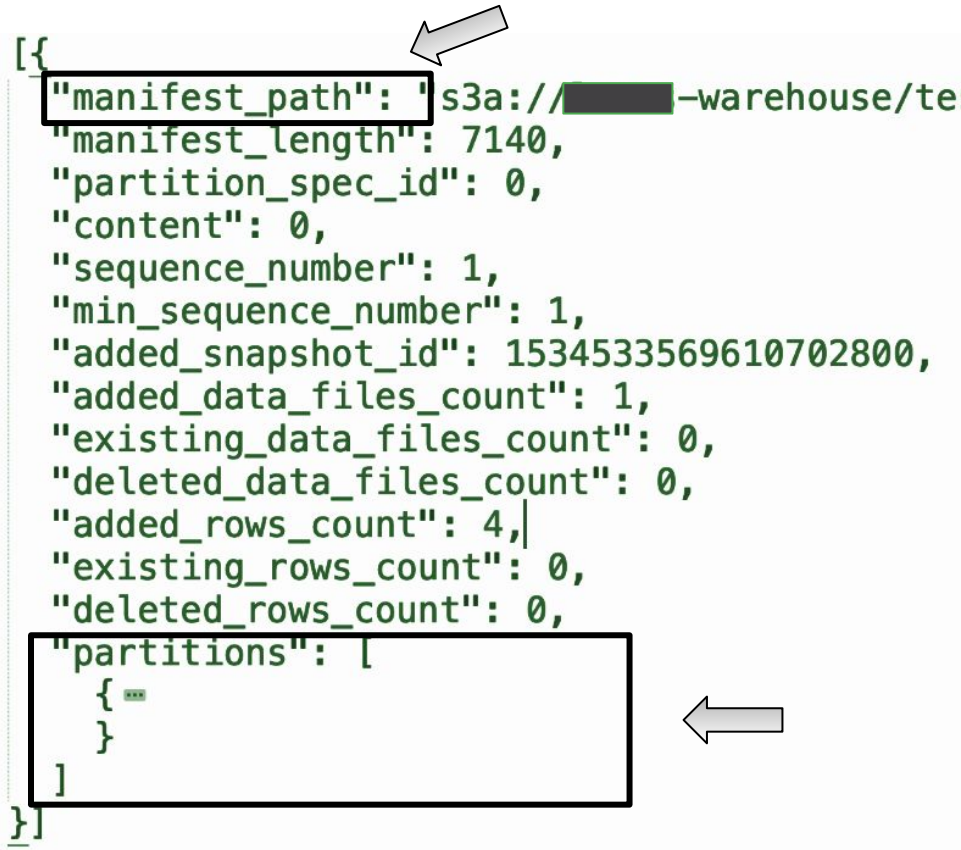
Bloomberg

Engineering



MANIFEST FILE

.AVRO



MANIFEST LIST

.AVRO

TechAtBloomberg.com

Bloomberg

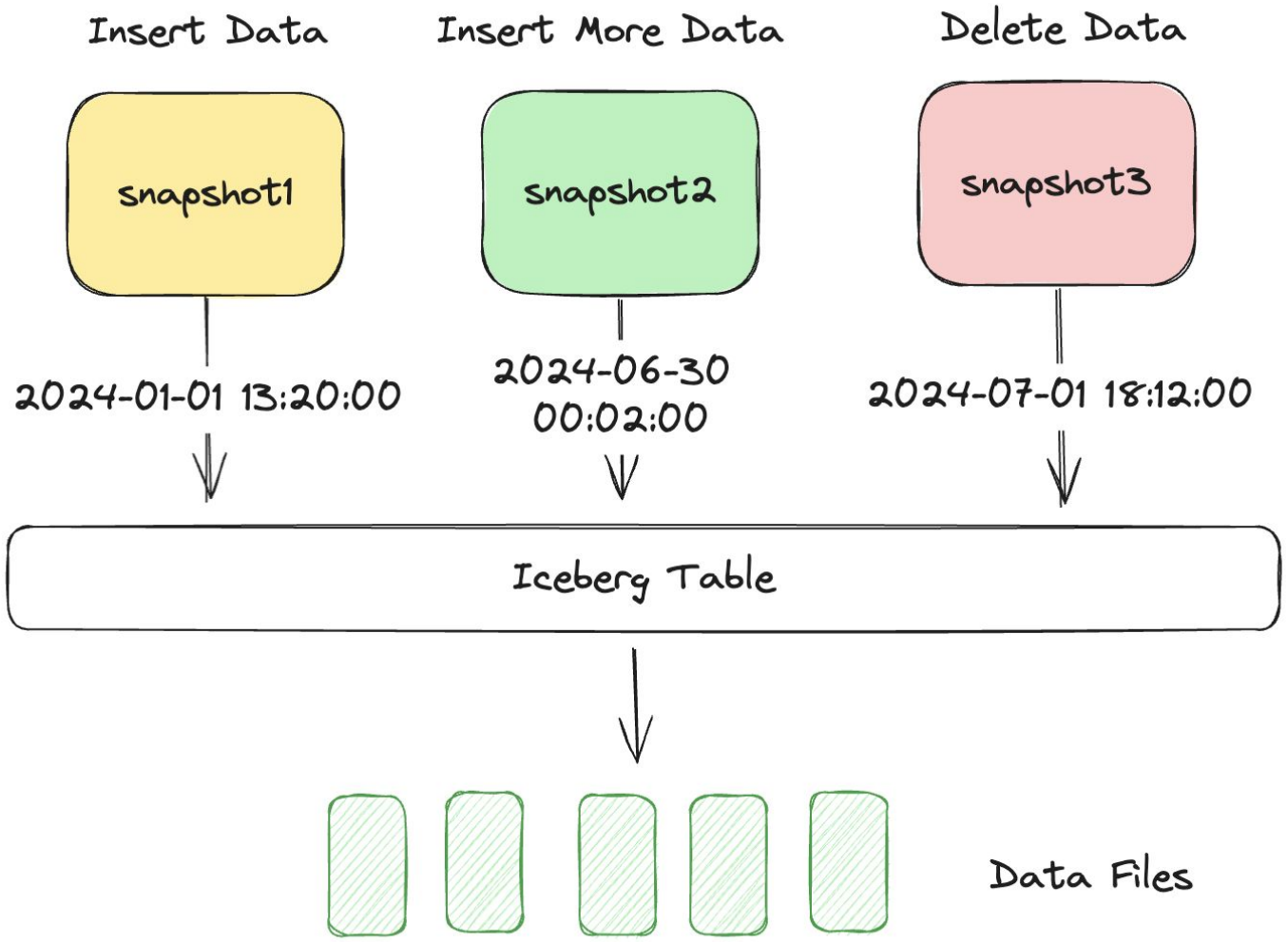
Engineering

Time Travel

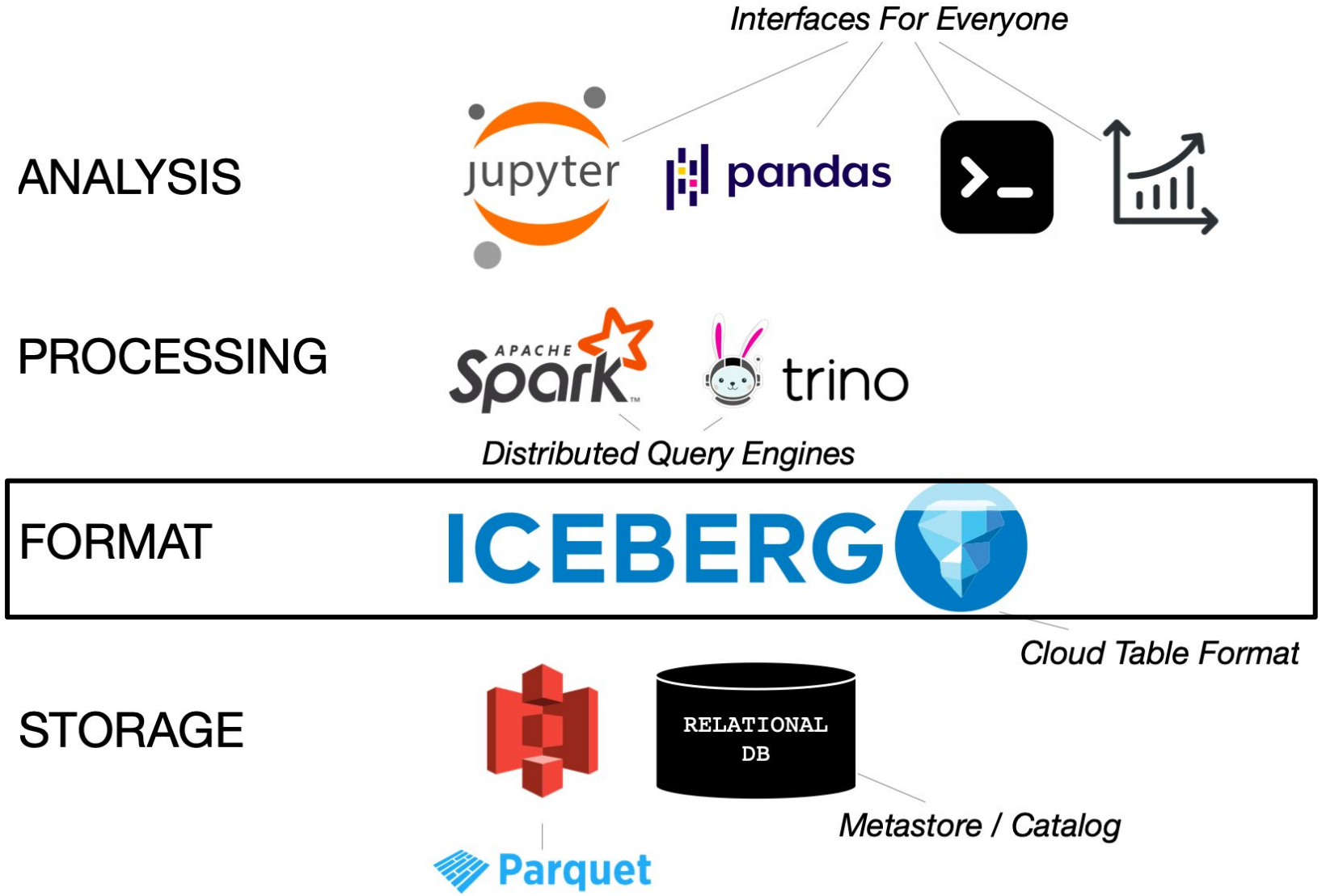
TechAtBloomberg.com

Bloomberg

Engineering




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



TechAtBloomberg.com

Bloomberg

Engineering

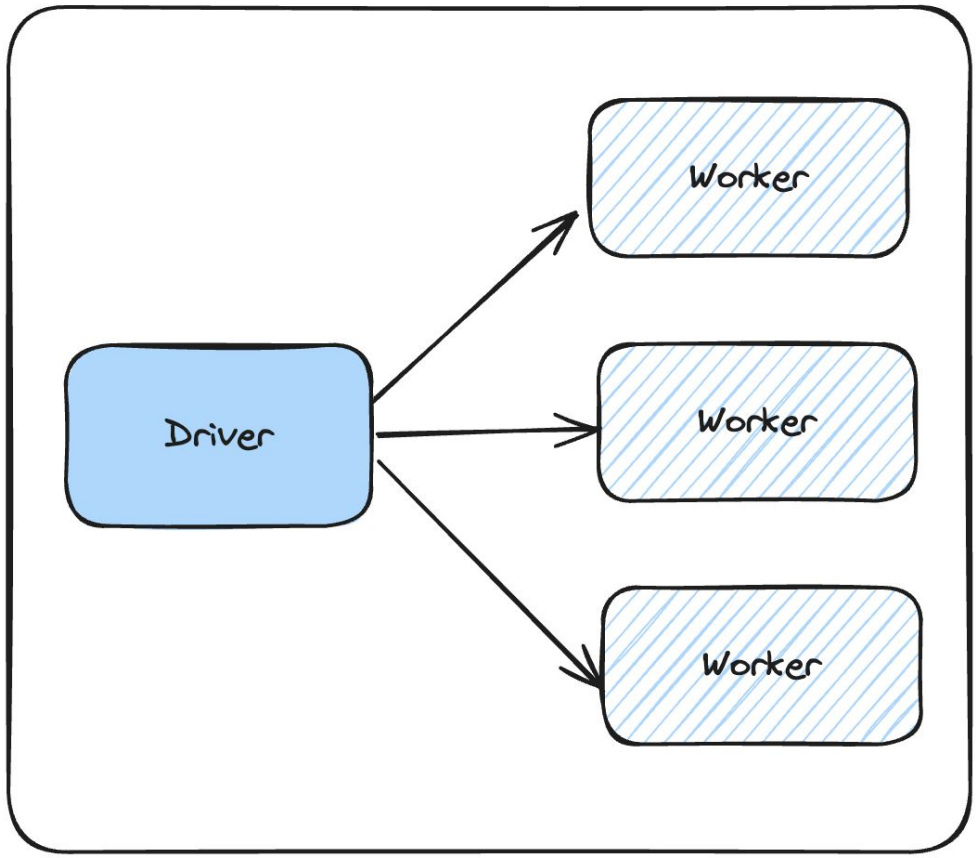
Using Python with Apache Iceberg

TechAtBloomberg.com

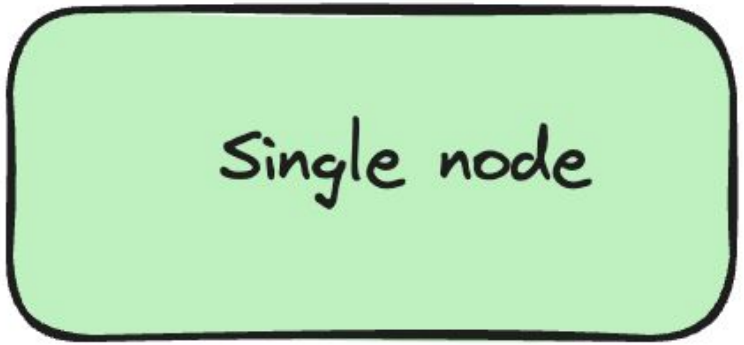
Bloomberg

Engineering

#1: PySpark



#2: Pylceberg



TechAtBloomberg.com

Bloomberg

Engineering

PySpark

TechAtBloomberg.com

Bloomberg

Engineering

```
JARS = [
"https://xxxxx/root-repos/org/apache/iceberg/iceberg-spark-runtime-3.5_X.XX/X.X.X/iceb
erg-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()
```

catalog namespace table

```
spark.sql("""
        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 [redacted].test.europython""")
```

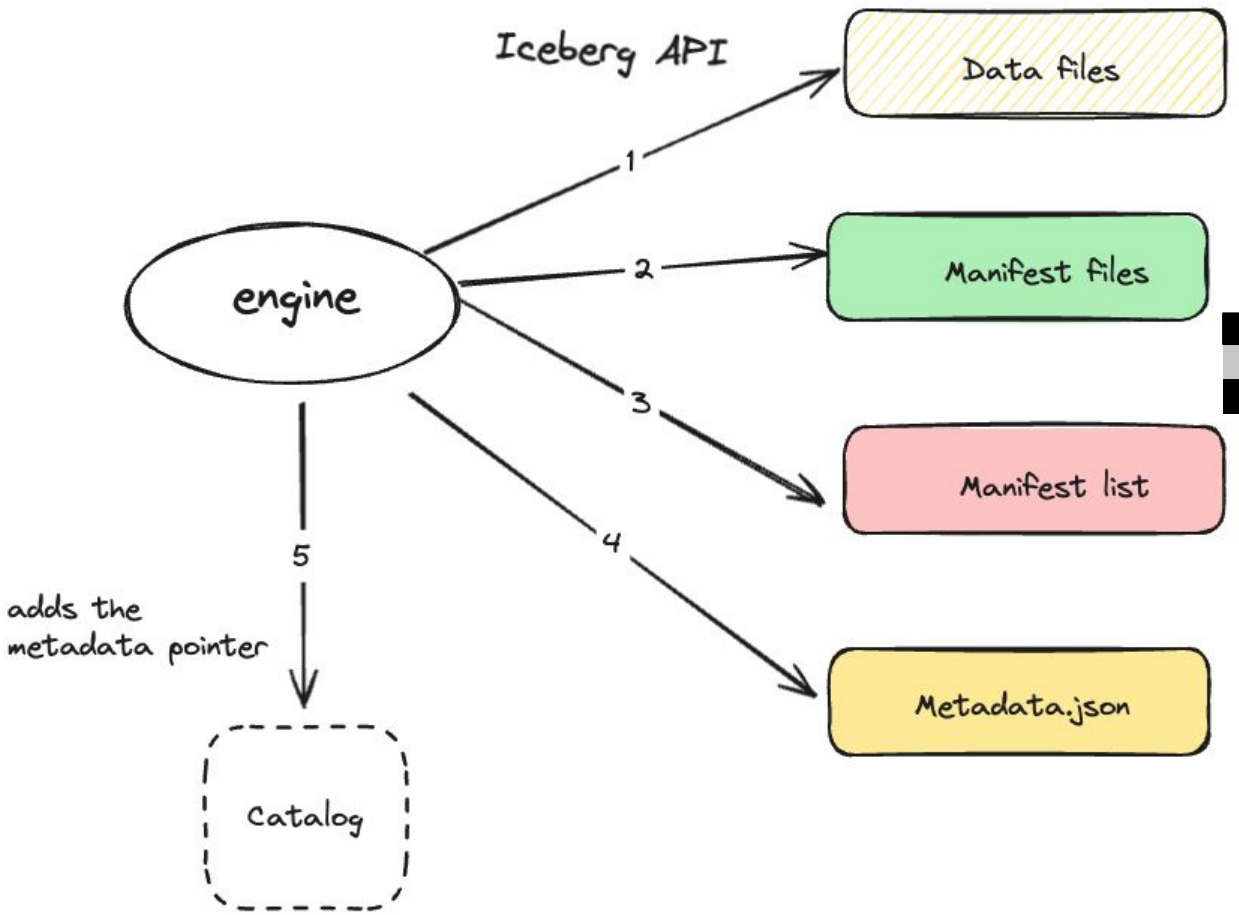
Last executed at 2024-06-27 16:58:01 in 4.53s

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)	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
```



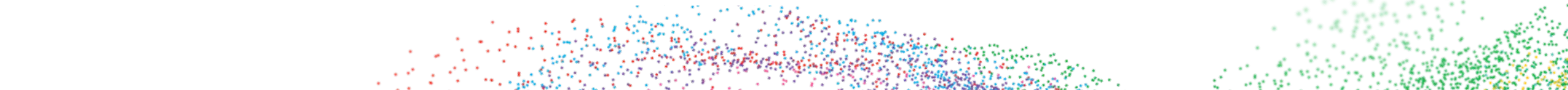

```
                    -warehouse/test/europython/data ------------
   1.362Ki [##########] /SESSION_DATE_year=2024
```

```
          :lacus-warehouse/test/europython/data/SESSION_DATE_year=2024 ----------------
1.362Ki [##########]  00000-6-12d78028-5ec4-4080-8f1d-adace111c2a8-00001.parquet
```

```
                -warehouse/test/europython/metadata ----------------------------------------
6.973Ki [#########]  8261a1ce-3376-4126-a200-4dbd0e028690-m0.avro
4.170Ki [######    ]  snap-1534533569610702955-1-8261a1ce-3376-4126-a200-4dbd0e028690.avro
```

00000-a1249670-2ddf-4734-8620-f3d6d6977c78.metadata.json

snapshot_id - 1534533569610702955



```
|-- 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)	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":"SES
SION_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":f
alse},{"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

PyIceberg API

TechAtBloomberg.com

Bloomberg

Engineering

```
>>> pip install “pyiceberg[sql-sqlite]”
```

```python
from pyiceberg.catalog.sql import SqlCatalog

warehouse_path = "/tmp/warehouse"
catalog = SqlCatalog(
    "default",
    **{
        "uri": f"sqlite:///{warehouse_path}/pyiceberg_catalog.db",
        "warehouse": f"file://{warehouse_path}",
    },
)
catalog.create_namespace("test")
```

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

```python
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]]
```

```python
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
```

```python
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-ee45bb2f-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
  - [x] Support partitioned appends: Support Appends with TimeTransform Partitions #784
  - [ ] Support partial deletes: Support partial deletes #569
- [x] Support for parallelizing writes: Parallel Table.append #428 Faster ingestion from Parquet #346
  - [x] Support parallelized writes: Bin-pack Writes Operation into multiple parquet files, and parallelize writing `WriteTask`s #444
- [x] 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: Add metadata tables #511
  - [ ] *Files* assigned to **@Gowthami03B**, PR in Add Files metadata table #614
  - [x] *Snapshots* assigned to **@Fokko** in Add Snapshots table metadata #524
  - [x] *History* assigned to **@ndrluis** in Add history inspect table #828
  - [x] *Metadata log entries* **@kevinjqliu** (issue in [feat request] Add `metadata_log_entries` metadata table #594): Metadata Log Entries metadata table #667
  - [x] *Manifests* **@geruh**: PR in Add manifests metadata table #717
  - [x] *Partitions* assigned to **@syun64** (issue in Support get partition table with filter #24): Add Partitions Metadata Table #603
  - [x] *References* assigned to **@geruh** in Add Refs metadata table #602
  - [x] *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

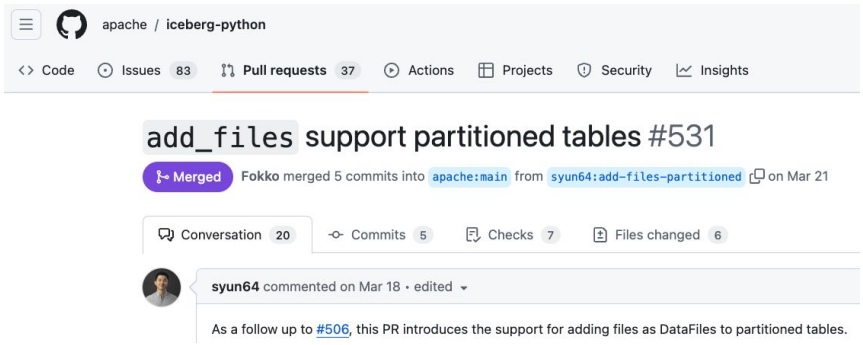
Table of contents

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



Partitioned Append on Identity Transform #555

Merged · **Fokko** merged 20 commits into `apache:main` from `jqin61:identity-partitioned-append` on Apr 5

Conversation 42 · Commits 20 · Checks 7 · Files changed 11

jqin61 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 · Contributor

No description provided.

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 · Contributor

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 · Contributor

Closes #234

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

TechAtBloomberg.com

Bloomberg

Engineering

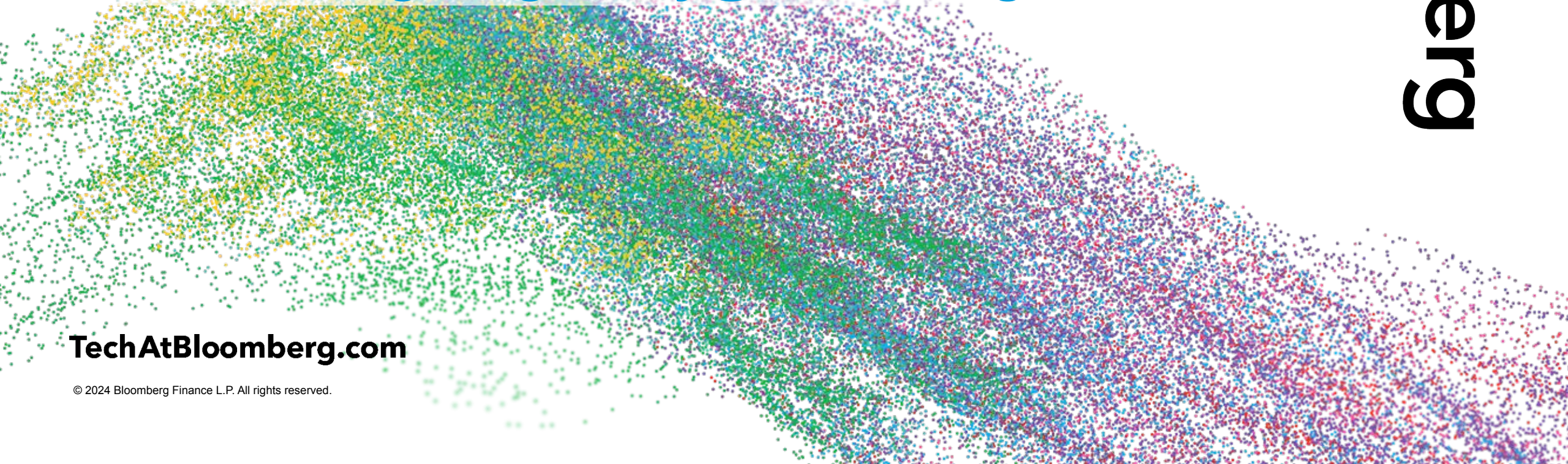
Thank you!

We are hiring: bloomberg.com/engineering

Contact me: gbhogireddy@bloomberg.net 📧

Bloomberg

Engineering

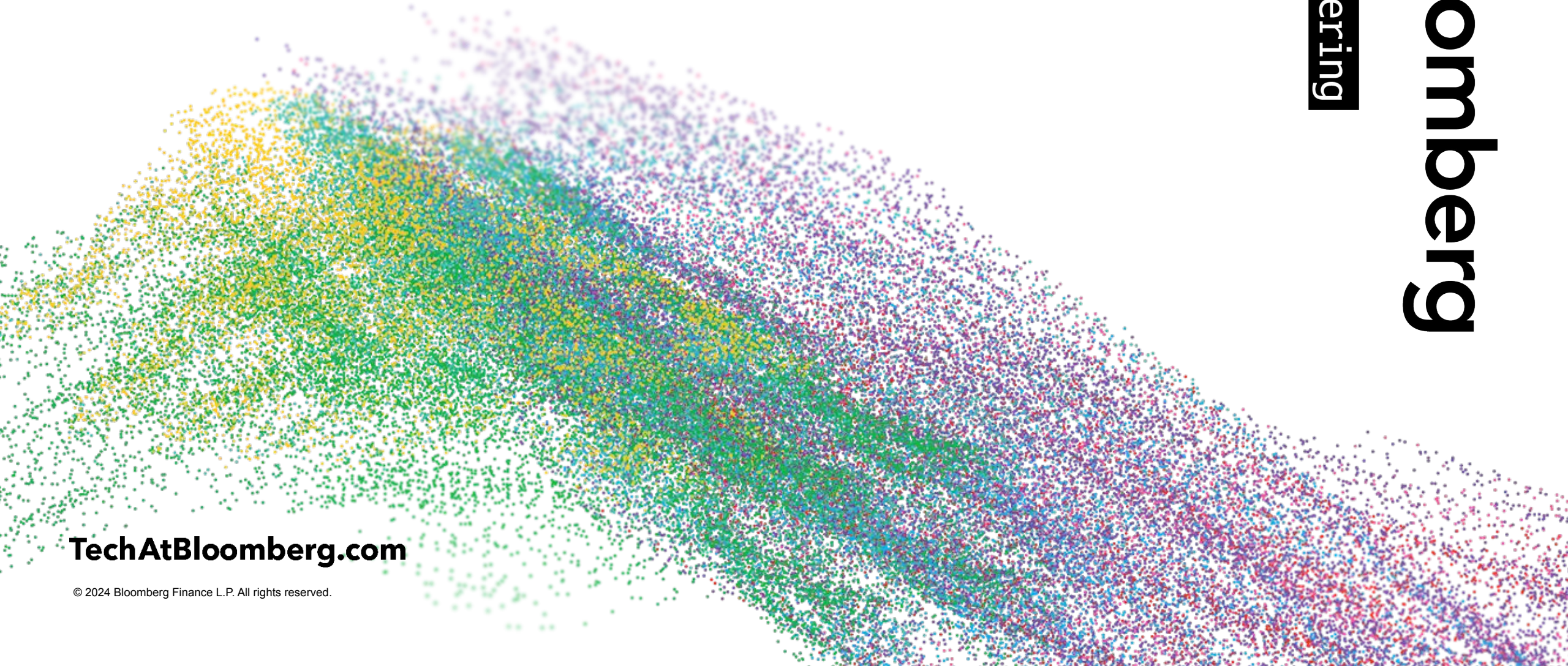


TechAtBloomberg.com

Q&A

Bloomberg

Engineering



TechAtBloomberg.com

Logo Attribution

Apache®, Apache Spark™, Apache Iceberg™, Apache Parquet™, Apache Kafka™ are either registered trademarks or trademarks of the Apache Software Foundation in the United States and/or other countries.

TechAtBloomberg.com