Fast results using Iceberg and Trino

Ryan Blue
Trino Summit 2021

Fast results using Iceberg and Trino

Ryan Blue
Trino Summit 2021

Same query, very different runtimes

- **Spark**: > 1 hour
- **Trino**: 5 minutes

- Same data: 1m tasks
- Filtering and aggregation

Why!?

Not a benchmark!

Spark problems:

- Coarse locking never fixed
- Tasks accidentally include lots of state
- Confusion between Spark scheduler, executor manager, and YARN
- Resources sharing via app scale up/down

Trino optimizations

- Planning and tasks run in parallel
- Light-weight tasks
- Workers ready to scale up processing for a query
- Vectorized reads go directly to Trino memory

Tabular

Trino works great, however your data is stored

Tabular

Tables are inefficient

- Query planning may take 10+ minutes
  - Aggregating the set of data files
  - Listing partition directories

- Splits may not contain any rows for the query
  - 2-3 orders of magnitude more tasks than needed
  - More sensitive to S3 hanging

Tabular

What if we fixed tables?

Tabular

That’s why Iceberg exists*!

* This isn’t quite true

Tabular

Iceberg exists to fix productivity

Tabular

Iceberg

- Add reliable transactions
- Unlock performance
- Fix usability

Tabular

Iceberg

- Add reliable transactions
- Unlock performance
- Fix usability

Tabular

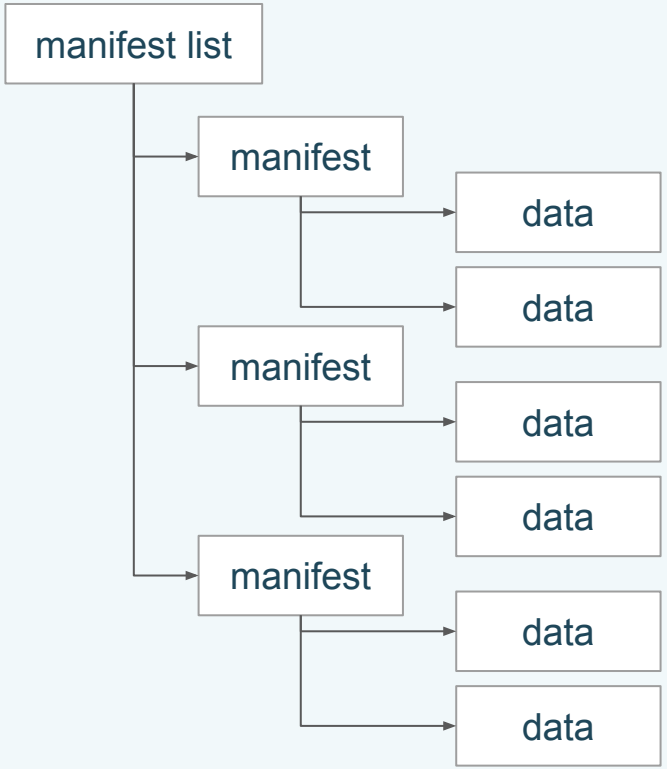
Iceberg optimizations

- Built for object stores
  - No list operations (slow)
  - No rename operations (unreliable!)
- Metadata tree
  - Faster planning, fewer tasks
  - Low write amplification
- Delta files

Tabular

Iceberg metadata

- Manifest file pruning by partition range
- Data file pruning by partition
- Data file pruning by column stats
- Data skipping with columnar structures



Tabular

Iceberg metadata tree

id = 74 and **ts > 2021-10-01** and **ts < 2021-10-04**

snapshot-1-manifest-list

manifest_path	partition_summaries		
	partition	**lower_bound**	**upper_bound**
manifest-1.avro	ts_day	2021-10-01	2021-10-15
	id_bucket	0	63
manifest-2.avro	ts_day	2021-10-15	2021-10-31
	id_bucket	0	63
…	…	…	…

manifest-1.avro

file_path	partition		lower_bounds	upper_bounds
01-data.parquet	2021-10-01	0	{ ts => …, id => … }	{ ts => …, id => … }
02-data.parquet	2021-10-01	1	{ ts => …, id => … }	{ ts => …, id => … }
…	…	…	…	…
64-data.parquet	2021-10-02	0	{ ts => …, id => … }	{ ts => …, id => … }
65-data.parquet	2021-10-02	1	{ ts => …, id => … }	{ ts => …, id => … }
…	…	…	…	…

manifest-2.avro

file_path	partition		lower	upper_bound
932-data.parquet	2021-10-15	36	{ ts => …, id => … }	{ ts => …, id => … }
02-data.parquet	2021-10-15	37	{ ts => …, id => … }	{ ts => …, id => … }
…	…	…	…	…

Tabular

Iceberg metadata tree

id = 74 and **ts > 2021-10-01** and **ts < 2021-10-04**

snapshot-1-manifest-list

manifest_path	partition_summaries		
	partition	**lower_bound**	**upper_bound**
manifest-1.avro	ts_day	2021-10-01	2021-10-15
	id_bucket	0	63
manifest-2.avro	ts_day	2021-10-15	2021-10-31
	id_bucket	0	63
…	…	…	…

manifest-1.avro

file_path	partition		lower_bounds	upper_bounds
01-data.parquet	2021-10-01	0	{ ts => …, id => … }	{ ts => …, id => … }
02-data.parquet	2021-10-01	1	{ ts => …, id => … }	{ ts => …, id => … }
…	…	…	…	…
64-data.parquet	2021-10-02	0	{ ts => …, id => … }	{ ts => …, id => … }
65-data.parquet	2021-10-02	1	{ ts => …, id => … }	{ ts => …, id => … }
…	…	…	…	…

manifest-2.avro

file_path	partition		lower	upper_bound
932-data.parquet	2021-10-15	36	{ ts => …, id => … }	{ ts => …, id => … }
02-data.parquet	2021-10-15	37	{ ts => …, id => … }	{ ts => …, id => … }
…	…	…	…	…

Tabular

Iceberg metadata tree

id = 74 and **ts > 2021-10-01** and **ts < 2021-10-04**

snapshot-1-manifest-list

manifest_path	partition_summaries		
	partition	**lower_bound**	**upper_bound**
manifest-1.avro	ts_day	2021-10-01	2021-10-15
	id_bucket	0	63
manifest-2.avro	ts_day	2021-10-15	2021-10-31
	id_bucket	0	63
…	…	…	…

manifest-1.avro

file_path	partition		lower_bounds	upper_bounds
01-data.parquet	2021-10-01	0	{ ts => …, id => … }	{ ts => …, id => … }
02-data.parquet	2021-10-01	1	{ ts => …, id => … }	{ ts => …, id => … }
…	…	…	…	…
64-data.parquet	2021-10-02	0	{ ts => …, id => … }	{ ts => …, id => … }
65-data.parquet	2021-10-02	1	{ ts => …, id => … }	{ ts => …, id => … }
…	…	…	…	…

manifest-2.avro

file_path	partition		lower	upper_bound
932-data.parquet	2021-10-15	36	{ ts => …, id => … }	{ ts => …, id => … }
02-data.parquet	2021-10-15	37	{ ts => …, id => … }	{ ts => …, id => … }
…	…	…	…	…

Tabular

Iceberg metadata tree

id = 74 and **ts > 2021-10-01** and **ts < 2021-10-04**

manifest-1.avro

X

file_path	partition		lower_bounds		upper_bounds	
			id	ts	id	ts
01-data.parquet	2021-10-01	0	299	…	959,446	…
02-data.parquet	2021-10-01	1	186	…	960,724	…
…	…	…	…	…	…	…
64-data.parquet	2021-10-02	0	357	…	962,984	…
65-data.parquet	2021-10-02	1	65	…	959,875	…
…	…	…	…	…	…	…

Tabular

Some things to note…

- Ordering data within each file makes scans even faster
  - Enables row group and page skipping

- If writes don’t align with reads, metadata may be mixed
  - Use `rewrite_manifests` to reorganize for faster planning:
    ```
    CALL system.rewrite_manifests('db.table')
    ```
  - Or use the table API:
    ```
    table.rewriteManifests()
    ```

Tabular

Iceberg & Trino results

- Metrics data: Query over years rather than days
  - Job *planning* time was 12 minutes
  - Total *running* time was reduced to 45 seconds

- Log data: Results in < 5 seconds from a multi-petabyte dataset
  - More than 2 million partitions
  - Metadata rewrites were needed for efficient planning

Tabular

Future work

- Fix Iceberg/Trino stats estimation
  - This prevents splits from running during planning
- Avoid unnecessary S3 calls
  - FileSystem requires more calls than FileIO
- Use table configuration
  - Supports automatic table-specific tuning
- Merge row-level deletes while reading

Tabular

Thanks for listening!
Questions?

Tabular

Big point

- Point
  - Sub
    - Sub-sub

Tabular

Section break

Tabular

Basic slide

- Point
  - Sub
    - Sub-sub

Tabular

Title only