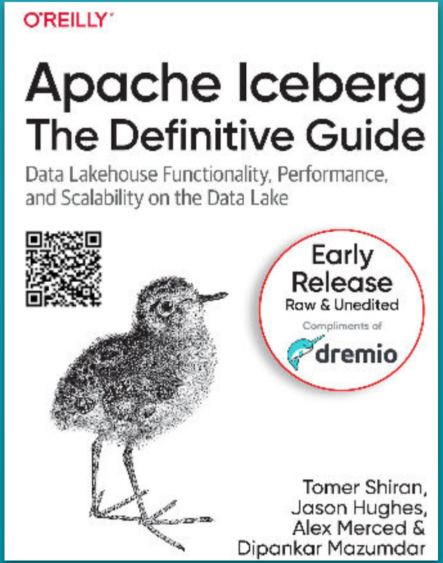
Optimizing Data & Files in Apache Iceberg

Performance strategies

About Me

- **Current:** Developer Advocate @Dremio
- **Open source:** Apache Iceberg, Arrow, Project Nessie
- **Past:** BI, Machine Learning, Data architecture
- **Upcoming Iceberg Book**



O'REILLY

Apache Iceberg
The Definitive Guide

Data Lakehouse Functionality, Performance, and Scalability on the Data Lake

Early Release
Raw & Unedited
Compliments of
dremio

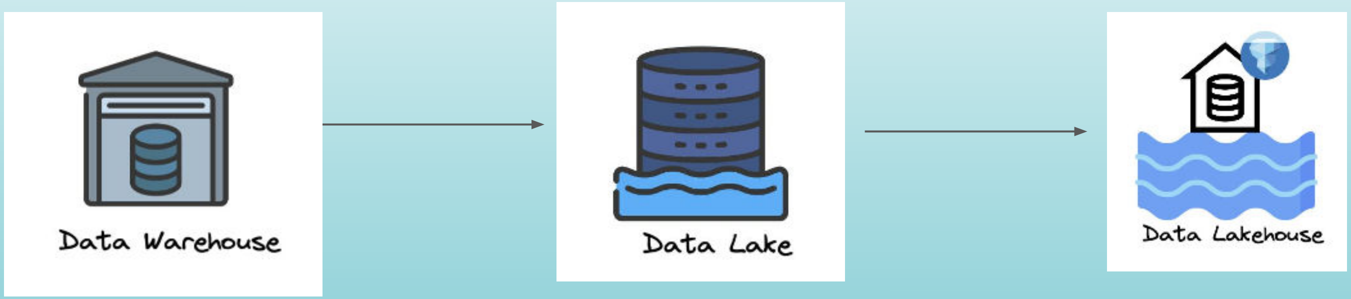
Tomer Shiran,
Jason Hughes,
Alex Merced &
Dipankar Mazumdar

dremio

Agenda

1. Intro to the Apache Iceberg Table format
2. Problem: Lot of files & Small files
3. Solution: File skipping → Partitioning, Compaction, Metrics filtering
4. Overlapping Metrics Problem
5. Solution: Reorganizing files → Sorting, Z-ordering

Evolution of Data architecture



- Centralized, reliable data platform
- Democratize data
- Data warehouse → Data Lakes → Lakehouse

NETFLIX: Motivation

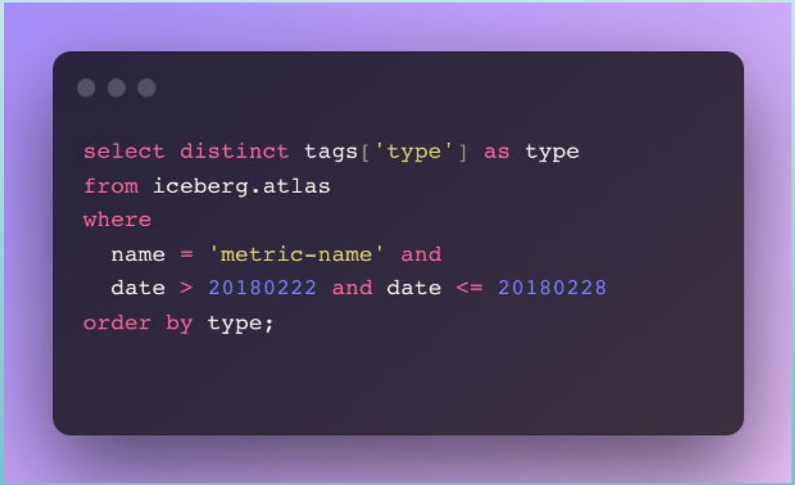
ATLAS: Time series metrics from Netflix's runtime system

1 month: 2.7 million files in 2,688 partitions

Problem: cannot process more than a few days of data

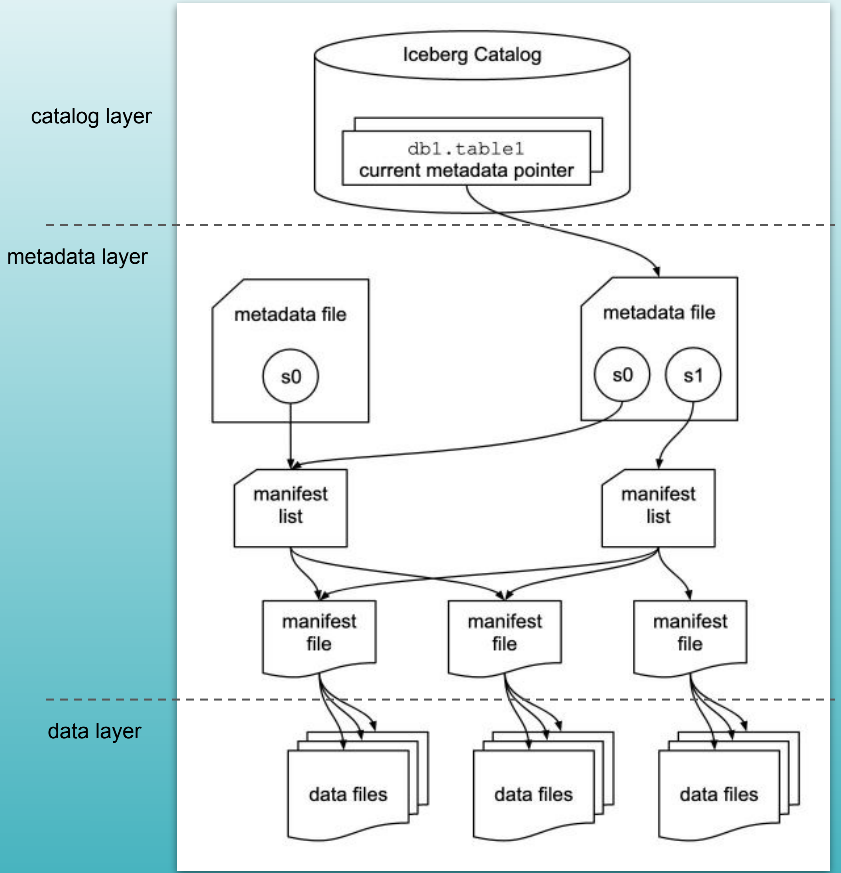
Hive table – with Parquet filters:
EXPLAIN query: 9.6 min (planning wall time)

Iceberg table – partition and min/max filtering:
42 sec (wall time) / 22 min (task time) / 25 sec (planning)



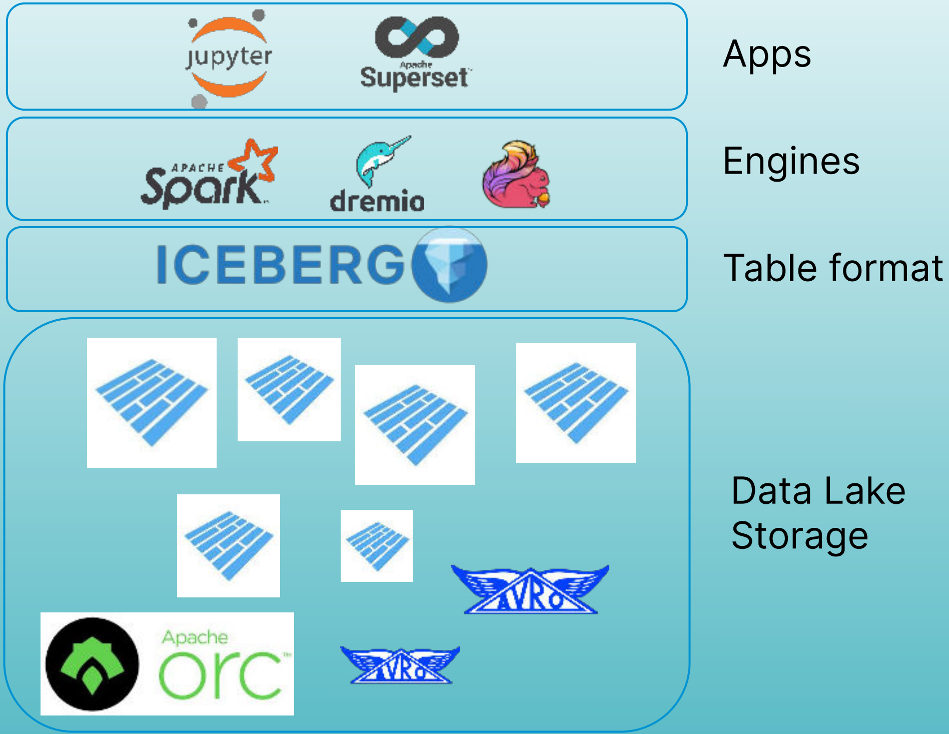
```
select distinct tags['type'] as type
from iceberg.atlas
where
  name = 'metric-name' and
  date > 20180222 and date <= 20180228
order by type;
```

Iceberg Table format

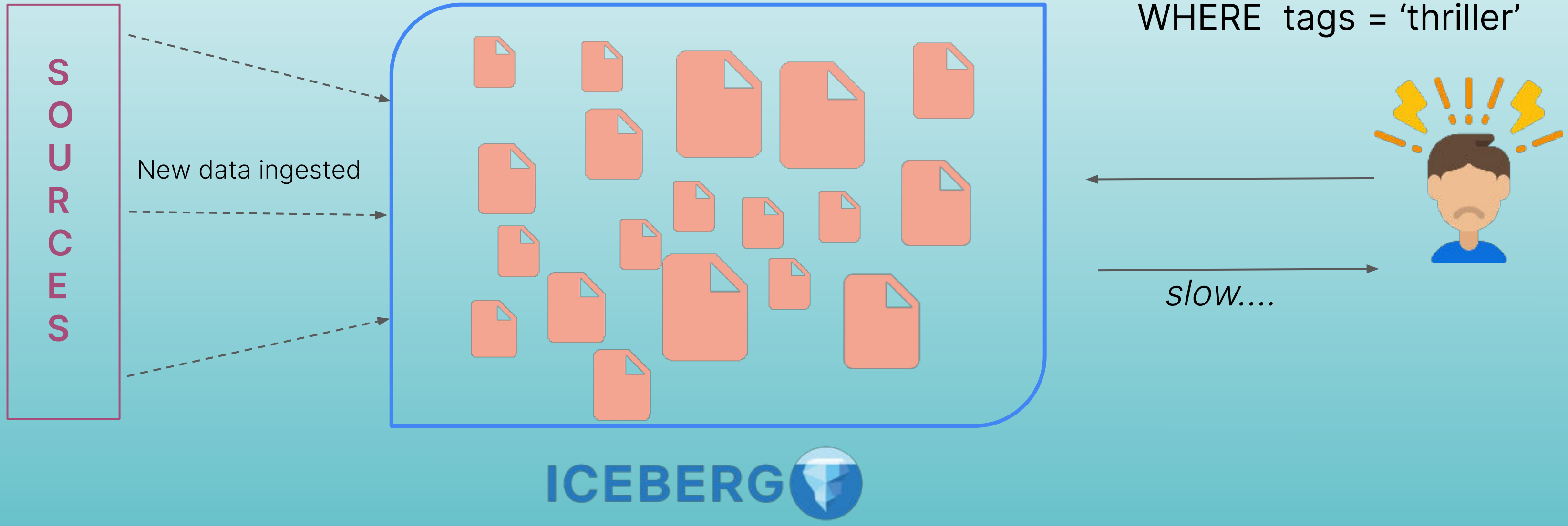


- an ‘open’ table format for analytical datasets in data lake
- Capabilities such as Expressive SQL, ACID compliant queries, schema evolution, time travel
- Multiple compute engines on same dataset at the same time

Iceberg in a Lakehouse



Problem in the Lake



```
SELECT movies
FROM iceberg.table
WHERE  tags = 'thriller'
```

Problem in the Lake

Querying 100s of Petabytes of data demands optimized query speed!

Your queries are fast today, but maybe not over time..

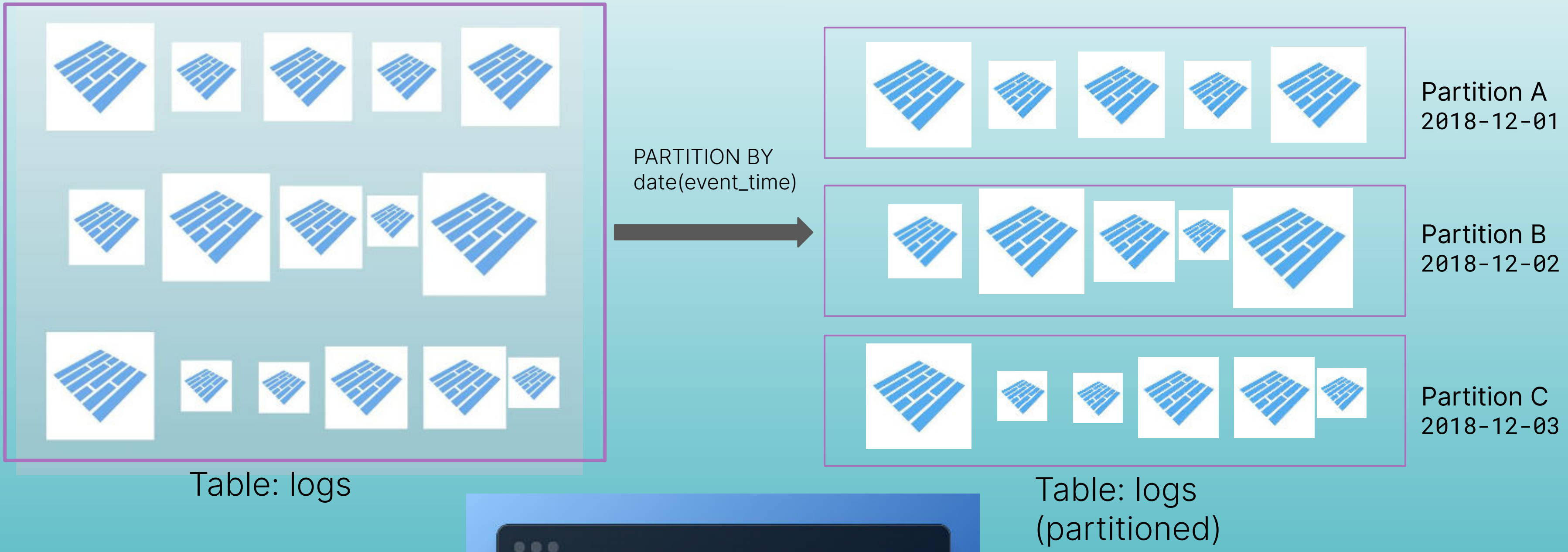
ISSUE: Unorganized & Small Files

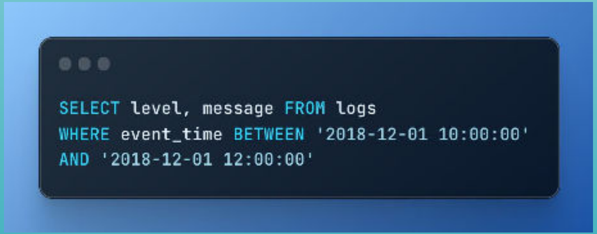
Solution

Optimizations

- Compaction
- Partitioning
- Min/Max Filtering
- Sorting
- Z-order clustering

Partitioning ICEBERG





```
SELECT level, message FROM logs
WHERE event_time BETWEEN '2018-12-01 10:00:00'
AND '2018-12-01 12:00:00'
```

Partitioning ICEBERG

Hidden Partitioning

- Iceberg handles the tedious and error-prone task of producing partition values for rows in a table.
- Hive → explicit partition columns
- Users need to know the physical layout & specify predicates in query
- Iceberg tracks these transformations without the need for extra columns

```
INSERT INTO logs PARTITION (event_date)

SELECT level, message, event_time,
format_time(event_time, 'YYYY-MM-dd')

FROM unstructured_log_source
```

Compaction ICEBERG

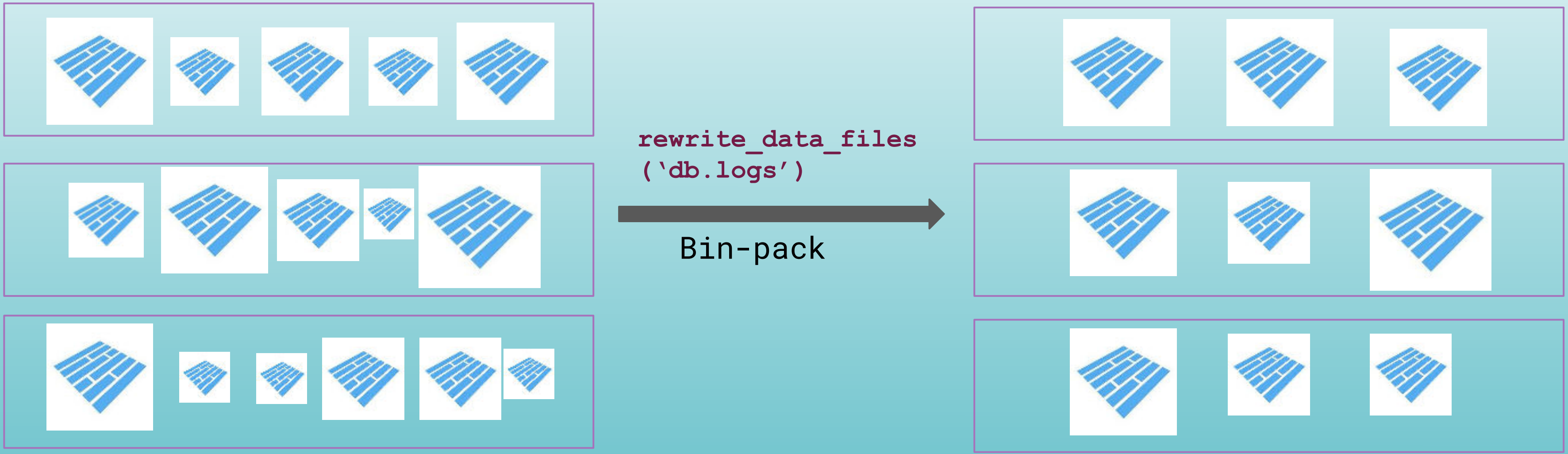
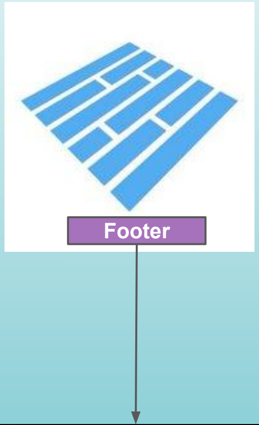


Table: logs
(small files)

Table: logs
(compacted)

Metrics Filtering ICEBERG

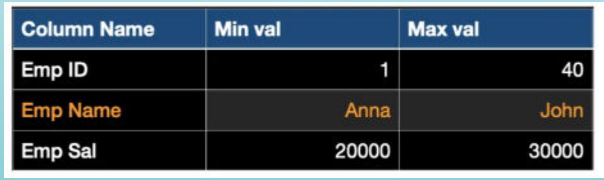


Column Name	Min val	Max val
Emp ID	1	40
Emp Name	Anna	Dmitry
Emp Sal	20000	30000

Metrics metadata = Iceberg Manifests

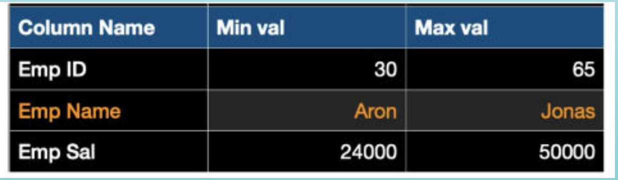
Overlapping Metrics

File 1



Column Name	Min val	Max val
Emp ID	1	40
Emp Name	Anna	John
Emp Sal	20000	30000

File 2



Column Name	Min val	Max val
Emp ID	30	65
Emp Name	Aron	Jonas
Emp Sal	24000	50000

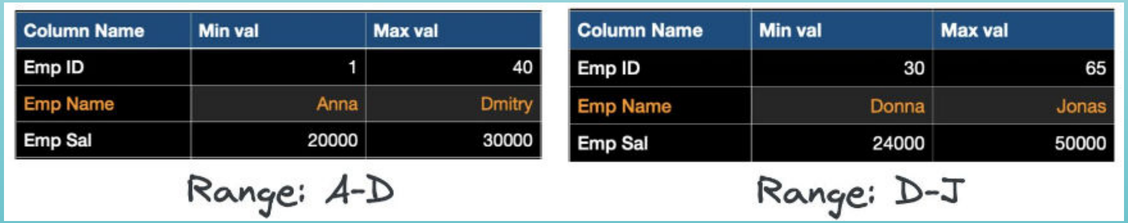
```
SELECT * FROM Employee
WHERE Emp Name = ‘Dennis’
```

Sorting

```
CALL rewrite_data_files(table ⇒
'db.emp', strategy ⇒ 'sort', sort_order ⇒
'emp_name ASC)
```

File 1

File 2



Column Name	Min val	Max val
Emp ID	1	40
Emp Name	Anna	Dmitry
Emp Sal	20000	30000

Range: A-D

Column Name	Min val	Max val
Emp ID	30	65
Emp Name	Donna	Jonas
Emp Sal	24000	50000

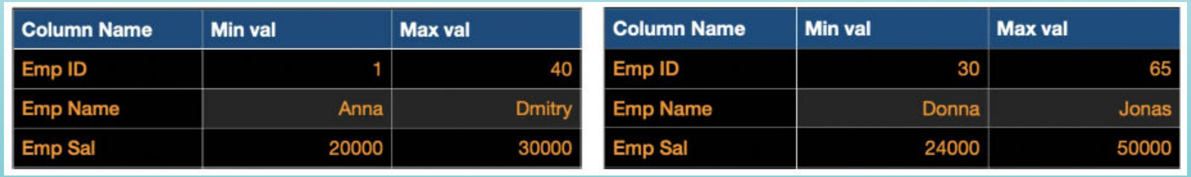
Range: D-J

Hierarchical Sorting

File 1

File 2

Sort order: (Emp Name, Emp Sal, Emp ID)



Column Name	Min val	Max val
Emp ID	1	40
Emp Name	Anna	Dmitry
Emp Sal	20000	30000

Column Name	Min val	Max val
Emp ID	30	65
Emp Name	Donna	Jonas
Emp Sal	24000	50000

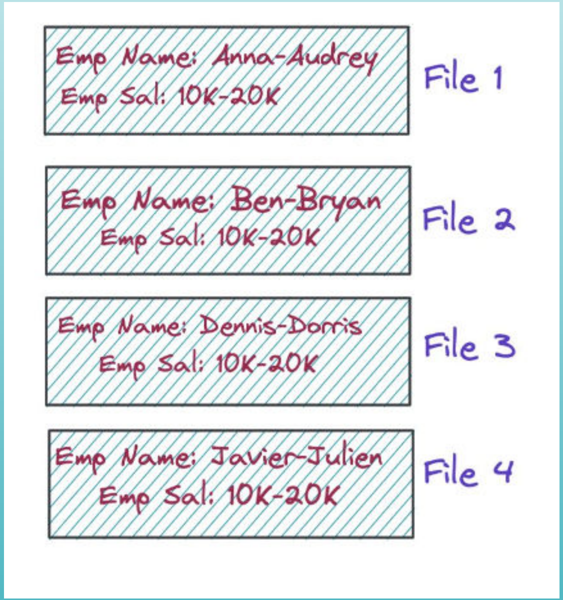
```
SELECT * FROM Employee WHERE
Emp Name='Dennis' AND Emp
Sal=25000 AND Emp ID=22
```

Hierarchical Sorting Problems

```
SELECT * FROM Employee WHERE
Emp Sal>18000
```

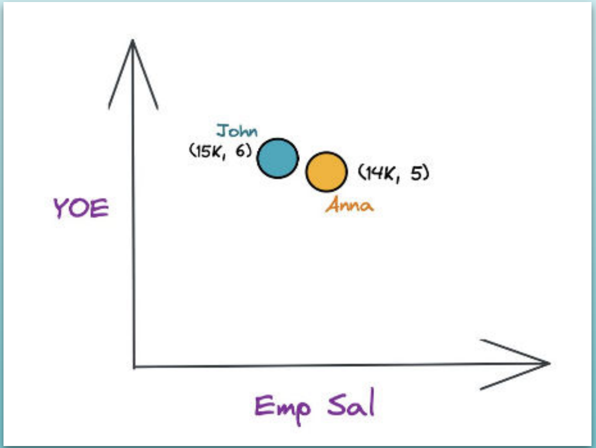
Column Name	Min val	Max val
Emp Name	Anna	Julien
Emp Sal	10000	20000

Sort order: (Emp Name, Emp Sal)



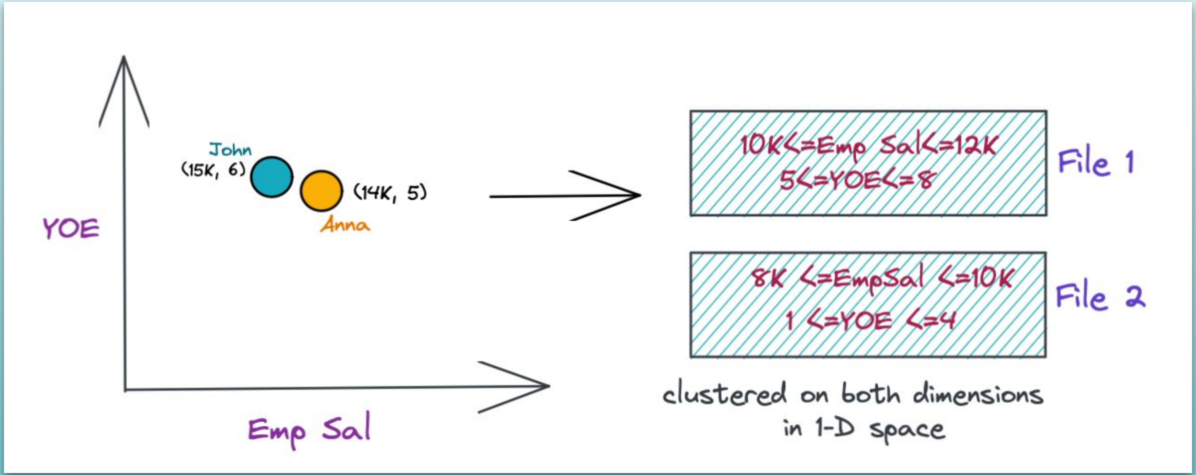
Colocality ICEBERG

Emp Name	Emp Sal	Emp ID	Years of exp(YOE)
Anna	14000	10	5
Dennis	8000	16	2
John	15000	13	6
Bryan	20000	14	3



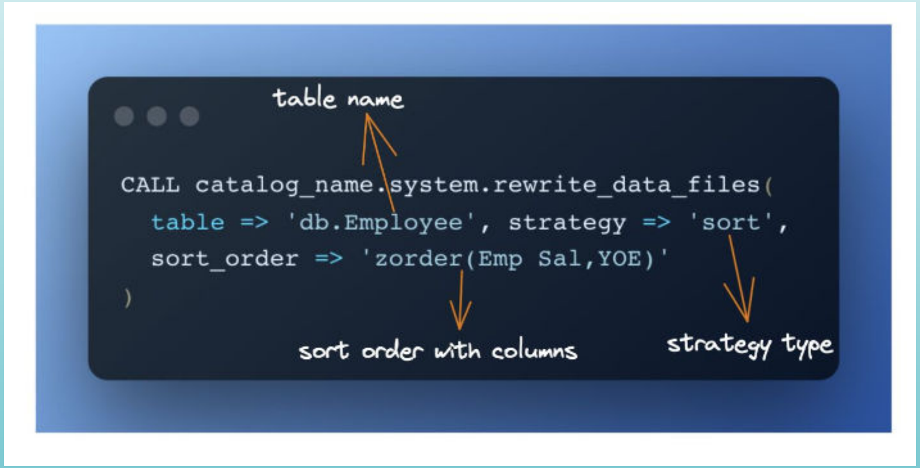
"Give me all the employees who are 5+ years experienced and have salaries over 10000"

Z-ordering ICEBERG



"Give me all the employees who are 5+ years experienced and have salaries over 10000"

Z-ordering ICEBERG



```
CALL catalog_name.system.rewrite_data_files(
  table => 'db.Employee', strategy => 'sort',
  sort_order => 'zorder(Emp Sal,YOE)'
)
```

SELECT * FROM Employee WHERE Emp Sal>10000 AND YOE>5

When to Z-order Vs When not?

- Frequently run queries when you want to filter data using multiple dimensions
- Z-order works best with data that have similar distribution and range
- Z-ordering on fields with a very small distribution range isn’t beneficial
- Columns with high cardinality are best suited for Z-ordering

Optimization methods

Compaction

Hidden Partitioning

Min/Max Filtering

Sorting

Z-ordering

Q&A