CLUSTERING

- By the end of this section you will understand how clustering works under the hood.
- Micro partitions and how it gets created in snowflake.
- You will understand how query processing happens in snowflake.
- Do and don’ts while using clustering.
- Drawbacks of clustering.

Micro partition in snowflake

type	name	country	date
2	A	UK	11/2
4	C	SP	11/2
3	C	DE	11/2
2	B	DE	11/2
3	A	FR	11/2
2	C	SP	11/2
3	Z	DE	11/2
2	B	UK	11/2
4	C	NL	11/2
5	X	FR	11/3
1	A	NL	11/3
5	A	FR	11/3
2	X	FR	11/2
4	Z	NL	11/2
2	Y	SP	11/2
1	B	SP	11/3
5	X	DE	11/3
3	A	UK	11/4
1	C	FR	11/3
4	Z	NL	11/4
5	Y	SP	11/4
5	B	SP	11/5
3	X	DE	11/5
2	Z	UK	11/5

Table file1:

TYPE	NAME	COUNTRY	DATE
2	A	UK	11/2
4	C	SP	11/2
3	C	DE	11/2
2	B	DE	11/2
3	A	FR	11/2
2	C	SP	11/2

Table file2:

TYPE	NAME	COUNTRY	DATE
3	Z	DE	11/2
2	B	UK	11/2
4	C	NL	11/2
5	X	FR	11/3
1	A	NL	11/3
5	A	FR	11/3

Table file3:

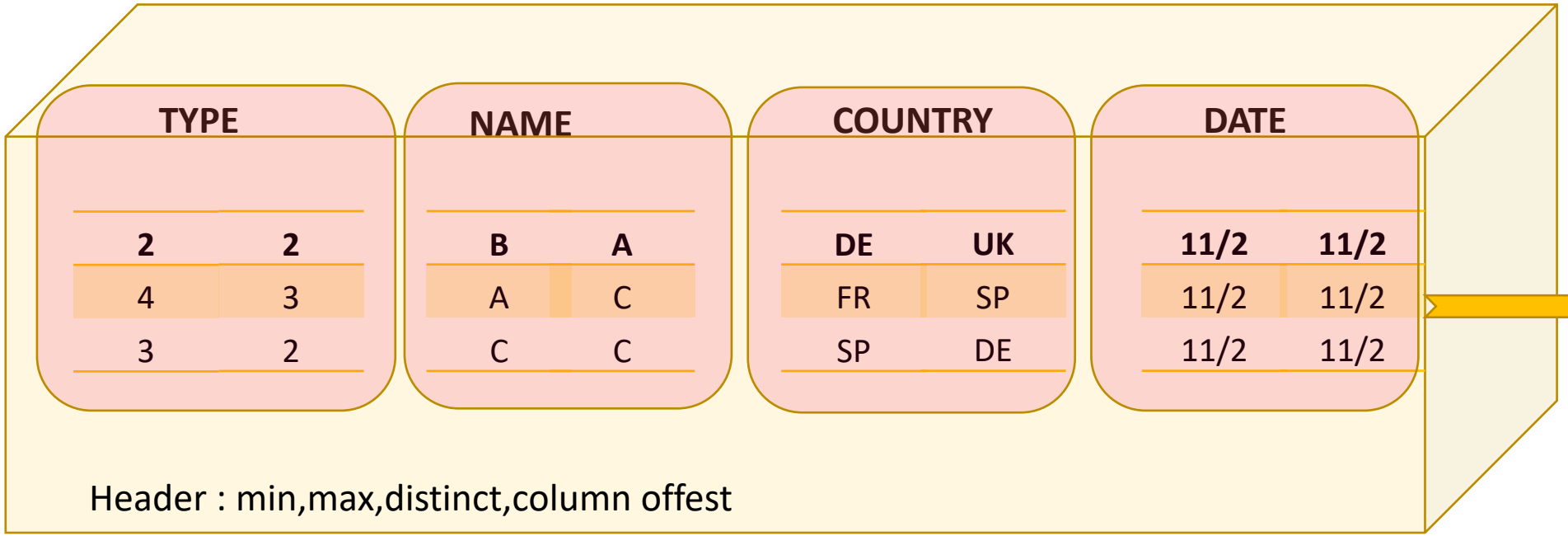
TYPE	NAME	COUNTRY	DATE
2	X	FR	11/2
4	Z	NL	11/2
2	Y	SP	11/2
1	B	SP	11/3
5	X	DE	11/3
3	A	UK	11/4

Table file4:

TYPE	NAME	COUNTRY	DATE
1	C	FR	11/3
4	Z	NL	11/4
5	Y	SP	11/4
5	B	DE	11/5
3	X	DE	11/5
2	Z	UK	11/5

MICRO PARTITIONS

Table File



Heavily compressed

Micro-partitioning is automatically **performed** on all Snowflake tables. Tables are transparently partitioned using the **ordering of the data as it is inserted/loaded**.

```
Select
Name,
Country
From employee
Where date='11/2'
```

This query will get submitted to cloud services layer.

Cloud service layer will do query optimization , creates execution plan and submits this plan to all virtual warehouse nodes.

Nodes will first download table file HEADER from all table files.

Based on the metadata information in the HEADER file, table files(Miro partitions) will be scanned.

Header1: column offset, min,max…

TYPE	NAME	COUNTRY	DATE
2	A	UK	11/2
4	C	SP	11/2
3	C	DE	11/2
2	B	DE	11/2
3	A	FR	11/2
2	C	SP	11/2

Micro partition 1

Header2: column offset, min,max…

TYPE	NAME	COUNTRY	DATE
3	Z	DE	11/2
2	B	UK	11/2
4	C	NL	11/2
5	X	FR	11/3
1	A	NL	11/3
5	A	FR	11/3

Micro partition 2

Header3: column offset, min,max…

TYPE	NAME	COUNTRY	DATE
2	X	FR	11/2
4	Z	NL	11/2
2	Y	SP	11/2
1	B	SP	11/3
5	X	DE	11/3
3	A	UK	11/4

Micro partition 3

Header4: column offset, min,max…

TYPE	NAME	COUNTRY	DATE
1	C	FR	11/3
4	Z	NL	11/4
5	Y	SP	11/4
5	B	DE	11/5
3	X	DE	11/5
2	Z	UK	11/5

Micro partition 4

```
Select
Name,
Country
From employee
Where date='11/2'
```

Virtual warehouse layer

Select

Name,

Country

From employee

Where date=’11/2’

TYPE	NAME	COUNTRY	DATE
2	A	UK	11/2
4	C	SP	11/2
3	C	DE	11/2
2	B	DE	11/2
3	A	FR	11/2
2	C	SP	11/2

TYPE	NAME	COUNTRY	DATE
3	Z	DE	11/2
2	B	UK	11/2
4	C	NL	11/2
5	X	FR	11/3
1	A	NL	11/3
5	A	FR	11/3

TYPE	NAME	COUNTRY	DATE
2	X	FR	11/2
4	Z	NL	11/2
2	Y	SP	11/2
1	B	SP	11/3
5	X	DE	11/3
3	A	UK	11/4

Selected

TYPE	NAME	COUNTRY	DATE
1	C	FR	11/3
4	Z	NL	11/4
5	Y	SP	11/4
5	B	DE	11/5
3	X	DE	11/5
2	Z	UK	11/5

Pruned

Select

Name,

Country

From employee

Where date=’11/2’

TYPE	NAME	COUNTRY	DATE
2	A	UK	11/2
4	C	SP	11/2
3	C	DE	11/2
2	B	DE	11/2
3	A	FR	11/2
2	C	SP	11/2

TYPE	NAME	COUNTRY	DATE
3	Z	DE	11/2
2	B	UK	11/2
4	C	NL	11/2
5	X	FR	11/3
1	A	NL	11/3
5	A	FR	11/3

TYPE	NAME	COUNTRY	DATE
2	X	FR	11/2
4	Z	NL	11/2
2	Y	SP	11/2
1	B	SP	11/3
5	X	DE	11/3
3	A	UK	11/4

Virtual warehouse will request AWS s3(data storage layer in our case)
to download only this part of the files or range of file.

Select

Name,

Country

From employee

Where date='11/2'

Micro-partitioning is automatically **performed** on all Snowflake tables. Tables are transparently partitioned using the **ordering of the data as it is inserted/loaded**.

TYPE	NAME	COUNTRY	DATE
2	A	UK	11/2
4	C	SP	11/2
3	C	DE	11/2
2	B	DE	11/2
3	A	FR	11/2
2	C	SP	11/2

TYPE	NAME	COUNTRY	DATE
3	Z	DE	11/2
2	B	UK	11/2
4	C	NL	11/2
5	X	FR	11/3
1	A	NL	11/3
5	A	FR	11/3

TYPE	NAME	COUNTRY	DATE
2	X	FR	11/2
4	Z	NL	11/2
2	Y	SP	11/2
1	B	SP	11/3
5	X	DE	11/3
3	A	UK	11/4

Virtual warehouse will request AWS s3(data storage layer in our case) to download only this part of the files or range of file.

Select

Name,

Country

From employee

Where date='11/2'

Micro-partitioning is automatically **performed** on all Snowflake tables. Tables are transparently partitioned using the **ordering of the data as it is inserted/loaded**.

50- 500 MB

TYPE	NAME	COUNTRY	DATE
2	A	UK	11/2
4	C	SP	11/2
3	C	DE	11/2
2	B	DE	11/2
3	A	FR	11/2
2	C	SP	11/2
3	Z	DE	11/2
2	B	UK	11/2
4	C	NL	11/2
2	X	FR	11/2
4	Z	NL	11/2
2	Y	SP	11/2

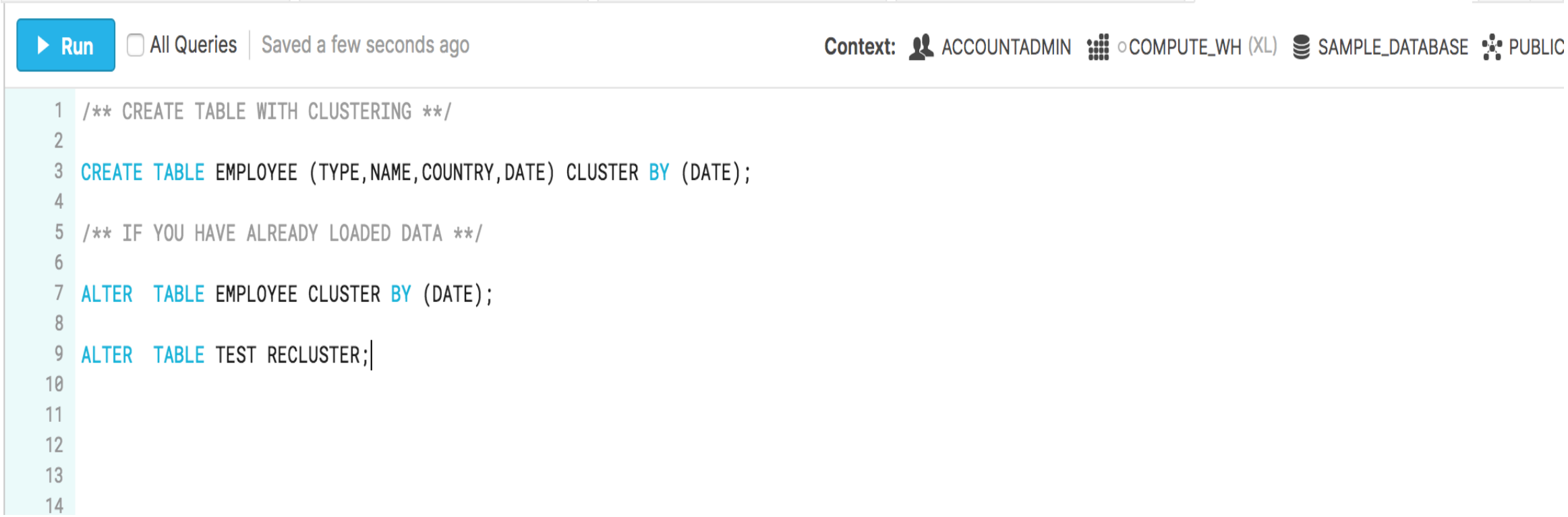
TYPE	NAME	COUNTRY	DATE
3	Z	DE	11/2
2	B	UK	11/2
4	C	NL	11/2
2	X	FR	11/2
4	Z	NL	11/2
2	Y	SP	11/2

TYPE	NAME	COUNTRY	DATE
5	X	FR	11/3
1	A	NL	11/3
5	A	FR	11/3
1	B	SP	11/3
5	X	DE	11/3
3	A	UK	11/4

prune

Virtual warehouse will request AWS s3(data storage layer in our case) to download only this part of the files or range of file.

CLUSTERING



MICRO PARTITION DEPTH

Select

Name,

Country

From employee

Where date='11/2'

Micro-partitioning is automatically **performed** on all Snowflake tables. Tables are transparently partitioned using the **ordering of the data as it is inserted/loaded**.

TYPE	NAME	COUNTRY	DATE
2	A	UK	11/2
4	C	SP	11/2
3	C	DE	11/2
2	B	DE	11/2
3	A	FR	11/2
2	C	SP	11/2

TYPE	NAME	COUNTRY	DATE
3	Z	DE	11/2
2	B	UK	11/2
4	C	NL	11/2
5	X	FR	11/3
1	A	NL	11/3
5	A	FR	11/3

TYPE	NAME	COUNTRY	DATE
2	X	FR	11/2
4	Z	NL	11/2
2	Y	SP	11/2
1	B	SP	11/3
5	X	DE	11/3
3	A	UK	11/4

Virtual warehouse will request AWS s3(data storage layer in our case) to download only this part of the files or range of file.

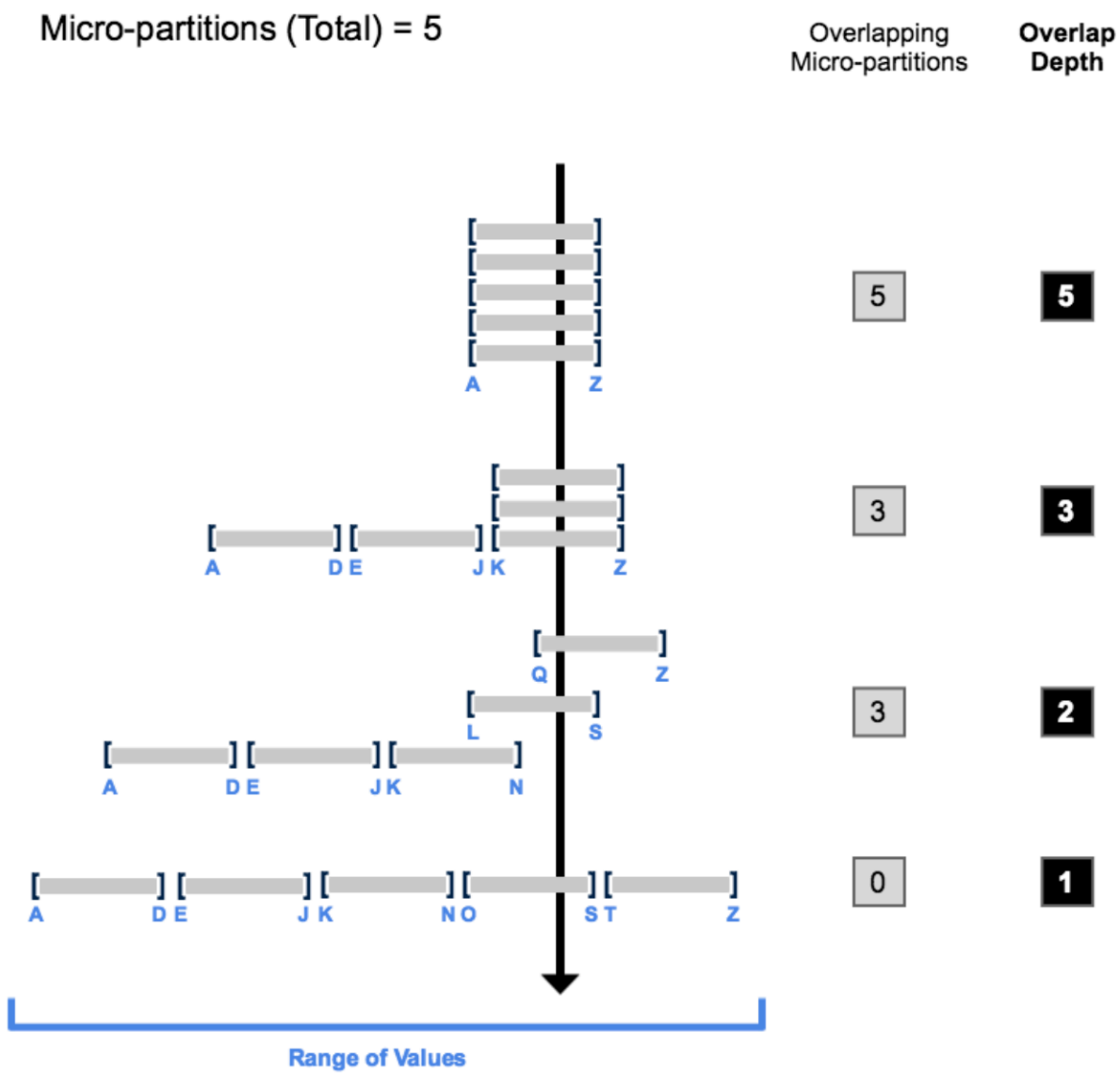


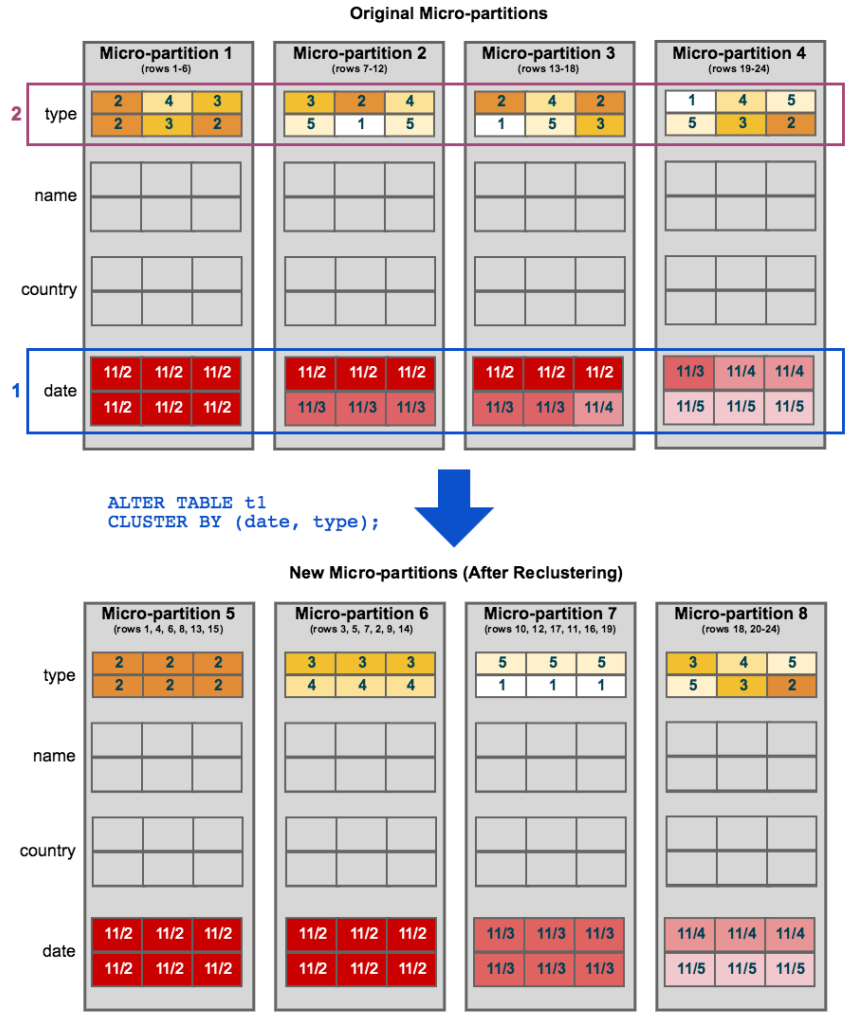
Table: `t1`

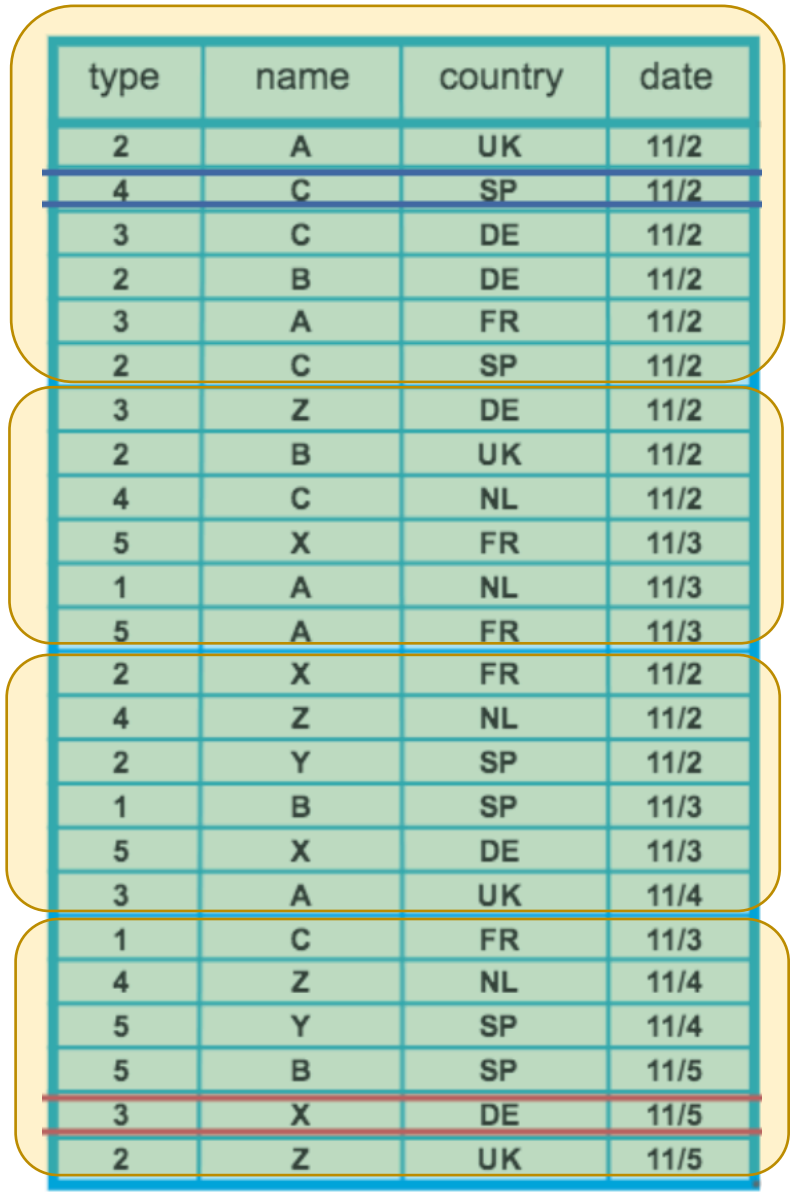
Logical Structure

type	name	country	date
2	A	UK	11/2
4	C	SP	11/2
3	C	DE	11/2
2	B	DE	11/2
3	A	FR	11/2
2	C	SP	11/2
3	Z	DE	11/2
2	B	UK	11/2
4	C	NL	11/2
5	X	FR	11/3
1	A	NL	11/3
5	A	FR	11/3
2	X	FR	11/2
4	Z	NL	11/2
2	Y	SP	11/2
1	B	SP	11/3
5	X	DE	11/3
3	A	UK	11/4
1	C	FR	11/3
4	Z	NL	11/4
5	Y	SP	11/4
5	B	SP	11/5
3	X	DE	11/5
2	Z	UK	11/5

```
SELECT name, country FROM t1
WHERE type = 2
AND date = '11/2';
```

Physical Structure





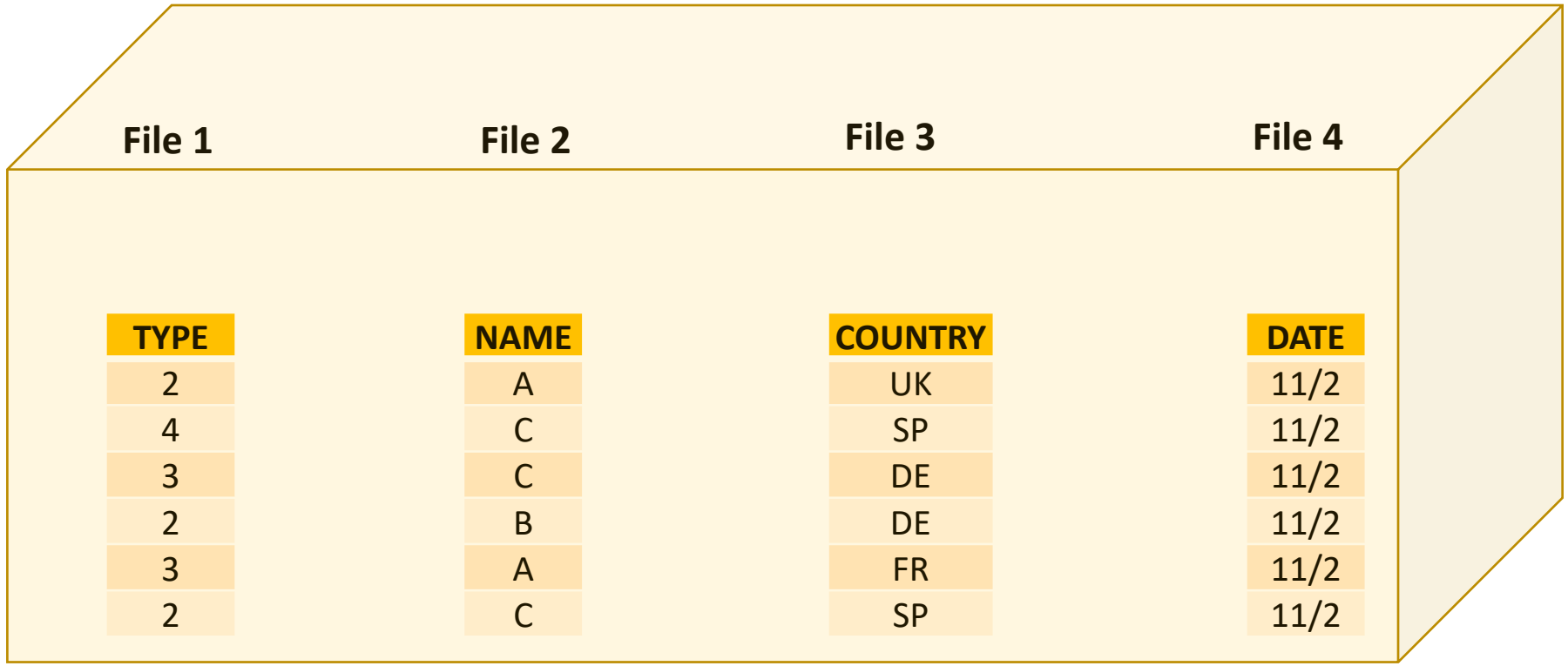
type	name	country	date
2	A	UK	11/2
4	C	SP	11/2
3	C	DE	11/2
2	B	DE	11/2
3	A	FR	11/2
2	C	SP	11/2
3	Z	DE	11/2
2	B	UK	11/2
4	C	NL	11/2
5	X	FR	11/3
1	A	NL	11/3
5	A	FR	11/3
2	X	FR	11/2
4	Z	NL	11/2
2	Y	SP	11/2
1	B	SP	11/3
5	X	DE	11/3
3	A	UK	11/4
1	C	FR	11/3
4	Z	NL	11/4
5	Y	SP	11/4
5	B	SP	11/5
3	X	DE	11/5
2	Z	UK	11/5

TYPE	NAME	COUNTRY	DATE
2	A	UK	11/2
4	C	SP	11/2
3	C	DE	11/2
2	B	DE	11/2
3	A	FR	11/2
2	C	SP	11/2

TYPE	NAME	COUNTRY	DATE
3	Z	DE	11/2
2	B	UK	11/2
4	C	NL	11/2
5	X	FR	11/3
1	A	NL	11/3
5	A	FR	11/3

TYPE	NAME	COUNTRY	DATE
2	X	FR	11/2
4	Z	NL	11/2
2	Y	SP	11/2
1	B	SP	11/3
5	X	DE	11/3
3	A	UK	11/4

TYPE	NAME	COUNTRY	DATE
1	C	FR	11/3
4	Z	NL	11/4
5	Y	SP	11/4
5	B	DE	11/5
3	X	DE	11/5
2	Z	UK	11/5



Each column will be stored as file.

Columns are stored independently within micro partition this is also called as columnar storage.

BAD Idea

- **Applying clustering on table will increase your query response time.**

- **Applying clustering will increase your processing cost and storage cost.**

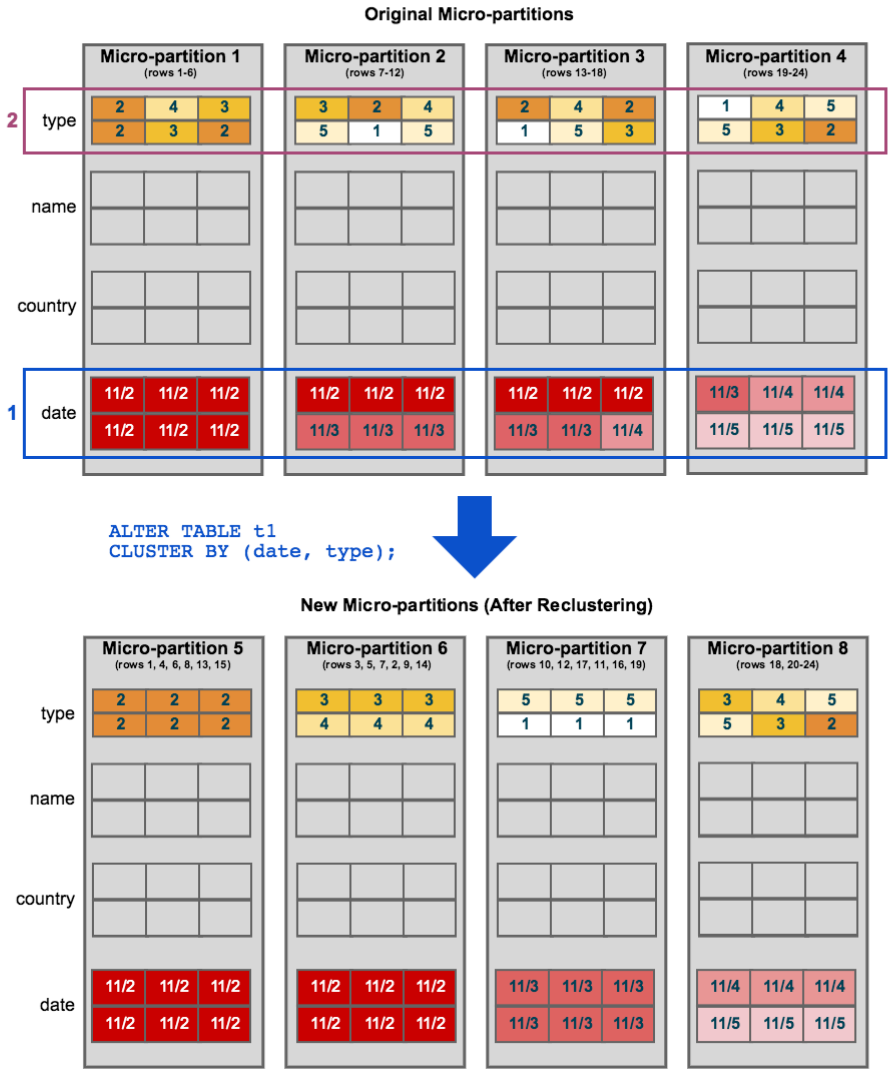
Table: `t1`

Logical Structure

type	name	country	date
2	A	UK	11/2
4	C	SP	11/2
3	C	DE	11/2
2	B	DE	11/2
3	A	FR	11/2
2	C	SP	11/2
3	Z	DE	11/2
2	B	UK	11/2
4	C	NL	11/2
5	X	FR	11/3
1	A	NL	11/3
5	A	FR	11/3
2	X	FR	11/2
4	Z	NL	11/2
2	Y	SP	11/2
1	B	SP	11/3
5	X	DE	11/3
3	A	UK	11/4
1	C	FR	11/3
4	Z	NL	11/4
5	Y	SP	11/4
5	B	SP	11/5
3	X	DE	11/5
2	Z	UK	11/5

```
SELECT name, country FROM t1
WHERE type = 2
AND date = '11/2';
```

Physical Structure



Pre-Cautions

- Clustering keys are not intended for all tables
- The size of a table, as well as the query performance for the table, should dictate whether to define a clustering key for the table.
- Table has to be large enough to consist of a sufficiently large number of micro-partitions, and the column(s) defined in the clustering key have to provide sufficient filtering to select a subset of these micro-partitions.
- In general, tables in the multi-terabyte (TB) range will experience the most benefit from clustering, particularly if DML is performed regularly/continually on these tables.

Pre-Cautions

- Clustering keys are not intended for all tables. **The size of a table**, as well as the **query performance for the table**, should dictate whether to define a clustering key for the table. In particular, to see performance improvements from a clustering key, a table has to be large enough to consist of a sufficiently **large number of micro-partitions**, and the column(s) defined in the clustering key have to provide **sufficient filtering** to select a subset of these micro-partitions.

- In general, tables in the multi-terabyte (TB) range will experience the most benefit from clustering, particularly if DML is performed regularly/continually on these tables.