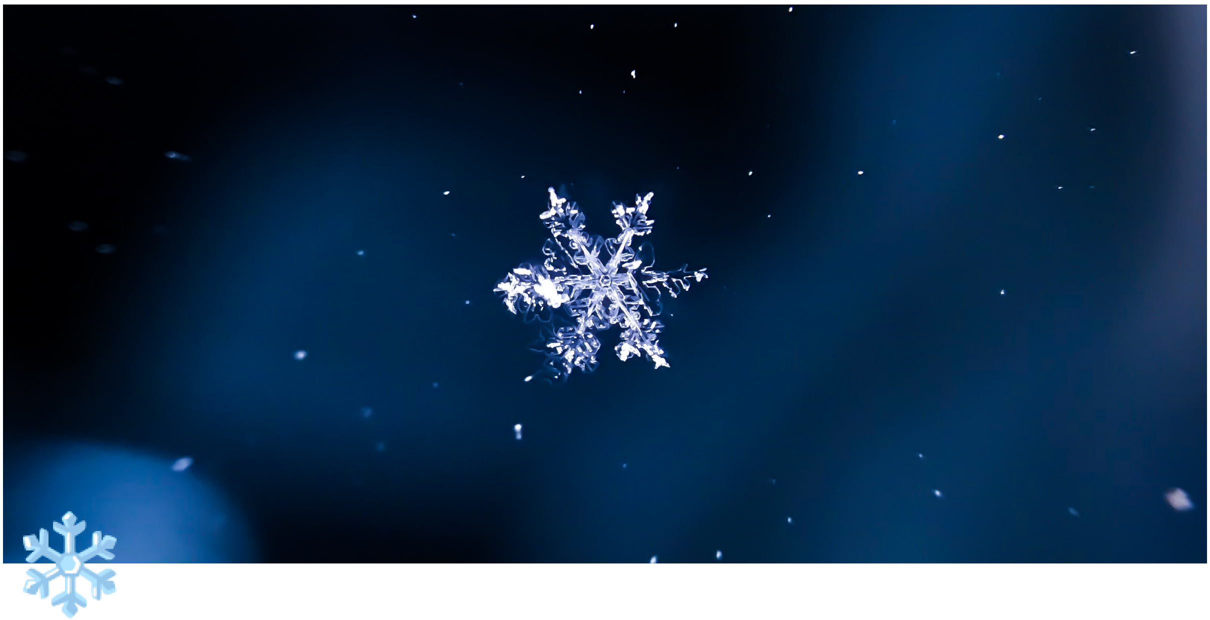
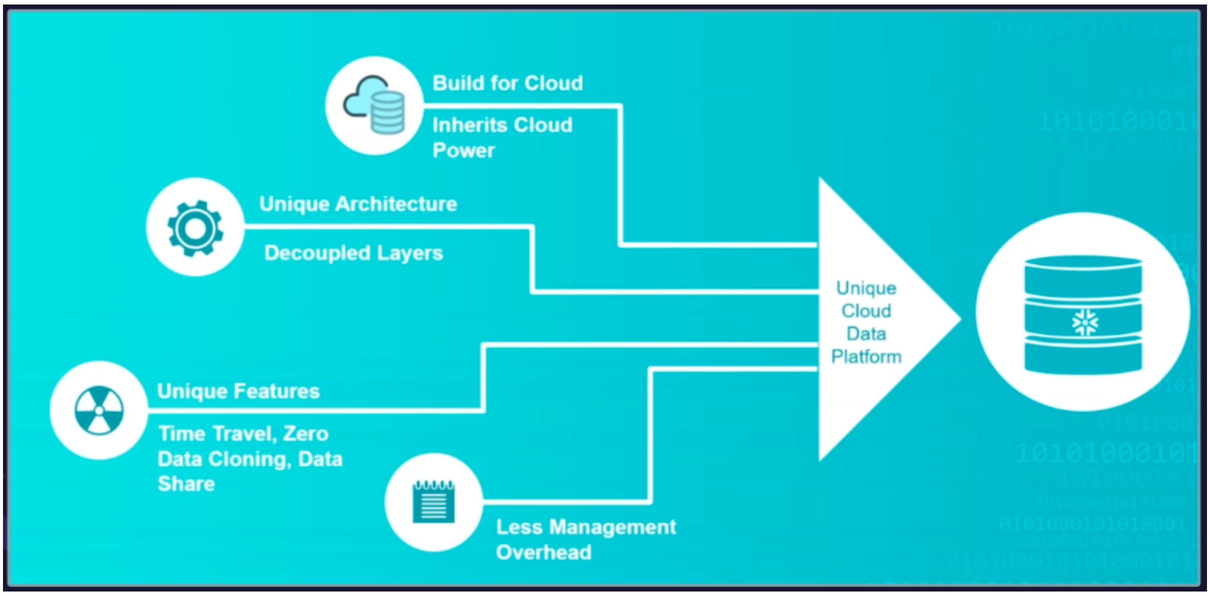


SNOWFLAKES

- Snowflake is an analytic data warehouse provided as Software-as-Services(SaaS). Snowflake provides data warehouse that is faster, easierto use and more flexible that other traditional data warehouses.
- Snowflake data warehouse is not built on existing databases or not on big data software platform as Hadoop.
- The snowflake data warehouse uses a new SQL database engine with unique architecture designed for the cloud.

Snowflakes key differentiator

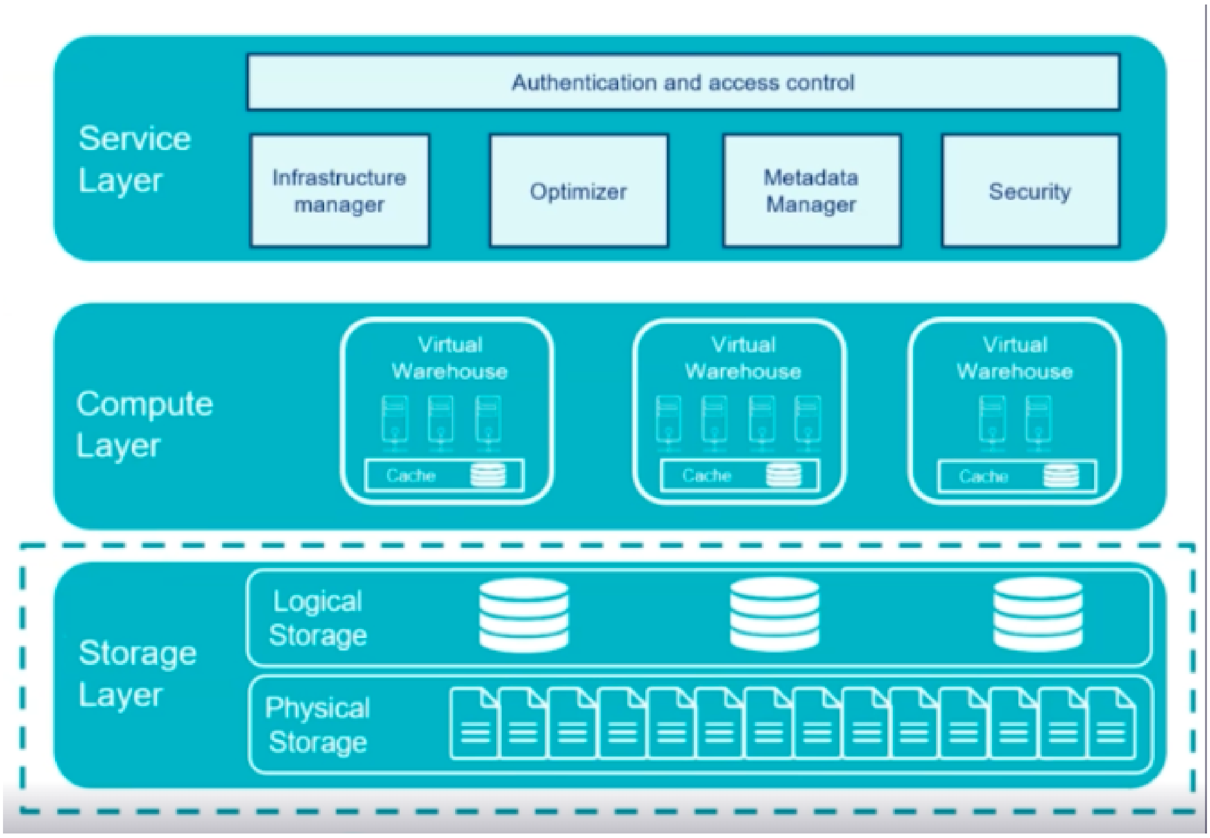


Key Concept and Architecture

Data Warehouse as Cloud Service:

- Snowflake data warehouse is true SaaS offering :
  - There is no hardware (virtual or physical) for you to select, install, configure and manage.
  - There is no software for you install, configure and manage.
  - Ongoing maintenance, management and tuning is handled by snowflake
- Snowflake completely runs on cloud infrastructure. All the component of the snowflake service runs on public cloud infrastructure
- Snowflake uses virtual compute instance for its compute need and storage service for storage of data. Snowflake can not be run on private cloud infrastructure(on primises)

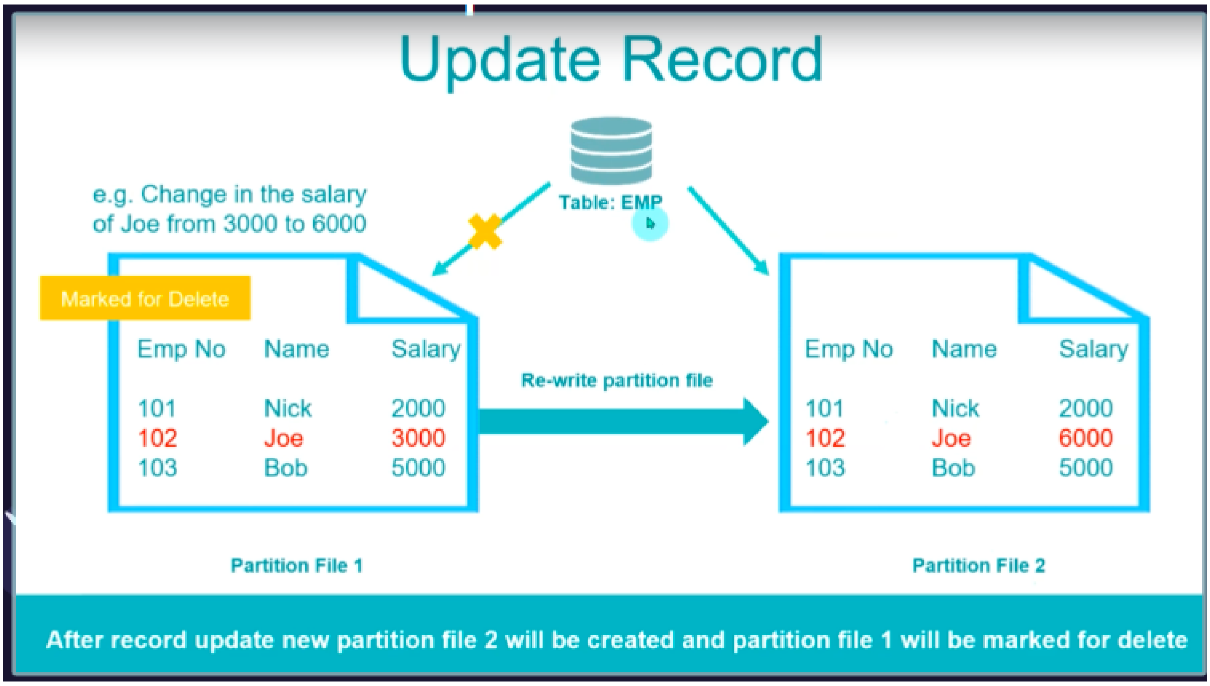
SNOWFLAKE ARCHITECTURE



Storage Layer

- This layer is independent from other 2 layer.
- Virtually unlimited storage capacity because it is connected to different cloud service provider eg. AWS, azure etc. in the backend.
- Data physically stored as micro partition files.
- Each micro partitioned file is of size 16 mb.
- Every file gets replicated 3 times to ensure high availlability.
- File format is propriety to snowflake means snowflake has not exposed in which format it save file.
- File are immutable
- Table definition are logical and stored in its metadata layer. Here logical means it is there but u can't see it eg. underground metro u know there it is , but can't see it but u can feel it.

UPDATE Record



Update Record

e.g. Change in the salary of Joe from 3000 to 6000

Table: EMP

Marked for Delete

Emp No	Name	Salary
101	Nick	2000
102	Joe	3000
103	Bob	5000

Partition File 1

Re-write partition file

Emp No	Name	Salary
101	Nick	2000
102	Joe	6000
103	Bob	5000

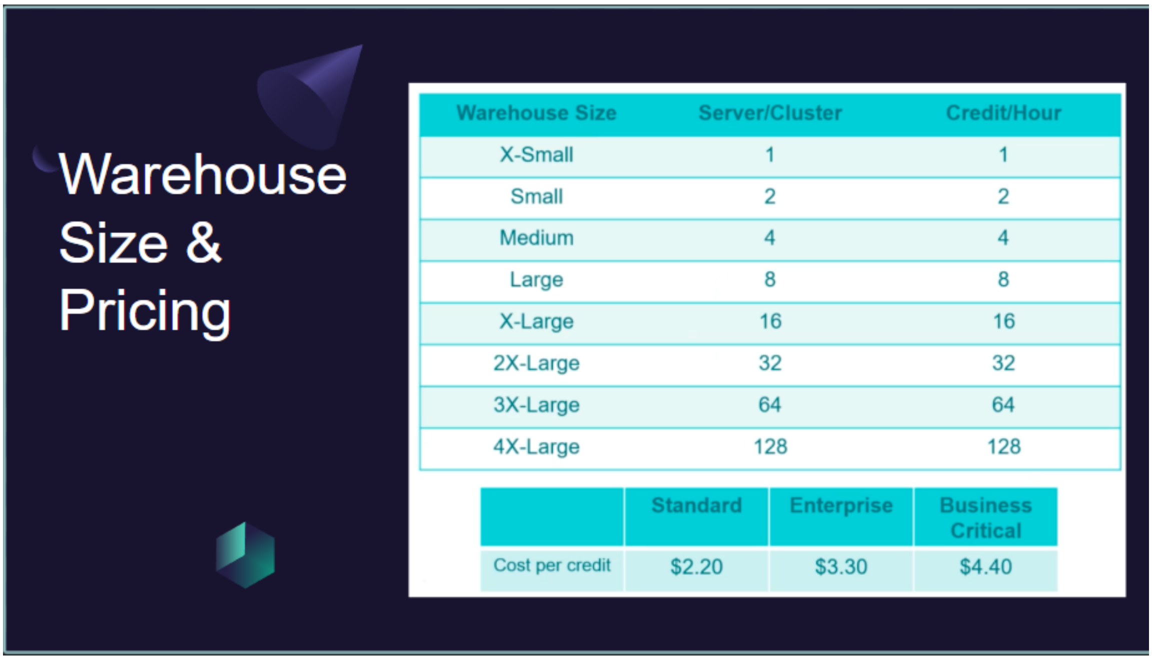
Partition File 2

After record update new partition file 2 will be created and partition file 1 will be marked for delete

The mark for deleteis done by snowflake itself and how it do it, thats not disclosed to real world.

Compute (a.k.a Warehouse) Layer

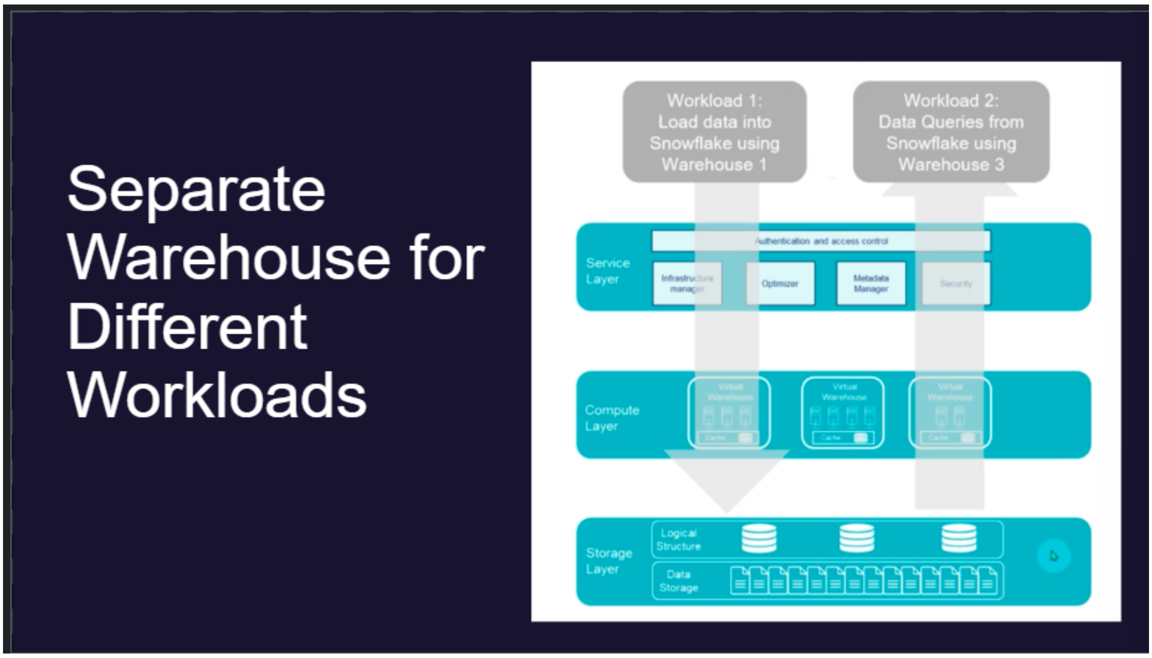
- Only layer to access data from storage layer
- It is scalable
    - Change the size of the warehouse at runtime
    - Add more nodes at the runtime
- Warehouse size vary from X-small(1 server/cluster) to 4X-Large (128 server/cluster)
- Warehouses are independent of each other.
- Every warehouse have its own small storage, which it use to cache query data for better performance
- Compute can be auto-suspended when idle.



Warehouse Size & Pricing

Warehouse Size	Server/Cluster	Credit/Hour
X-Small	1	1
Small	2	2
Medium	4	4
Large	8	8
X-Large	16	16
2X-Large	32	32
3X-Large	64	64
4X-Large	128	128

	Standard	Enterprise	Business Critical
Cost per credit	$2.20	$3.30	$4.40



Separate Warehouse for Different Workloads

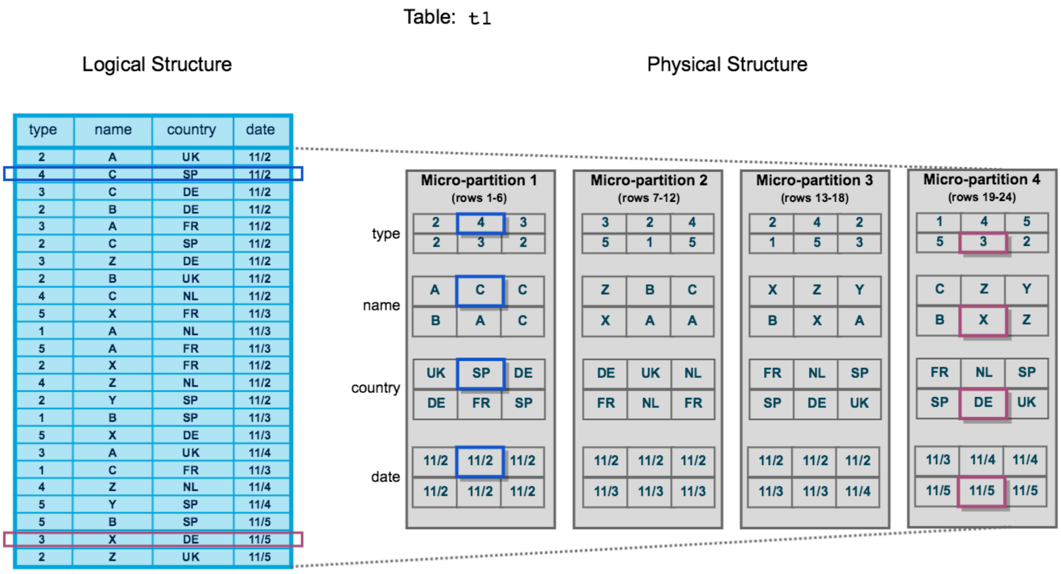
Sevice Layer

- Brain of the Snowflake.
- All the interaction with snowflake established via this layer
- Stores query output in a result cache
- Snowflake doesn’t expose its service layer and users can’t get insight and access their metadata.

- During data load service layer keeps track of which data stored in which partition file
- Maintain transaction consistency across Warehouse.

What are Micro partitions?

- All data in Snowflake tables is automatically divided into micro partitions, which are contiguous units of storage.
- Each Micro partitions contains between 50 mb to 500 mb of uncompressed data (note that the actual size in snowflake is smaller because data is always stored compressed.)
- Group of rows in tables are mapped into individual micro partitions, organized in a columnar fashion.



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

	Micro-partition 1 (rows 1-6)	Micro-partition 2 (rows 7-12)	Micro-partition 3 (rows 13-18)	Micro-partition 4 (rows 19-24)
type	2 4 3 2 3 2	3 2 4 5 1 5	2 4 2 1 5 3	1 4 5 5 3 2
name	A C C B A C	Z B C X A A	X Z Y B X A	C Z Y B X Z
country	UK SP DE DE FR SP	DE UK NL FR NL FR	FR NL SP SP DE UK	FR NL SP SP DE UK
date	11/2 11/2 11/2 11/2 11/2 11/2	11/2 11/2 11/2 11/3 11/3 11/3	11/2 11/2 11/2 11/3 11/3 11/4	11/3 11/4 11/4 11/5 11/5 11/5

- Snowflake stores metadata about all rows stored in a micro-partition, including:
    - The range of values for each of the columns in the micro-partition.
    - The number of distinct values.
    - Additional properties used for both optimization and efficient query processing.

```
Note

Micro-partitioning is automatically performed on all Snowflake tables.
```

```
Tables are transparently partitioned using the ordering of the data as
it is inserted/loaded.
```

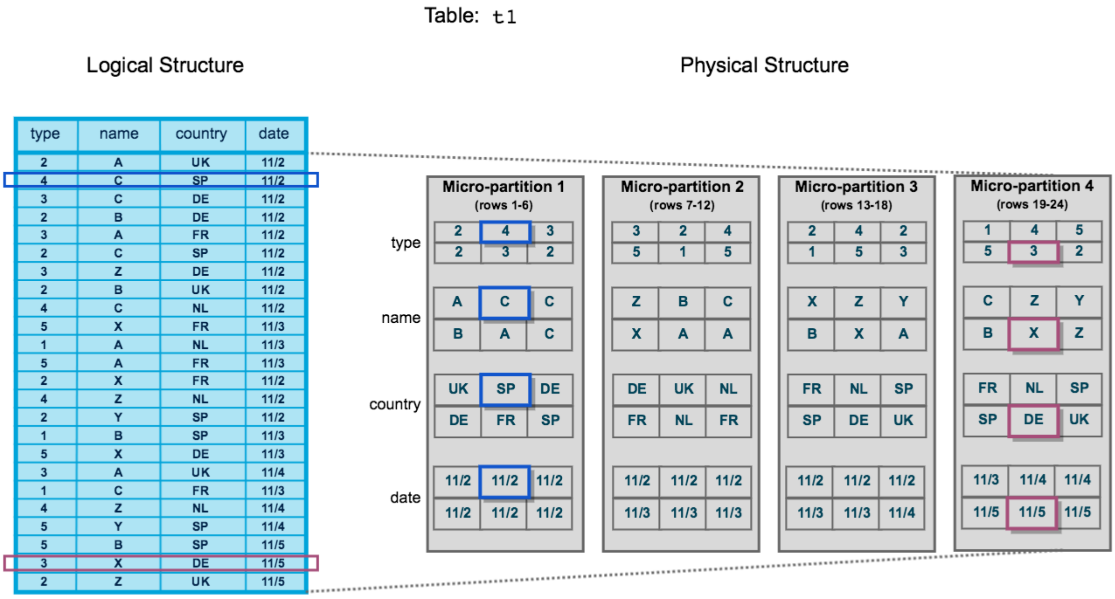
Benefits of Micro-partitioning

The benefits of Snowflake’s approach to partitioning table data include:

- In contrast to traditional static partitioning, Snowflake micro-partitions are derived automatically; they don’t need to be explicitly defined up-front or maintained by users.
- As the name suggests, micro-partitions are small in size (50 to 500 MB, before compression), which enables extremely efficient DML and fine-grained pruning for faster queries.
- Micro-partitions can overlap in their range of values, which, combined with their uniformly small size, helps prevent skew.
- Columns are stored independently within micro-partitions, often referred to as *columnar storage*. This enables efficient scanning of individual columns; only the columns referenced by a query are scanned.
- Columns are also compressed individually within micro-partitions. Snowflake automatically determines the most efficient compression algorithm for the columns in each micro-partition.

Important

In Snowflake, as data is inserted/loaded into a table, clustering metadata is collected and recorded for each micro-partition created during the process. Snowflake then leverages this clustering information to avoid unnecessary scanning of micro-partitions during querying, significantly accelerating the performance of queries that reference these columns.



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

The table consists of 24 rows stored across 4 micro-partitions, with the rows divided equally between each micro-partition. Within each micro-partition, the data is sorted and stored by column, which enables Snowflake to perform the following actions for queries on the table:

1. First, prune micro-partitions that are not needed for the query.
2. Then, prune by column within the remaining micro-partitions.

Note that this diagram is intended only as a small-scale conceptual representation of the data clustering that Snowflake utilizes in micro-partitions. A typical Snowflake table may consist of thousands, even millions, of micro-partitions.