Clustering keys considerations

“In general, if a column (or expression) has higher cardinality, then maintaining clustering on that column is more expensive.”

🔹 Cardinality: What It Is

Cardinality refers to the number of **distinct values** in a column.

- **High cardinality** = many unique values (e.g., a `UUID`, `email address`, or `timestamp`)
- **Low cardinality** = few unique values (e.g., `gender`, `status`, or `country`)

🔹 Clustering in Snowflake

Clustering in Snowflake means organizing the underlying data storage to group similar rows together based on the values of certain columns.

This improves **query performance** (especially for large tables with selective filters), but Snowflake must actively **maintain** the clustering as data changes.

🔹 Why High Cardinality Increases Cost

When clustering by a **high-cardinality column**, Snowflake has to:

1. Track a **large number of distinct values**.
2. Reorganize the data often to keep values in proper "clusters."
3. Split and rewrite many small micro-partitions to maintain clustering.

This becomes computationally expensive, both in **processing time** and **storage I/O**.

🔹 Example

Let’s say you have a table with 1 billion rows:

- Clustering by `customer_id` (millions of distinct values → **high cardinality**) = ❌ **Expensive**
- Clustering by `region` (only 10 distinct values → **low cardinality**) = ✅ **Less expensive**