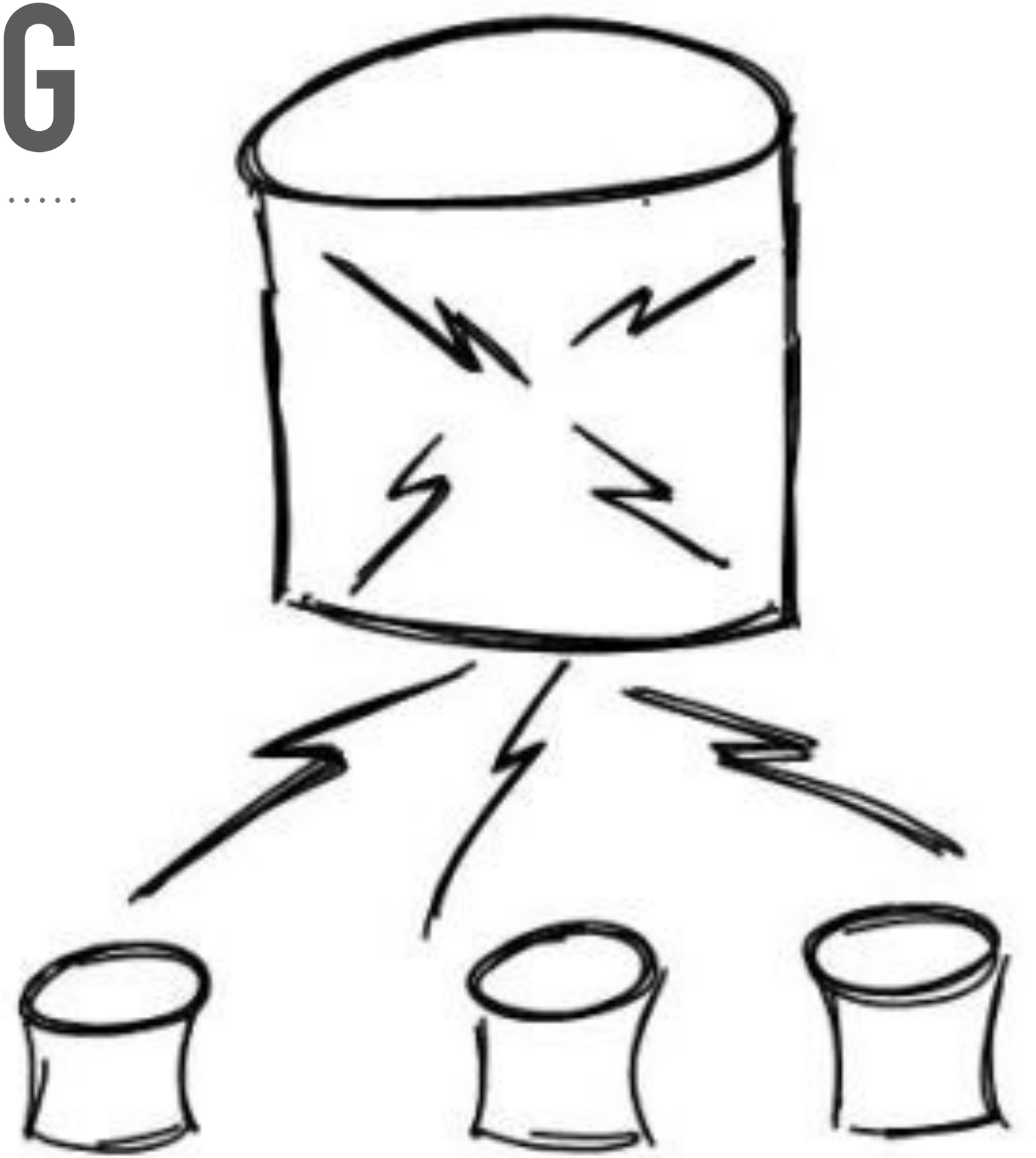
SHARDING

UNIT 5



Sharding is a type of database partitioning that separates very large databases the into smaller, faster, more easily managed parts called data shards.

ETYMOLOGY

ETYMOLOGY

- The word "shard" in a database context may have been introduced by the Computer Corporation of America's "System for Highly Available Replicated Data".
- However, the SHARD system appears to have used its redundant hardware only for replication and not for horizontal partitioning

VERTICAL PARTITIONING

VERTICAL PARTITIONING

- Vertical partitioning involves creating tables with fewer columns and using additional tables to store the remaining columns.
- Normalization also involves this splitting of columns across tables.
- Vertical partitioning goes beyond that and partitions columns even when already normalized.
- Example : PROJECT operation in SQL

HORIZONTAL PARTITIONING

HORIZONTAL PARTITIONING

- Horizontal partitioning is a database design principle whereby rows of a database table are held separately, rather than being split into columns.
- Each partition forms part of a shard, which may in turn be located on a separate database server or physical location.
- This way, horizontal partitioning can be seen to be synonymous to sharding but they are not the same.
- Example: SELECT operation in SQL

EXAMPLE OF HORIZONTAL AND VERTICAL PARTITIONING

EXAMPLE

➤ Consider a table “example_table” as follows,

a : b : c : d : e

---

1 : 2 : 3 : 4 : 5

2 : 2 : 3 : 4 : 5

2 : 2 : 3 : 4 : 5

EXAMPLE

- We can write a query to access the columns and rows separately as,
  - PROJECT a, b (SELECT a=1 (example_table))
  - SELECT a, b FROM example_table WHERE a=1

a : b || c : d : e

---

1 : 2 || 3 : 4 : 5

1 : 2 || 3 : 4 : 5

==========

2 : 2 || 3 : 4 : 5

2 : 2 || 3 : 4 : 5

EXAMPLE

➤ The result would be

a : b

---

1 : 2

1 : 2

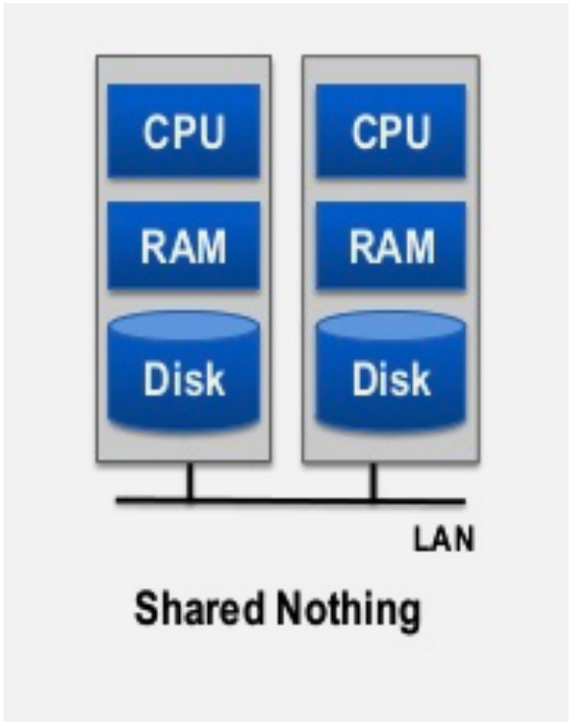
SHARDING

SHARDING

- Horizontal partitioning splits one or more tables by row, usually within a single instance of a schema and a database server.
- Sharding goes beyond this: it partitions the problematic table(s) in the same way, but it does this across potentially multiple instances of the schema.
- The obvious advantage would be that search load for the large partitioned table can now be split across multiple servers (logical or physical), not just multiple indexes on the same logical server.

SHARDING

- Sharding is related to a shared nothing architecture.
- Once sharded, each shard can live in a totally separate logical schema instance / physical database server / data center / continent.
- There is no ongoing need to retain shared access (from between shards) to the other unpartitioned tables in other shards.



SHARDING

➤ ADVANTAGES

- ➤ Since the tables are divided and distributed into multiple servers, the total number of rows in each table in each database is reduced.
- ➤ This reduces index size, which generally improves search performance.
- ➤ The load can be spread out over multiple machines, greatly improving performance.
- ➤ If the database shard is based on some real-world segmentation of the data, relevant data can be accessed easily.

SHARDING

➤ DISADVANTAGES

- ➤ **Increased complexity of queries** - Increased bugs because the developers have to write more complicated queries to handle sharding logic.
- ➤ **Single point of failure** - Corruption of one shard due to network/hardware/systems problems causes failure of the entire table.
- ➤ **Failover servers more complex** - Failover servers must themselves have copies of the fleets of database shards.
- ➤ **Backups more complex** - Database backups of the individual shards must be coordinated with the backups of the other shards.
- ➤ **Operational complexity added** - Adding/removing indexes, adding/deleting columns, modifying the schema becomes much more difficult.

HOW IS SHARDING DONE?

HOW IS SHARDING DONE? – DRIVING PRINCIPLES

- **How the data is read**—Databases are used to store and retrieve data.Data retrieval requirements (or lack thereof) heavily influence the sharding strategy.
- **How the data is distributed**—Once you have a cluster of machines acting together, it is important to ensure that data and work is evenly distributed. Uneven load causes storage and performance hotspots.
- Once sharding is employed, **redistributing data** is an important problem. Once your database is sharded, it is likely that the data is growing rapidly. Adding an additional node becomes a regular routine.

HOW IS SHARDING DONE? – CATEGORIES

- There are various categories based on which sharding can be achieved.
  - Algorithmic Sharding
  - Dynamic Sharding
  - Entity Groups
  - Hierarchical Keys and Column-Oriented Databases

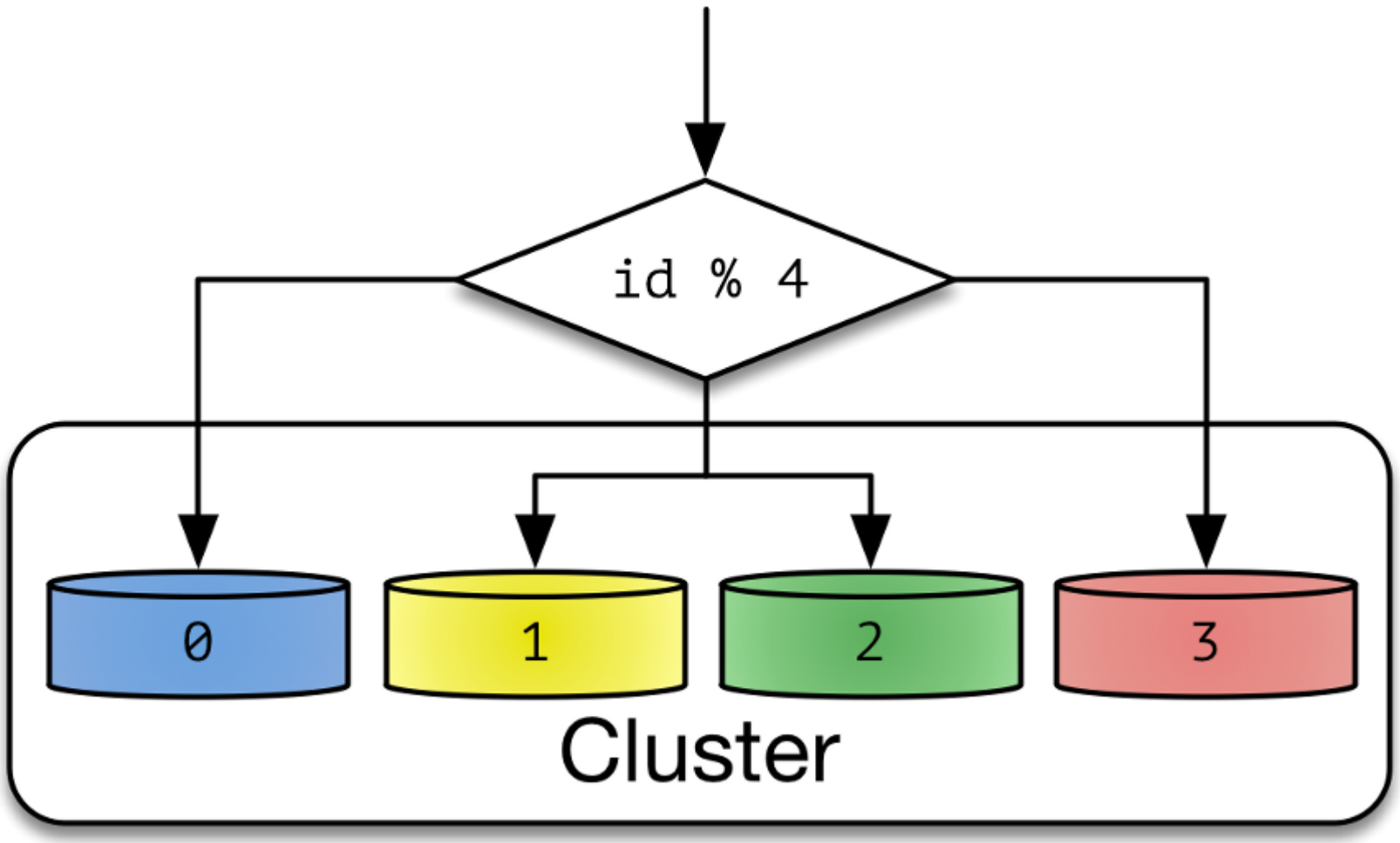
HOW IS SHARDING DONE? – ALGORITHMIC SHARDING

- In algorithmic sharding, the client can determine a given partition's database without any help.
- Algorithmically sharded databases use a sharding function (partition_key) -> database_id to locate data.
- A simple sharding function may be "hash(key) % NUM_DB".
- Algorithmic sharding distributes data by its sharding function only.
- It doesn't consider the payload size or space utilization. To uniformly distribute data, each partition should be similarly sized.
- EXAMPLE: Memcached

HOW IS SHARDING DONE? – ALGORITHMIC SHARDING

- Fine grained partitions reduce hotspots—a single database with many partitions, and the sum of data between databases is statistically likely to be similar.
- For this reason, algorithmic sharding is suitable for key-value databases with homogeneous values.
- Resharding data can be challenging.
- It requires updating the sharding function and moving data around the cluster.
- Doing both at the same time while maintaining consistency and availability is hard.

HOW IS SHARDING DONE? – ALGORITHMIC SHARDING



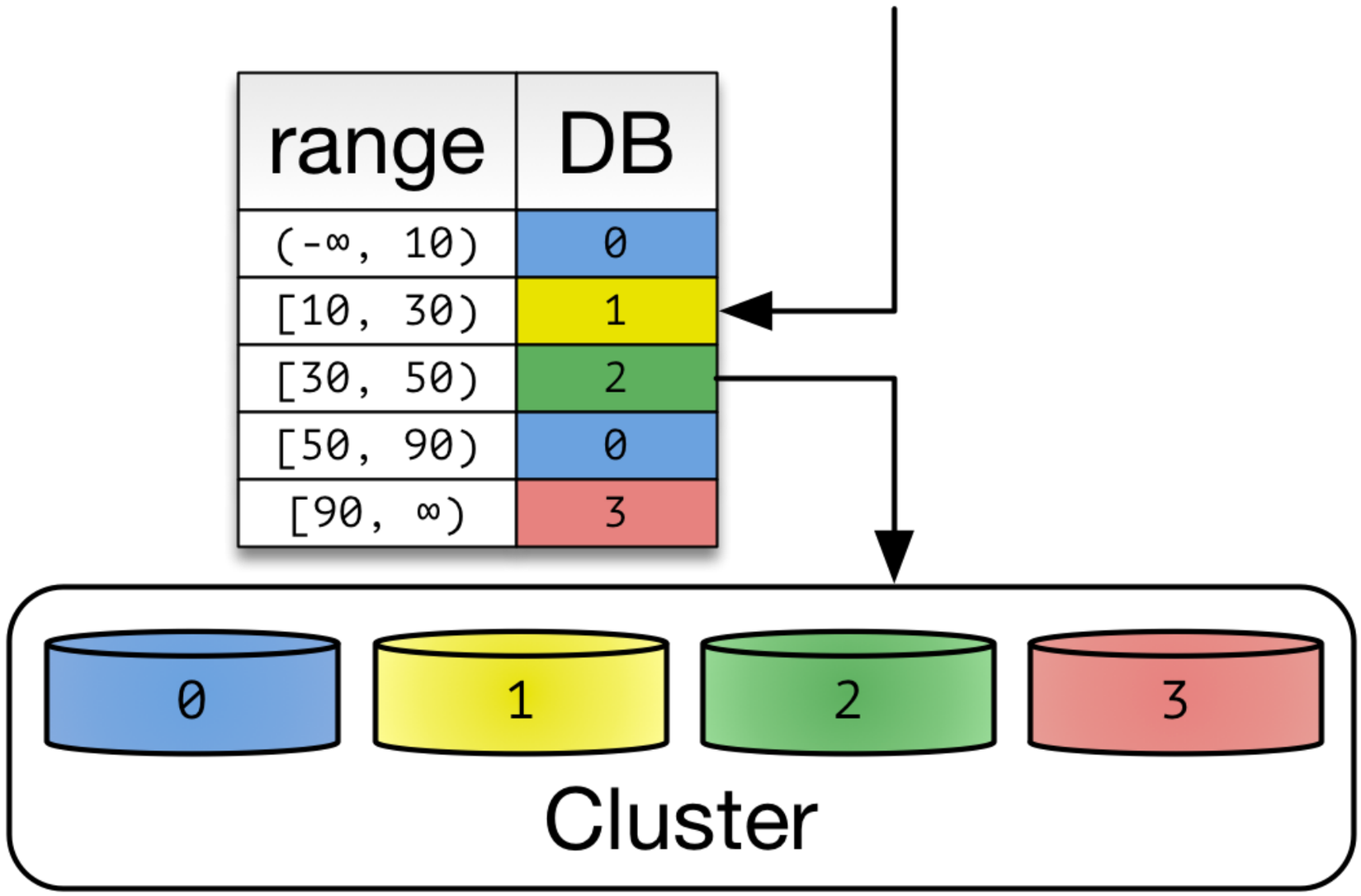
HOW IS SHARDING DONE? – DYNAMIC SHARDING

- In dynamic sharding, an external locator service determines the location of entries.
- If the cardinality of partition keys is relatively low, the locator can be assigned per individual key.
- Otherwise, a single locator can address a range of partition keys.
- In the example of range-based partition keys, range queries are efficient because the locator service reduces the number of candidate databases.
- To read and write data, clients need to consult the locator service first.

HOW IS SHARDING DONE? – DYNAMIC SHARDING

- Dynamic sharding is more resilient to nonuniform distribution of data.
- Locators can be created, split, and reassigned to redistribute data.
- However, relocation of data and update of locators need to be done in unison.
- The locator service becomes a single point of contention and failure. Every database operation needs to access it, thus performance and availability are a must.
- EXAMPLE: HDFS, HBase, MongoDB

HOW IS SHARDING DONE? – DYNAMIC SHARDING



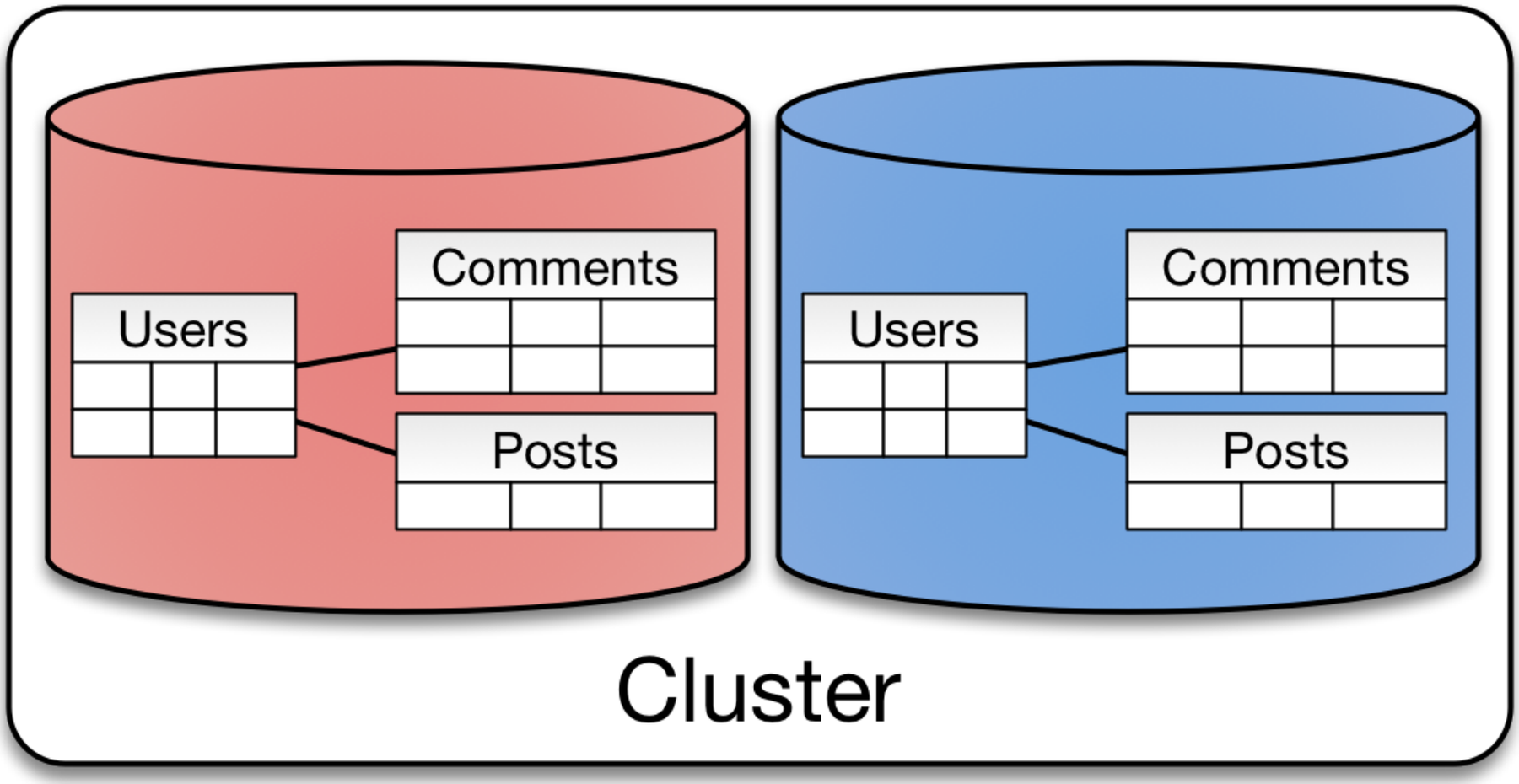
HOW IS SHARDING DONE? – ENTITY GROUPS

- Previous examples are geared towards key-value operations. However, many databases have more expressive querying and manipulation capabilities.
- CONCEPT: Store related entities in the same partition to provide additional capabilities within a single partition.
- This is advantageous as features such as joins, indexes and transactions reduce complexity for an application.
- Queries within a single physical shard are efficient.
- Stronger consistency semantics can be achieved within a shard.

HOW IS SHARDING DONE? – ENTITY GROUPS

- For example, in a typical web application, data is naturally isolated per user. Partitioning by user gives scalability of sharding while retaining most of its flexibility.
- Entity groups can be implemented either algorithmically or dynamically.
- They are usually implemented dynamically since the total size per group can vary greatly.
- EXAMPLE: Google Megastore

HOW IS SHARDING DONE? – ENTITY GROUPS



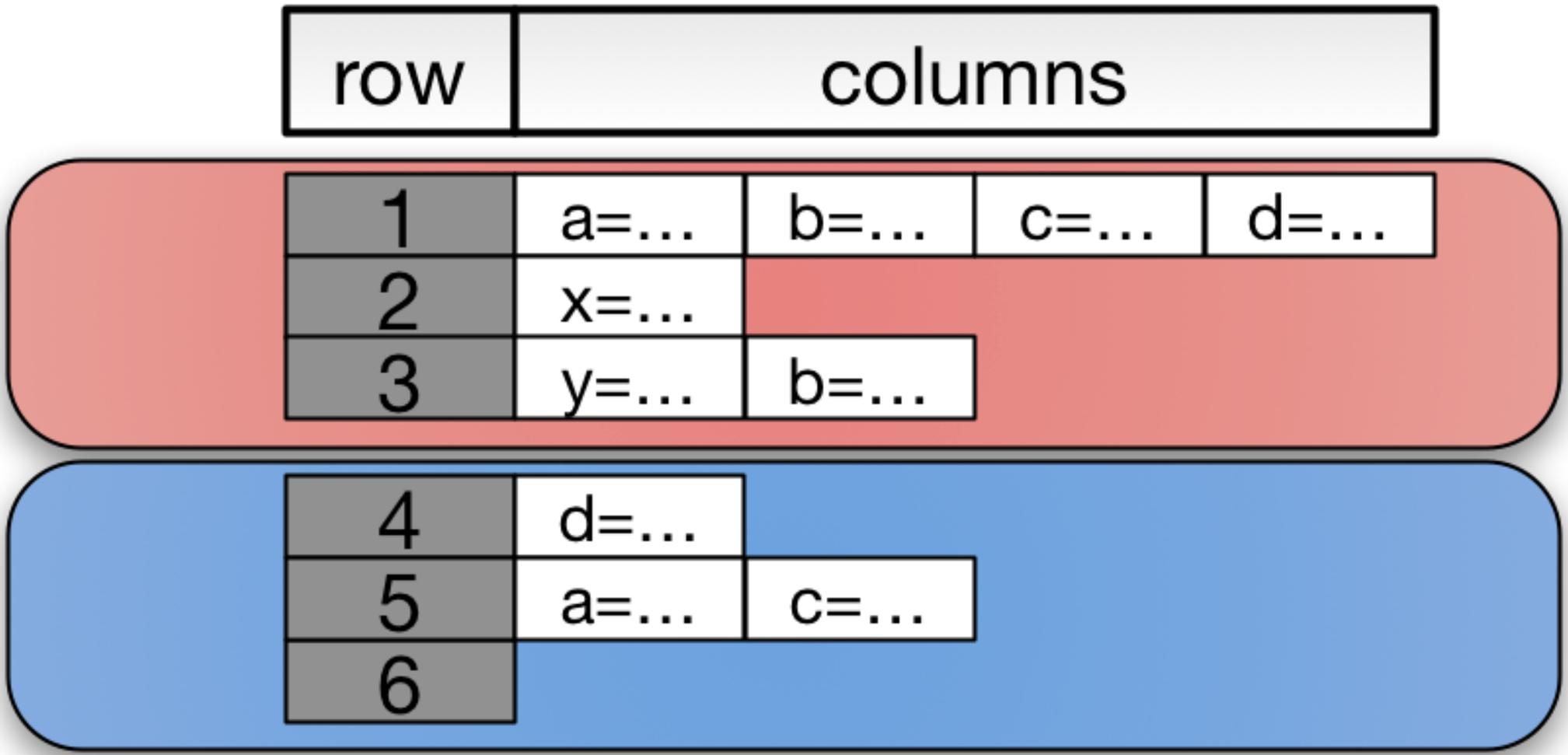
HOW IS SHARDING DONE? – COLUMN ORIENTED DATABASES

- Column-oriented databases are an extension of key-value stores.
- They add expressiveness of entity groups with a hierarchical primary key.
- A primary key is composed of a pair (row key, column key).
- Entries with the same partition key are stored together.
- The restriction given by hierarchical keys allows databases to implement data-agnostic sharding mechanisms and efficient storage engines.

HOW IS SHARDING DONE? – COLUMN ORIENTED DATABASES

- Column-oriented databases can be sharded either algorithmically or dynamically.
- With small and numerous small partitions, they have constraints similar to key-value stores.
- Otherwise, dynamic sharding is more suitable.
- EXAMPLE: Amazon DynamoDB is a platform-as-a-service offering of Dynamo.
- DynamoDB uses (hash key, range key) as its primary key.
- The term column database is losing popularity. Both HBase and Cassandra once marketed themselves as column databases, but not anymore. They can be called hierarchical key-value stores, since this is the most distinctive characteristic between them.

HOW IS SHARDING DONE? – COLUMN ORIENTED DATABASES



HOW IS SHARDING DONE? – PITFALLS

- A **logical shard** (data sharing the same partition key) must fit in a single node. This is the most important assumption, and is the hardest to change in future. A logical shard is an atomic unit of storage and cannot span across multiple nodes. In such a situation, the database cluster is effectively out of space.
- Many web applications **shard data by user**. This may become problematic over time, as the application accumulates power users with a large amount of data.

HOW IS SHARDING DONE? – PITFALLS

- Even though **dynamic sharding** is more resilient to unbalanced data, an unexpected workload can reduce its effectiveness. In a range-partitioned sharding scheme, inserting data in partition key order creates hot spots. Only the last range will receive inserts. This partition range will split as it becomes large. However, out of the split ranges, only the latest range will receive additional writes.
- In the case of dynamic sharding, it is bad to have a **large number of locators.** Since the locators are frequently accessed, they are normally served directly from RAM. HDFS's Name Node needs at least 150 bytes of memory per file for its metadata, thus storing a large number of files is prohibitive.

SUMMARY

SUMMARY

- Partitioning - Horizontal and Vertical
- Sharding - Definition
- How sharding is different from partitioning
- Advantages and Disadvantages of sharding
- Various methods for sharding
- Pitfalls in the various methods

THANK YOU