**Database Sharding**

Sharding means partitioning.

Suppose you have only 1 DB which stores 1million users at max.

If, your user base increases more than 1 million, suppose 4 million, in that case, we need to have more number of database and divide our data in multiple DBs.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | City | Age | ID | Role |
| Nik | Delhi | 22 | 1 | SDE1 |
| Ansh | Mumbai | 12 | 2 | Intern |
| Sharad | Pune | 42 | 3 | PM |

If we divide our data (tables) by keeping **different columns** in different DBs, this is called **vertical partitioning**.

If we divide our data such that, **different number of rows** are stored on different DBs, this is called **horizontal partitioning**.

**Horizontal Partitioning = Database Sharding.**

**Shard** is a subset of data.

When we divide 4 million users into 4 shards, each of 1 million, we say, that we have created 4 logical shards, each of 1 million.

Now I can store 4 million users on two physical DBs (physical shards), where each DB will store 2 logical shards that is 2 million users each.

Advantages of sharding

1. While querying the data, searches become faster, we can query multiple instances of DB at same time.
2. We can store data at geographically different locations, so that users querying around an area will get faster result when they have a physical shard present near them.
3. If anyone physical shard goes down, we ca still serve other users till that time.

Sharding Strategies.

**Algorithmic Sharding**

When the application knows, which shard it needs to communicate and queries to that particular shard. This is called algorithmic sharding where app takes care of determining shard.

S4

S3

S2

2

S1

APP

**Dynamic Sharding**

Here, application talks to a middleware called Service Locator which looks up into all shards and determines, which shard the query needs to be sent too.

Here, app is not responsible for determining the shard. We can add and remove shards in this strategy, hence called dynamic sharding.

S2

2

S1

S3

S4

Service locator

APP

Disadvantages of Sharding

1. If we don’t choose a sharding strategy correctly, some shards may have less data, some may have a lot of data (hotspots), which will make querying slower.
2. It’s very difficult to switch back to non-sharded architecture, to combine all data. Sharding should be last way of optimization after caching and all other ways. Once data is sharded, it can’t be merged.
3. If a query requires join operation on multiple shards, then it can be quite complex and time taking process.