LEC-18: Partitioning & Sharding in DBMS (DB Optimisation)

1. **A big problem** can be solved easily when it is chopped into several smaller sub-problems. That is what the partitioning technique does. It divides a big database containing data metrics and indexes into smaller and handy slices of data called partitions. The partitioned tables are directly used by SQL queries without any alteration. Once the database is partitioned, the data definition language can easily work on the smaller partitioned slices, instead of handling the giant database altogether. This is how partitioning cuts down the problems in managing large database tables.
2. **Partitioning** is the technique used to divide stored database objects into separate servers. Due to this, there is an increase in performance, controllability of the data. We can manage huge chunks of data optimally. When we horizontally scale our machines/servers, we know that it gives us a challenging time dealing with relational databases as it's quite tough to maintain the relations. But if we apply partitioning to the database that is already scaled out i.e. equipped with multiple servers, we can partition our database among those servers and handle the big data easily.
3. **Vertical Partitioning**
    1. Slicing relation vertically / column-wise.
    2. Need to access different servers to get complete tuples.
4. **Horizontal Partitioning**
    1. Slicing relation horizontally / row-wise.
    2. Independent chunks of data tuples are stored in different servers.
5. **When Partitioning is Applied?**
    1. Dataset become much huge that managing and dealing with it become a tedious task.
    2. The number of requests are enough larger that the single DB server access is taking huge time and hence the system's response time become high.
6. **Advantages of Partitioning**
    1. Parallelism
    2. Availability
    3. Performance
    4. Manageability
    5. Reduce Cost, as scaling-up or vertical scaling might be costly.
7. **Distributed Database**
    1. A single logical database that is, spread across multiple locations (servers) and logically interconnected by network.
    2. This is the product of applying DB optimisation techniques like **Clustering, Partitioning and Sharding.**
    3. Why this is needed? READ Point **5**.
8. **Sharding**
    1. Technique to implement Horizontal Partitioning.
    2. The **fundamental idea** of Sharding is the idea that instead of having all the data sit on one DB instance, we split it up and introduce a Routing layer so that we can forward the request to the right instances that actually contain the data.
    3. **Pros**
        1. Scalability
        2. Availability
    4. **Cons**
        1. Complexity, making partition mapping, Routing layer to be implemented in the system, Non-uniformity that creates the necessity of Re-Sharding
        2. Not well suited for Analytical type of queries, as the data is spread across different DB instances. (Scatter-Gather problem)