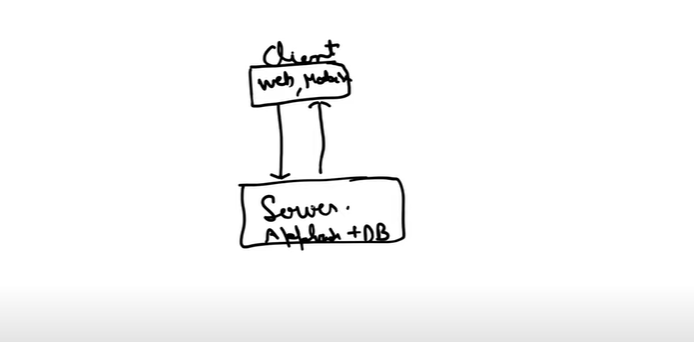
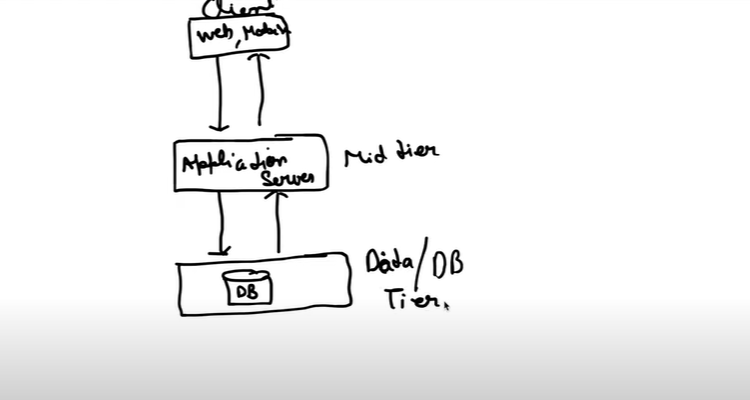
**How to scale from zero to 1 million Users in detail**

1. Single Server
2. Application and DB server application
3. Load Balancer and Multiple APP server
4. Database Replication
5. Cache
6. CDN
7. Data Centre
8. Messaging Queue
9. Single Sever



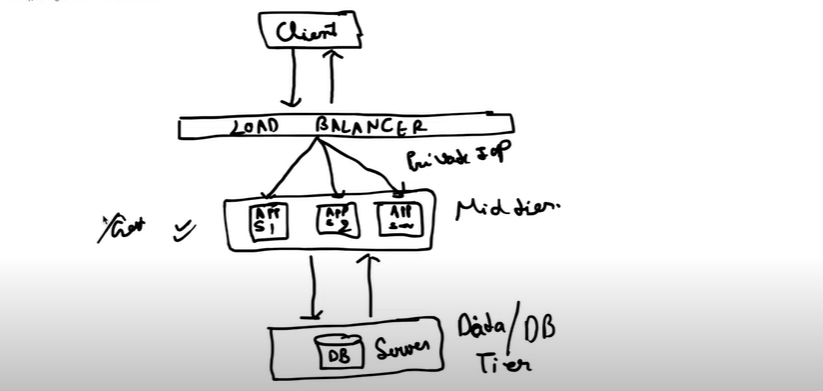
1. Application and DB server application
2. If I want to scale the application or DB server independently then we first need to segregate



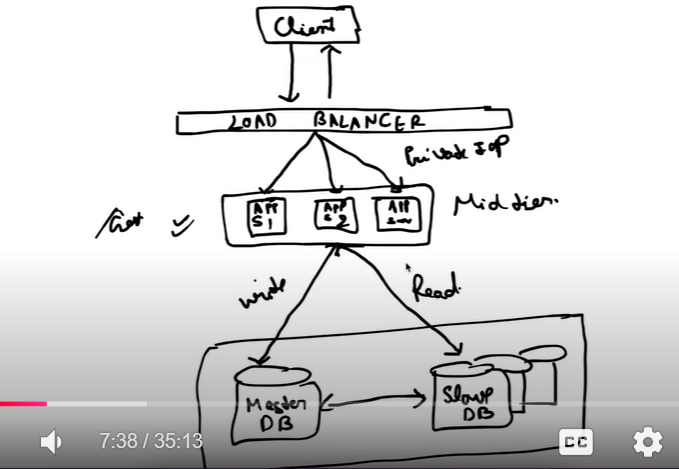
1. Load Balancer and Multiple App server

If I need to take more traffic then we need to add the multiple apps in the middle tier

1. Since we will have multiple application in middle tier. So, the client can’t talk directly to server.
2. Now, Loadbalancer will come into the picture which will equally divide the traffic between the application using the private IP.

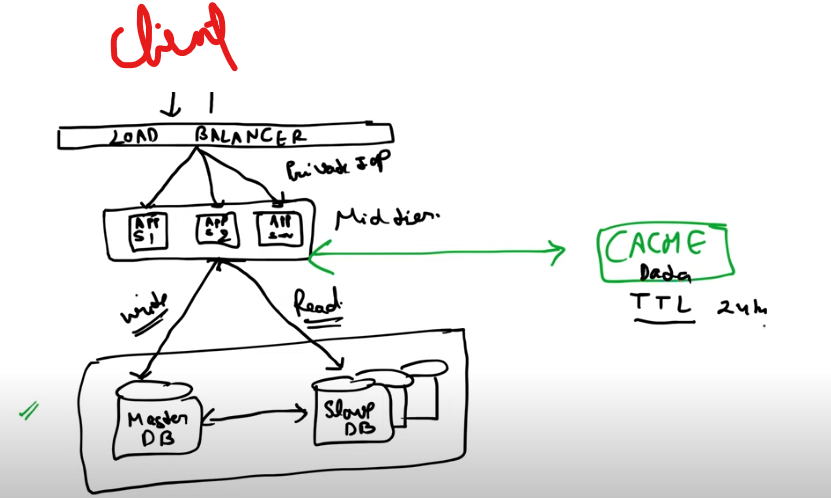


1. Database Replication
2. In Database Replication, the concept of master and slave comes into the picture
3. The Master DB takes care of write operation and slave DB takes care of Read operation
4. If Master DB goes down, then one of slave DB will promote as Master DB
5. If any slave DB is down, then we have multiple slave DB.

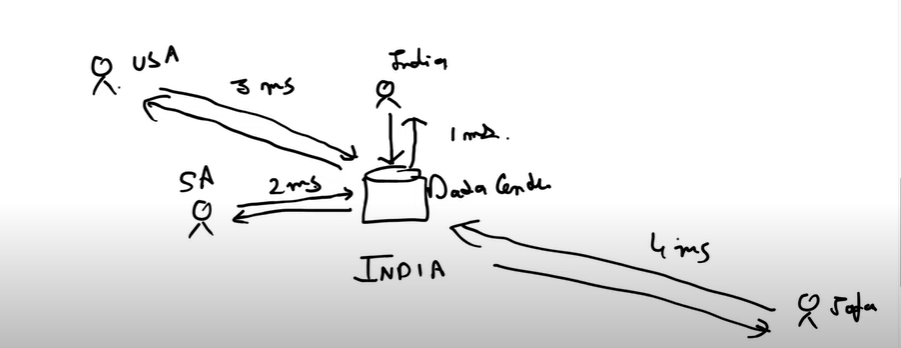


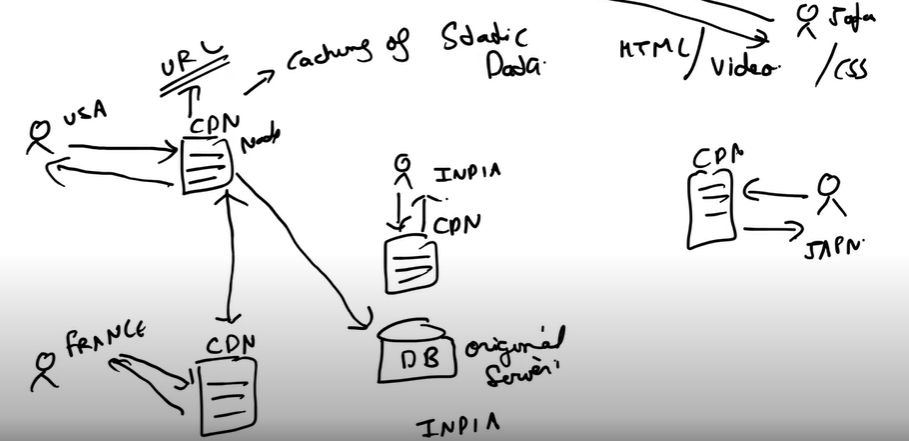
1. **Cache**

It is used for performance improvement



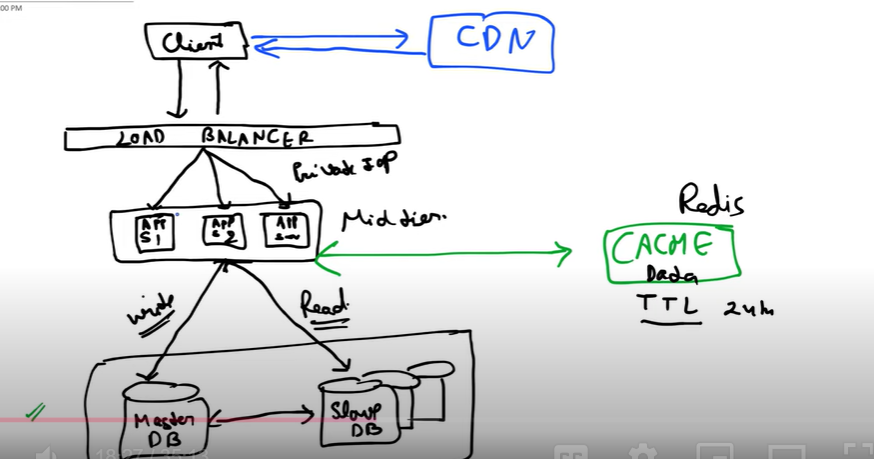
1. CDN
2. Content Delivery Network
3. CDN does caching but all those who does caching are not CDN
4. CDN does the caching of static data like HTML pages, videos, CSS files





Every country has its own CDN. Let us suppose if customer in USA wants to hit the URL which he already access then it goes to CDN and give the repose.

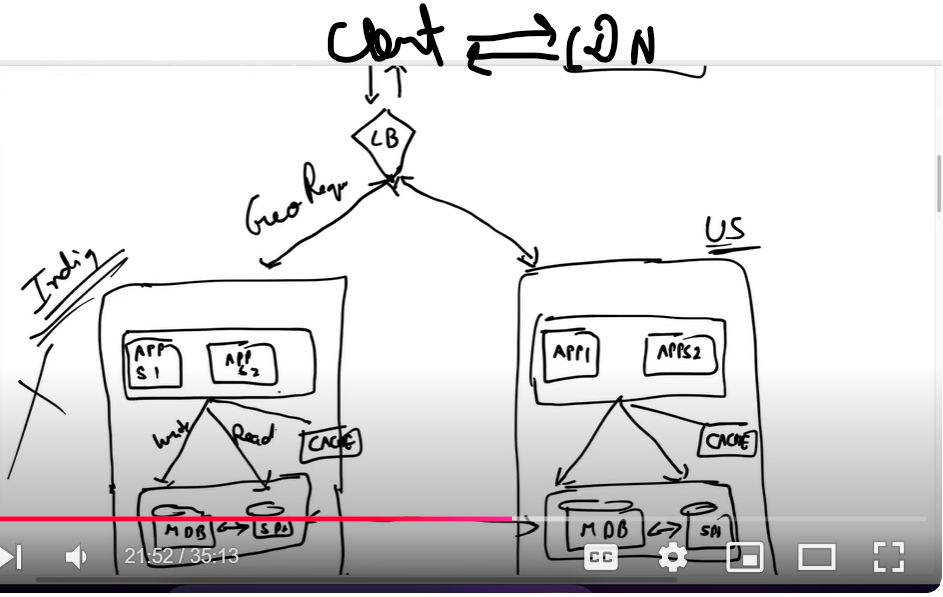
Let us suppose if URL was not hit earlier, then CDN asked to neighbour CDN whether you have. If it is not present then it will go to the original server.



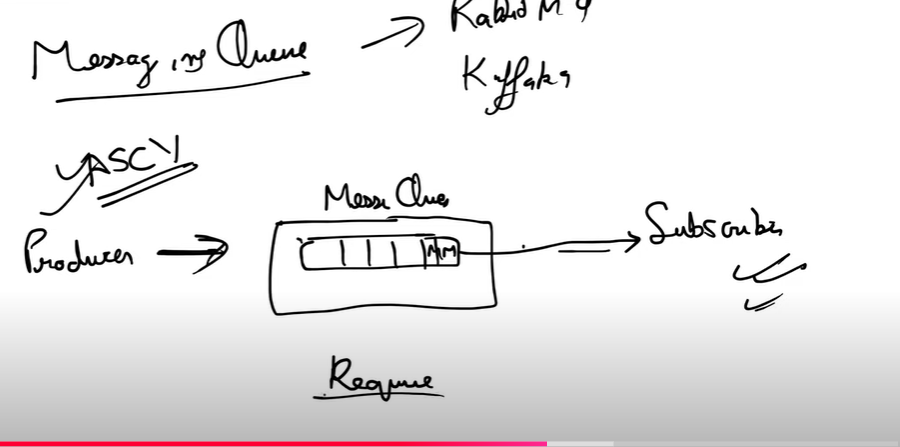
1. Data Centre

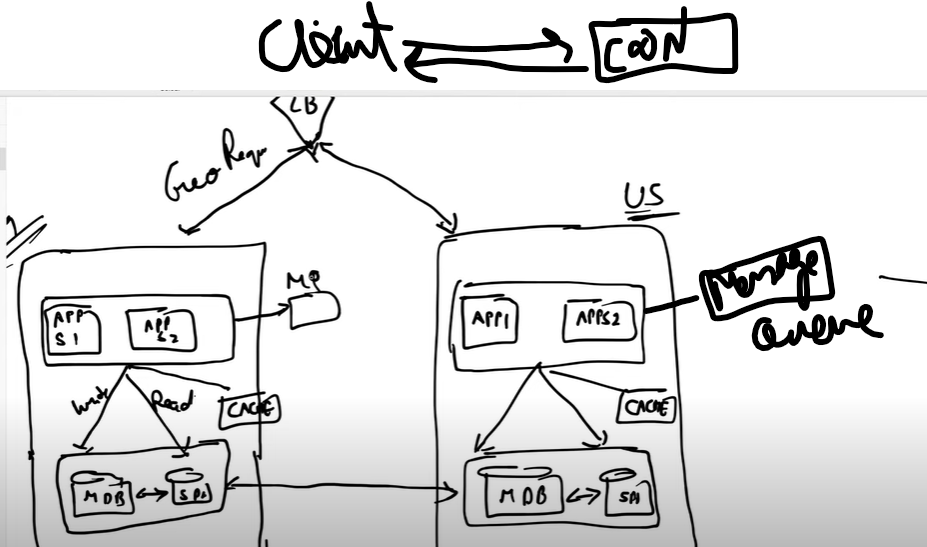
Request will be going through the Geo

Database replication will happen between 2 data center

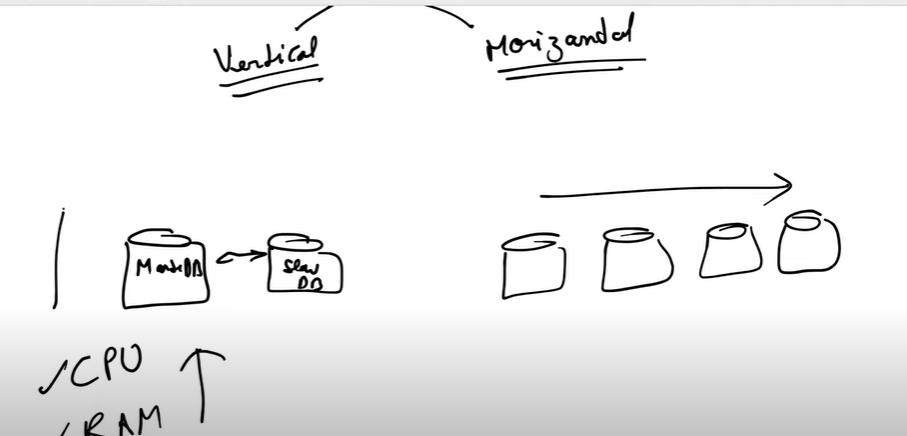


1. Messaging Queue





1. **Database scaling**
2. Vertical Scaling---Vertical scaling means we can increase the RAM and CPU of database. But there is a limit of increasing the RAM and CPU



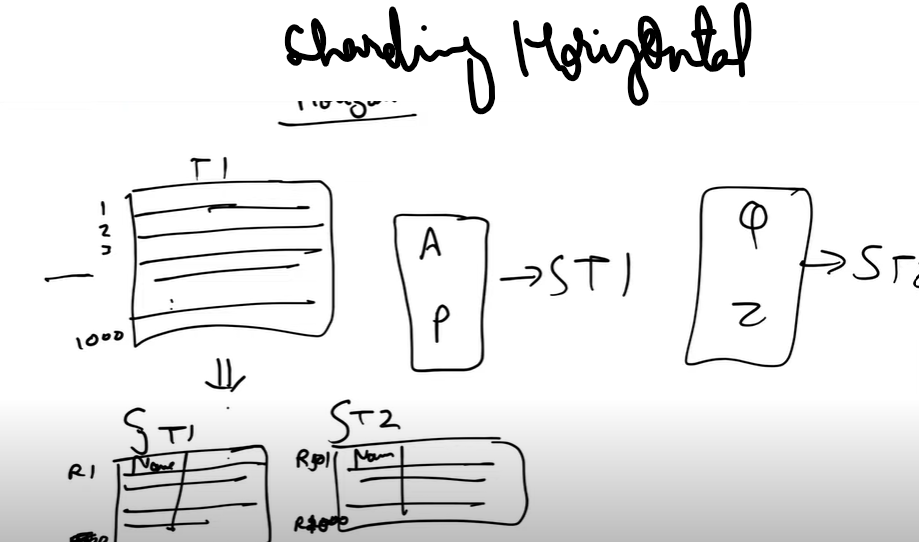
1. Horizontal scaling

In Horizontal Database scaling, the Sharding comes into the picture

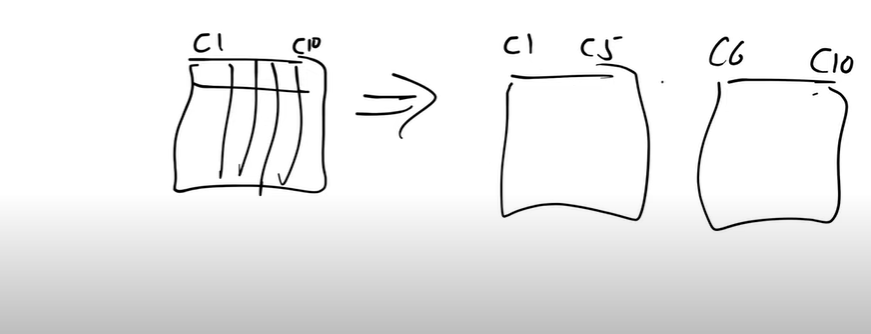
Sharding is of 2 types

1. Horizontal Sharding
2. Vertical Sharding

Horizontal Sharding—dividethe table in row wise

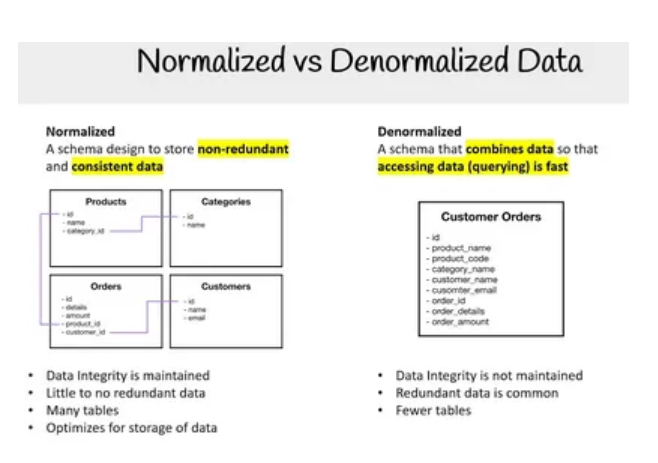


Vertical Sharding



The 1st Drawback of sharding is if all the data whose name start with A, it will only goes to SF1 sharding. So, SF1 will fill up very soon. So we need to shard again SF1 and so on which will create like a tree. Now This problem can be fixed by **Consistent Hashing**

**The 2nd drawback is since we devide in multiple tables. Now, I can’t use the join for which I need to do the de normalization.**



**Consistent Hashing**