In this topic we will discuss different type of data present in your system and how you will store.

for ex: structured data, semi structured data, Non structured data etc.

And to cater these different type of data, you might require different type of database like, SQL database (Ex. mySQL, oracle, Postgres etc.). Similarly we have No-SQL (Casandra, mongo db, HBase, Amazon DynamoDB etc.

Going forward, we may discuss, how you can choose right database according to the given constraint in your system.

As an example

- If your application required super low latency
- If your data is unstructured and no relationship data
- If you only need to serialize or deserialize your document for ex: JSON, XML, YAML etc.
- You need to store massive amount of data

These all are confirming that you need database where ACID properties guarantee is not on top priority

And hence, NOSQL database could be your candidate of choice.

Also you are clear about your choice of database and reason or logic of choosing perticular database. It also clear your thought of data significance in your system.

So, now next point of discussion could be, how much scale do you really need and also being an architect what would be your estimate of scale.

OK so now we are moving towards the discussion of do we prefer vertical scaling or horizontal scaling. Also we should discuss pros and cons of approach choosen over approach not chosen.

vertical scaling or scale up

Process of adding more power to the server for ex: more CPU, more RAM, more GPU etc.

Horizontal scaling or scale out

Process of adding more same box parallel to your existing box and create a group of your task executor. So instead of one box you have now n number of box who can execute task for you as they given the chance.

so you have two different option, and so you have confusion also, am I right? Confusion is which option do you choose and when?

So to handle your confusion, I encourage you to think more and understand the benefit of each type And once you know the benefit of each type, it helps you in your decision.

Vertical scaling

Pros

- If traffic is low to your system (specially when you are starting your system) for ex: Dropbox has not even 1000 users at the time they ~~started~~ and for initial year.
- when you have no experience in distributed system along with low traffic.

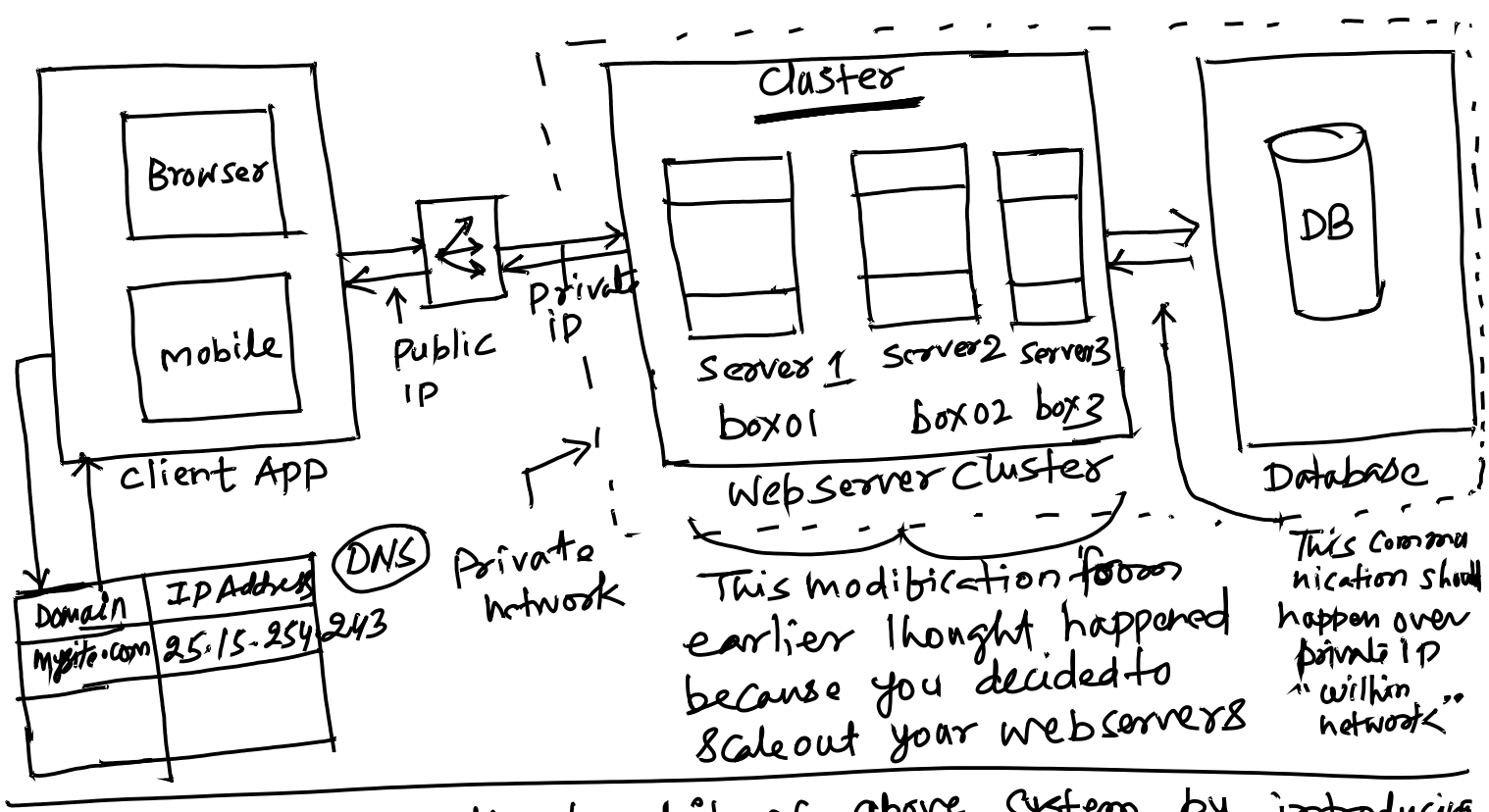
Cons

- It is limited with scale. Impossible to add unlimited resource to one server machine.
- What if your single machine Fail? You loose everything right?

So, by logic vertical scaling is only choice if you ~~have~~ very specific and purposeful decision to do so.

So, what is the alternative then, yes you are right - horizontal scaling is the answer to this problem

So in your last image you have seen that - you have only one server and so your system fail if your that perticular server is down. You have no other choice.



Let's understand the benefit of above system by introducing horizontal scale into your system.

- Even your server 1 is down, still your system is up and running and handling your user request. which was not the case in our earlier decision.

- Due to this abstraction, now you have liberty to scale dynamically (meaning as per load add more box and if load decreased, removed some box safely.

Still we have single DB and single point of failure in your system. So next is to understand how we will handle database failure?

Yes you guessed it right, for handling database failure you need database replication in your system.

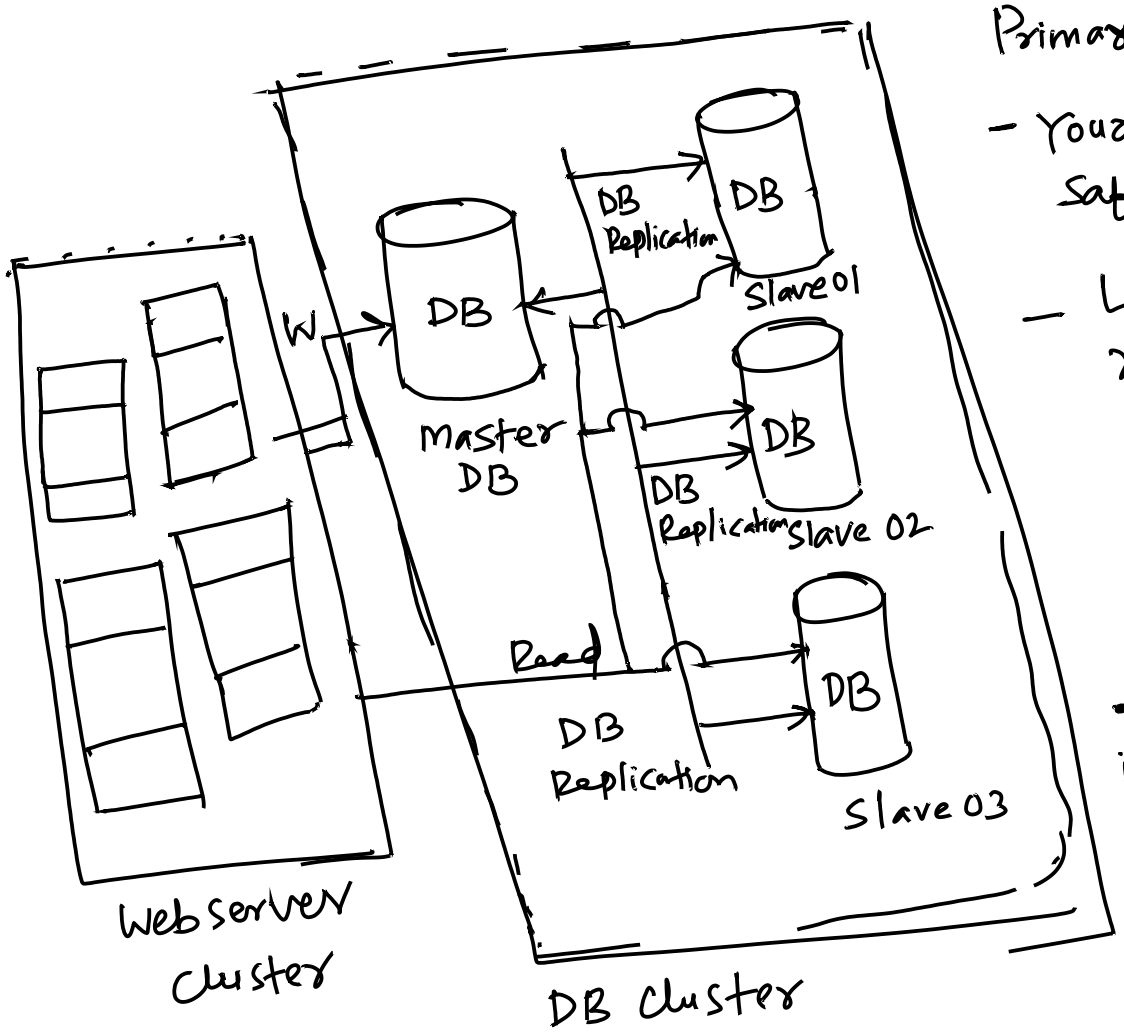
Database Replication

19/Feb/2022

There are different kind of strategy we are choosing under Database replication. we will learn each one by one in coming discussion. for now as an example you can think like master-slave, master-master etc.

OK confused ? Let's draw one example through diagram,

You can observe that below is an example of master-slave strategy for Database Replication



Primary notable benefit

- Your data is more safe and reliable
- Load on write & read distribution help in serving more User and with better performance.
- It also helps incase if some db server is down for a moment. So it improves your system availability

One thing I missed in above diagram to introduce loadbalancer infront of DB cluster, but in reality it must be their. So safely assume that you introduced load balancer before your DB cluster.