What is Scaling?

It is the ability to expand or concise the capacity of the system by either adding or removing the resources from it, in order to alter the usage of our application.

There is mainly two types of scaling:

➔ Horizontal Scaling.

➔ Vertical Scaling.

Horizontal Scaling:

Horizontal Scaling, also known as **Scale out,** refers to addition of extra servers/nodesto distribute the load among them.

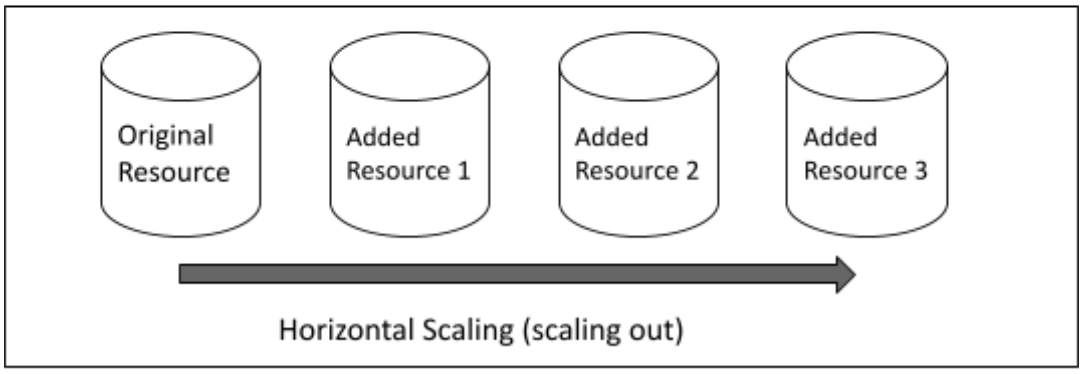
Now, this type of scaling puts up a little difficulties when dealing with relational databases because maintaining those relations while spreading the data out is tough to work out.

Although if we take into account the non-relational databases, then scaling out doesn't seem to be a bad option.

Therefore, to be little precise Horizontal Scaling refers to the addition of resources tothe existing system to optimize the performance of the system accordingly.

A small example:

Original System



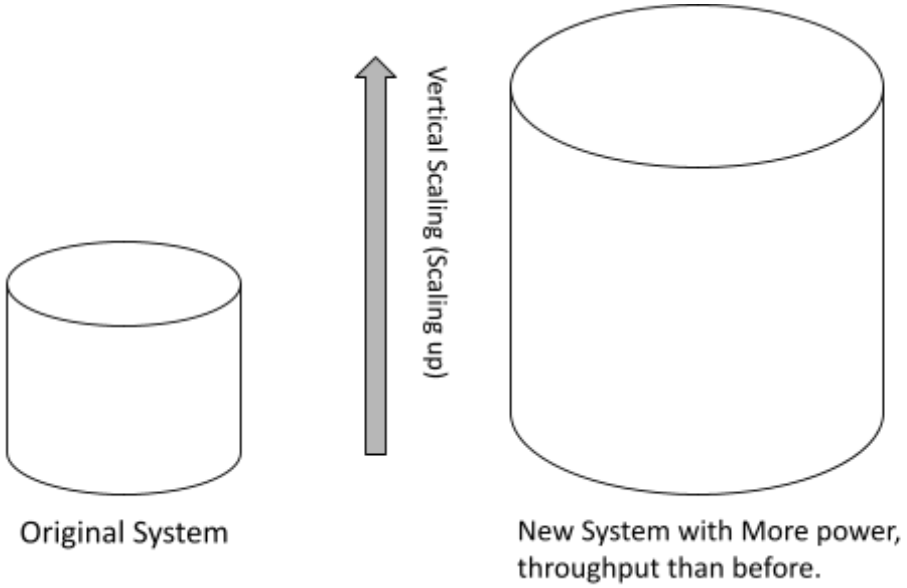
A new System with added resources

Three resources were added in the resources pool to build up a new system.

Vertical Scaling:

Vertical Scaling also known as **Scale up**, refers to increasing the power of the existingsystem/machine.
This type of scaling can be adapted by both relational as well as non-relational databases. But as access to everything is bad, scaling up also has a certain limit after which doing it further makes no sense.
Instead of combining multiple machines/servers we build one huge machine/serverhaving much more power and throughput than the previous system.

Below is how scaling up is done.



Difference between Horizontal Scaling and Vertical Scaling:

Horizontal Scaling	Vertical Scaling
Horizontal Scaling refers to the additionof resources to the existing system.	Vertical Scaling refers to increasing the computational power of the existing system/machine.
Difficult to implement.	Relatively, easy to implement.
Cost is expensive.	Not that expensive.

Advantages of NoSQL:

- **Horizontal Scaling (scaling out):** Databases have been scaling out for years and DBAs had been dependent on scaling up relational databases so that they could increase the efficiency. Scaling up means stacking up larger servers like the load raises or increasing the head count of hardware assets for the existing machine to overcome its limitations.
  But a point will arrive when more scaling up will not be possible as a machine can handle a particular amount of hardware assets and scaling out is not possible with relational databases because of it's technical restrictions. Scaling out means distributing the database throughout multiple computers when the load increases. Therefore, it can handle more traffic simply by adding more servers to the database.
  They have the ability to become larger and much more powerful, also it is quite cost effective. Hence, making them the preferred choice for large or constantly evolving data sets.

- **High Availability:** NoSQL databases are highly available due to its auto replication feature i.e. whenever any kind of failure happens data replicates itself to the preceding consistent state.
  Not only this, but whenever a server fails, we can access that data from another server as well, as in nosql database data is stored at multiple servers.

- **Flexible Schema:** Relational databases are used to have a defined schema, which is quite an issue because in case you need to make any modification or addition to the database we need to change the schema as well every time. Whereas in NoSQL databases, we could start working with the data without schema.

- **Shrading:** It is a database partitioning technique used by b lockchain companies with the purpose of scalability. In case of NoSQL it is an automated process that is the database is partitioned among a random number of the servers on it's own, with no requirement for application to be aware of it.\

- **Integrated Caching**: It's a technique of keeping frequently used data in the system memory for its instant use which removes the need for a separate

caching layer. NoSQL is equipped with such intelligent integrated caching capability.

- **Less Management:** NoSQL databases require minimum to zero management, they are equipped to auto-repair data distribution and due to it having flexible data models that results in reducing administration and performance difficulties.

Note: NoSQL also has certain other benefits like more compatible for cloud, helps raise funding for business as it is one reliable source for data storage and performing analytics as well.

Disadvantage of NoSQL:

- When compared to SQL or relational databases, NoSQL database is designed for storing data of many types but it lacks functionality like it doesn't support Transaction properties like ACID.
- The process of managing big data on NoSQL is quite complex as compared to RDBMS.
- It is also not compatible with SQL, although few NoSQL databases do use SQL but many don't. Hence, will require manual query language which is quite slow and highly complex.
- It doesn't support data entry with constraints like RDBMS.
  For eg: There's an on-campus placement going on for a company in multiple colleges. Now, they maintain a database of all students applied for the Job role. Now we need to enter only those whose CGPA is above 7.5.
  Here, NoSQL wouldn't be able to get the job done, whereas RDBMS will get the job done.
- Other concerns for NoSQL are standardization, Interfaces and Interoperability and also less community support(due to it being new in the industry).