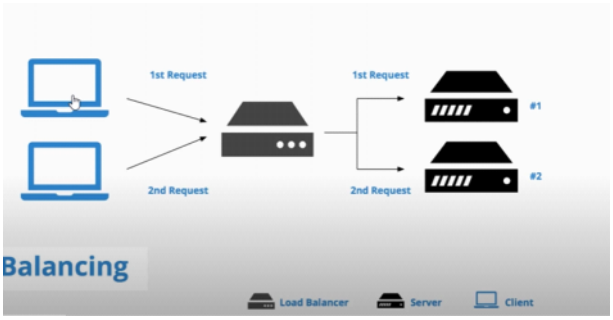
Load Balncer

Monday, February 15, 2021 4:22 AM



Load Balancers: Where it can be added

- User – Web Server
- Web server – Internal Server
- Internal Server - Database

Load Balancers: Algorithms

- Round Robin
- Round Robin with Weighted Server
- Least connections
- Least Response Time
- Source IP hash
- URL hash

Round robin with weighted server-> The system which has more capacity. Like 32 gb ram and other 16 gb so 32 gb will handel more

Ip address hash then apply as per that encoding

- Hardware LB:
  - They are hardware which works as LB, but are very expensive.
  - Even big companies use them only as first point of contact & use other mechanism for load-balancing.
- Software Load balancers:
  - It's hybrid approach. HAProxy is popular open source software LB.
  - Every client request on this port (where HAProxy is running) will be received by proxy & then passed to the backend service in efficient way.
  - HAProxy manages health check & will remove or add machines to those pools.
  - We should start with these.

****

Cache: What is Cache/ Caching

- Caching works on locality of reference principle: recently requested data is likely to be requested again.
- It's like short-term memory which has limited space but is faster & contains most recently accessed items.

Cache: Types

- Application Server Cache
- Distribute Cache
- Global Cache
- CDN

Cache: Where it can be added

- Cache can be used in almost every layer: hardware, OS, Web browsers, web application, but are often found nearest to the front end.

Mostly presnet on front pages. To retrive static pages

- Application Server Cache
  - Placing cache on request layer node enables local storage.
  - But when you've load balancer, it can send request to any node which can increase **cache miss**
  - Two choice to overcome: Distribute Cache & Global Cache

So there will load balancer. There will be two server 1 and server 2.
If server1 has cache then server 2doesnt know the cahce present.

So again if request come but this time if it goes to serevr 2 as load balncer route to 2 then there will be cache miss	then

- Distribute Cache
  - Each of its nodes own part of cached data.
  - The cache is divided up using a consistent hashing function, such that if a request node is looking for a certain piece of data, it can quickly know where to look within distributed cache to check if data is available.

- CDN

- Distribute Cache
  - Each of its nodes own part of cached data.
  - The cache is divided up using a consistent hashing function, such that if a request node is looking for a certain piece of data, it can quickly know where to look within distributed cache to check if data is available.
  - Easily we can increase the cache space just by adding nodes to the request pool

- Global Cache
  - All nodes use the same single cache space.
  - There can two type of global cache:
    - First, when a cached response not found in cache, cache itself becomes responsible for retrieving the missing piece of data.
    - Second, it's the responsibility of request nodes to retrieve any data that is not found

Product price ->1000
So salle then- now ->900
If cache stored 1000 then it need to update and invalidate cache

So again if request come but this time if it goes to serevr 2 as load balncer route to 2 then there will be cache miss	then

- CDN
  - It's Content Distribution Network for serving large amount of static media which is common to all.
  - First request ask the CDN for data, if not, cdn will query the back-end servers & then cache it locally

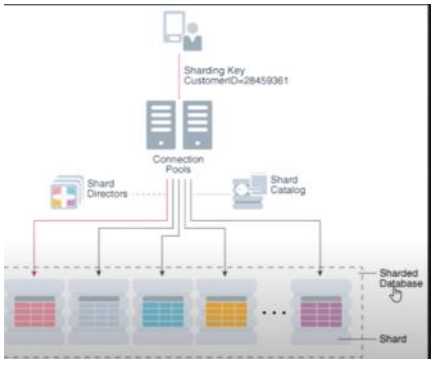
Static data like image

Cache Invalidation

- When data is modified in DB, it should be invalidated in the cache.
- Types
  - Write-through Cache
    - Data is written into the cache & the DB at the same time.
    - This minimizes the risk of data loss, but has disadvantage of higher latency for write operations.
  - Write Around Cache
    - Data is written directly to storage, bypassing the cache.
    - This reduces flooded write operations but has disadvantage that a read request for recently written data will create a cache miss & must be read from slower back-end.
  - Write Back Cache
    - Data is written to cache alone & completion is immediately confirm to client & write to permanent storage is done after specified intervals.
    - Risk of Data loss in case of crash

****

Sharding-> Divide big databases to small databases



Handling all data in single databases is not good. Single failure. If databses fail all data will go

Replciate the data

Sharding: What is Sharding

- Sharding is a technique to break up a big database into many smaller parts
- Horizontal scaling means adding more machines, which is cheaper & more feasible
- Vertical scaling means improving servers

Sharding Methods

- Horizontal partitioning
  - Putting different rows into different DBs.
  - Range based Sharding , i.e Based on Alphabetically names
- Vertical Partitioning:
  - Divide tables based of features i.e 1 for user, 1 for location
- Directory based Partitioning
  - we query directory server that holds the mapping between each tuple key to its DB server.

Horizonatl partioning-> Starting 4 character will be paritioned range can be pincode or anything, database paritioned
SO A to d so many charcter other can be low so it can imbalance

Vertical-> 5 schema.
1 feature in 1 db 1 photo 1 user info 1 user for location
So it can be 1 feature has more data other have less

Sharding Criteria

- Hash based
  - Using hashcode on any entity value
- List Partitioning:
  - Based on List i.e. Regions: APAC, EMEA, US
- Round-Robin Partitioning
- Composite Partitioning: Combining any above Schemes

Sharding Challanges

- ACID Compliance
- Joins Inefficient

Based on region list partion as well as hasmap in composite partiton

****

Indexes: What are Indexes

1. Search query is slow

****

Indexes: What are Indexes

- Useful in database to improve the speed of data retrieval.
- Index makes trade-off of increased storage overhead, slower writes for the benefit of faster read.
- Index is a data structure of table of contents that points us to the location where actual data lives, so when we create an index on a column of a table, we store that column & a pointer to the whole row in the index.

1. Search query is slow.
2. Indexes to get data faster.
3. If we have to cloumn then we can store ascending order or descending order. Or we can store using hashing
4. So search know where is data
5. Mongodb can do index on single index or compund index. Or in text particualr term

Type of indexing

- Ordered Indexing
  - Column is sorted as per ascending order.
- Hash Indexing:
  - Indexing is as per Hash function & Hash table

Proxy: What is Proxy Server

- Proxy server is intermediary hardware/software that sits between the client & server
- Used to filter requests:
  - Log requests
  - Transform request by adding/ removing headers
  - Encrypting/decrypting or compression

Proxy: Benefit of using Proxy Server

- Cache: Which can serve a lot of requests.
- It can coordinate requests from multiple servers & can be used to optimize request traffic.

Proxy: Benefit of using Proxy Server

- It collapse requests for data that is spatially close together in the storage, which'll decrease request latency.
- Proxies are useful under high load situations or when we have limited caching.
- Provides anonymity and may be used to bypass IP address blocking.

How Message Queue Works

- Messages are stored on the queue by Producer until they are processed by Consumer and deleted .
- Each message is processed only once, by a single consumer.
- Queues are also used as fault tolerance; In case of any service outage, they can retry the requests.