High Availability & Redundancy Study Notes

I. High Availability (HA) & The "9s"

High Availability is the characteristic of a system which aims to ensure an agreed level of operational performance, usually uptime, for a higher than normal period.

- **SPOF (Single Point of Failure):** If one component fails and the whole system stops, that is a SPOF. We kill SPOFs by using **Redundancy** (having backups).
- **The Goal:** We often measure HA in "Nines." For example, **99.9%** (Three Nines) allows for about 9 hours of downtime per year, while **99.999%** (Five Nines) allows only about 5 minutes.

> Interview Talk Track:
>
> "High Availability isn't about building a system that never fails—it's about building a system that is resilient to failure. I look at every layer of the stack, from the DNS to the Database, and ask: 'If this box dies right now, does the user notice?' If the answer is yes, I need to add redundancy."

II. Replication Strategies

Replication is about keeping copies of your data or services in different places.

1. Active-Passive (Failover)

- The **Active** node handles all traffic.
- The **Passive** node (Standby) stays in sync but does nothing until the Active node dies.
- **Pros:** Easy to manage, consistent data.
- **Cons:** Wasted resources (you are paying for a server that is doing nothing).

2. Active-Active (Multi-Master)

- Both nodes handle traffic simultaneously.
- **Pros:** Great for scaling and high traffic; no wasted resources.
- **Cons:** Very complex to keep data perfectly synchronized between them.

III. Database Replication (Primary-Replica)

This is the most common way to scale databases for web applications, which are usually **read-heavy**.

- **Primary Node:** Handles all **Writes** (INSERT, UPDATE, DELETE).
- **Replica Nodes:** Handle all **Reads** (SELECT). Data is copied from the Primary to the Replicas.

Interview Talk Track:

"For a system like a news site or social media, where users read much more than they post, I would implement Primary-Replica replication. This allows us to scale horizontally by adding more read-replicas as our user base grows, while the primary node focuses purely on data integrity and writes."

IV. Failover & Health Checks

Failover is the process of switching to a redundant or standby computer server, system, hardware component, or network upon the failure of the previously active one.

- **Automatic Failover:** Requires a monitoring system (Health Checks) to detect the failure and a "Heartbeat" mechanism to confirm a node is alive.
- **The Challenge:** The "Split-Brain" problem, where two nodes both think they are the leader and start overwriting each other's data.

V. Availability vs. Reliability

They sound similar, but they are different:

- **Availability:** Is the system "up"? (A system can be available but returning wrong data).
- **Reliability:** Does the system work "correctly" over a period of time?

Interview Talk Track:

"I like to distinguish between the two by saying: A system that crashes once a year for an hour is highly Available (99.98%), but if it loses your data during that hour, it's not Reliable. We want a system that is available so users can reach it, but reliable so they can trust it."

Practice Interview Questions (Day 11–12)

1. "What is the 'Split-Brain' problem in a failover scenario?"

- **Answer:** It happens when the network between two nodes fails, but both nodes are still running. Both nodes think the other is dead, so they both try to become the "Primary." This leads to data corruption because both start writing to the database independently. We solve this using a **Quorum** (a majority vote system).

2. "Why would you choose Active-Passive over Active-Active?"

- **Answer:** Simplicity and **Consistency**. Active-Active is notoriously difficult to keep in sync (data conflicts). If the system doesn't have massive traffic that requires multiple active masters, Active-Passive is safer and easier to maintain.

3. "If our Primary database dies and a Replica is promoted, is there a risk of data loss?"

- **Answer:** Yes, if we are using **Asynchronous Replication**. If the Primary dies before it finished sending the latest data to the Replica, that data is lost. For 100% safety, we use **Synchronous Replication**, but that makes the system slower.

4. "How do you achieve High Availability for a Load Balancer?"

- **Answer:** We use a **Floating IP** (or Virtual IP). We have two load balancers. If the primary fails, the secondary takes over the Floating IP address via a protocol like VRRP or Keepalived. The user never sees a change in the IP address.

5. "Describe a scenario where a system is Available but not Reliable."

- **Answer:** Imagine a search engine. You can open the page (it's Available), but no matter what you search for, it returns "No results found" or "Internal Error." The system is up, but it is failing to perform its intended function correctly—thus, it is not Reliable.