🔷 AWS Real-World Scenarios for High-Traffic & Load Balancing (Q&As)

• Section 1: Website Scaling & Load Balancing

Q1. Your website is facing high traffic and crashing — what's your first step?
Answer: Add an **Elastic Load Balancer (ALB)** and enable **Auto Scaling** for EC2 instances.
💡 Tip: Highlight elasticity + high availability.

Q2. How do you ensure even traffic distribution across instances?
Answer: Use **Application Load Balancer (ALB)** for HTTP/HTTPS or **NLB** for TCP.
💡 Tip: Mention sticky sessions if needed.

Q3. What if one EC2 instance fails?
Answer: Auto Scaling replaces unhealthy instances automatically.
💡 Tip: Tie health checks to ALB.

Q4. How do you handle sudden traffic spikes (e.g., flash sale)?
Answer: Configure Auto Scaling with **dynamic policies** based on CPU/requests.
💡 Tip: Add CloudFront CDN for caching.

Q5. How do you improve global traffic performance?
Answer: Use **Route 53 + CloudFront CDN** for geo-routing + caching.
💡 Tip: Always mention latency-based routing.

• Section 2 : Databases & Storage

Q6. Your DB is overloaded with read queries — solution?
Answer: Add **RDS Read Replicas** or **ElastiCache**.
💡 Tip: Mention read/write splitting.

Q7. How do you scale RDS vertically vs horizontally?
Answer: Vertically → increase instance size.
Horizontally → use read replicas/sharding.
💡 Tip: Horizontal scaling = preferred for big traffic.

Q8. What if writes are bottlenecking your database?
Answer: Use **Aurora Multi-Master** or **DynamoDB** for scale-out writes.
💡 Tip: DynamoDB scales better for unpredictable workloads.

Q9. How do you handle millions of file uploads daily?
Answer: Store in **S3**, use **pre-signed URLs**.
💡 Tip: Offload storage away from EC2.

Q10. How to serve static website files efficiently?
Answer: Host on **S3 + CloudFront**.
💡 Tip: Cost-effective + fast.

• Section 3 : Security & Reliability

Q11. How do you secure a high-traffic web app?
Answer: Use **WAF + Shield** for DDoS protection.
💡 Tip: Mention IAM least-privilege policies.

Q12. How to isolate production from development traffic?
Answer: Use **separate VPCs** or subnets with strict IAM.
💡 Tip: Mention environment separation.

Q13. How to ensure data durability in case of failure?
Answer: Replicate across **Multi-AZ** (RDS, S3).
💡 Tip: Mention 11 nines durability of S3.

Q14. What if a region goes down?
Answer: Use **Multi-Region deployment** + Route 53 failover.
💡 Tip: Disaster Recovery strategy = important.

Q15. How do you detect unusual spikes or attacks?
Answer: Enable **CloudWatch Alarms** + **GuardDuty**.

💡 Tip: Security + monitoring must be linked.

- ## Section 4 : Monitoring & Optimization

Q16. How do you troubleshoot sudden latency in your app?
Answer: Use **X-Ray** for tracing + **CloudWatch Logs**.
💡 Tip: Mention distributed tracing.

Q17. How to optimize high-cost infrastructure under traffic load?
Answer: Use **Spot Instances + Auto Scaling** for cost savings.
💡 Tip: Blend with On-Demand for reliability.

Q18. How to cache frequently accessed content?
Answer: Use **CloudFront** for edge caching.
Or **ElastiCache (Redis/Memcached)** for DB queries.
💡 Tip: Caching = core AWS optimization.

Q19. How to handle high traffic API requests?
Answer: Use **API Gateway + Lambda** (serverless).
Scales automatically with demand.
💡 Tip: No server mgmt required.

Q20. How to monitor user activity & app performance?
Answer: Use **CloudWatch metrics, dashboards, alarms**.
💡 Tip: Mention proactive monitoring.

-

- ## Section 5: Failover & Disaster Recovery

Q21. How do you ensure zero downtime deployment?
Answer: Use **Blue/Green deployments** with Elastic Beanstalk/CodeDeploy.
💡 Tip: Mention Canary as alternative.

Q22. How to recover from a DB crash?
Answer: Use **automated backups + snapshots**.

Restore RDS from latest snapshot.
💡 Tip: Add Multi-AZ for higher availability.

Q23. How to prepare for disaster recovery in AWS?
Answer: Define **RTO + RPO**.
Implement **Pilot Light or Multi-Site** strategy.
💡 Tip: Interviewers love DR strategies.

Q24. How to prevent one AZ failure from crashing app?
Answer: Deploy across **Multi-AZ** with ALB + Auto Scaling.
💡 Tip: Fault tolerance = must.

Q25. How to scale messaging under heavy load?
Answer: Use **SQS (queue)** or **SNS (pub-sub)**.
Decouples app components.
💡 Tip: Improves resilience under spikes.

• Section 6: Real-World Business Scenarios

Q26. How would you handle millions of concurrent users?
Answer: Multi-Region, Auto Scaling, CloudFront, DynamoDB.
💡 Tip: Mention “horizontal scaling + CDN”.

Q27. How do you architect an e-commerce app on AWS?
Answer: ALB + Auto Scaling EC2, RDS Multi-AZ, ElastiCache, S3 + CloudFront.
💡 Tip: Sketching the architecture diagram = bonus.

Q28. How do you reduce downtime during maintenance?
Answer: Use **Rolling Updates** with Auto Scaling.
Or **Blue/Green deployments**.
💡 Tip: Continuous delivery best practice.

Q29. How do you handle mobile app backend traffic spikes?
Answer: Use **API Gateway + Lambda + DynamoDB**.
Serverless scales seamlessly.
💡 Tip: Mention pay-per-use efficiency.

Q30. How to design a high-availability blogging platform?

Answer: S3 + CloudFront for static assets.

ALB + EC2 + RDS Multi-AZ for dynamic content.

💡 Tip: Add caching for comments (ElastiCache).