Good question 👍. In AWS/DevOps interviews, **“Why” questions** are very common because they test your **practical reasoning** (not just definitions). For **5 years experience**, interviewers expect you to explain **why you chose one AWS service over another** in real-time projects.

Here’s a list of **frequently asked “Why” questions in AWS interviews** with **how to answer them**:

**🔹 EC2 & Compute**

**Q1: Why did you choose EC2 instead of on-premises servers?**  
👉 Because EC2 provides on-demand, scalable, and pay-as-you-go compute without upfront hardware costs. We can scale up/down in minutes, unlike physical servers.

**Q2: Why did you choose Auto Scaling instead of manually adding instances?**  
👉 Auto Scaling automatically adjusts instance count based on demand (CPU, memory, custom metrics). This ensures cost optimization + high availability without manual intervention.

**Q3: Why did you choose Spot Instances vs On-Demand?**  
👉 Spot is **cheap (up to 90% savings)** for non-critical workloads (batch jobs, testing), whereas On-Demand is used for **production workloads** where guaranteed availability is required.

**🔹 Storage (S3, EBS, EFS)**

**Q4: Why S3 instead of EBS/EFS?**  
👉 S3 is object storage → best for static files, backups, and data lakes.  
👉 EBS is block storage → used when you need OS-level access like databases.  
👉 EFS is file storage → best for shared access across multiple EC2 instances.

**Q5: Why S3 Standard vs S3 Glacier?**  
👉 Standard = Frequently accessed data (web apps, images).  
👉 Glacier = Archival storage, cheap but retrieval is slow (good for compliance, logs).

**🔹 Networking (VPC, ELB, Route 53)**

**Q6: Why did you create a VPC instead of using the default VPC?**  
👉 Custom VPC gives control over CIDR ranges, subnets, route tables, NAT Gateways, and security – which is critical for production environments. Default VPC is only for quick testing.

**Q7: Why use ALB instead of NLB?**  
👉 ALB is best for HTTP/HTTPS traffic (Layer 7), supports routing rules, host-based routing, and SSL termination.  
👉 NLB is best for TCP/UDP traffic (Layer 4) and when we need **static IPs** + ultra-high performance.

**Q8: Why Route 53 instead of GoDaddy/other DNS?**  
👉 Route 53 integrates deeply with AWS, supports health checks, latency-based routing, and automatic failover – which third-party DNS often lacks.

**🔹 Security (IAM, KMS, WAF)**

**Q9: Why IAM roles instead of hardcoding access keys?**  
👉 Hardcoding = insecure + risk of credential leaks.  
👉 IAM Role = temporary credentials, auto-rotated, more secure.

**Q10: Why use KMS for encryption instead of managing your own keys?**  
👉 KMS provides secure, centralized key management, automatic rotation, and AWS compliance (PCI-DSS, HIPAA). Managing your own keys increases risk + operational overhead.

**🔹 Monitoring & Automation (CloudWatch, CloudTrail, Lambda)**

**Q11: Why CloudWatch instead of third-party monitoring?**  
👉 CloudWatch is tightly integrated with AWS services (metrics, logs, alarms).  
👉 Third-party tools (Datadog, Grafana) may require agents and extra cost.

**Q12: Why CloudTrail?**  
👉 CloudTrail records **who did what, when, and from where** in AWS → critical for security auditing and compliance.

**Q13: Why Lambda instead of EC2 for small tasks?**  
👉 Lambda = serverless, no need to manage servers, scales automatically, pay-per-use. EC2 is better for long-running apps.

**🔹 Databases (RDS, DynamoDB, Aurora)**

**Q14: Why RDS instead of hosting DB on EC2?**  
👉 RDS automates backups, patching, replication, and scaling. Hosting DB on EC2 means managing everything manually.

**Q15: Why DynamoDB instead of RDS?**  
👉 DynamoDB is NoSQL, serverless, highly scalable for key-value workloads (IoT, gaming, caching). RDS is relational and better for structured data + joins.

**🔹 Content Delivery & Global Services (CloudFront, SNS, SQS)**

**Q16: Why CloudFront instead of just S3?**  
👉 CloudFront caches data in edge locations → reduces latency and improves performance globally. S3 alone may cause high latency for global users.

**Q17: Why SNS vs SQS?**  
👉 SNS = pub/sub (fan-out messages to multiple subscribers at once).  
👉 SQS = message queue (decoupling, one-to-one message delivery).

✅ **Tip for answering “Why” questions:**

* Always compare **two AWS services** or **AWS vs non-AWS approach**.
* Use **real-time project examples**: “In my last project, we used ALB because…”

👉 Bro, do you want me to prepare a **full Q&A list of "Why" questions with sample project-based answers** (so you can explain like you actually implemented them)?

**You said:**

what are the why,when ,which,how and give example type question may expect in an interview

**ChatGPT said:**

🔥 Great point bro! Interviewers **don’t just test definitions**, they test your **decision-making** skills with **Why, When, Which, and How** type questions. These reflect **real-world problem solving** and are very common for **5 years+ AWS/DevOps experience**.

Let’s break it down:

**✅ 1. “Why” Questions (Decision-making, reasoning)**

👉 Focus: Why you chose one AWS service over another.

**Examples:**

* Why did you use **ALB instead of NLB**?
  + *Answer:* ALB supports Layer 7 routing (host/path based). We used it because our app had multiple microservices (api.myapp.com, app.myapp.com) that needed routing.
* Why did you choose **S3 instead of EBS/EFS**?
  + *Answer:* S3 is highly durable (11 9’s), scalable, and cost-effective for object storage. Perfect for storing static files and backups, unlike EBS (block storage tied to one instance).
* Why do we need **Auto Scaling**?
  + *Answer:* To handle unpredictable traffic automatically → reduces cost and ensures high availability.

**✅ 2. “When” Questions (Situational usage)**

👉 Focus: When to use a specific service.

**Examples:**

* When would you use **DynamoDB instead of RDS**?
  + *Answer:* When we need high throughput, low-latency NoSQL data (IoT, gaming leaderboards, session data) without worrying about scaling DB servers.
* When should you use **CloudFront**?
  + *Answer:* When your users are spread globally and you want low-latency access to static/dynamic content.
* When would you use **Spot Instances**?
  + *Answer:* For fault-tolerant, non-critical workloads like batch processing, testing, CI/CD pipelines.

**✅ 3. “Which” Questions (Choosing between options)**

👉 Focus: Compare services and justify selection.

**Examples:**

* Which ELB should you use for **real-time gaming traffic**?
  + *Answer:* **NLB** because it works at Layer 4, supports TCP/UDP, static IPs, and handles millions of requests per second with ultra-low latency.
* Which storage type would you choose for **a database**?
  + *Answer:* **EBS (gp3/Provisioned IOPS)** because DB requires block-level storage with high performance.
* Which AWS service would you use for **decoupling microservices**?
  + *Answer:* **SQS** (message queue) → ensures one service’s failure won’t affect another.

**✅ 4. “How” Questions (Implementation & real-time usage)**

👉 Focus: Explain hands-on, step-by-step process.

**Examples:**

* How do you configure **Route 53 with ALB** for myapp.com?
  + *Answer:*
    1. Create ALB → note DNS name.
    2. In Route 53, create a hosted zone (myapp.com).
    3. Add an **Alias record** pointing myapp.com → ALB DNS.
    4. Now requests to myapp.com route to EC2s behind ALB.
* How do you **secure S3 buckets**?
  + *Answer:* Block public access, use bucket policies + IAM roles, enable server-side encryption (SSE-S3 or SSE-KMS), enable logging, use CloudTrail for access auditing.
* How does **Auto Scaling** work with ELB?
  + *Answer:* ELB distributes traffic → Auto Scaling launches/terminates EC2s → ELB health checks ensure only healthy instances get traffic.

**✅ 5. “Give Example” Questions (Practical, real-time scenarios)**

👉 Focus: Explain actual project experience.

**Examples:**

* Give me an example where you used **CloudWatch alarms**.
  + *Answer:* In one project, we set a CloudWatch alarm when CPU > 80% on EC2 for 5 minutes → it triggered an Auto Scaling policy to add a new instance.
* Give an example of **how you used Lambda**.
  + *Answer:* We used Lambda for image processing → whenever a user uploaded an image to S3, an S3 event triggered Lambda, which resized the image and stored it back in another S3 bucket.
* Give an example of **multi-AZ RDS failover**.
  + *Answer:* In production, our RDS primary failed in AZ-A. The standby in AZ-B automatically became primary within 1 minute, and Route 53 DNS redirected traffic → app had minimal downtime.

✅ **How to answer in interviews:**

1. Always **start with service definition** (1 liner).
2. Then **explain scenario/use case** (real-time).
3. Compare with alternatives (**why not others**).
4. If possible, add **project example** (“In my last project, we used this for…”).