Explain Lambda versioning and aliases. When would you use aliases?

Think of a **version** as a frozen snapshot of your function (code + configuration) and an **alias** as a named bookmark that points to one of those snapshots. Aliases make it safe and practical to release, test, and roll back changes.

Versions — the snapshots

- **What it is:** When you **publish** a version you create an **immutable** (cannot be changed) copy of the function code and configuration (memory, timeout, env vars, etc.).
- **Key points:**
    - Versions are numbered (1, 2, 3, ...).
    - Once published, a version can't be changed. If you change code you must publish a *new* version.
    - There is also $LATEST — the editable working copy you update while developing.

Why use versions?

- You get a stable reference for production traffic and audits.
- You can run tests against a specific version without risking accidental changes.

Aliases — named pointers

What it is: An alias is a friendly name (for example dev, staging, prod, blue, green) that points to a specific published version.

- **Stable ARN:** An alias gives you a stable function ARN like arn:aws:lambda:region:acct:function:MyFunc:prod — callers can invoke the alias instead of a raw version number.
- **Traffic shifting:** Aliases can route a percentage of traffic to a second version (weighted routing), enabling canary or phased rollouts.

Analogy

- **Version = photo of a document.** Once taken, the photo never changes.
- **Alias = sticky note labeled "prod" stuck to one photo.** You can move the sticky note to a new photo when you're ready. You can also tear a tiny corner off a second photo and let 10% of readers see it (weighted traffic).

Typical Use Cases for Aliases

Environment Separation

- Assign aliases like "dev", "test", and "prod" to different versions for each environment, making it easy to manage changes and promotion across environments without breaking integrations.

Stable References for Services

- Use aliases as stable ARNs for event sources (such as S3 or API Gateway), so that consumers or permissions policies do not need updating every time a new Lambda version is deployed.

Deployment Safety and Rollbacks

- Enable canary and blue/green deployments by shifting a percentage of traffic to a new version using alias traffic splitting, allowing careful rollout and easy rollback if issues arise.

Fine-Grained Permissions

- Grant specific AWS services or accounts access to only certain aliases (and thus certain versions), supporting permission management at a granular level.

Why Use an Alias?

- Aliases let engineers update which version is active for a use case (e.g., production)—without modifying consumer settings, event source mappings, or permissions tied to that alias.
- They offer deployment flexibility while ensuring reliability and auditability for production and other environments.

Aliases are especially helpful for teams practicing continuous deployment, automated testing, and needing clear separation of application stages.

In **AWS Lambda aliases**, the **weight** setting is used for **traffic shifting between versions**.

- An **alias** normally points to a single Lambda **version**.
- But you can also configure the alias to **split traffic** between two versions, by assigning weights.

Q1. In AWS Lambda, what does it mean when a version is described as *immutable*?
a) You can update its code and configuration anytime
b) It cannot be changed once published
c) It can only be deleted but not executed
d) It automatically updates when $LATEST changes
✅ **Answer:** b) It cannot be changed once published

Q2. What is the purpose of $LATEST in AWS Lambda?
a) It represents the oldest published version
b) It is an alias pointing to production
c) It is the editable working copy of the function
d) It is the version that receives 100% of traffic by default
✅ **Answer:** c) It is the editable working copy of the function

---

Q3. Which of the following is a **reason to use versions** in AWS Lambda?
a) To allow continuous modification of deployed code
b) To get stable references for audits and production traffic
c) To avoid publishing new code versions
d) To automatically route traffic
✅ **Answer:** b) To get stable references for audits and production traffic

---

Q4. Which of the following best describes an **alias** in AWS Lambda?
a) A permanent number assigned to a function
b) A pointer to $LATEST only
c) A friendly name that points to a specific version
d) A copy of function code with traffic shifting disabled
✅ **Answer:** c) A friendly name that points to a specific version

---

Q5. Which use case is **NOT** correct for aliases?
a) Environment separation (dev, test, prod)
b) Stable ARNs for event sources
c) Traffic shifting for canary deployments
d) Updating code directly in production without versioning
✅ **Answer:** d) Updating code directly in production without versioning

---

Q6. How do aliases help with **deployment safety**?
a) By blocking changes to function code
b) By splitting traffic between two versions for canary or blue/green rollout
c) By automatically rolling back to $LATEST on failure
d) By deleting old versions during deployment
✅ **Answer:** b) By splitting traffic between two versions for canary or blue/green rollout

---

Q7. What happens if you want to deploy new code to production but keep the same ARN used by consumers?
a) You must reconfigure all event sources to point to the new version
b) You update the alias (e.g., “prod”) to point to the new version
c) You overwrite the old version with new code
d) You can only use $LATEST
✅ **Answer:** b) You update the alias (e.g., “prod”) to point to the new version

---