DVA-C02 Cheat Sheet

High-yield AWS Developer Associate review for the last pass before practice tests.

BEST FOR	Last review before mini test or app session
FOCUS	Serverless patterns, security, CI/CD, and troubleshooting
USE WITH	Quick Summary + Service Map

1. Domain weights and mindset

32 / 26 / 24 / 18 — Development, Security, Deployment, Troubleshooting are the current scored domain weights.
Think like a builder — Prefer managed services, event-driven patterns, retries, idempotency, and least-ops choices.
Code + AWS service fit — Questions often test the right integration pattern rather than raw syntax.
Observability matters — Expect logs, traces, metrics, alarms, and structured troubleshooting to appear often.
Scope discipline — Some services are explicitly out of scope, so do not overfit to advanced platform services.

2. Lambda and event-driven core

Timeout / memory / concurrency — Tune Lambda with the right runtime settings before guessing the service is wrong.
DLQ vs destinations — Both handle failed invocations, but destinations are richer for async workflows and routing.
EventBridge vs SNS vs SQS — EventBridge for routing by pattern; SNS for fanout push; SQS for decoupled pull and buffering.
Idempotency — Critical for retries, duplicate events, and at-least-once delivery patterns.
VPC access — Attach Lambda to subnets only when private resource access is needed because cold starts and networking can change.

3. Data stores and performance

DynamoDB keys — Partition key design drives scale, latency, and hot-partition risk.
Query vs Scan — Query is targeted and cheaper; scan reads broadly and should raise suspicion.
TTL and lifecycle — Use retention and lifecycle controls instead of manual cleanup logic when possible.
Caching — ElastiCache and app-side caching help reduce repeated database or API reads.
OpenSearch fit — Choose it for search or analytics-style access patterns, not generic transactional storage.

4. Security in application code

Cognito — Common choice for user sign-up, sign-in, tokens, and federation in app scenarios.
IAM roles — Prefer temporary credentials through roles rather than embedded long-term access keys.
Secrets Manager vs Parameter Store — Secrets Manager is the stronger default for rotating secrets and credentials.
KMS — Know when data is encrypted at rest automatically versus when code or config must call KMS directly.
Data masking — Protect logs, events, and payloads from leaking tenant or sensitive information.

5. CI/CD and release patterns

CodePipeline — Coordinates stages across source, build, test, and deploy.
CodeBuild — Build and test environment; often paired with artifact generation and unit/integration checks.
Blue/green vs canary — Both reduce release risk; canary shifts partial traffic, blue/green swaps full environments.
IaC — CloudFormation and CDK are exam-friendly defaults for repeatable environments.
Config per environment — AppConfig and parameterization help avoid hard-coded environment differences.

6. Troubleshooting and optimization

CloudWatch logs/metrics — First stop for runtime failures, throttling, latency, and alarm-based clues.
X-Ray — Use for request tracing across distributed components and latency bottlenecks.
Structured logging — Makes filtering and root-cause analysis faster than free-form logs.
Concurrency — Reserved or provisioned concurrency can solve startup behavior and noisy-neighbor issues.
Health checks — Readiness and health endpoints matter for containers, load balancers, and safe deployments.