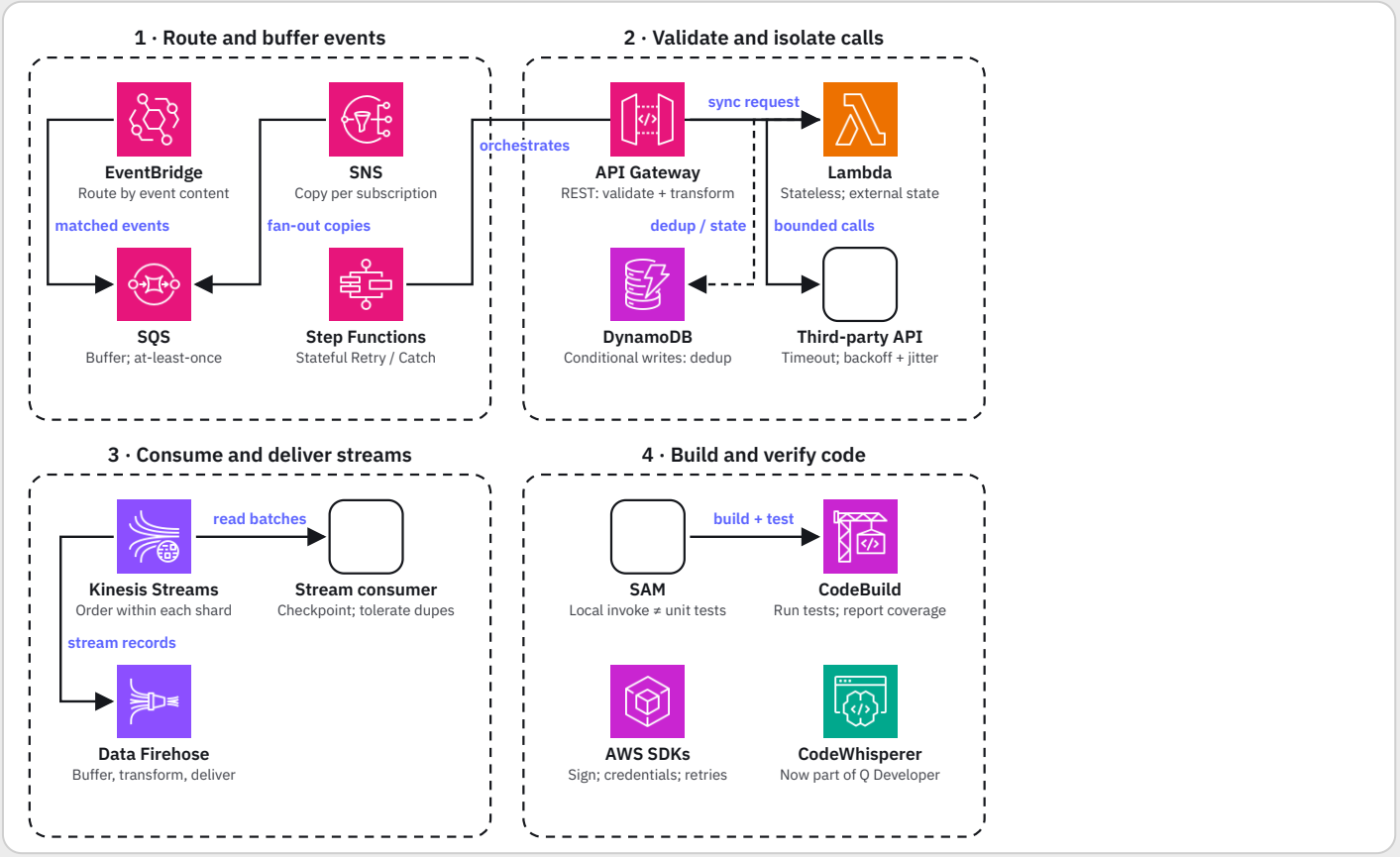
DOMAIN 1 · DEVELOPMENT WITH AWS SERVICES · 32% OF THE EXAM · TASK STATEMENT 1.1

Develop code for applications hosted on AWS

Route, isolate, stream, verify



1 Route and buffer events

Choose **orchestration** for central workflow state; event-driven choreography has no central coordinator. SNS fan-out to separate SQS queues gives each consumer a durable copy; one shared queue distributes work.

2 Validate and isolate calls

Use **stateless handlers** with external session state; stickiness is not durability. REST mappings can override response status. Bound retries; circuit breakers fail fast and allow safe degraded responses.

3 Consume and deliver streams

Choose **Kinesis Data Streams** for independently replayable consumers; partition keys determine shard placement, not global order. Checkpoint after successful processing. Firehose buffers and delivers to destinations.

4 Build and verify code

Use **mocked SDK calls** for isolated unit tests; SAM invokes locally and CodeBuild runs CI. Code must follow pagination tokens and handle service errors. Review and test Q-generated code before accepting it.

Exam traps

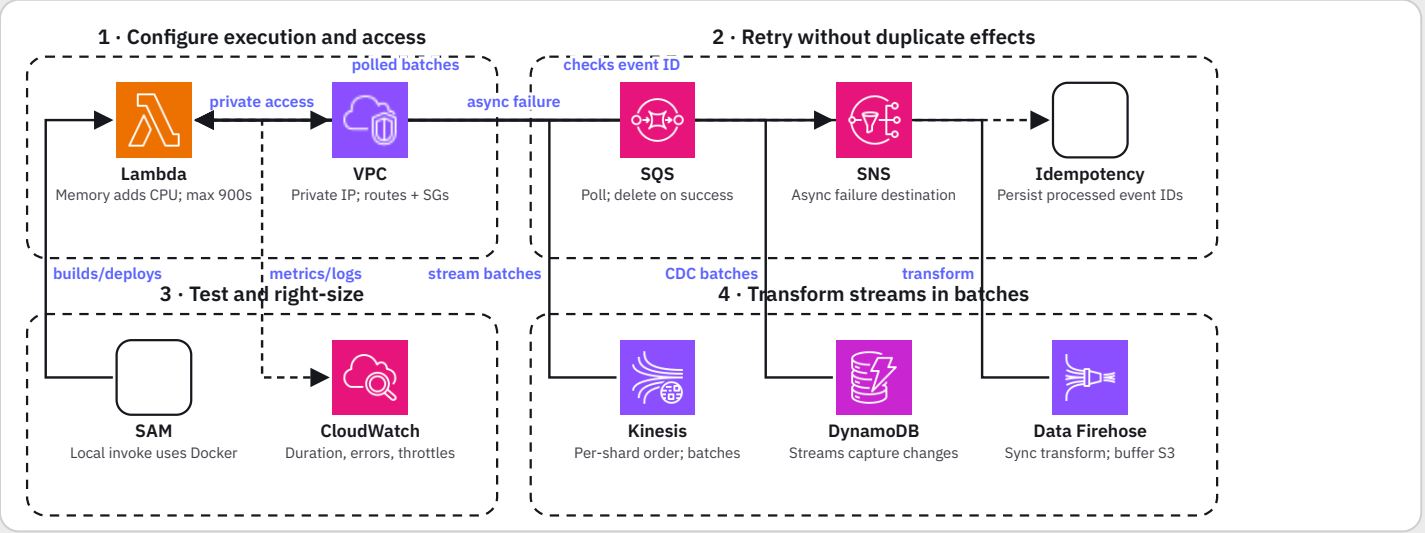
- **SQS FIFO** orders messages within a message group and deduplicates sends; it does not guarantee exactly-once business side effects. Standard queues can duplicate and reorder.
- **Synchronous Lambda** returns errors without automatic retries. SQS failures cause redelivery after visibility expires; partial batch responses prevent successful records from being retried.
- **Loose coupling** permits independent releases and failure isolation, not automatic consistency. Preserve API compatibility or version breaking changes; high test coverage does not prove correctness.

Get practice exams and lessons for AWS Certified Developer - Associate

DOMAIN 1 · DEVELOPMENT WITH AWS SERVICES · TASK STATEMENT 1.2

Develop code for AWS Lambda

Configure, recover, test, transform



1 Configure execution and access

An **execution role** grants API access; VPC routes/SGs control reachability. Private IPv4 internet egress needs NAT. Environment variables are KMS-encrypted at rest; AWS Systems Manager Parameter Store holds external config.

2 Retry without duplicate effects

For **async failures**, destinations carry invocation details; DLQs retain the original event. SQS retries after visibility timeout; partial batch responses identify failed items. Retry transient dependencies with backoff and jitter.

3 Test and right-size

Use **mocking** for unit tests; SAM local does not validate cloud IAM/networking. Run deployed integration tests in AWS CodeBuild. Profile with AWS X-Ray; right-size memory and reuse clients/connections outside the handler.

4 Transform streams in batches

For **stream failures**, failed batches block shard progress by default. Partial batch responses and bisecting isolate failures; retries can replay records. Larger batches amortize overhead but increase buffering latency.

Exam traps

- **A public subnet does not give Lambda a public IP**. An internet-gateway route alone cannot provide IPv4 internet access; use private-subnet routing through NAT.
- **Reserved concurrency is not prewarming**: it reserves and caps concurrency. Provisioned concurrency prepares execution environments to reduce cold-start latency.
- **Amazon API Gateway invokes synchronously**: Lambda does not retry function errors. Amazon S3 invokes asynchronously; Lambda retries function errors twice by default.

Get practice exams and lessons for AWS Certified Developer - Associate

AWS Certified Developer - Associate Cheat Sheet

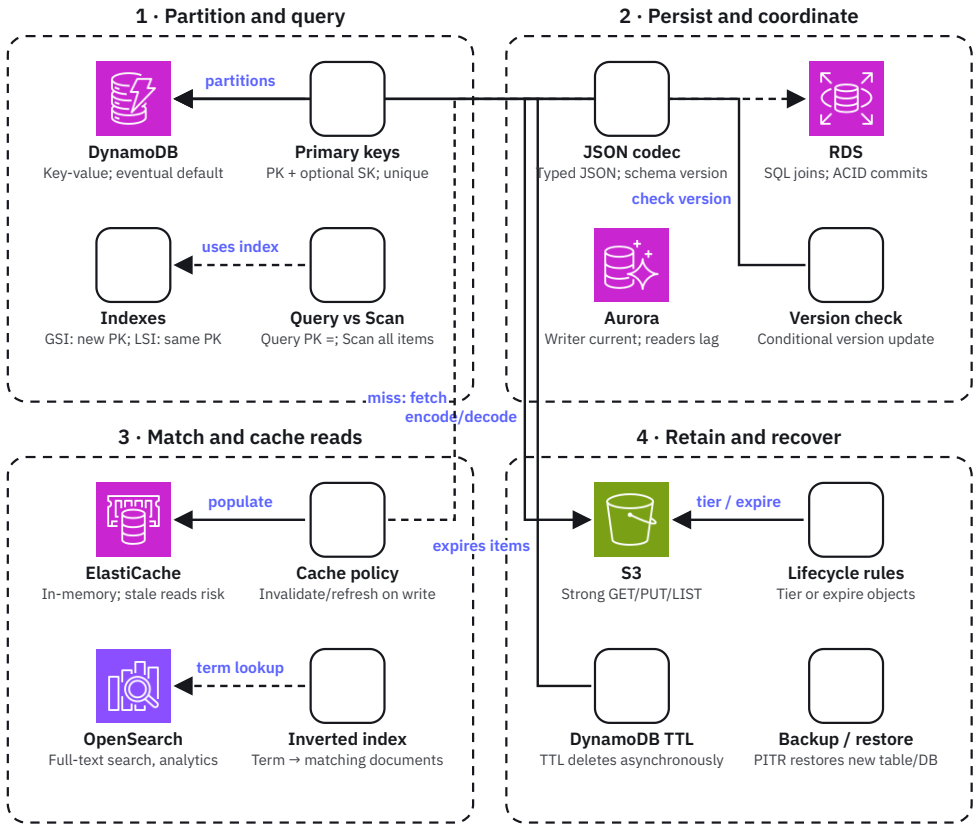
DVA-C02 · Domain 1 of 4

BlowStack

DOMAIN 1 · DEVELOPMENT WITH AWS SERVICES · TASK STATEMENT 1.3

Use data stores in application development

Partition, persist, cache, retain



1 Partition and query

Hot key: high-cardinality partition keys need evenly distributed access. Salt hot keys to spread writes; reads must combine results across salts. Indexes accelerate access but add storage and write overhead.

2 Persist and coordinate

Lost updates: condition writes on the stored version; retry conflicts. DynamoDB transactions atomically change multiple items. Serialize on write, deserialize on read; version schemas so readers can interpret older records.

3 Match and cache reads

Cache-aside: a miss fetches from the store and populates cache. Write-through updates store and cache on writes, adding latency and caching unused data. TTL and invalidation limit staleness; they do not eliminate it.

4 Retain and recover

Retention: S3 Lifecycle transitions storage classes or expires objects; versioned buckets need noncurrent-version rules. Enable DynamoDB PITR or RDS/Aurora automated backups; restore creates a new table/DB.

Exam traps

- **Query and Scan** page at 1 MB before filtering. Filters do not reduce read capacity consumed. Follow LastEvaluatedKey even after an empty result page; Query requires partition-key equality.
- **Strong reads** are optional on DynamoDB tables/LSIs, never GSIs; default reads are eventual. RDS/Aurora reader replicas may lag; use the writer for read-after-write.
- **DynamoDB TTL** uses epoch seconds and deletes asynchronously, usually within days; expired items remain readable until deletion. It is not an exact timer. S3 Lifecycle expiration is asynchronous too.

Get practice exams and lessons for AWS Certified Developer - Associate

DOMAIN 1 · DEVELOPMENT WITH AWS SERVICES · WHAT TO REMEMBER

Concepts and services of Development with AWS Services

Concepts to know

Idempotency

Idempotency makes repeated processing produce the same effect as one successful operation. At-least-once delivery requires **duplicate-safe consumers**; conditional writes with an event identifier can prevent repeated business updates.

Loose Coupling

Loose coupling separates components through contracts rather than shared implementation or execution timing. Choose **queues for buffered work**, publish-subscribe for independent subscribers, and orchestration for centrally coordinated workflows.

Database Indexing

Indexes provide alternate access paths at a storage and write cost. In DynamoDB, choose a **GSI for a different partition key**; an LSI retains the table partition key, must exist at table creation, and supports strongly consistent reads.

Core services



AWS Lambda

More memory also allocates more CPU. Choose **provisioned concurrency for preinitialized environments**; reserved concurrency reserves capacity and caps a function's concurrent executions.



Amazon DynamoDB

Choose **Query for a known partition key**, not Scan. Filters do not reduce read capacity consumed. Strongly consistent reads are available on tables and LSIs, but not GSIs.



Amazon Simple Queue Service (Amazon SQS)

Choose **FIFO for ordering within message groups**. Visibility timeout hides in-flight messages; delete after success. A redrive policy moves repeatedly received messages to a dead-letter queue.