## 1. CloudWatch Metrics & Custom Metrics

- **Namespaces & Metrics:**

*Namespace:* A logical container that isolates metrics from different applications.
*Metric:* A time-ordered set of data points (e.g., CPU usage, page view counts).

- **Dimensions & Resolution:**

*Dimensions:* Key-value pairs that add context (e.g., InstanceId, VehicleID).
*Resolution:*
**Standard:** 1-minute granularity (default for AWS metrics).
**High-resolution:** Up to 1-second granularity for detailed monitoring.

- **Publishing Custom Metrics:**

Use the AWS CLI or API (e.g., `aws cloudwatch put-metric-data`) to send metrics.
Metrics can be aggregated and later retrieved using commands like `get-metric-statistics`.

- **Metric Retention:**

Data points are stored with different resolutions for varying durations (e.g., 1-minute data for 15 days, 1-hour data for 15 months).

## 2. Data Aggregation, Alarm Creation & Anomaly Detection

- **Data Aggregation & Statistics:**

Raw data can be aggregated into statistic sets (e.g., Sum, Average, Minimum, Maximum).
Aggregation reduces API calls and simplifies analysis.

- **Alarms:**

*Static Alarms:* Triggered when a metric crosses a fixed threshold.
*Anomaly Detection Alarms:* Use machine learning to compare current metric values against expected baselines; the alarm state is based on deviations rather than static numbers.

- **Cross-Account & Cross-Region Monitoring:**

Enable centralized monitoring across multiple AWS accounts and Regions using IAM roles (e.g., `CloudWatch-CrossAccountSharingRole` and `AWSServiceRoleForCloudWatchCrossAccount`).

## 3. CloudWatch Metrics Insights

- **Metrics Insights Query Language:**

Use an SQL-based language to query up to 10,000 metrics at scale.
Automatically adapts to new resources so that alarms can track metrics dynamically across resources.

- **Creating Insights Alarms:**

Build complex queries that monitor aggregate metrics (e.g., average CPU utilization across an entire EC2 fleet).
Set alarms based on the query results rather than individual metrics.

## 4. Container & Lambda Insights

**CloudWatch Container Insights:**

- **Purpose:** Monitors containerized applications on Amazon ECS, EKS, and Kubernetes.
- **Features:**

Provides cluster-wide, node-level, and container-level views.
Supports troubleshooting through dashboards, graphs, and logs.

- **Setup:**

Enable via the ECS console or update-cluster-settings command.
For EKS, use the CloudWatch Observability EKS add-on.

**CloudWatch Lambda Insights:**

- **Purpose:** Offers deep observability into AWS Lambda functions.
- **Features:**

Collects system-level metrics (CPU, memory, network) and diagnostic data (cold starts, errors).
Supports both multi-function overviews and detailed single-function views.

- **Setup:**

Enable via the Lambda console or AWS CLI; may require attaching specific IAM policies and adding the LambdaInsightsExtension layer.

## 5. Internet & Network Monitoring

**CloudWatch Internet Monitor:**

- **Purpose:** Provides a global view of internet-facing traffic performance and availability.
- **Features:**

Monitors client locations (city-networks) and ISPs (ASNs) to detect health events.
Offers metrics such as performance scores, availability scores, round-trip time, and bytes transferred.

- **Usage:**

Create monitors to associate your AWS resources (e.g., VPCs, CloudFront distributions) and get alerts for connectivity issues.

**CloudWatch Network Monitor:**

- **Purpose:** Focuses on monitoring hybrid network connections between AWS and on-premises resources.
- **Features:**

Uses agentless probes to measure latency and packet loss.
Publishes metrics and a Network Health Indicator (NHI) to quickly pinpoint degradation in network paths.

- **Usage:**

Set up probes from AWS subnets to your on-premises IP addresses and create dashboards/alarms based on round-trip time and packet loss metrics.