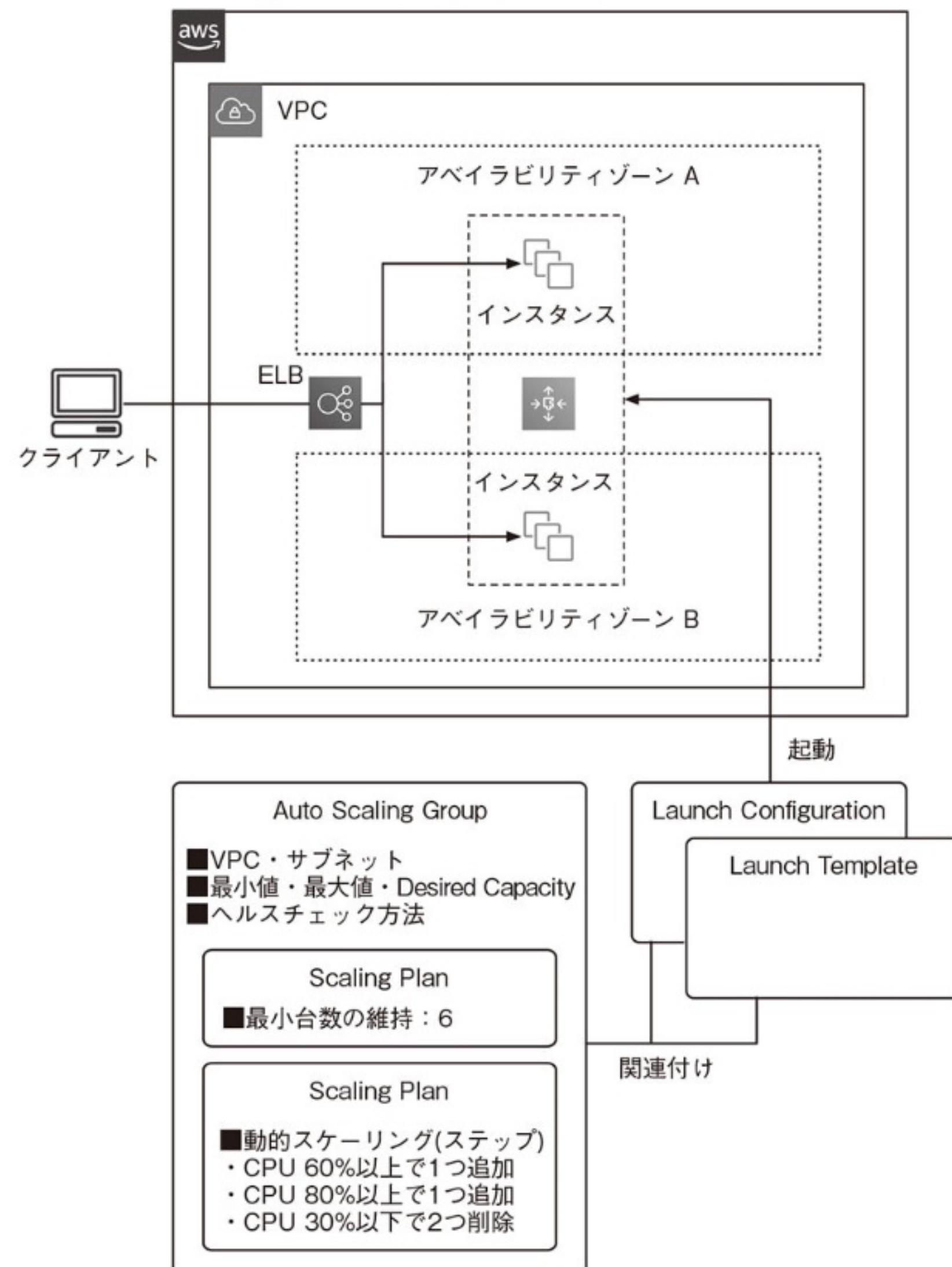


次の図は、EC2 Auto Scalingと構成要素がどのように関わるかを示した図です。複数のScaling Planを持つAuto Scaling GroupがLaunch ConfigurationやLaunch Templateと関連付けられて、インスタンスを起動することを示しています。

### 【EC2 Auto Scalingと構成要素のイメージ】



代表的なScaling Planは次のとおりです。

#### ●最小台数の維持

Auto Scaling Groupに設定した配置するインスタンスの最小値を維持するオプションで、いわゆるAuto Healingを実現します。インスタンスに発生した障害がヘルスチェックで検出されると、自動的に障害インスタンスを切り離します。その後、Launch Configurationで定義したルールにもとづき、新たなインスタンスを追加します。

#### ●手動スケーリング

Auto Scaling GroupのDesired Capacity設定を、障害対応時などに手動で変更したい場合に設定するオプションです。また起動済みのインスタンスを手動でアタッチ・デタッチすることも可能です。

#### ●スケジューリング

指定した日時や定時実行スケジュールで自動的にスケールを行うオプションです。スケーリングが完了するまで猶予時間があるため、それを見越した実行スケジュールの設定が必要になります。

#### ●動的スケーリング

CloudWatchで監視しているリアルタイムのメトリクスと、あらかじめ定義したスケーリングポリシーのルールを評価して動的にスケーリングを行うオプションです。スケーリングポリシーでは、次の3つを選択できます。

- ・シンプルスケーリングポリシー：1つのメトリクスが条件を満たすとスケールする
- ・ステップスケーリングポリシー：複数の条件を定義でき、段階的にスケーリングを行う
- ・ターゲット追跡スケーリングポリシー：定められたメトリクスを維持するようEC2インスタンス数を調整する

#### ●予測スケーリング

2週間分のメトリクスを分析し、時間帯別の需要を予測して自動的にスケールを行うプランです。24時間ごとに次の48時間の予測値を作成し、キャパシティの増減をスケジュールします。予測を開始するには最低24時間分のデータが必要です。

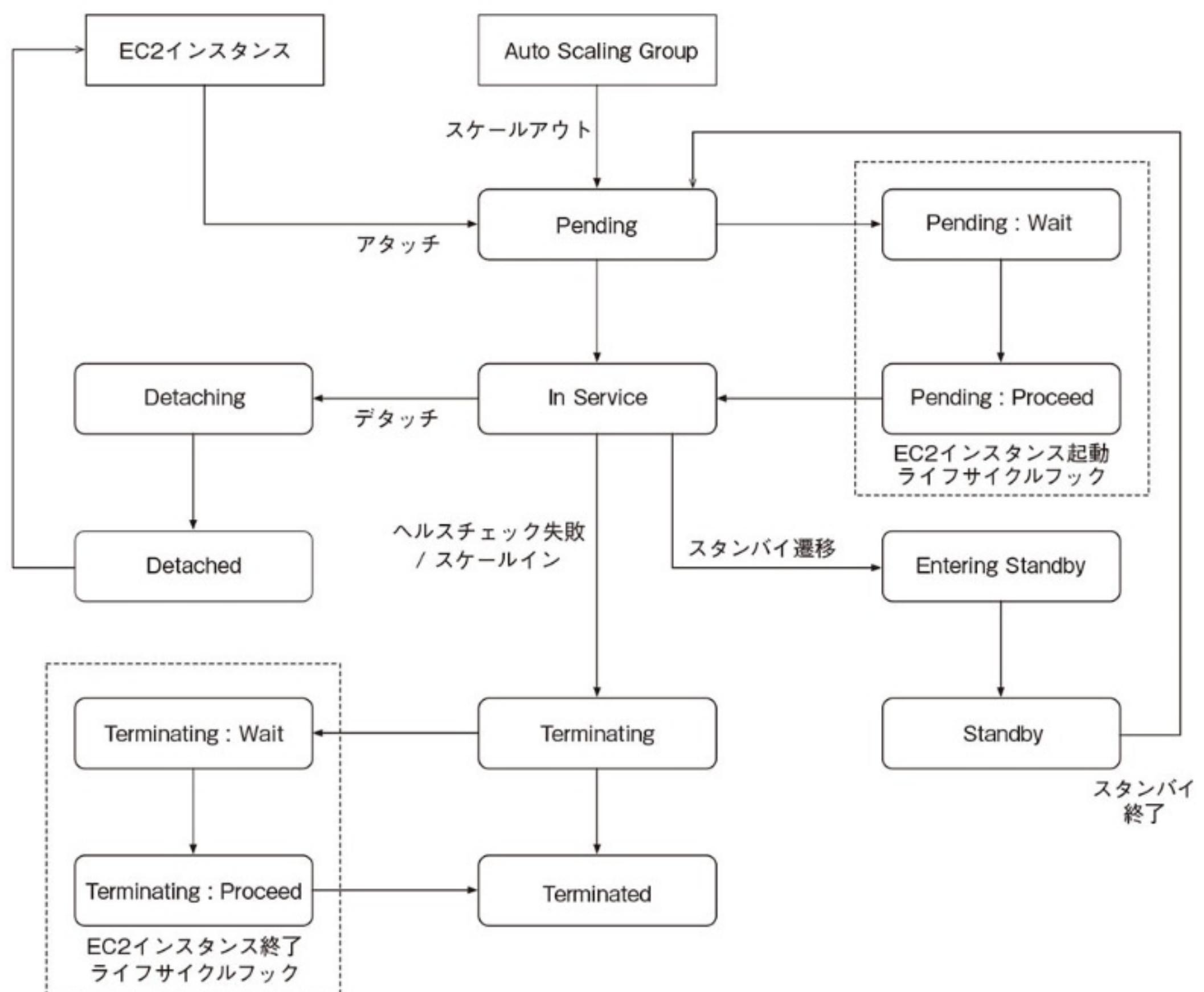


ターゲット追跡スケーリングポリシーでは、CPU使用率のほか、ネットワークI/O、ALBで設定したターゲットグループのターゲットごとの完了リクエスト数などをメトリクスに指定できます。また、新しく起動されたインスタンスのウォームアップにかかる秒数を指定できます。指定されたウォームアップ期間が終了するまで、そのインスタンスは収集メトリクス対象から除外されます。そのため、アプリケーション起動処理などで高い負荷がかかるなどして、余剰にインスタンスが追加されるリスクを軽減できます。

## 2 スケーリングされたインスタンスのライフサイクル

Auto Scaling Groupに組み込まれたインスタンスにはすべて何らかのステータスが付与され、各ステータスでさまざまなイベントやアクションが実行されます。こうした起動時から終了までに起こるインスタンス状態の遷移のことを、「インスタンスのライフサイクル」といいます。Scaling Planにもとづいてインスタンスがスケーリングされる場合、インスタンスのステータスは次の図のように遷移します。

【EC2 Auto Scalingにおけるインスタンス状態の遷移<sup>※53</sup>】



各ステータスの説明は次のとおりです。インスタンスのステータスが変化することで、どのようなアクションやイベントが発生し得るのか、正確に意味を押さえておくようしましょう。

### ● Pending

インスタンスの起動や初期化処理を行っている段階です。通常は初期化処理が完了すると次のInServiceステータスに遷移しますが、オプションとして、インスタンス起動時にカスタムアクションを実行するライフサイクルフックを追加できます。ライフサイクルフックが追加されると、Pending:Waitへ移行し、指定したハートビートタイムアウトの時間までステータスが保留します。その間、カスタムアクションを実行します。手動でインスタンスにログインし、コマンド実行してもよいですし、通知を受けて、EventBridgeを経由して、Lambda関数やSystems Manager RunCommandなどでコマンド実行することもできます。アクションが完了したら、CompleteLifecycleActionコマンドを実行し、Pending:Proceedに遷移させます。ヘルスチェックが成功するとAuto Scaling Groupへ追加されて、InServiceへ遷移します。

なお、ハートビートタイムアウト時間をカスタムアクションの処理中に延長したい場合は、RecordLifecycleActionHeartbeatコマンドを実行します。

### ● InService

インスタンスが正常起動されている状態です。以下のイベントが発生するまでServiceはこの状態を保ち続けます。

- Scaling Planにもとづいてスケールインが発生する
- ユーザによりスタンバイ操作が実行される
- ユーザによりインスタンスがデタッチされる
- 何らかの理由でインスタンスに障害が発生し、インスタンスのヘルスチェックが失敗する

### ● Terminating

スケールインやヘルスチェックの失敗通知により、インスタンスの終了処理を行っている段階です。Auto Scalingの設定で、インスタンスが終了したときにカスタムアクションを実行するためのライフサイクルフックを追加できます。Pending:Waitと同様、カスタムアクションを実行したのち、Terminating:Waitへ移行し、アクションが完了するとTerminating:Proceedに遷移します。インスタンスが完全に終了するとTerminatedへ遷移します。

### ● Terminated

インスタンスが終了した状態です。

### ● Detaching

インスタンスがユーザからの操作によりAuto Scaling Groupからデタッチ処理されている状態です。

※53 [https://docs.aws.amazon.com/ja\\_jp/autoscaling/ec2/userguide/AutoScalingGroupLifecycle.html](https://docs.aws.amazon.com/ja_jp/autoscaling/ec2/userguide/AutoScalingGroupLifecycle.html)

### ● Detached

インスタンスのデタッチが完了した状態です。Auto Scaling Groupからは外れているもののインスタンス自体は起動されたままとなります。

### ● Entering Standby

インスタンスがユーザからの操作によりStandbyへ移行されている状態です。Standby状態ではトラブルシューティングや変更を加えてから、再びアタッチせずにInServiceに戻すことができます。

### ● Standby

インスタンスがAuto Scaling Groupで管理されながらも一時的に削除されている状態です。



EC2 Auto Scalingでは、2021年6月にEC2 インスタンスを事前に初期化してプールしておくウォームプール機能が実装されました。従来、オートスケール要求があってから初期化を行い、起動まで要していた時間がこの機能で大幅に削減できます。ただし、プール期間中も起動と同様の費用が必要になるので、トラフィックの増加が確実に見込まれ、かつ速やかなスケールが必要な場合に利用を検討したほうがよいでしょう。

## 3 Application Auto Scaling

Application Auto ScalingはAPIリクエストの需要数にもとづき、AWSの各サービスでスケーリングする機能です。以下のサービスでAuto Scalingがサポートされています。

### ● AppStream 2.0フリート

AppStream 2.0フリートは、完全マネージド型のデスクトップアプリケーションストリーミングサービスです。ストリーミングを実行しているインスタンス群をフリートと呼び、ユーザ数が増加すると自動的にフリートがスケールアウトします<sup>\*54</sup>。

### ● Aurora DBクラスタ

2章8.3節「Amazon Aurora」でも解説したとおり、Auroraリードレプリカは、メトリクスに応じてスケーリングします。Auroraはターゲット追跡スケーリングポリシーを使用して、最大・最小レプリカ数を定義して調整します。Auto Scaling設定では、CloudWatchアラームを作成し、CPU使用率など事前定義されたメトリ

\*54 [https://docs.aws.amazon.com/ja\\_jp/appstream2/latest/developerguide/autoscaling.html](https://docs.aws.amazon.com/ja_jp/appstream2/latest/developerguide/autoscaling.html)

クスまたはカスタムメトリクスとそのメトリクスのターゲット値を指定します。メトリクスとターゲット値にもとづいて、最小・最大値の範囲内でスケーリングが行われます<sup>\*55</sup>。

### ● Amazon Comprehend

Amazon Comprehendは機械学習を使用してテキスト内で固有名詞・キーワード抽出や感情分析を行う自然言語処理サービスです。ドキュメントの分類エンドポイントとエンティティ認識のエンドポイント用にプロビジョニングされた推論単位の数をスケーリングできます。ターゲット追跡スケーリングポリシーおよびスケジューリングによるスケーリングが可能です<sup>\*56</sup>。

### ● DynamoDB テーブルとグローバルセカンダリインデックス

2章3.3節「キャパシティユニット」では、テーブルの読み書きのパフォーマンスをスループットとして定義するプロビジョンドモードおよびキャパシティユニットについて説明しました。DynamoDBでは読み込み・書き込みキャパシティユニットに対し、ターゲット追跡スケーリングポリシーを使用して、最大・最小キャパシティユニット数を定義してスケーリングの調整を行います。Auto Scaling設定では、CloudWatchアラームを作成し、キャパシティユニットの使用率ターゲット値を指定します。メトリクスとターゲット値にもとづいて、最小・最大値の範囲内でスケーリングが行われます。この設定は、テーブルおよびグローバルセカンダリインデックス（オプション）が対象となります<sup>\*57</sup>。

### ● Amazon EMR (Elastic Map Reduce) クラスタ

EMRは、オープンソースのApache Spark、Apache Hive、Apache HBase、Apache Flink、Apache Hudi、Prestoといったビッグデータ処理プラットフォームのサービスです。EMRのAuto Scalingでは、カスタムポリシーを使用してコアノードとタスクノードに対しスケーリング設定します。カスタムポリシーでは、ノードで使用する最大・最小インスタンス数を定義して調整します。Auto Scaling設定は、CloudWatchアラームを作成し、CPU使用率などカスタムメトリクスとそのメトリクスのターゲット値を指定します。メトリクスとターゲット値にもとづいて、最小・最大値の範囲内でスケーリングが行われます<sup>\*58</sup>。

### ● Amazon ECSサービス

2章7.3節「ECSタスクとサービス」で解説したとおり、ECSサービスを実行する際に、AutoScaling設定をオプションで選択できます。ECSではターゲット追跡スケーリングポリシーもしくはステップスケーリングポリシーを使用して、最大・

\*55 [https://docs.aws.amazon.com/ja\\_jp/AmazonRDS/latest/AuroraUserGuide/Aurora.Integrating.AutoScale.html](https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/AuroraUserGuide/Aurora.Integrating.AutoScale.html)

\*56 [https://docs.aws.amazon.com/ja\\_jp/comprehend/latest/dg/comprehend-autoscaling.html](https://docs.aws.amazon.com/ja_jp/comprehend/latest/dg/comprehend-autoscaling.html)

\*57 [https://docs.aws.amazon.com/ja\\_jp/amazondynamodb/latest/developerguide/AutoScaling.html](https://docs.aws.amazon.com/ja_jp/amazondynamodb/latest/developerguide/AutoScaling.html)

\*58 [https://docs.aws.amazon.com/ja\\_jp/emr/latest/ManagementGuide/emr-automatic-scaling.html](https://docs.aws.amazon.com/ja_jp/emr/latest/ManagementGuide/emr-automatic-scaling.html)

最小タスク数を定義してスケーリングを調整します。Auto Scaling設定では、CloudWatchアラームを作成し、CPU使用率など事前定義されたメトリクスまたはカスタムメトリクスとそのメトリクスのターゲット値を指定します。メトリクスとターゲット値にもとづいて、最小・最大値の範囲内でスケーリングが行われます<sup>※59</sup>。

### ●Amazon Keyspaces テーブル

Amazon KeyspacesはDynamoDBと同じアーキテクチャを持つオープンソースNoSQLデータベースApache Cassandraのマネージドサービスです。AutoScalingはDynamoDBと同じく、テーブルに対する読み込み・書き込みキャパシティユニットに対し、ターゲット追跡スケーリングポリシーを使用して、最大・最小キャパシティユニット数を定義してスケーリングを調整します。Auto Scaling設定では、CloudWatchアラームを作成し、キャパシティユニットの使用率ターゲット値を指定します。メトリクスとターゲット値にもとづいて、最小・最大値の範囲内でスケーリングが行われます<sup>※60</sup>。

### ●AWS Lambda 同時実行性

2章2.1節「Lambda関数」で解説したとおり、Lambda関数には同時実行数を設定します。リクエスト数に応じて、実行数の最大値までスケールします。

### ●Amazon MSK (Managed Service for Kafka) クラスタストレージ

MSKは、オープンソースの大規模メッセージングミドルウェアKafkaのマネージドサービスです。自動スケーリングポリシーを使用して、メッセージプローカーとなるクラスタストレージの使用率にターゲットを設定することで、その値を保つように自動的にクラスタのストレージが拡張されます<sup>※61</sup>。

### ●Amazon SageMaker

機械学習サービス SageMakerの学習モデルをデプロイするインスタンスに対し、AutoScalingが適用できます。SageMakerは、ターゲット追跡スケーリングポリシーもしくはステップスケーリングポリシーを使用して、最大・最小インスタンス数を定義してスケーリングを調整します。Auto Scaling設定では、CloudWatchアラームを作成し、CPU使用率など事前定義されたメトリクスまたはカスタムメトリクスとそのメトリクスのターゲット値を指定します。メトリクスとターゲット値にもとづいて、最小・最大値の範囲内でスケーリングが行われます<sup>※62</sup>。

### ●EC2スポットインスタンスのリクエスト

EC2ではスポットインスタンスをリクエストして廉価な価格でEC2インスタンスを利用することができますが、リクエストされたスポットインスタンス群（スポットフリート）にもAutoScalingが適用できます。スポットフリートも、オンデマンド同様、ターゲット追跡スケーリング、ステップスケーリング、スケジューリングによるスケーリングが選択できます<sup>※63</sup>。

※59 [https://docs.aws.amazon.com/ja\\_jp/AmazonECS/latest/developerguide/service-auto-scaling.html](https://docs.aws.amazon.com/ja_jp/AmazonECS/latest/developerguide/service-auto-scaling.html)

※60 [https://docs.aws.amazon.com/ja\\_jp/keysaces/latest/devguide/autoscaling.html](https://docs.aws.amazon.com/ja_jp/keysaces/latest/devguide/autoscaling.html)

※61 [https://docs.aws.amazon.com/ja\\_jp/msk/latest/developerguide/msk-autoexpand.html](https://docs.aws.amazon.com/ja_jp/msk/latest/developerguide/msk-autoexpand.html)

※62 [https://docs.aws.amazon.com/ja\\_jp/sagemaker/latest/dg/endpoint-auto-scaling.html](https://docs.aws.amazon.com/ja_jp/sagemaker/latest/dg/endpoint-auto-scaling.html)

※63 [https://docs.aws.amazon.com/ja\\_jp/AWSEC2/latest/UserGuide/spot-fleet-automatic-scaling.html](https://docs.aws.amazon.com/ja_jp/AWSEC2/latest/UserGuide/spot-fleet-automatic-scaling.html)

## 演習問題

1 ある開発者が、EC2上で実行されているアプリケーションパフォーマンスを監視するために、CloudWatch Metricsでメトリクスを収集しようとしています。次のうち、デフォルトで収集されず、CloudWatchエージェントを使用して収集する必要があるメトリクスを2つ選択してください。

- A. メモリ使用率(MemoryUtilization)
- B. ネットワークIO(NetworkIn/Out)
- C. ディスク使用率(DiskUsage)
- D. ディスクIO(DiskRead/Write)
- E. EBSボリュームIO(EBSRead/Write)

2 ある会社が、ECS上でアプリケーションを運用しています。アプリケーションは負荷の変化が激しく、担当者は30秒間隔でメトリクスを収集したいと考えています。開発者が設定すべきオプションで正しいものを選択してください。

- A. 標準解像度の設定で10秒間隔でメトリクスを収集するよう設定します。
- B. 標準解像度の設定で30秒間隔でメトリクスを収集するよう設定します。
- C. 高解像度の設定で10秒間隔でメトリクスを収集するよう設定します。
- D. 高解像度の設定で30秒間隔でメトリクスを収集するよう設定します。

3 ある機関で断続的にバッチ処理が実行されるEC2インスタンスを運用しています。指定された条件でサーバの負荷が高まったときに、EC2 AutoScalingによりEC2インスタンスを追加するよう、CloudWatch Alarmsで設定したいと思っています。瞬間にCPU使用率が高まることが頻繁にあるため、継続的に5分弱程度、75%以上にCPU使用率が高まった場合にサーバをスケールさせるアクションを実行し、負荷を平準化したい場合に、CloudWatch Alarmsをどのように構成すべきか正しいものを選択してください。

- A. メトリクスとしてCPUUtilization、期間を1分、評価期間を5つのデータポイントとし、しきい値を75と定義します。
- B. メトリクスとしてCPUUtilization、期間を5分、評価期間を1分とし、しきい値を0.75と定義します。
- C. メトリクスとしてCPUUtilization、期間を5分、評価期間を1つのデータポイントとし、しきい値を75と定義します。
- D. メトリクスとしてCPUUtilization、期間を1分、評価期間を5分とし、しきい値を0.75と定義します。

4 開発チームはAWS Lambdaで同期実行中にエラーが発生した場合、CloudWatch Logsにログを出力し、運用担当者と共有するサードパーティ製コミュニケーションツールへエラー内容を通知したいと考えています。最も低コストで構築可能なオプションで正しいものを選択してください。

- A. CloudWatch Logsに出力されたロググループにメトリクスフィルタを設定し、エラー文字列のフィルタパターンと、メトリクス値に名前付きフィールド識別子を指定してパブリッシュする設定を行います。
- B. CloudWatch Logsに出力されたロググループにサブスクリプションフィルタを設定し、エラー文字列のフィルタパターンと、コミュニケーションツールにメッセージを通知するLambda関数を設定します。
- C. CloudWatch AlarmsでLambdaのError(Errors)メトリクスでエラーを検知し、SNSトピックを定義してLambda関数を経由してコミュニケーションツールへメッセージ通知する設定を行います。
- D. CloudWatch Logsに出力されたロググループとは別に、Lambda関数の送信先(Destination)で、条件として「失敗時(On Failure)」の送信タイプを指定し、コミュニケーションツールにメッセージを通知するLambda関数を設定します。

5 開発チームはAWSクラウド上でEKSを使用してアプリケーション環境を構築していますが、最近、急激に増加したネットワーク通信コストの原因を早急に調査する必要があります。CloudWatchを使用して、最も迅速で効率的な対応方法を選択してください。

- A. すべてのVPCフローログを有効化し、CloudWatch Logsにエクスポートする。CloudWatch Logs Insightsで「bytes」順でソートするクエリコマンドを発行し、ネットワーク負荷が高い原因を調査します。
- B. すべてのVPCフローログを有効化し、S3にエクスポートする。エクスポートしたデータにCloudWatch Logs Insightsで「bytes」順でソートするクエリコマンドを発行し、ネットワーク負荷が高い原因を調査します。
- C. CloudWatch Metricsを使用して、EKSクラスタが実行されているインスタンスノードのメトリクスでネットワークIO「(NetworkIn/NetworkOut)」が高いインスタンスを特定し、ネットワーク負荷が高い原因を調査します。
- D. CloudWatch Metricsを使用して、EKSクラスタが実行されているインスタンスノードのメトリクスで仮想ENIトラフィック「(ENIIn/ENIOut)」が高いコンテナを特定し、ネットワーク負荷が高い原因を調査します。

6 開発チームはAWSクラウド上で監査のために必要な操作をCloudTrailを使って記録しようとしています。CloudTrailを使って実現できないものを2つ選択してください。

- A. クロスリージョンレプリケーションにより、1箇所にログを集約できます。
- B. 90日間以上保存してマネジメントコンソール上のイベント履歴から確認できるオプションを有料で選択できます。
- C. CloudWatch Logsサブスクリプションフィルタと組み合わせ、特定の操作が実行された際に管理者へ通知できます。
- D. イベントを削除する際に、MFAを使った2段階認証により不正な操作を防止できます。
- E. SHA-256ハッシュアルゴリズムを使用して、ログが改竄されていないことを証明する署名を付与できます。validate-logsコマンドで、公開鍵を指定して実行することにより、ログが改竄されていないことを検証できます。

7 AWSクラウドを利用している中で、アカウント内のユーザにより、特定の時間にEC2を使って不正な操作が行われていることが判明しました。リアルタイムで被害が発生していることからできるだけ迅速に原因を特定したいと考えています。CloudTrailを使って履歴調査する際に最も適した方法を選択してください。

- A. S3へ出力したイベント履歴をAmazon Athenaを使って、不正な操作を検出します。
- B. マネジメントコンソールの「イベント履歴」から不正な操作を検出します。
- C. CloudWatch Logs Insightsを使って、不正な操作を検出します。
- D. Amazon OpenSearch Serviceを組み合わせて、不正な操作を検出します。

8 ある企業はAWS Lambdaを使ってコンシューマー向けのWebサービスをサーバレス環境で提供しています。CloudTrailを使用して、サービスの中で最も利用頻度の高い処理をできるだけ低コストで特定したいと思っています。最も適した方法を選択してください。

- A. インサイトイベントの出力オプションを有効化し、S3へ出力したイベント履歴をAmazon Athenaを使って、呼び出し回数の多いLambda関数を特定します。
- B. インサイトイベントの出力オプションを有効化し、Amazon OpenSearch Serviceを組み合わせて、呼び出し回数の多いLambda関数を可視化します。
- C. データイベントの出力オプションを有効化し、S3へ出力したイベント履歴をAmazon Athenaを使って、呼び出し回数の多いLambda関数を特定します。
- D. データイベントの出力オプションを有効化し、Amazon OpenSearch Serviceを組み合わせて、呼び出し回数の多いLambda関数を可視化します。

**9** ある企業はコンシューマー向けのWebサービスをワールドワイドで提供しています。CloudTrailを使用して、IPアドレスなどの情報から、さまざまなデータにアクセスしているサービスの中で最も利用頻度の高い国や地域を特定したいと思っています。最も適した方法を選択してください。

- A. インサイトイベントの出力オプションを有効化し、S3へ出力したイベント履歴をAmazon Athenaを使って、国・地域ごとのサービス利用数を特定します。
- B. インサイトイベントの出力オプションを有効化し、Amazon OpenSearch Serviceを組み合わせて、国・地域ごとのサービス利用数を可視化します。
- C. データイベントの出力オプションを有効化し、S3へ出力したイベント履歴をAmazon Athenaを使って、国・地域ごとのサービス利用数を特定します。
- D. データイベントの出力オプションを有効化し、Amazon OpenSearch Serviceを組み合わせて、国・地域ごとのサービス利用数を可視化します。

**10** 開発チームは、アプリケーションの実装中にリレーショナルデータベースアクセスを含む処理のパフォーマンスを検証するため、X-Rayを使って各処理の実行時間を計測しようとしています。必要な設定や実装の中で誤っているものを選択してください。

- A. 開発用のユーザに、X-RayのAPIへトレースデータをPutする権限を付与し、認証情報を開発端末に保存します。
- B. X-Rayデーモンを実行するために、DockerのインストールおよびX-Rayデーモンのコンテナイメージを作成します(イメージ名:xxxxx/xray-daemon:latest)。
- C. X-Rayデーモンを以下のコマンドで起動します。

```
docker run --attach STDOUT -v ~/.aws:/root/.aws/ --net=host -e AWS_REGION=ap-northeast-1 --name xray-daemon -p 2000:2000/udp xxxx/xray-daemon:latest -o
```

- D. アプリケーションにサンプリングルールの設定や受信リクエスト、セグメント・カスタムサブセグメントの開始終了およびRDSへアクセスするためのSDKクライアントへX-Rayの設定を行います。

**11** 開発チームは、アプリケーションのリレーショナルデータベースアクセスを含む処理パフォーマンスを検証するため、X-Rayを使って各処理の実行時間を計測しようとしています。フィルタ式を使って計測対象となるデータベースアクセス処理を一覧化して抽出する方法で最も簡易的なやり方を選択してください。

- A. CONTAINSオペレータを使って、データベースアクセスを含む処理をターゲットとするカスタムサブセグメントを抽出します。トレースデータのセグメントにインデックスとなるキーを含めます。
- B. メタデータを使って、データベースアクセスを含む処理をターゲットとするカスタムサブセグメントを抽出します。トレースデータのセグメントにインデックスとなるキーを含めます。
- C. 注釈を使って、データベースアクセスを含む処理をターゲットとするカスタムサブセグメントを抽出します。トレースデータのセグメントにインデックスとなるキーを含めます。
- D. CloudWatch ServiceLensを使ってデータベースアクセスを含む処理をターゲットとするカスタムサブセグメントを抽出します。トレースデータのセグメントにインデックスとなるキーを含めます。

**12** 開発チームは、ステージング環境にデプロイしたアプリケーションでX-Rayを有効化しようと設定を行っていますが、トレースデータがX-Rayコンソールから確認できません。アプリケーションログにエラーは出力されておらず、開発環境で実行されているアプリケーションからは正常にトレースデータが表示されていることを確認できています。この原因として考えられる正しいものを2つ選択してください。

- A. アプリケーションをデプロイしたEC2インスタンスのユーザデータでX-Rayデーモンがインストールされていません。
- B. アプリケーションをデプロイしたEC2インスタンスのインスタンスプロファイルにPutTraceSegments APIにアクセスする権限が含まれていません。
- C. アプリケーションをデプロイしたEC2インスタンスがプライベートサブネットに配置されており、X-RayのAPIへ送信できません。
- D. マネジメントコンソール上でX-Rayのトレースデータを表示するための権限を持っていません。
- E. アプリケーションに設定したX-Ray APIのエンドポイントがステージング環境以外のものが設定されています。

13 ある開発者は複数のインスタンスやコンテナで実行されているセッションデータをElastiCache(Redis)を使って共有しようとしています。基本的にアプリケーションのダウンは最小限に、ElastiCacheの障害発生時にもなるべく早くアプリケーションを復旧することが求められます。この要件を満たすための必要なElastiCacheの設定のうち、最も低コストで実現できるオプションを選択してください。

- A. クラスタモードを無効にして、自動フェイルオーバー機能も無効に設定しておきます。複数のレプリカでマルチAZ構成とし、障害時は手動フェイルオーバーにより迅速に切り替えます。
- B. クラスタモードを無効にして、自動フェイルオーバー機能を有効にします。
- C. クラスタモードを有効にして、自動フェイルオーバー機能は無効に設定しておきます。複数のレプリカでマルチAZ構成とし、障害時は手動フェイルオーバーにより迅速に切り替えます。
- D. クラスタモードを有効にして、自動フェイルオーバー機能も有効にします。

14 設問13の環境で実際にElastiCache(Redis)のプライマリノードダウンが発生し、フェイルオーバーが完了するまでアプリケーションが使用不能になってしまい、アプリケーションのユーザからクレームが入ってしまいました。再発防止のために開発者が対応するべきアクションで正しいものを選択してください。

- A. クラスタモードを有効にして、アプリケーションの参照を設定エンドポイントに設定することでゼロダウンタイムを実現します。
- B. フェイルオーバーしたタイミングでプライマリエンドポイントが指示するノードを切り替えるよう、Lambda関数で実装します。
- C. フェイルオーバーしたタイミングで読み込みポイントが指示するノードを切り替えるよう、Lambda関数で実装します。
- D. どのモードでもフェイルオーバー時に一時的に書き込みアクセスは不可能になるため、アプリケーション側でElastiCacheのアクセス不可を検知したら、しばらく時間をおいてアクセスする旨をユーザ側に通知するようにエラーハンドリング処理を実装します。

15 ある開発チームは、Webアプリケーションから要求を受け付け、SQSを使って非同期でディレードバッチ処理を実行することを検討しています。バッチ処理は数秒程度で終わる簡単な物ですが、リクエストを受け付けた後、1分程度でバッチ処理を完了することが求められています。この要件を満たすために必要なSQSの設定として、最も適切なものを選択してください。

- A. メッセージ受信待機時間を0秒に設定します。
- B. メッセージ受信待機時間を30秒に設定します。
- C. 配信遅延時間を0秒に設定します。
- D. 配信遅延時間を30秒に設定します。

16 あるプロデューサーアプリケーションではファイルのマージ処理を実行した後に、ファイルをEFSに保存して、SQSへメッセージを送信します。複数のコンシューマーアプリケーションがキューにポーリングを行っており、メッセージを取得したそのうちの1つがファイルの内容をデータベースにロードします。ロード処理は幂等性が担保されており、30秒程度で完了するのですが、時々エラーが発生します。原因と正しい対処として適切なものを選択してください。

- A. 処理が完了する前に、別のコンシューマーアプリケーションが実行して、データベースへのロードが失敗している。可視性タイムアウトの設定を1分程度に変更する。
- B. 処理が完了する前に、別のコンシューマーアプリケーションが実行して、ReceiptHandleが書き換えられている。可視性タイムアウトの設定を1分程度に変更する。
- C. 処理が完了する前に、別のコンシューマーアプリケーションが実行して、データベースへのロードが失敗している。メッセージの配信遅延設定を1分程度に変更する。
- D. 処理が完了する前に、別のコンシューマーアプリケーションが実行して、ReceiptHandleが書き換えられている。メッセージの配信遅延設定を1分程度に変更する。

17 あるリテール企業では、新しく開発するECサイトの注文受付処理をSQSを使ってキューイングし、在庫確認や決済、配送などの処理コストの高い処理を非同期で実行することを検討しています。注文処理はFIFOキューを使って、順序性を担保しつつ、2重送信など重複が発生しないように構成する必要があります。SQSの設定やプロデューサーアプリケーションの送信処理として、パフォーマンス効率が高い、適切なものを選択してください。

- A. すべての商品で共通のメッセージグループIDを構成します。注文処理ごとにユニークなメッセージ重複排除IDを発行し、2重送信による重複注文を抑止します。
- B. すべての商品で共通のメッセージ重複排除IDを構成します。注文処理ごとにユニークなメッセージグループIDを発行し、2重送信による重複注文を抑止します。
- C. 商品ごとにメッセージグループIDを構成します。注文処理ごとにユニークなメッセージ重複排除IDを発行し、2重送信による重複注文を抑止します。
- D. 商品ごとにメッセージ重複排除IDを構成します。注文処理ごとにユニークなメッセージグループIDを発行し、2重送信による重複注文を抑止します。

18 あるプロデューサーアプリケーションが、SQSに対し、断続的に10件程度のメッセージをまとめて送信しています。コンシューマーアプリケーションは10秒でロングポーリングを行っていますが、キュー内に蓄積されたメッセージを処理できず、コンシューマーアプリケーションをオートスケールして対応する状況がたびたび発生しています。このアプリケーションの処理効率を最適化するために、開発者が実施すべき対処で最も正しいものを選択してください。

- A. ポーリング間隔を短くするよう、キューを設定します。
- B. SQSのキューを追加して、半分の5件ずつ送信するようにプロデューサーを構成します。コンシューマーアプリケーションは2つのキューからメッセージを取得して、多重実行可能ないように構成します。
- C. キューのメッセージ受信待機時間を20秒に設定します。
- D. コンシューマーアプリケーションがReceiveMessage APIを呼び出す際のMaxNumberOfMessagesパラメータを10に設定し、多重実行可能なようにアプリケーションを構成します。

19 SQSのキューに蓄積されたメッセージを1件ずつ取得して処理する、複数のコンシューマーバッチアプリケーションがあります。アプリケーションはたびたび重複したメッセージを処理し、トランザクションを書き換えてしまうことが問題となっています。対処として誤っているものを選択してください。

- A. バッチ処理を幕等性となるよう実装します。
- B. FIFOキューを使用して、重複処理が発生しないよう構成します。
- C. 可視性タイムアウトをバッチの処理時間より大きめに設定します。
- D. スタンダードキューに蓄積されたメッセージを取得する際に、メッセージ重複排除IDを指定して取得し、重複処理を抑止します。

20 あるWebサービスを提供する企業はパフォーマンス改善のため、CloudFrontを使ってコンテンツをキャッシングするようにアプリケーションをリファクタリングしようとしています。サービスは現在、www.webserviceA.comでS3の静的Webサイトホスティング機能により提供されています。CloudFrontへ移行する設定のうち、誤っているものを選択してください。

- A. CloudFrontでディストリビューションを作成します。オリジンパスにS3のバケットを設定し、代替ドメイン名にwww.webserviceA.comを指定します。
- B. AWS Certificate Manager(ACM)を使用して、SSL/TLS証明書をサービス展開するリージョンでインポートします。
- C. Route 53の設定を変更します。webserviceA.comゾーン内にCNAMEレコードに作成されたCloudFrontディストリビューションのドメイン名を割り当てます。
- D. パスパターンごとにビヘイビアを設定し、キャッシュポリシーやオリジンリクエストポリシーを構成します。

21 あるWebサービスを提供する企業はパフォーマンス改善のため、CloudFrontを使ってコンテンツをキャッシングするようにアプリケーションをリファクタリングしようとしています。オリジンで静的なHTMLや画像イメージが格納されているパスパターンに対し、設定すべきキャッシングポリシーで最も適切なものを2つ選択してください。なお、設定されている対象のオリジンのコンテンツには、リリース後1日程度で変更後に反映されればよく、特に更新後に1時間ほどで反映したいものに対して、Cache-Control "max-age=3600"を設定しているものとします。

- A. Min TTL:0, Max TTL:86400, Default TTL:0
- B. Min TTL:86400, Max TTL:315360000, Default TTL:86400
- C. Min TTL:3600, Max TTL:315360000, Default TTL:31560000
- D. Min TTL:1, Max TTL:315360000, Default TTL:86400
- E. Min TTL:0, Max TTL:315360000, Default TTL:86400

22 あるWebサービスを提供する企業はパフォーマンス改善のため、CloudFrontを使ってコンテンツをキャッシングするようにアプリケーションをリファクタリングしようとしています。オリジンに保存されている、ユーザごとに固有の画像データのサイズが大きくなりがちでパフォーマンスに問題が生じているため、キャッシングの対象として設定しようとしています。これに必要な設定で最も適切なものを1つ選択してください。

- A. 画像データを格納しているパスパターンにアタッチしたキャッシングポリシーのみにCookieを含めて構成します。
- B. 画像データを格納しているパスパターンにアタッチしたキャッシングポリシーおよびオリジンリクエストポリシーにCookieを含めて構成します。
- C. 画像データを格納しているパスパターンにアタッチしたオリジンリクエストポリシーのみにCookieを含めて構成します。
- D. 画像データを格納しているパスパターンにアタッチしたキャッシングポリシーにCookieを、オリジンリクエストポリシーにOriginヘッダを含めて構成します。

23 あるWebサービスを提供する企業はパフォーマンス改善のため、CloudFrontを使ってコンテンツをキャッシングするようにアプリケーションをリファクタリングしようとしています。フロントに配置したキャッシングデータは秘匿性が高く、制限付きアクセスオプションを使ってビューアーの接続を制限します。これに必要な設定で誤っているものを1つ選択してください。

- A. 公開鍵と秘密鍵のキーペアを作成し、「信頼されたキーグループ」に公開鍵を追加します。
- B. SDKを使って署名付きURLやCookieを生成するアプリケーションを実装します。
- C. パスパターンごとにビヘイビアを設定し、ビューアー接続で「信頼されたキーグループ」を指定します。
- D. アクセスする接続元IPアドレスや有効期限を短く設定したキャッシングポリシーを定義して署名に含めるよう実装します。

24 あるWebサービスを提供する企業はパフォーマンス改善のため、CloudFrontを使ってコンテンツをキャッシングするようにアプリケーションをリファクタリングしようとしています。ビューアーのUser-Agentヘッダに応じて、レスポンスする画像をリサイズしてコンテンツ整形して返却する処理をLambda@Edgeで実現しようとしています。これに必要な設定で最も適切なものを1つ選択してください。

- A. 画像をリサイズするLambda関数を実装し、ビューワーレスポンスに関連付けます。
- B. User-Agent値に応じてURLを切り替えるLambda関数を実装し、オリジンリクエストに関連付けます。
- C. 複数のサイズを保存する画像パスを用意して、できるだけ短いTTLを設定します。
- D. 複数のサイズを保存する画像パスを用意して、リサイズ処理を実行した後、オリジンに永続化します。

25 あるWebサービスを提供する企業はアプリケーションを通常3台のEC2インスタンス上で実行しています。このWebサービスは不特定な時間帯でリクエストが急激に増減するため、Auto Scalingを構成して各インスタンスで処理するトラフィックをコントロールしたいと思っています。アプリケーションが実行されているEC2インスタンスの平均CPU使用率が60%を超えたときにインスタンスを追加し(最大10台まで)、30%を下回ったときにインスタンスを削除する(最小3台)のに、必要な設定で誤っているものを1つ選択してください。

- A. AutoScalingGroupを作成して、最小台数の維持として3、最大キャパシティ10を設定します。
- B. CloudWatchアラームを作成し、CPU使用率が60%を超えたとき、および30%を下回ったときにアラームする設定を2つ作成します。
- C. 作成したAutoScalingGroupの動的スケーリングポリシーの設定でシンプルスケーリングポリシーを作成し、アラームを設定します。
- D. EC2起動テンプレートを関連付けて、AutoScalingGroupを作成します。グループメトリクスの収集を有効化しておきます。

26 あるWebサービスを提供する企業はアプリケーションを通常複数台のEC2インスタンス上で実行しています。このWebサービスは平日の12:00-13:00ごろにリクエストが急激に増加するため、ピーク時間に合わせて各インスタンスを増加させて、各インスタンスで処理するトラフィックをコントロールしたいと思っています。必要な設定で正しいものを1つ選択してください。

- A. 予測スケーリングを使用します。予測スケーリングはAutoScalingGroupに、予測に必要なメトリクスをスケーリングポリシーとして設定するだけですぐに利用できます。
- B. スケジューリングを使用します。スケジューリングは日時や定時実行スケジュール、DesiredCapacityを「予定されたアクション」として設定するだけですぐに利用できます。
- C. CloudWatch Alarmsを使用します。指定した時間のアラームをシンプルスケーリングポリシーに設定し、動的スケーリングを構成します。
- D. CloudWatch Eventを使用します。指定した時間にイベントを発生してAuto Scalingを起動するように構成します。

27 あるWebサービスを提供する企業はアプリケーションを通常複数台のEC2インスタンス上で実行しています。このWebサービスは不特定な時間帯でリクエストが急激に増減するため、ターゲット追跡ポリシーを設定した動的スケーリングを構成することにより、各インスタンスで処理するトラフィックをコントロールしたいと思っています。ターゲット追跡ポリシーの設定で誤っているものを1つ選択してください。

- A. ターゲット追跡ポリシーでは、EC2インスタンスで使用されているメモリ使用率がメトリクスとして指定できます。
- B. ターゲット追跡ポリシーでは、EC2インスタンスで使用されているネットワークIOがメトリクスとして指定できます。
- C. ターゲット追跡ポリシーでは、ALBで指定したターゲットグループのターゲットごとの完了リクエスト数がメトリクスとして指定できます。
- D. ターゲット追跡ポリシーでは、ウォームアップ時間を指定することで、必要以上にインスタンスが追加されるリスクを軽減できます。

28 あるWebサービスを提供する企業はアプリケーションを通常複数台のEC2インスタンス上で実行しています。アプリケーションはさまざまな最新のソフトウェアパッチを取得して構成しなければならず、Auto Scaling Groupのライフサイクルフック機能を使ってセットアップすることを検討しています。ライフサイクルフックに関する内容や設定について誤っているものを1つ選択してください。

- A. Auto Scalingにより起動されたインスタンスは通常ステータス "Pending"から"In Service"へ移行するところ、間に"Pending:Wait"および"Pending:Proceed"ステータスの状態が追加されます。
- B. "Pending:Wait"ステータスでカスタムアクションを実行できます。EventBridgeでライフサイクルフックイベントを受け取る設定を作成し、最新パッチのインストールスクリプトを実装したLambda関数やSSM RunCommandを設定します。
- C. スクリプトの中でインストールが完了すると、ハートビートタイムアウトまで待機され、その後、InServiceへ移行します。
- D. 処理時間がタイムアウト時間を上回る場合は、RecordLifecycleActionHeartbeatコマンドを実行してタイムアウト時間を延長します。

**29** あるWebサービスを提供する企業はアプリケーションを通常複数台のEC2インスタンス上で実行しています。EC2インスタンスだけでなく、アプリケーションが利用するサービスにもコスト・パフォーマンス最適化を図るために、Auto Scalingの導入を検討しています。以下の選択肢の中で、Application Auto Scalingをサポートしていないものを1つ選択してください。

- A. Amazon ECSのサービス
- B. Amazon RDSのリードレプリカ
- C. Amazon DynamoDBのテーブルおよびグローバルセカンダリインデックス
- D. AWS Lambdaの同時実行数

## 解答

**1 A, C**

⇒ 問題：438 ページ、本文解説：364 ページ

- ・選択肢A：メモリ使用率はEC2の標準メトリクスには含まれていません。CloudWatchエージェントで収集することができます<sup>※64</sup>。
- ・選択肢B：ネットワークIOはEC2の標準メトリクスには含まれています<sup>※65</sup>。
- ・選択肢C：ディスク使用率はEC2の標準メトリクスには含まれていません。CloudWatchエージェントで収集することができます。
- ・選択肢D：ディスクIOはEC2の標準メトリクスには含まれています。
- ・選択肢E：EBSボリュームIOはEC2の標準メトリクスには含まれています。

**2 D**

⇒ 問題：438 ページ、本文解説：363 ページ

- ・選択肢A、B：標準解像度は最小1分の間隔でメトリクスを収集するオプションです。
- ・選択肢C、D：高解像度は1秒、5秒、10秒、30秒の間隔でメトリクスを収集することができます。選択肢Cの10秒だと余剰にデータ収集しています。

**3 A**

⇒ 問題：438 ページ、本文解説：366 ページ

- ・選択肢A：期間が1分で、5つのデータポイントを評価期間とすると、1分間隔で5回データ収集した状態で評価されます。
- ・選択肢B：しきい値が0.75%のCPU使用率として評価される設定となっています。
- ・選択肢C：期間が5分で、1つのデータポイントを評価期間とすると、5分間で1回しかデータ評価されていないため要件に合致しません。
- ・選択肢D：しきい値が0.75%のCPU使用率として評価される設定となっています。

※64 [https://docs.aws.amazon.com/ja\\_jp/AmazonCloudWatch/latest/monitoring/metrics-collected-by-CloudWatch-agent.html](https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/metrics-collected-by-CloudWatch-agent.html)

※65 [https://docs.aws.amazon.com/ja\\_jp/AWSEC2/latest/UserGuide/viewing\\_metrics\\_with\\_cloudwatch.html](https://docs.aws.amazon.com/ja_jp/AWSEC2/latest/UserGuide/viewing_metrics_with_cloudwatch.html)

**4****B**

➡ 問題：439 ページ、本文解説：368 ページ

- 選択肢A：メトリクスフィルタで、特定の文字列を含むエラーをピックアップしてパブリッシュすることができますが、発生件数などのメトリクスであり、エラーの内容を通知することはできません。
- 選択肢B：サブスクリプションフィルタで、特定の文字列を含むエラーをピックアップして、エラーログをLambda関数やSNSに渡すことができます。Lambda関数の中でサードパーティ製コミュニケーションツールのWebhookエンドポイントへ通知する処理を実装します。
- 選択肢C：CloudWatch AlarmsでLambdaのErrorメトリクスを使って発生したエラーを検知し、通知することはできますが、エラー内容を通知することはできません。
- 選択肢D：Lambda関数はエラー発生時の送信先(Destination)で別のLambda関数を呼び出すことができ、こちらも簡易に実装することができます<sup>※66</sup>、S3からのイベント契機など非同期に実行されたLambda関数が対象になります。API Gatewayからの呼び出しなど、同期的に実行されたLambda関数では、CloudWatch Logsのサブスクリプション機能を使って、別のLambda関数を呼び出す必要があります。

**5****A**

➡ 問題：440 ページ、本文解説：369-370 ページ

- 選択肢A：アカウント内のすべてのVPCフローログを収集することで、最も費用がかかっているVPCを絞り込むことができ、CloudWatch Logs Insightsを使って、通信データ量順でトラフィックデータをソートすることで、大きなデータサイズの通信を実行しているENIを特定することができます。また標準メトリクスで送受信元のIPアドレスがわかるため、費用が発生することになるアベイラビリティゾーンのアウトバウンド通信の特定が容易です。
- 選択肢B：VPCフローログはS3にエクスポートすることができますが、CloudWatch Logs InsightsはS3にエクスポートしたデータにクエリを発行することができません。
- 選択肢C：CloudWatch MetricsでネットワークIOが高いインスタンスを迅速に特定することができます。ただし、特定した結果、表示されるのがデータ量のみで、送受信元のIPアドレスがわからないため、どのコンテナで実行されているかや、費用が発生することになるアベイラビリティゾーンのアウトバウンド通信の特定が難しく、調査に時間がかかります。
- 選択肢D：CloudWatch Metricsには仮想ENI単位でトラフィックを出力する標準のオプションはありません。インスタンス単位で特定はできますが、どのコンテナで実行されているかまで特定するのに時間がかかります。

※66 <https://aws.amazon.com/jp/about-aws/whats-new/2019/11/aws-lambda-supports-destinations-for-asynchronous-invocations/>

**6****B、E**

➡ 問題：440 ページ、本文解説：377-379 ページ

- CloudTrailはAWSアカウント内のユーザ、ロール、サービスなどがマネジメントコンソールやCLI、SDKから実行したAPIアクションをイベントとして記録します。実現できない選択肢は、以下のとおりです。
- 選択肢B：90日間以上イベントを保存するにはS3へ出力する必要があります。90日以降のイベント履歴を確認できる有料オプションはありません。
  - 選択肢E：validate-logsコマンドの実行時に公開鍵を指定する必要はありません。署名の検証はコマンド実行を通して透過的に行われます。

**7****C**

➡ 問題：441 ページ、本文解説：382 ページ

CloudTrailでイベント履歴を調査する方法はいくつかありますが、このユースケースでは、できる限りリアルタイムで特定するという点がポイントです。そのため、イベントのロードが不要なCloudWatch Logs Insightsを使って検索する方法が最も適しています。

**8****C**

➡ 問題：441 ページ、本文解説：380 ページ

AWS Lambda関数の実行アクティビティをCloudTrailで記録するには、データイベントを収集する必要があります。イベント履歴を調査する方法はいくつかありますが、このユースケースでは、できる限り低コストという点がポイントです。Athenaで指定する際は、読み込むデータの対象を絞るようターゲットを絞ったパーティションにより、イベントの特定の属性、例えばイベントソースをLambdaに限定して、データのスキャン量を削減します。

**9****D**

➡ 問題：442 ページ、本文解説：381 ページ

コンシューマーのさまざまなデータアクセス処理を収集するには、CloudTrailでデータイベントを収集する必要があります。イベント履歴を調査する方法はいくつかありますが、このユースケースでは、IPアドレスなどのアクセス統計を可視化するという点がポイントです。そのため、解析と可視化に優れたサービスAmazon OpenSearch Serviceを使って検索する方法が最も適しています。

**10 D**

→ 問題：442 ページ、本文解説：389 ページ

開発端末やサーバでX-Rayを使用する場合、デーモンをインストールしたり、Dockerを使ってデーモンを実行するコンテナを起動することで使用できます。X-Ray APIへアクセス権限を持つユーザの認証情報(アクセスキーやシークレットアクセキー)をデーモンプロセスが実行される環境へ設定しておきます(EC2インスタンスであればIAMロールでも可)。アプリケーションではサンプリングルールや受信リクエスト、セグメント・カスタムセグメントの開始終了を実装しますが、RDSなどリレーショナルデータベースへのアクセスは、データベース接続のドライバやSQLクライアントに対し、X-RayのSDKを設定します。RDSへアクセスするためのSDKクライアントではありません。

**11 C**

→ 問題：443 ページ、本文解説：390-391 ページ

注釈は上記のフィルタ式で使用するためのインデックス化されたキーバリューペアです。例えばアプリケーションで、注釈に特定の値を設定し、フィルタ式のキーワードで、「annotations.key」を使用すると、該当のトレースデータだけをフィルタできます。トレースごとに最大50個の注釈がインデックス化され検索キーワードとして利用できます。なお、他の選択肢の不正解理由は、以下のとおりです。

- ・選択肢A：CONTAINSオペレータだけでは、データベースアクセスを特定するフィルタのインデックスを含めることはできません。注釈の設定が必要です。
- ・選択肢B：メタデータはトレースデータに追加情報を含める用途であり、フィルタのインデックスにすることはできません。
- ・選択肢D：CloudWatch ServiceLensは、CloudWatchからX-Rayトレースデータを一元的に参照できる機能であり、関係ありません。

**12 A、B**

→ 問題：443 ページ、本文解説：387-388 ページ

X-Rayへトレースデータを送信するには、X-Rayデーモンのインストールとデーモンが実行されるサーバやインスタンスに PutTraceSegmentsのアクセス権限を設定しておく必要があります。開発環境で正しく送信できているので、アプリケーション内の設定実装に不備はないと考えると、不正が疑われる原因是上記の2点です。その他の選択肢の不正解理由は、以下のとおりです。

- ・選択肢C：プライベートサブネットに配置され、NAT Gatewayが設置されていないなど確かに送信ができないケースはありますが、この場合、アプリケーションに通信エラーログが出力されるはずです。
- ・選択肢D：マネジメントコンソールにトレースデータを表示する権限は存在しません。
- ・選択肢E：X-Ray APIのエンドポイントはデーモン起動時にリージョンを指定することによりデーモン内で設定されます。デーモン経由で送信するアプリケーションで指定することはできません。

**13 B**

→ 問題：444 ページ、本文解説：395-397 ページ

ElastiCacheでデータが保存されているプライマリノードに障害が発生した場合、自動フェイルオーバー機能により迅速に復旧できます。自動フェイルオーバーを利用するには、1つ以上のリードレプリカを作成し、マルチAZを有効化します。また、クラスタモードを有効化するとデータを保存するノードが分散され、シャーディングされますが、これは高速化を図るためのもので、本設問の必須の要件ではありません。その他の選択肢の不正解理由は以下のとおりです。

- ・選択肢A：手動フェイルオーバーを実行するにはレプリカをシングルAZ構成にしておく必要があります。また、迅速に復旧するにはマルチAZ構成にして自動フェイルオーバーを利用したほうがより効果的です。
- ・選択肢C：クラスタモードを有効化すると自動的にフェイルオーバー機能は有効化されます。また、本設問でシャーディングする必要はありません。
- ・選択肢D：本設問でシャーディングする必要はありません。できるだけ低コストで実現するには、クラスタモードを無効化したほうがよい選択肢です。

**14 D**

→ 問題：444 ページ、本文解説：396 ページ

フェイルオーバーが実行されたとしても、切り替わるまでの数秒から数分間、ElastiCacheへのアクセスは不能になります。ElastiCacheの利用シーンにもありますが、マスターデータのキャッシング用途であれば、キャッシングの元データを参照するなど、要件に応じて、ElastiCacheにアクセス不可の際の適切なエラーハンドリング方法をアプリケーション側へ実装しておく必要があります。この設問では、セッションデータの共有にElastiCacheを利用しており、かつノードダウンにより、ユーザがアプリケーションを利用不能になってクレームが発生したと考えられるため、ElastiCacheへアクセス不能になった場合は、自動フェイルオーバーが完了することを見込んで、しばらく時間をおいてアクセスするようにユーザ側へ通知するのが適当な方法だと考えられます。その他の選択肢の不正解理由は以下のとおりです。

- ・選択肢A：クラスタモードを有効化し、シャーディングしても、フェイルオーバーが発生したノードでは一時的にアクセス不可になるのでゼロダウンタイムは実現できません。
- ・選択肢B、C：自動フェイルオーバーした際、ノードを指示するエンドポイントは自動的に切り替わるのでLambdaで実装する必要はありません。

**15 A**

→ 問題：445 ページ、本文解説：405-406 ページ

SQSでは、キューを作成する際にメッセージ受信待機時間を設定して、ショートポーリングかロングポーリングかメッセージの取得方式を設定できます。ショートポーリングでは0を、ロングポーリングでは1~20秒で指定します。配信遅延時間はキューにメッセージが登録されてから、コンシューマーがメッセージ受信できるようになるまで待機する時間であり、この場合、特に要件に影響は及ぼしません。

**16 B**

→ 問題：445 ページ、本文解説：408 ページ

このケースでは、ロード処理が完了してメッセージを削除する前に、可視性タイムアウトが経過して、別のコンシューマーアプリケーションがメッセージを取得して実行しているものと考えられます。ロード処理は幂等性が担保されているので、ロードが失敗することはありませんが、別のアプリケーションがメッセージを取得したことにより、メッセージのReceiptHandle属性が書き換えられ、メッセージの削除処理でエラーが発生しているため、可視性タイムアウトを余裕を持って設定するのが正解です。

**17 C**

→ 問題：446 ページ、本文解説：409 ページ

FIFOキューでは順序性を保つためにメッセージグループという単位で管理します。FIFOキューへ送信するには、メッセージグループIDを指定することが必要です。よりパフォーマンス効率が高いのは商品ごとに順序性を担保するよう、商品ごとにメッセージグループを構成することです。メッセージ重複排除IDはメッセージの重複を除外するために、同一のメッセージ単位に指定される必要があるIDで、送信時に指定するか、SQSがメッセージの内容をもとに割り振るオプションがあります。

**18 D**

→ 問題：446 ページ、本文解説：406 ページ

SQSでは、ReceiveMessage APIを呼び出す際にMaxNumberOfMessagesパラメータを指定して、最大10件までメッセージを取得することができます。その他の選択肢の不正解理由は以下のとおりです。

- ・選択肢A：ポーリング間隔を短くしても処理効率が向上するとは限りません。プロデューサーアプリケーションが10件程度まとめて送信しているためです。間隔を短くしても取得できないタイミングがあるので効果は限定的です。
- ・選択肢B：キューを2つに分離しても、メッセージを取得できる頻度は変わらないので、効果は限定的です。
- ・選択肢C：このオプションではポーリング間隔を逆に大きめに設定しています。

**19 D**

→ 問題：447 ページ、本文解説：409 ページ

メッセージ重複IDはFIFOキューにメッセージを蓄積する場合に、メッセージに対して指定されるものです。コンシューマーがスタンダードキューでメッセージ重複排除IDを指定したとしても、重複処理の抑止にはつながりません。

**20 B**

→ 問題：447 ページ、本文解説：419 ページ

AWS Certificate Manager(ACM)を使って作成したSSL/TLS証明書は「米国東部(バージニア北部)us-east-1」で作成しておく必要があります。

**21 D、E**

➡ 問題：448 ページ、本文解説：420-423 ページ

この問題で考えなければならないキャッシュの期間は2つあります。1つは(1)オリジンでCache-Control "max-age=3600"が設定されている、更新後に1時間ほどで反映したいコンテンツと、もう1つは(2)オリジンでヘッダが何も設定されていないコンテンツです。

- ・選択肢A：(1)はMin TTL=0のため、Max TTL(86400)がCache-Control(3600)と比較してより小さいCache-Controlの設定値が有効になります。(2)はMin TTL=0なので、デフォルトキャッシュが有効です。したがって、キャッシュ期間が0となり、キャッシュされないので要件を満たしていません。
- ・選択肢B：(1)はMin TTL=86400であり、また、Cache-Control(3600) < Min TTL(86400)であるため、Min TTLがキャッシュ期間として有効になり、更新後に1時間ほどで反映したい要件を満たしていません。
- ・選択肢C：(1)はMin TTL=3600であり、Min TTL(3600) = Cache-Control(3600) < Max TTL(315360000)です。したがって、キャッシュはCache-Controlの設定値が有効になります。(2)は、Min TTL=3600かつDefault TTL=315360000なので、より大きいデフォルトキャッシュが有効です。したがって、キャッシュ期間は1年(315360000秒)となり、1日程度で変更後に反映されればよい要件を満たしていません。
- ・選択肢D：(1)はMin TTL=1であり、Min TTL(1) < Cache-Control(3600) < Max TTL(315360000)です。したがって、キャッシュはCache-Controlの設定値が有効になります。(2)は、Min TTL=1かつDefault TTL=86400なので、より大きいデフォルトキャッシュが有効です。したがって、キャッシュ期間は1日(86400秒)です。
- ・選択肢E：(1)はMin TTL=0のため、Max TTL(315360000)がCache-Control(3600)と比較してより小さいCache-Controlの設定値が有効になります。(2)はMin TTL=0なので、デフォルトキャッシュが有効です。したがって、キャッシュ期間は1日(86400秒)です。

**22 A**

➡ 問題：448 ページ、本文解説：424 ページ

動的なコンテンツをキャッシュ対象にするには、コンテンツが格納されているパスに設定するキャッシュポリシーのキャッシュキーにCookieを指定する必要があります。なお、キャッシュキーに指定されたパラメータはオリジンリクエストに自動的に含まれます。そのため、オリジンリクエストポリシーにCookieを含める必要はありません。

**23 D**

➡ 問題：449 ページ、本文解説：425 ページ

CloudFrontで制限付きアクセスを実現するには、署名付きURL/Cookieを利用します。このオプションを利用するには、まずローカル端末などで、公開鍵・秘密鍵のキーペアを作成します。署名者として作成された「信頼されたキーグループ」に対し、公開鍵を追加します。対となる秘密鍵を使って、SDKを使ってアプリケーションの処理などで署名付きURLやCookieを生成します。署名を作成する際にアクセスポリシーを含めます。キャッシュポリシーではありません。

**24 D**

➡ 問題：449 ページ、本文解説：427 ページ

Lambda@Edgeでは、ビューワーリクエスト、オリジンリクエスト、オリジンレスポンス、ビューワーレスポンスへLambda関数を設定できます。User-Agentに応じて画像をリサイズする処理は、ビューワーリクエストでURLを切り替え、オリジンレスポンスで画像リサイズ処理やオリジン・キャッシュへ保存する処理を設定するのが最も効率的な構成です<sup>※67</sup>。誤った選択肢の不正解理由は、以下のとおりです。

- ・選択肢A：画像をリサイズするLambda関数は、再利用目的でオリジンへ保存したりするため、オリジンレスポンスへ関連付けたほうが効果的です。
- ・選択肢B：User-Agentに応じてURLを切り替えるのは、HTTPヘッダをデフォルトインプットパラメータとして利用できるビューワーリクエストへ関連付けたほうが効果的です。
- ・選択肢C：キャッシュができるだけ長く利用するために、できるだけ長いTTLを設定します。

※67 <https://aws.amazon.com/jp/blogs/news/resizing-images-with-amazon-cloudfront-lambdaedge-aws-cdn-blog/>

**25 C**

➡ 問題：450 ページ、本文解説：431 ページ

シンプルスケーリングポリシーは1つのスケーリング条件しか設定できないポリシーです。複数の条件を設定するときは、ステップスケーリングポリシーを設定します。なお、単一の条件しか設定することがなくてもステップスケーリングポリシーを使用することが推奨されています。

**26 B**

➡ 問題：450 ページ、本文解説：431 ページ

特定の時間帯に負荷が集中することが見込まれる場合は、スケジューリングを設定することにより、指定した条件でスケールアウト・スケールインが可能です。他の選択肢の不正解理由は、以下のとおりです。

- 選択肢A：予測スケーリングは予測に必要なデータが集まるまで時間が必要なので、すぐ利用することはできません。
- 選択肢C：シンプルスケーリングポリシーでは条件が1つしか設定できないため、スケールアウトできてもスケールインができません。
- 選択肢D：CloudWatchイベントでターゲットにAutoScalingを設定することはできません。

**27 A**

➡ 問題：451 ページ、本文解説：431 ページ

ターゲット追跡ポリシーでメモリ使用率は選択できません。

## 第6章

# 総仕上げ問題

**■問題数：65問****■試験時間：130分****28 C**

➡ 問題：451 ページ、本文解説：433 ページ

ライフサイクルフックでアクションを実行した後は、CompleteLifecycleActionを実行し、ステータスを進行させます。タイムアウト時間まで待っても、継続することはできますが、通常終了コマンドを実行します。

**29 B**

➡ 問題：451 ページ、本文解説：434 ページ

Amazon AuroraでリードレプリカのApplication AutoScalingはサポートされていますが、RDSでサポートされるのはストレージのスケーリングだけです。

## 6-1 問題

**1** 開発者は、実行するたびに新しいファイルを生成するAWS Lambda関数を作成しています。新しいファイルはそれぞれ、同じAWSアカウントでホストされているAWS CodeCommitリポジトリにチェックインする必要があります。開発者はこれをどのように達成する必要があるか、選択してください。

- A. Lambda関数が起動したら、Git CLIを使用してリポジトリのクローンを作成します。新しいファイルを複製されたリポジトリにチェックインし、変更をプッシュします。
- B. Lambdaで新しいファイルが作成されたら、cURLを使用してCodeCommit APIを呼び出します。ファイルをリポジトリに送信します。
- C. AWS SDKを使用して、CodeCommitクライアントをインスタンス化します。put\_fileメソッドを呼び出して、ファイルをリポジトリに追加します。
- D. 新しいファイルをAmazon S3バケットにアップロードします。S3イベントを受け入れるAWS Step Functionsを作成します。Step Functionsで、新しいファイルをリポジトリに追加します。

**2** 開発者は開発中のアプリケーションのビルド処理にCodeBuildを利用しようとしていますが、ビルド環境構築時にネットワーク接続エラーが発生していました。原因として考えられるものとして最も適切なものを選択してください。

- A. CodeBuildを実行するためのサービスロールにssm:GetParametersアクションが追加されていません。
- B. CodeBuildを実行する際に、Privilegedモードにしていなかったために、ENIをアタッチできていません。
- C. 実行するプロジェクトに実行環境のVPCおよびサブネットが設定されていません。
- D. 実行環境のサブネットにNAT Gatewayがアタッチされていません。

**3** ある会社ではリリースパイプラインを自動化するためにAWS CodePipelineを実装しようとしています。開発チームは実行されたアクションでステージの変更を通知するAWS Lambda関数を開発しています。どのステップでLambda関数とイベントソースを関連付けるべきか選択してください。

- A. Lambdaコンソールで、CodePipelineをイベントソースとして選択し、Lambda関数を実行するトリガーを作成します。
- B. CodePipelineコンソールで、Lambda関数を指定してイベントトリガーを作成します。
- C. CodePipelineとLambda関数のトリガーのステータス変化を監視するCloudWatchアラームを作成します。
- D. イベントソースとしてCodePipelineを使用するCloudWatchイベントを作成します。

**4** ある社内システムでは、アプリケーションをELBにより負荷分散されたEC2インスタンス上に、CodeDeployを使ってデプロイしています。デプロイはIn-Placeデプロイ方式で行っていますが、あるときデプロイに失敗し、環境がロールバックされました。このときの挙動として誤っている記述を選択してください。

- A. デプロイが失敗した場合、または監視しきい値が満たされた場合に、自動的にロールバックするようデプロイグループを設定されました。
- B. ELBで向き先を切り替えることにより、ロールバック前の環境へは迅速に復旧されます。
- C. アプリケーションは直前のリビジョンにロールバックされます。
- D. デプロイに失敗した原因となるログはロールバックされた環境に上書きされて残っていません。

5 ある企業では、提供するアプリケーションの環境構築をCloudFormationを使って実装しています。テンプレートはネットワーク環境やアプリケーションを実行するインスタンスなど、再構築されるかの有無で分離して実装されています。インスタンス構築テンプレートはVPCやサブネットの情報を必要としており、開発者はスタックからそれらの情報を取得したいと考えています。次の選択肢のうち、別テンプレートの情報を取得できる最も適切な方法を選択してください。

- A. Fn:ImportValueを使って、クロススタックリファレンスで、別テンプレートの情報を参照します。
- B. {{resolve:xxxx}}を使って、ダイナミックリファレンスで、別テンプレートの情報を参照します。
- C. Fn:GetAttを使って、ネスティッドスタックで、別テンプレートの情報を参照します。
- D. Custom::XXXXを使って、カスタムリソースで、別テンプレートの情報を参照します。

6 ある開発者がCloudFormationテンプレートを実装しており、複数のリージョンで展開されるEC2インスタンスリソースがあります。このEC2インスタンスはリージョンごとに異なるAMIで構成されます。テンプレートの記述を簡素化したい場合に最適なセクションの構成要素はどれか、選択してください。

- A. Metadata
- B. Parameters
- C. Mappings
- D. Conditions

7 ある開発者が実装したCloudFormationテンプレートを、シェルスクリプト内でCLI実行しようとしています。スタックは構築された後もたびたび更新されるため、スクリプト化して同一のコマンドで実行されることが求められています。この場合、スタックの構築に最も適したコマンドを選択してください。

- A. aws cloudformation build-stack
- B. aws cloudformation create-stack
- C. aws cloudformation upsert-stack
- D. aws cloudformation deploy

8 AWS Serverless Application Model(SAM)によって定義されたオブジェクトをAWS CloudFormationテンプレートに含めるには、Resourcesに加えて、どのセクションをドキュメントルートに含める必要があるか、選択してください。

- A. Conditions
- B. Globals
- C. Transform
- D. Properties

9 開発者はAWS Serverless Application Model(SAM)を使って、デプロイを実行しようとしています。アプリケーションをデプロイするために必要なコマンドの組み合わせで正しいものを選択してください。

- A. sam package, sam deploy
- B. sam init, sam validate, sam build, sam deploy
- C. sam build, sample deploy
- D. sam init, sam build, sam deploy

10 ある開発者は、新しいモバイルアプリケーションのバックエンド処理をLambdaを使って実装しようとしています。作成したLambda関数を、AWSLambdaBasicExecutionRole管理ポリシーがアタッチされた既存のロールを使用して実行し、Amazon CloudWatchコンソールでLambda関数のログを表示しようとすると、「Log group does not exist」というエラーが表示されます。考えられる原因として誤っているものを選択してください。

- A. 書き込みアクションCreateLogGroupおよびCreateLogStreamが許可されたポリシーがロールにアタッチされていません。
- B. 作成したLambda関数は、デフォルトでCloudWatch Logsは有効化されていません。
- C. 既存ロールにアタッチされたポリシーのResourceで指定されたARNで異なるリージョンが指定されている可能性があります。
- D. 既存ロールにアタッチされたポリシーのResourceで指定されたARNで異なるLambda関数のロググループが指定されている可能性があります。

11 ある企業では、オンプレミスサーバに配置されたアプリケーションがAmazon S3にあるデータへアクセスする要件があります。アプリケーションにS3へのアクセス権限を付与するために必要な方法を選択してください。

- A. S3へのアクセス権限ポリシーをアタッチしたIAMユーザとアクセスキー、シークレットキーを作成し、サーバへクレデンシャルとして設定します。
- B. S3へのアクセス権限ポリシーをアタッチしたIAMロールを作成し、割り当てます。
- C. S3へのアクセス権限ポリシーをアタッチしたIAMグループを作成し、オンプレミスサーバをグループへ追加します。
- D. S3へのアクセス権限ポリシーをアタッチしたIAMユーザとキーペアを作成し、S3アクセス時にキーペアを使用します。

12 ある企業では、プロダクション環境と開発環境に対し、AWSアカウントを分離して管理しています。プロダクション環境のデプロイのために、プロダクションアカウントから開発アカウントのS3バケット内にある資材にアクセスしようとしています。必要な設定のうち正しいものを選択してください。

- A. プロダクションアカウントと開発アカウントでそれぞれIAMロールを作成し、開発アカウントのIAMロールの信頼ポリシーのプリンシパルにプロダクションアカウントのIAMロールを設定し、AssumeRole APIを呼び出します。
- B. プロダクションアカウントと開発アカウントでそれぞれIAMロールを作成し、プロダクションアカウントのIAMロールの信頼ポリシーのプリンシパルに開発アカウントのIAMロールを設定し、AssumeRole APIを呼び出します。
- C. プロダクションアカウントと開発アカウントでそれぞれIAMロールを作成し、開発アカウントのIAMロールの信頼ポリシーのプリンシパルにプロダクションアカウントのIAMロールを設定し、SessionToken APIを呼び出します。
- D. プロダクションアカウントと開発アカウントでそれぞれIAMロールを作成し、プロダクションアカウントのIAMロールの信頼ポリシーのプリンシパルに開発アカウントのIAMロールを設定し、SessionToken APIを呼び出します。

- 13 ある企業では、プロダクション環境と開発環境に対し、AWSアカウントを分離して管理しています。プロダクション環境のデプロイのために、プロダクションアカウントから開発アカウントのS3バケット内にある資材にアクセスしようとしています。開発アカウントのロールにおける、S3バケットへアクセスするポリシーで必要な設定のうち正しいものを選択してください。

A. アクセスポリシー

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "AllowDevelopmentAccountBucketAccess",
        "Effect": "Allow",
        "Action": [
            "s3:GetObject"
        ],
        "Principal": {"AWS": "arn:aws:iam::XXXX:role/ProductionRole"}
        "Resource": "arn:aws:s3:::SampleDevelopmentBucket/*"
    }
}
```

B. アクセスポリシー

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "AllowDevelopmentAccountBucketAccess",
        "Effect": "Allow",
        "Action": [
            "s3:GetObject"
        ],
        "Resource": "arn:aws:s3:::SampleDevelopmentBucket/*"
    }
}
```

C. 信頼ポリシー

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "AllowDevelopmentAccountBucketAccess",
        "Effect": "Allow",
        "Action": [
            "s3:GetObject"
        ],
        "Resource": "arn:aws:s3:::SampleDevelopmentBucket/*"
    }
}
```

D. 信頼ポリシー

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "AllowDevelopmentAccountBucketAccess",
        "Effect": "Allow",
        "Action": [
            "s3:GetObject"
        ],
        "Principal": {"AWS": "arn:aws:iam::XXXX:role/ProductionRole"}
        "Resource": "arn:aws:s3:::SampleDevelopmentBucket/*"
    }
}
```

14 開発チームはユーザがAmazon S3バケットから画像を取得して表示するEC2アプリケーションを開発しています。画像へのアクセスは大量に発生することが見込まれます。最もパフォーマンス効率のよい安全な実装方法を選択してください。

- A. アプリケーションの中でSDKを使って、STS AssumeRole APIを呼び出し、一時認証情報を使ってS3バケットにある画像データにアクセスし、クライアントユーザへ画像データを返却します。
- B. アプリケーションの中でSDKを使って、STS AssumeRole APIを呼び出し、一時認証情報が設定されたS3クライアントを使って、有効期限が設定された署名付きURLを生成し、クライアントユーザへURLを返却します。
- C. アプリケーション向けのIAMユーザを作成し、EC2にアクセスキー、シークレットキーを設定した上で、SDKを使ってS3バケットにある画像データにアクセスし、クライアントユーザへ画像データを返却します。
- D. EC2向けのIAMロールを作成し、S3へアクセスするポリシーをアタッチしておきます。インスタンスプロファイルからIAMロールを引き受けて、SDKを使ってS3バケットにある画像データにアクセスし、クライアントユーザへ画像データを返却します。

15 アプリケーションは、Amazon EC2インスタンスのクラスタで実行されています。Amazon S3バケット内に保存されているオブジェクトを読み取ろうとしたときに、アプリケーションがエラーを受け取りました。バケットは、サーバ側の暗号化とAWS KMS管理対象キー(SSE-KMS)で暗号化されています。エラーは次のとおりです。

サービス : AWS KMS; ステータスコード : 400、エラーコード :

ThrottlingException

この障害を防ぐには、どの手順の組み合わせを実行する必要があるか、2つ選択してください。

- A. AWSサポートに連絡して、AWS KMSレート制限の引き上げをリクエストします。
- B. アプリケーションコードでエクスボンシャルバックオフを使用してエラーの再試行を実行します。
- C. AWSサポートに連絡して、S3レート制限の引き上げをリクエストします。
- D. より大きなキーサイズのカスタマーマスターキー(CMK)をインポートします。
- E. S3データを暗号化するために複数回のカスタマーマスターキー(CMK)を使用します。

16 ある地方公共団体では、住民の個人情報を含むPDFファイルをAmazon S3へ保存しようとしています。このファイルは法律上暗号化して保存されなければならず、また暗号化に使用するキーは、団体の首長が責任を持って保存することが要件として求められています。団体が採用すべき暗号化戦略のうち、適切なものを選択してください。

- A. SSE-S3によりサーバサイド暗号化を使用します。
- B. SSE-KMSによりサーバサイド暗号化を使用します。
- C. SSE-Cによりサーバサイド暗号化を使用します。
- D. カスタマーマスターキー(CMK)を使用したクライアントサイド暗号化を使用します。

17 ある開発者はモバイルアプリケーションにおいて、サインアップとサインインを追加する必要があります。ソリューションはソーシャルアイデンティティプロバイダとSAML IdPsと統合しなければなりません。どのサービスをユーザは利用できるか、選択してください。

- A. AWS Cognitoユーザプール
- B. AWS CognitoIDプール
- C. Lambdaオーソライザ付きAPI Gateway
- D. AWS IAMとSTS

18 ある企業はAmazon API Gateway、AWS LambdaおよびAmazon DynamoDBを使ってサーバレスアプリケーションを構築しています。このアプリケーションはさまざまなデバイスから構成されますが、セキュリティ要件として2段階認証が必要です。この要件を実現するために最も適切なサービスはどれか選択してください。

- A. AWS IAM
- B. AWS KMS
- C. Amazon Cognito
- D. AWS Lambdaオーソライザ

19 ある企業では社員の税務書類データをAmazon S3に保存しています。データは保存時に暗号化する必要があります、暗号キーは毎年ローテーションする必要があります。必要最小限の労力でこの要件を実現するために最も適したサービスを選択してください。

- A. AWS Secret Manager
- B. AWS KMS
- C. AWS Systems Manager Parameter Store
- D. AWS Cloud HSM

20 ECSコンテナ上に実行されているアプリケーションはAmazon RDSへアクセスしています。アプリケーションは30日ごとに接続文字列を自動ローテーションする必要があります。この要件を満たすために、最も簡単な方法はどれか選択してください。

- A. ECSの環境変数に接続文字列を保存し、自動でローテーションするデーモンスクリプトを起動しておきます。
- B. Systems Manager Parameter Storeに接続文字列を保存し、自動ローテーションを有効にします。
- C. AWS Secret Managerに接続文字列を保存し、自動ローテーションを有効にします。
- D. AWS KMSに接続文字列を保存し、自動ローテーションを有効にします。

21 ある企業では、来月リリース予定のアプリケーションのセキュリティ担保のために、セキュリティホールや脆弱性、ベストプラクティスからの逸脱がないか評価したいと考えています。アプリケーションはEC2上にデプロイされます。これに最も適したサービスを1つ選択してください。

- A. Amazon Detective
- B. Amazon Macie
- C. Amazon GuardDuty
- D. Amazon Inspector

22 ある企業では、来月リリース予定のアプリケーションのセキュリティ担保のために、DDoS対策を施したいと考えています。アプリケーションはEC2上にデプロイされ、ELBがフロントに配置されます。これに最も適したサービスを1つ選択してください。

- A. AWS WAF
- B. AWS Shield
- C. Amazon GuardDuty
- D. AWS Firewall Manager

23 ある企業では、来月リリース予定のアプリケーションのセキュリティ担保のために、AWS CloudTrail、Amazon VPCフローログ、DNSログなどを分析して、不正な攻撃を検知したいと考えています。これに最も適したサービスを1つ選択してください。

- A. Amazon Inspector
- B. Amazon Detective
- C. Amazon Macie
- D. Amazon GuardDuty

24 ある企業では、来月リリース予定のアプリケーションのセキュリティを向上させたいと考えています。アプリケーションはS3上に個人情報を含むデータが保存されます。そのため、暗号化が行われていないデータなどコンプライアンスに沿わない不備を検出したいと考えています。これに最も適したサービスを1つ選択してください。

- A. Amazon Inspector
- B. Amazon Detective
- C. Amazon Macie
- D. AWS CloudTrail

25 ある企業では、来月リリース予定のアプリケーションの運用方法を検討しています。セキュリティインシデントがあった際にAWS CloudTrail、Amazon VPCフローログ、DNSログなどを分析して、効率的な調査を行うためにインテラクティブに視覚化しておきたいと考えています。これに最も適したサービスを1つ選択してください。

- A. Amazon Inspector
- B. Amazon Detective
- C. Amazon CloudWatch Logs
- D. Amazon CloudWatch Insights

26 ある企業は、不特定多数のクライアントに対し、自社のデータを取得できるREST APIをAPI Gatewayで公開しています。一部のデータは非常に希少性が高いことから、オーナーはこのAPIへのアクセスを有料サービス化して、アクセス回数の上限を設定したいと考えています。これを実現するために最も簡単な方法を選択してください。

- A. CognitoとAPI Gatewayを組み合わせて、アクセストークンを払い出します。アクセストークンを使って有料サービスを契約したユーザのみが指定した回数分だけアクセスできるよう構成します。
- B. LambdaとAPI Gatewayを組み合わせて、リクエストの認証情報をチェックします。アクセス回数を判定するロジックを関数内に実装し、指定した回数分だけアクセスできるよう構成します。
- C. APIキーを使用して、アクセス可能な使用プランを作成し、APIステージと関連付けます。
- D. OIDCプロバイダとAPI Gatewayを連携して、IDトークンを払い出します。トークンのユーザクレームから有料サービスを契約したユーザを判定し、指定した回数分だけアクセスできるよう構成します。

27 ある企業は、不特定多数のクライアントに対し、自社のデータを取得できるREST APIをAPI Gatewayで公開しています。一部のAPIにアクセスが集中しており、他のリクエストを含めてスロットリングエラーが頻発しているため、開発者はアクセスが多いAPIに対し、単位時間のリクエスト数を制限したいと考えています。これを実現するために最も簡単な方法を選択してください。

- A. LambdaとAPI Gatewayを組み合わせて、リクエスト数をチェックします。実行中のリクエスト数を判定するロジックを関数内に実装し、指定したリクエスト数分だけアクセスできるよう構成します。
- B. APIがデプロイされているステージに対し、リクエストの制限値を設定します。
- C. AWSサポートに連絡し、リクエストの制限値を設定してもらいます。
- D. AWS WAFを設定し、指定されたリクエストの流量を制御します。

28 ある企業は、不特定多数のクライアントに対し、自社のデータを取得できるAPIをAPI Gatewayで公開しようとしています。APIはX(旧Twitter)やFacebook、Instagramなどのソーシャルメディアと連携してアクセス制御したいと考えています。開発者が選択する実装方法のうち、最も低コストで実現可能な方法を選択してください。

- A. API GatewayでREST APIを構成します。OIDCプロバイダとしてソーシャルメディアを設定し、アクセス制御します。
- B. API GatewayでHTTP APIを構成します。OIDCプロバイダとしてソーシャルメディアを設定し、アクセス制御します。
- C. API GatewayでLambdaオーソライザを構成します。Lambda関数内でソーシャルメディアへアクセスし、認証情報にもとづいてアクセス制御します。
- D. API GatewayでCognitoオーソライザを構成します。CognitoのIDプロバイダとしてソーシャルメディアを設定し、アクセス制御します。

29 ある企業は、不特定多数のクライアントに対し、自社のデータを取得できるAPIをAPI Gatewayで公開しています。APIをアップデート後に、ユーザから古いデータが見えているとクレームがありました。問題を解決する事項で正しいものを選択してください。

- A. APIを呼び出すリクエストで”Cache-Control: max-age = 0”のHTTPヘッダを設定するように依頼する。
- B. APIを呼び出すリクエストでFlush entire cache APIを呼び出すように依頼する。
- C. API Gatewayの設定でキャッシングエントリを無効化する。
- D. APIを呼び出すリクエストでInvalidateCacheポリシーをBase64エンコードして、SUCCEED\_WITHOUT\_RESPONSE\_HEADERヘッダに設定するよう依頼する。

30 ある開発者はAmazon SNSによって実行され、連続的に失敗し続けているLambda関数のトラブルシューティングを実行しています。先に進めるために廃棄されたイベントを保護する方法を選択してください。

- A. Lambda関数のためのCloudWatch Logsを有効化します。
- B. デッドレターキュー(DLQ)を設定します。
- C. Lambda Streamsを有効化します。
- D. イベント失敗時のSNS通知を有効化します。

31 開発者がLambda関数のコードを更新しています。開発者は、少量のトランザクションを新しいバージョンに転送することで、コードの更新をテストしようと注力しています。どのように達成するのがベストな方法か選択してください。

- A. 関数コードの2つのバージョンを作成します。リクエストのサブセットを新しいバージョンに転送するようにアプリケーションを構成します。
- B. 関数コードの新しいバージョンと以前のバージョンの両方を指すエイリアスを作成し、トランザクションの一部を新しいバージョンに送信するための重みを割り当てます。
- C. API Gatewayを使用してAPIを作成し、ステージ変数を使用してLambda関数のさまざまなバージョンをポイントします。
- D. 新しいコードを使用して新しい関数を作成し、アプリケーションを更新して、新しい関数間でリクエストを分割します。

32 開発者は、標準のLambdaライブラリに含まれていない外部ライブラリを使用する複数のAWS Lambda関数を作成しています。これらのライブラリを関数で使用できるようにするための最良の方法を選択してください。

- A. 関数コードに外部ライブラリを含めます。
- B. 外部ライブラリを含むデプロイメントパッケージを作成します。
- C. ファイルをAmazon S3に保存し、関数コードから参照します。
- D. 外部ライブラリを含むLambdaレイヤーを作成します。

33 Lambda関数は完了までに長い時間がかかることがあります。開発者は関数が割り当てられているコンピューティングのキャパシティが不十分なことに気づきました。開発者がよりコンピューティングのキャパシティを確保して関数を割り当てるための適切な方法を選択してください。

- A. 関数により大きなメモリを割り当てます。
- B. より高性能なCPUのインスタンスタイプを使用します。
- C. 実行時間を最大化します。
- D. 予約並行性を増加させます。

34 アプリケーションは複数のLambda関数を使用して、Amazon RDSデータベースにデータを書き込んでいます。Lambda関数は同じ接続文字列を共有する必要があります。セキュリティと運用効率を確保するための最良のソリューションを選択してください。

- A. 各Lambda関数内でKMS暗号化環境変数を使用します。
- B. AWS Systems Manager Parameter Storeを使用してセキュアな文字列パラメータを作成します。
- C. 関数間で共有されるCloudHSM暗号化環境変数を使用します。
- D. 接続文字列をLambda関数コードに埋め込みます。

35 ある開発者がAmazon EC2インスタンスのオートスケーリンググループ上で実行されるアプリケーションをデプロイしました。アプリケーションのデータはDynamoDBテーブルに保存され、レコードはすべてのインスタンスにより常に更新されています。インスタンスは時々古いデータを参照しています。開発者は強い一貫性読み込みによってこれを補正したいと考えています。開発者はこれをどのように達成できるか、選択してください。

- A. GetItemをコールするとき、ConsistentReadをTrueにセットします。
- B. 新しくDynamoDB Accelerator (DAX) テーブルを作成します。
- C. UpdateTableをコールするとき一貫性をStrongに設定します。
- D. GetShardIteratorコマンドを使用します。

36 開発者は、Amazon DynamoDBにアクセスするAWS Lambda関数に取り組んでいます。Lambda関数は、項目を取得してその属性の一部を更新するか、項目が存在しない場合は作成する必要があります。Lambda関数は主キーにアクセスできます。この機能を実現するために、開発者はLambda関数に対してどのIAM権限を要求する必要があるか、選択してください。

- A. "dynamodb:DeleteItem", "dynamodb:GetItem", "dynamodb:PutItem"
- B. "dynamodb:UpdateItem", "dynamodb:GetItem", "dynamodb:DescribeTable"
- C. "dynamodb:GetRecords", "dynamodb:PutItem", "dynamodb:UpdateTable"
- D. "dynamodb:UpdateItem", "dynamodb:GetItem", "dynamodb:PutItem"

37 開発者は、OAuth 2.0のリフレッシュトークンを保存するために、Amazon DynamoDBテーブルを利用するアプリケーションを作成しています。保存するデータサイズは約4.5KBだと見積もられており、トークンは結果整合性のある読み込みが1秒あたり10回、標準の書き込みを6回行います。この場合のRCUs/WCUsで正しい組み合わせを選択してください。

- A. 20RCUs / 30WCUs
- B. 20RCUs / 60WCUs
- C. 10RCUs / 30WCUs
- D. 5RCUs / 15WCUs

38 あるECサイトは商品の在庫情報をプロビジョンドモードのDynamoDBに保存しています。在庫状況を整理するために、在庫テーブルに対してスキャンAPIを実行するバッチ処理がたびたび実行されますが、この処理の実行中にECサイト側からのパフォーマンスが低下していると報告がありました。パフォーマンス改善のために、最も適切な方法を選択してください。

- A. スキャン時にページサイズを小さくするよう設定します。
- B. スキャンAPIをパラレル実行します。
- C. 在庫テーブルにソートキーを設定します。
- D. 在庫テーブルにグローバルセカンダリインデックスを設定します。

39 あるECサイトは購買トランザクションデータの保存にAmazon DynamoDBを利用しています。トランザクションは一定時間に集中するため、キャパシティユニットを動的に変更したいと考えています。これを実現するために必要な正しいものを選択してください。

- A. リクエスト数に応じたオンデマンドキャパシティモードを選択します。
- B. プロビジョンドスループットでオートスケーリングを設定します。
- C. CLIやSDKでプロビジョンドスループットを変更します。
- D. 動的プロビジョンドスループットを選択します。

40 あるグローバル企業は、EC事業をさまざまなリージョンで展開しており、サイトを訪れるユーザに対し、最もハイパフォーマンスなレスポンスで応答したいと考えています。この場合、最も適当なルーティングポリシーはどれか、選択してください。

- A. 位置情報ルーティングポリシーを設定します。
- B. 地理的近接性ルーティングポリシーを設定します。
- C. レイテンシベースドルーティングポリシーを設定します。
- D. 複数回答ルーティングポリシーを設定します。

41 ある会社が、Elastic Load Balancer(ELB)の背後でAmazon EC2上でWebアプリケーションを実行しています。同社はWebアプリケーションのセキュリティを懸念しており、SSL証明書でアプリケーションを保護したいと考えています。このソリューションは、EC2インスタンスのパフォーマンスに影響を与えることはありません。Webアプリケーションを保護するためにどのような手順を実行する必要があるか、2つ選択してください。

- A. SSLパスルートを使用してElastic Load Balancerを構成します。
- B. SSL証明書をElastic Load Balancerに追加します。
- C. EC2インスタンスにSSL証明書をインストールします。
- D. SSLターミネーション用にElastic Load Balancerを構成します。
- E. KMS管理キーを使用してサーバ側の暗号化を構成します。

42 ある企業では、オンプレミスサーバで動作している社内業務アプリケーションをAWSへ移行することを考えています。移行はアプリケーション自体はそのままEC2へリフトし、ロードバランサーとRDBをAWSマネージド管理とする予定です。アプリケーションはステートフルに動作するため、リクエストは常に同じEC2インスタンスで処理される必要があります。この場合、必要な設定で正しいものを選択してください。

- A. ELBでスティッキーセッションを有効にします。
- B. ELBへのRoute 53ルーティングポリシーをセッションアフィニティポリシーで設定します。
- C. ELBのターゲットグループのルーティングアルゴリズムをLOR(Least Outstanding Requests: 最小未処理リクエスト)で設定します。
- D. L4負荷分散を実現するNLBを導入し、Elastic IPアドレスを固定するよう設定します。

43 ある開発者がEC2インスタンスのオートスケーリンググループを構成しており、カスタムメトリクスをAmazon CloudWatchに送信する必要があります。この場合の設定で最も適切なものを選択してください。

- A. CloudWatchAgentServerPolicyがアタッチされたIAMロールを作成して、そのロールを使用してEC2インスタンスを起動します。
- B. CloudWatchAgentServerPolicyがアタッチされたIAMロールを作成して、そのロールを使用してオートスケーリンググループ起動構成を作成します。
- C. CloudWatchAgentServerPolicyがアタッチされたIAMユーザを作成して、インスタンスのユーザデータに認証情報を設定してEC2インスタンスを起動します。
- D. CloudWatchAgentServerPolicyがアタッチされたIAMユーザを作成して、インスタンスのユーザデータに認証情報を設定してオートスケーリンググループ起動構成を作成します。

44 開発チームがAmazon ECS上にコンテナワークロードを実行させたいと考えています。それぞれのアプリケーションコンテナはログやメトリクスの収集を行うために別のコンテナでデータ共有する必要があります。開発チームはこれらの要求を満たすために何をすべきか、選択してください。

- A. 2つのPod Specificationを作ります。1つはアプリケーションコンテナを含むもので、もう1つは別のコンテナを含むもので、2つのポッドをリンクさせます。
- B. 2つのタスク定義を作成します。1つはアプリケーションコンテナを含むもので、もう1つは別のコンテナを含むもので、2つのタスクで共有のボリュームをマウントします。
- C. 1つのタスク定義を作成します。定義内で両方のコンテナを指定し、2つのコンテナの間で共有ボリュームをマウントします。
- D. 1つのPodSpecificationを作ります。Specification内に2つのコンテナを使用し、両方のコンテナに永続ボリュームをマウントします。

**45** ある会社がAmazon ECS上で実行される新しいオンラインゲームを開発しています。4つの異なるECSサービスはアーキテクチャの一部となり、それぞれがさまざまなServiceへの特定の権限を必要としています。会社はメモリ予約にもとづきコンテナをbin packingによってEC2インスタンス上に配置して使用を最適化したいと考えています。どの設定が開発チームの要求を安全に満たすことができるか、選択してください。

- A. さまざまなECSサービスにおいて必要な権限を含むIAMインスタンスプロファイルを作成し、配置されるEC2インスタンスへロールを関連付けます。
- B. 4つの異なるIAMロールを作成し、それぞれに関連するECSサービスに必要な権限を含めてから、関連するIAMロールを参照するように各ECSサービスを構成します。
- C. 4つの異なるIAMロールを作成し、それぞれに関連するECSサービスに必要な権限を含めてから、IAMグループを作成してグループへECSクラスタを参照するよう設定します。
- D. 4つの異なるIAMロールを作成し、それぞれに関連するECSサービスに必要な権限を含めてから、それぞれECSタスクにIAMロールを関連付けて設定します。

**46** ある企業では、オンプレミスサーバで動作しているアプリケーションをAWSへ移行することを考えています。アプリケーションはセッションにさまざまなビジネスデータを保存しており、ステートフルに動作します。そのため、セッションデータの保存先を、共有キャッシュであるElastiCache (Redis)へ移行することを検討しています。共有キャッシュがダウンするとクリティカルなエラーとなるため、可能な限り最小コストで可用性を向上させたいと考えています。どの設定が最も適した解決策か、選択してください。

- A. レプリカを作成せず、クラスタモードを有効化し、設定エンドポイントを参照します。
- B. レプリカを作成して、自動フェイルオーバーおよびマルチAZを有効にし、プライマリエンドポイントを参照します。
- C. レプリカを作成せず、自動フェイルオーバーを有効にし、プライマリエンドポイントを参照します。
- D. レプリカを作成して、クラスタモードおよびマルチAZを有効化し、設定エンドポイントを参照します。

**47** ある企業では、既存のWebアプリケーションをマイクロサービスアーキテクチャに移行しています。データベースをAmazon Auroraとし、CQRSパターンにもとづき、サービスをコマンドとクエリで責務分離することを検討しています。次の構成で適切なものを選択してください。

- A. 読み取りのサービスはAuroraの読み取りエンドポイントを参照し、書き込みのサービスはクラスタエンドポイントを参照します。
- B. 読み取り、書き込みのサービスの双方で、プライマリエンドポイントを参照します。
- C. 読み取りのサービスはAuroraのリードレプリカエンドポイントを参照し、書き込みのサービスはDBエンドポイントを参照します。
- D. 読み取り、書き込みのサービスの双方で、設定エンドポイントを参照します。

48 あるリテール企業では自社で取り扱う商品をECサイトでも販売しています。ECサイトはオムニチャネルで構成され、モバイルやPCなどさまざまなクライアントがサポートされています。この企業では、ECサイトの商品画像をAmazon S3で管理していますが、今回画像をアップロードした後にさまざまなクライアント向けに画像をリサイズする処理を実装したいと考えています。プロセスはなるべく自動化し、リサイズ処理の成否を管理者に通知する必要があります。最も低コストで実装できる適切な構成を選択してください。

- A. 特定のS3バケットを監視するLambda関数を実装します。画像がアップロードされたら、監視する関数がリサイズするLambda関数を同期的に呼び出し、リサイズ処理が完了したら、管理者への通知を行うSNSトピックを、正常終了とエラー終了双方、通知先に設定します。
- B. 特定のS3バケットへのPutObjectに対しイベントを設定して、SQS標準キューを通知先として設定します。画像をリサイズするLambda関数がSQSキューをポーリングして呼び出し、リサイズ処理が完了したら、管理者への通知を行うSNSトピックを、正常終了とエラー終了双方、送信先に設定します。
- C. 特定のS3バケットへのPutObjectに対しイベントを設定して、SQS標準キューを通知先として設定します。画像をリサイズするLambda関数がSQSキューをポーリングして呼び出し、リサイズ処理が完了したら、管理者への通知を行う処理を呼び出し、エラーの場合、SQSのデッドレターキューに設定されて、管理者に通知されます。
- D. 特定のS3バケットを監視するLambda関数を実装します。画像がアップロードされたら、監視する関数がリサイズするLambda関数を同期的に呼び出し、リサイズ処理が完了したら、管理者への通知を行う処理を呼び出し、エラーの場合、SQSのデッドレターキューに設定して管理者に通知します。

49 ある企業では、自社セミナーの参加者向けに資料をS3で共有しようとしています。セミナー資料には、オープンな情報しか含まれていないため、参加者にパブリックアクセスできるURLを通知します。S3バケットのアクセス設定で最も適切なものを選択してください。

- A. 説明用の資料オブジェクトに対し、事前定義されたACLで、パブリックアクセスを付与します。
- B. バケットポリシーでAction要素に「s3:\*」の実行権限と、Principalに「\*」、Resourceへ説明用の資料オブジェクトを指定します。
- C. バケットポリシーでAction要素に「s3:GetObject」の実行権限と、Principalに「\*」、Resourceへ説明用の資料オブジェクトを指定します。
- D. 説明用の資料オブジェクトに対し、被付与者として「All Usersグループ」を指定し、アクセス許可として「FULL\_CONTROL」を指定します。

50 ある企業では、VPCプライベートサブネット上で展開するアプリケーションで、処理中に扱うファイルを保存するためにS3を利用することを検討しています。ファイルは機密性が高く、ネットワークはインターネット環境に接続することが許されていません。アクセス設定で最も適切なものを選択してください。

- A. S3へのアクセスを可能にするピアツーピアVPNを設定します。
- B. S3へのアクセスでインターフェース型VPCエンドポイントを設定します。
- C. S3へのアクセスでゲートウェイ型VPCエンドポイントを設定します。
- D. S3へのアクセスを可能にするNAT Gatewayを設定します。

51 あるベンチャー企業は、有料会員のみが利用できるプレミア配信動画を自社のWebサイトで提供しています。動画はAmazon S3で管理されており、Webサイトを訪れる不特定多数の無料ユーザが閲覧できないようアクセス制御を施しています。次の選択肢のうち、最もこの要件に沿ったアクセス制御設定を選択してください。

- A. Authenticated Usersグループとして、認証済みのユーザに対しアクセスを許可するようACLを構成します。
- B. 有料会員からの要求に従ってプレミア動画をダウンロードさせるよう、有効期限が設定された署名URLをSDKで生成します。
- C. IAM認証済みのユーザに対し、アクセスを許可するようバケットポリシーを構成します。
- D. IAM認証済みのユーザに対し、アクセスを許可するようユーザポリシーを構成します。

52 ある企業では、既存のオンラインから起動されるバッチアプリケーションをクラウドネイティブ化しようとしています。オンラインからの起動要求をSQSキューに蓄積し、EC2インスタンス上に配置されたバッチアプリケーションがポーリングして処理を実行します。起動タイミングは30秒ほどの間隔で断続的に続いている。ポーリングによるAPI実行コストを最小化するために最も適切なアプローチを選択してください。

- A. キューのVisibilityTimeoutを20[s]に設定します。
- B. キューのReceiveMessageWaitTimeSecondsを20[s]に設定します。
- C. キューのMessageRetentionPeriodを20[s]に設定します。
- D. キューのDelaySecondsを20[s]に設定します。

53 あるECサイトでは、顧客からの注文受付にAmazon SQS標準キューを使用しています。注文を受け付けた後の在庫の引き当てや決済、配送処理は幕等性が担保されておらず、たびたび2度注文される問題が発生しています。この問題を最小限のコストで解決する方法を選択してください。

- A. 標準キューのままで、MessageDeduplicationIdが重複しないようコンシューマーを構成します。
- B. 標準キューのままで、MessageDeduplicationIdが重複しないようプロデューサーを構成します。
- C. FIFOキューを使用して、MessageDeduplicationIdが重複しないようプロデューサーを構成します。
- D. FIFOキューを使用して、MessageDeduplicationIdが重複しないようコンシューマーを構成します。

54 ある開発者は、SQSキューにポーリングするLambda関数が同期的に実行される場合に、発生したエラーを管理者へ通知する方法を検討しています。Lambda関数で3回エラーが発生した際に、SQSのデッドレターキューを用いて、SNSトピックを使って管理者に通知しようと考えています。この場合に必要な設定で正しいものを選択してください。

- A. Lambda関数のデッドレターキュー設定で、SNSトピックを設定しておきます。
- B. Lambda関数の送信先設定で、OnFailure時に、SNSトピックを設定しておきます。
- C. SQSでRedrivePolicyを設定します。最大受信回数(MaxReceiveCount)を3に設定し、デッドレターキューのARNで新たなキューを指定します。
- D. SQSでRedrivePolicyを設定します。最大受信回数(MaxReceiveCount)を3に設定し、デッドレターキューのARNでLambdaのデッドレターキューを指定します。

55 ある電力事業者は、スマートメーターのデータを収集し、Amazon Kinesis Data Streamで処理を実行しようと考えています。ストリームは複数のシャードで構成され、EC2インスタンスが各シャードのストリームデータを処理します。パフォーマンスを最大化するためのEC2インスタンスの構成について正しい考え方を選択してください。

- A. シャードの数とKCLワーカーが実行されるEC2インスタンスの数を1:1にします。
- B. シャードの数とKCLワーカーの数を一致させます。複数のKCLワーカーが1つのシャードを並列処理できるようEC2インスタンスを構成します。
- C. 複数のシャードに対し、複数のKCLワーカーを対応付けます。複数のKCLワーカーが1つのシャードを並列処理できるようEC2インスタンスを構成します。
- D. 複数のシャードに対し、複数のKCLワーカーを対応付けます。1つのKCLワーカーが1つのシャードを処理できるようEC2インスタンスを構成します。

56 開発者はアプリケーションを更新しました。アプリケーションは世界中でユーザーへサービスを提供し、CloudFrontを使用してより近いユーザーにコンテンツをキャッシュしています。アプリケーションを更新してデプロイしたのち、最新の変更が反映されないユーザーがいるとの報告を受けました。開発者はこの問題をどう解決すべきか、選択してください。

- A. CloudFront設定からオリジンを排除して再度追加します。
- B. CloudFront Distribution設定からクエリ文字列のフォワードとリクエストヘッダを無効化します。
- C. エッジキャッシュからのアプリケーションオブジェクトを無効化します。
- D. CloudFront Distributionを無効化し、エッジロケーションの反映を取り込むよう有効化します。

57 会社のWebアプリケーションは、Amazon S3バケットに写真や動画を安全にアップロードすることが求められています。コンテンツのアップロードは、署名されたユーザのみに許可する必要があります。アプリケーションはブラウザを介して、オブジェクトをアップロードするために署名付きURLを生成します。ところがWebアプリケーションを利用している多数のユーザは、100MBを超えるオブジェクトのアップロードにかかる時間が遅いと報告しています。アップロードのパフォーマンスを改善し、認証されたユーザのみにコンテンツのアップロードを許可するための最適な方法を選択してください。

- A. リージョンAPIエンドポイントをS3プロキシのリソースに接続し、Amazon API GatewayのエンドポイントタイプをリージョンAPIエンドポイントに設定します。リソースに対するメソッドとして、Amazon S3バケットにオブジェクトをアップロードするPutObjectを公開します。AWS Lambdaオーソライザを使用して、API Gatewayを保護します。ブラウザを介して、署名付きURLではなく、API Gatewayを使用してオブジェクトをアップロードします。
- B. Amazon CloudFront ディストリビューションを作成して、オリジンサーバにAmazon S3バケットを設定します。PUTおよびPOSTメソッドを処理するようにCloudFrontを構成し、CloudFrontオリジンを更新して、オリジンアクセスアイデンティティ(OAI)を使用します。OAIユーザには、バケットポリシーのS3 PutObject権限を付与します。ブラウザを介してCloudFrontディストリビューションを使用してオブジェクトをアップロードします。
- C. S3バケットでS3 Transfer Accelerationエンドポイントを有効にします。エンドポイントを使用して、署名付きURLを生成します。S3マルチパートアップロードAPIを使用して、ブラウザを介してこのURLにオブジェクトをアップロードします。
- D. エッジ最適化APIエンドポイントをS3プロキシのリソースに接続し、Amazon API Gatewayのエンドポイントタイプをエッジ最適化APIエンドポイントに設定します。リソースに対するメソッドとして、Amazon S3バケットにオブジェクトをアップロードするPutObjectを公開します。また、COGNITO\_USER\_POOLSタイプのオーソライザを使用して、API Gatewayを保護します。ブラウザを介して、署名付きURLではなく、API Gatewayを使用してオブジェクトをアップロードします。

58 企業はAmazon CloudFrontを使用して、世界中のユーザにアプリケーションコンテンツを配信しています。開発者がオリジン内のいくつかのファイルを更新しましたが、ユーザはまだ古いファイルを取得していると報告しています。開発者は、キャッシング内の古いファイルが中断を最小限に抑えて置き換えられるようにするにはどうすればよいか、選択してください。

- A. エッジキャッシングからのファイルを無効にします。
- B. 新しいファイルで新しいオリジンを作成し、古いオリジンを削除します。
- C. CloudFrontディストリビューションを無効にし、再度有効にして、すべてのエッジロケーションを更新します。
- D. キャッシュ内のファイルを更新するコードをLambda@Edgeに追加します。

59 開発チームはEC2インスタンス上で実行されているアプリケーションの負荷が高まるとオートスケーリングするように設定を行いたいと考えています。常に、CPU使用率が60%を維持し続けるようにオートスケーリンググループを構成するには、どのタイプのスケーリングポリシーが最適か、選択してください。

- A. ステップスケーリングポリシー
- B. シンプレスケーリングポリシー
- C. 予測スケーリングポリシー
- D. ターゲット追跡スケーリングポリシー

60 ある会社が最近複数のアプリケーションをAWSへ移行しました。Web層はEC2インスタンスのオートスケーリンググループ、データベース層はDynamoDBを使用しています。データベース層は極めて高いパフォーマンスが要求され、ほとんどのリクエストは繰り返しの読み取りリクエストです。どのサービスがデータベースを最もパフォーマンスよくスケールさせるために使用されるか、選択してください。

- A. Amazon CloudFront
- B. Amazon ElastiCache
- C. Amazon DynamoDB Accelerator (DAX)
- D. Amazon SQS

61 開発チームは、ハイブリッドクラウド環境を管理しています。オンプレミスサーバとAmazon EC2インスタンスからシステムレベルの指標を収集したいと考えています。開発チームはどのようにしてこの情報を最も効率的に収集できるか、選択してください。

- A. CloudWatchを使用して、put-metric-data APIを使用して、EC2インスタンスとカスタムAWS CLIスクリプトを監視します。
- B. オンプレミスサーバとEC2インスタンスにCloudWatchエージェントをインストールします。
- C. CloudWatchエージェントをEC2インスタンスにインストールし、オンプレミスサーバでcronジョブを使用します。
- D. EC2インスタンスとオンプレミスサーバの両方にCloudWatchの詳細なモニタリングを使用します。

62 チケット販売代理店を営むある企業では、チケット発売時に販売Webサイトのアクセスが集中する問題への対処を検討しています。発売のタイミングでサイトへのアクセス数を30秒間隔で収集したいと考えています。この要件を実現するための正しい構成を選択してください。

- A. Webサイトへのアクセス数を収集する標準メトリクスを高解像度で送信します。
- B. Webサイトへのアクセス数をCloudWatchエージェントにより収集し、標準解像度で送信します。
- C. Webサイトへのアクセス数を収集する標準メトリクスを標準解像度で送信します。
- D. Webサイトへのアクセス数をCloudWatchエージェントにより収集し、高解像度で送信します。

63 開発者はS3に保存されたオブジェクトに対してCloudTrailで操作オペレーションを記録したいと考えています。必要な設定で正しいものを選択してください。

- A. 管理イベントでS3に対する操作オペレーションを有効化します。
- B. 管理イベントの有料オプションを有効化します。S3に対する操作オペレーションが記録されるようになります。
- C. データイベントを有効化します。
- D. インサイトイベントを有効化します。

64 開発チームは運用しているマイクロサービスアプリケーションのサービスの実行状況を可視化しようとしています。マイクロサービスは相互にHTTP通信で呼び出されます。どのAWSサービスが最も低コストで要件を達成できるか、選択してください。

- A. CloudWatch ServiceLens
- B. CloudTrail
- C. AWS Audit Manager
- D. AWS Detective

65 開発チームは特定のサービスの実行状況を可視化するためにAWS X-Rayを使用したいと考えています。サービスのトレースデータはメソッド単位で大量になることから、フィルタ式を利用してインデックスを作成し、任意の処理をピックアップすることを検討しています。このときに必要な設定で正しいものを選択してください。

- A. カスタムサブセグメントにメタデータを追加します。
- B. カスタムサブセグメントに注釈を追加します。
- C. リクエストヘッダにTraceIDを設定します。
- D. トレースデータの出力時に「keyword operator value」形式で条件を指定します。

## 6-2 解答と解説

1 C

⇒ 問題：464 ページ、本文解説：282 ページ

AWS CodeCommitではSDKやCLIを使って、リポジトリの追加やブランチの更新、ファイル追加(put\_file)ができます。その他の選択肢の不正解理由は以下のとおりです。

- ・選択肢A：リポジトリのクローンを作成する必要はなく、新しいファイルをすでにホストされているCodeCommitリポジトリに追加します。
- ・選択肢B：cURLは使用されません。
- ・選択肢D：Step FunctionsはS3イベントを受け入れません。EventBridgeなどを経由する必要があります。

2 D

⇒ 問題：464 ページ、本文解説：287 ページ

CodeBuildは通常、AWSマネージド環境で実行されますが、必要に応じてオプションでVPCに接続するよう構成することができます。その際、実行するサブネットにNAT Gatewayがアタッチされて、インターネットに接続できるよう構成します。その他の選択肢の不正解理由は以下のとおりです。

- ・選択肢A：これはSystems Manager Parameter Storeを利用する際に必要な設定です。
- ・選択肢B：PrivilegedモードはDockerイメージなどを作成する際に有効化するオプションです。
- ・選択肢C：VPCの設定は必須ではなく、これが原因でネットワーク接続エラーにはなりません。

6

**3 D** ➔ 問題：465 ページ、本文解説：292 ページ

CodePipelineは、パイプラインの状態変化をモニタリングするために、CloudWatch Events(EventBridge)イベントソースとして構成できます。キャンセル・失敗・再開・開始・停止・停止中・成功などのパイプラインの状態変化イベントに応じたアクションを設定できます。その他の選択肢の不正解理由は、以下のとおりです。

- ・選択肢A：CodePipelineをイベントソースとして選択できません。
- ・選択肢B：Lambdaのイベントトリガーを作成できません(Invokeステージでカスタム実行アクションとして、Lambdaを指定することはできます)。
- ・選択肢C：CloudWatchアラームはメトリクスを監視対象とします。

**4 B** ➔ 問題：465 ページ、本文解説：297 ページ

In-Placeデプロイは、Blue-Greenデプロイとは異なり、既存環境に更新を行います。ロールバックには再び直前のリビジョンで上書き更新されます。ELBで切り替える向き先はありません。

**5 A** ➔ 問題：466 ページ、本文解説：309 ページ

別のテンプレートのスタック情報を参照するにはクロススタックリファレンスが用いられます。

**6 C** ➔ 問題：466 ページ、本文解説：305 ページ

Mappingsでは、テンプレート内で使用する2次元キーバリューマッピングテーブルを記述できます。EC2インスタンスマシンイメージIDなど、リージョンやユーザの入力パラメータにより、使用したい項目が変わってくる場合などに利用されます。

**7 D** ➔ 問題：467 ページ、本文解説：308 ページ

スタックの作成や更新はaws cloudformation deployで実行することができます。create-stackは、スタックの新規作成時に利用可能で、更新することはできません。他は架空のコマンドです。

**8 C** ➔ 問題：467 ページ、本文解説：315 ページ

SAMテンプレートでは、Transform、Resourcesセクションが必須要素になります。

**9 A** ➔ 問題：467 ページ、本文解説：320 ページ

SAMでアプリケーションをデプロイするためには、sam packageおよびsam deployコマンドが必要です。なお、このコマンドはaws cloudformation packageとaws cloudformation deployコマンドと等価です。

**10 B** ➔ 問題：468 ページ、本文解説：60 ページ

Lambdaでは、デフォルトでCloudWatch Logsが有効化されています。考えられるのは、既存ロールに設定されているポリシーが正しくなく、ログが出力されていない可能性です。

**11 A** ➔ 問題：468 ページ、本文解説：212 ページ

EC2やECSで実行されるアプリケーションにはIAMロールを割り当てるのが最適ですが、オンプレミスサーバでは、IAMユーザとアクセスキー、シークレットキーを作成し、サーバへクレデンシャルとして設定します。

**12 A** ➔ 問題：469 ページ、本文解説：217 ページ

このユースケースでは、商用アカウント→開発アカウントへのアクセスとなりますので、S3へのアクセス権限を付与した開発アカウントのIAMロールの信頼ポリシーに、プロダクションアカウントのIAMロールを設定し、AssumeRole APIを呼び出します。

**13 B** ➔ 問題：470 ページ、本文解説：219 ページ

このユースケースでは、開発アカウントには、プロダクションアカウントのIAMロールがAssumeRole APIを実行するための信頼ポリシーと、S3へアクセスポリシーを付与しておく必要があります。他の選択肢の不正解理由は、以下のとおりです。

- ・選択肢A：アクセスポリシーにはPrincipalを含める必要がありません。
- ・選択肢C、D：信頼ポリシーはプロダクションのIAMロールがAssumeRole APIを実行できるような設定が行われるべきです。

**14 B** ➔ 問題：472 ページ、本文解説：228 ページ

このユースケースでは、ユーザが直接S3にある画像データにアクセスするのが最適です。有効期限が設定された署名付きURLをクライアントユーザに渡すことによってこれを実現します。その他の選択肢は、すべてEC2アプリケーションが画像にアクセスして、クライアントにデータを返却しており、アプリケーションのパフォーマンス的に好ましくありません。

**15 A, B** ➔ 問題：472 ページ、本文解説：240 ページ

KMSは1秒あたりの呼び出し回数が制限されています。そのため、リクエストがクオータを超えた場合には、ThrottlingExceptionが発生します。これを回避するには、アプリケーションコードでエクスponentialバックオフを使用して再実行するか、サポートに連絡してKMSの制限を引き上げるようリクエストするかのいずれかの対応をとる必要があります。

**16 C** ➔ 問題：473 ページ、本文解説：239 ページ

AWSユーザ側で用意したキーを暗号化で利用するのはSSE-Cです。

**17 A** ➔ 問題：473 ページ、本文解説：241 ページ

Cognitoユーザプールは、モバイルアプリケーションの認証に利用できるサービスです。なお、IDプールはAWSリソースへのアクセスを許可する場合に利用されます。

**18 C** ➔ 問題：473 ページ、本文解説：241 ページ

Cognitoは、サーバレスアプリケーションで、さまざまなデバイスを記憶でき、SMSなどをを利用して多要素認証を実現できます。

**19 B** ➔ 問題：474 ページ、本文解説：239 ページ

KMSにはデータの暗号化に使用するキーを自動ローテーションできます。この要件に最も適合するサービスです。

**20 C** ➔ 問題：474 ページ、本文解説：255 ページ

AWS Secret Managerは接続文字列として保存したパラメータの自動ローテーションをサポートします。この要件に最も適合するサービスです。

**21 D** ➔ 問題：474 ページ、本文解説：254 ページ

Amazon InspectorはEC2に設定したエージェントをもとにセキュリティ診断を行い脆弱性を検出するサービスです。この要件に最も適合します。

**22 B** ➔ 問題：475 ページ、本文解説：253 ページ

AWS ShieldはELB、CloudFront、Route 53におけるDDoS対策を提供するサービスです。この要件に最も適合します。

**23 D** ➔ 問題：475 ページ、本文解説：254 ページ

Amazon GuardDutyは、悪意のある操作や不正な動作を継続的にモニタリングする脅威検出サービスです。この要件に最も適合します。

**24 C** ➔ 問題：475 ページ、本文解説：256 ページ

Amazon Macieは、機械学習を用いて、機密データを自動的に検出、分類、保護し、アクセス履歴を記録するセキュリティサービスです。この要件に最も適合します。

**25 B** ➔ 問題：476 ページ、本文解説：256 ページ

Amazon DetectiveはAWS CloudTrail、Amazon VPCフローログ、DNSログなどセキュリティに関するデータを分析、可視化しインシデントの原因特定をサポートするサービスです。

**26 C** ➔ 問題：476 ページ、本文解説：56 ページ

API Gatewayでは、APIキーを使用してスロットリングやアクセス回数を制限できます。その他の選択肢の不正解理由は、以下のとおりです。

- ・選択肢A：アクセストークンには、APIへアクセス回数や上限など高度にカスタマイズされた情報を含むことはできません。また、API Gatewayだけでアクセストークンの内容から要件を満たす判定ロジックを組み込むことはできません。
- ・選択肢B：実現できないことはありませんが、アクセス回数の上限値や現在のアクセス数などを保持するデータベースを構築する必要があり、最も簡単な方法とはいえない。
- ・選択肢D：IDトークンを使った判断ロジックもAPI Gatewayだけで実装することはできません。

**27 B** ➔ 問題：477 ページ、本文解説：55 ページ

API Gatewayでは、特定のステージもしくはAPIの個別のメソッドに対して、AWSアカウントレベルの同時アクセス制限値を上限としてオーバーライド（上書き）することができます。この方法が要件を満たす最も簡易な方法です。

**28 B** ➔ 問題：477 ページ、本文解説：54 ページ

API GatewayのHTTP APIでは、OIDC準拠のソーシャルメディアの認証情報であるIDトークンを使って、JWTオーソライザによるアクセス制御が可能です。その他の選択肢の不正解理由は、以下のとおりです。

- ・選択肢A：REST APIはJWTオーソライザをサポートしません。
- ・選択肢C：Lambdaオーソライザ内で、ソーシャルメディアへのアクセス・認証を実装できないことはありませんが、余計にコストがかかります。
- ・選択肢D：この方法も簡単に構築することはできますが、Cognitoを中継する分、余計にコストがかかります。

**29 A** ➔ 問題：477 ページ、本文解説：57 ページ

API Gatewayのクライアントは「Cache-Control: max-age=0」をリクエストヘッダに設定することにより、個々のリクエストでキャッシュを無効化できます。その他の選択肢の不正解理由は、以下のとおりです。

- ・選択肢B：このようなAPIはありません。
- ・選択肢C：キャッシュ自体を無効化するのは適切でない対処方法です。クライアントが古いキャッシュデータを参照しているため、一時的にクライアントからのキャッシュを無効化するのが正しい方法です。
- ・選択肢Dで、これはクライアントのキャッシュを無効化する方法ではありません。

**30 B** ➔ 問題：478 ページ、本文解説：68 ページ

Amazon SNSのようにLambdaを非同期実行するAWSサービスがイベントトリガーとなる場合は、エラーのイベント記録にLambdaのデッドレターキューを使用できます。

**31 B** ➔ 問題：479 ページ、本文解説：64 ページ

AWS Lambdaでは、関数のエイリアスを1つ以上作成できます。Lambdaのエイリアスは、特定のLambda関数バージョンへのポインターのようなものです。ユーザは、エイリアスARNを使用して機能バージョンにアクセスできます。関数コードの複数のバージョンにエイリアスを指定してから、重みを割り当てて、特定の量のトラフィックを各バージョンに転送できます。これにより、Blue/Greenデプロイメントスタイルが可能になり、問題が発生した場合に重み付けを更新するだけで、古いバージョンに簡単にロールバックできます。その他の選択肢の不正解理由は、以下のとおりです。

- ・選択肢A、D：この方法では関数のコード内で、新しいバージョンに転送するような分岐条件を加えなければなりません。
- ・選択肢C：ステージ変数は環境変数のように利用できる変数です。ステージ変数を使用して、ステージ固有の設定メタデータをクエリパラメータや入力マッピングテンプレートで生成されるペイロードとしてバックエンドのLambda関数に渡すこともできますが、少量のトラフィックを別のバージョンへ転送する場合には使用されません。

**32 D** ➔ 問題：479 ページ、本文解説：63 ページ

Lambdaレイヤーは、ライブラリ、カスタムランタイム、またはその他の依存関係を含むZIPアーカイブです。レイヤーを使用すると、デプロイメントパッケージにライブラリを含める必要なしに、関数でライブラリを使用できます。

**33 A** ➔ 問題：479 ページ、本文解説：60 ページ

Lambdaでは、設定したメモリスペックに応じたvCPUが割り当てられます。その他の選択肢の不正解理由は以下のとおりです。

- ・選択肢B：Lambdaにインスタンスタイプを設定することはできません。
- ・選択肢C：設問ではタイムアウトになっている状況ではありません。
- ・選択肢D：並列実行してもより多くのリクエストを処理できるだけです。

**34 B** → 問題：480 ページ、本文解説：255 ページ

Lambda関数内の処理で、環境に依存したり、セキュアなパラメータは Systems Manager Parameter Storeと連携して、パラメータをアプリケーションから外出しして利用できます。その他の選択肢の不正解理由は以下のとおりです。

- ・選択肢A：KMSは環境変数の暗号化には使用しません。
- ・選択肢C：CloudHSMも環境変数の暗号化を想定したサービスではありません。
- ・選択肢D：この方法はセキュリティの観点からも望ましい方法ではありません。

**35 A** → 問題：480 ページ、本文解説：80 ページ

.GetItemは、デフォルトで結果整合性のある読み取りを提供します。アプリケーションで強い整合性のある読み取りが必要な場合は、ConsistentReadパラメータをtrueに設定します。

**36 D** → 問題：480 ページ、本文解説：79 ページ

設問の要件を満たすためには、以下の権限が必要になります。

- ・GetItem  
テーブルから単一の項目を取り出します。目的の項目のプライマリキーを指定する必要があります。項目全体またはその属性のサブセットのみを取り出すことができます。
- ・UpdateItem  
項目の1つ以上の属性を変更します。変更する項目のプライマリキーを指定する必要があります。新しい属性を追加したり、既存の属性を変更または削除したりできます。ユーザ定義の条件を満たす場合にのみ更新が成功するように、条件付きの更新を実行できます。
- ・PutItem  
項目を作成します。プライマリキーは一部の属性ではなくすべての属性を指定する必要があります。

**37 C** → 問題：481 ページ、本文解説：77 ページ

設問では、4.5KBの結果整合性ある読み込みが1秒あたり10回、標準書き込みが6回です。結果整合性の読み込みでは4KBまでのデータは1RCUsで2回、標準書き込みでは1KBまでのデータが1WCUsで1回ですので、要件を満たすには10RCUs、30WCUsが必要になります。

**38 A** → 問題：481 ページ、本文解説：80 ページ

この設問では、Scan APIが実行されることにより、在庫テーブルで大量のデータ読み込みが発生し、スループットが消費されて、ECサイト側に影響が出ていると考えられます。そのため、1回のスキャンオペレーションの読み込みを制限し、単位時間あたりの読み込みキャパシティを下げて、ECサイト側へ影響が出ないように対処するのが効果的だと考えられます。Scanオペレーションでは、page-sizeパラメータによりサイズを指定して実行することが可能です。その他の選択肢の不正解理由は以下のとおりです。

- ・選択肢B：パラレルでスキャンAPIを実行してしまうと、より負荷がかかりECサイト側への影響が大きくなります。
- ・選択肢C：ソートキーはノード内のデータ順序などを規定するために用いられるもので、スキャンによるパフォーマンス低下の改善には寄与しません。
- ・選択肢D：グローバルセカンダリインデックスはテーブル内の特定の項目をキーに高速に検索するためのもので、スキャンによるパフォーマンス低下の改善には寄与しません。

**39 A** → 問題：481 ページ、本文解説：77 ページ

キャパシティユニットはオンデマンドキャパシティで動的に変更するか、プロビジョンドモードで設定したキャパシティユニットの範囲内でスケーリングすることができます。ワークロードが急激に急増して変動幅が予測できない場合は、オンデマンドモードが適しており、予測した範囲内でトラフィック量が推移する場合は、Auto Scalingでプロビジョニングモードを設定するほうが適しています。

**40 C** ➔ 問題：482 ページ、本文解説：98 ページ

最もハイパフォーマンスなレスポンスで応答する場合は、レイテンシベースドルーティングポリシーを設定します。位置情報ルーティングポリシーや地理的近接性ルーティングも選択できますが、位置情報にもとづいたルーティングなので、位置が近いからといって必ずしもすべてのケースで最適なルーティングがされているとは限りません。

**41 B、D** ➔ 問題：482 ページ、本文解説：105 ページ

通信の暗号化を有効にするためにSSL/TLS証明書を追加する必要があります。データの暗号と復号のプロセスはCPUに負荷がかかるため、EC2インスタンスに証明書を追加しないようにする必要があります。したがって、解決策は、Elastic Load BalancerでSSL証明書を構成してから、SSL終了を構成することです。これは、ロードバランサーのHTTPSリスナーに証明書を追加することで実行できます。

**42 A** ➔ 問題：482 ページ、本文解説：107 ページ

ステートフルなアプリケーションに対し、ELBで同一のクライアントからのリクエストを同一のサーバに振り分ける場合はスティッキーセッションを利用します。他の選択肢の不正解理由は、以下のとおりです。

- ・選択肢B：Route 53にはセッションアフィニティポリシーはありません。
- ・選択肢C：ステートフルに動作するよう、同一のサーバにリクエストを振り分けるのにターゲットグループのアルゴリズムは関係ありません。
- ・選択肢D：Elastic IPアドレスを割り当てるのはNLBに対するものなので、同一のサーバにリクエストを振り分けることは関係ありません。

**43 B** ➔ 問題：483 ページ、本文解説：39、364 ページ

カスタムメトリクスを送信する場合は、CloudWatchAgentServerPolicyがアタッチされたIAMロールを作成し、オートスケーリンググループ起動構成を作成します。EC2に権限を割り当てるのにIAMユーザを利用したり、オートスケーリンググループを構成するのに直接EC2インスタンスを起動することはできません。

**44 C** ➔ 問題：483 ページ、本文解説：121 ページ

ECSのタスク間で共有のボリュームをマウントする場合は、タスク定義を1つ作成し、2つのコンテナ間で共有ボリュームをマウントするよう「コンテナの定義」にて設定します。選択肢A、DはEKSに対する説明です。

**45 D** ➔ 問題：484 ページ、本文解説：120 ページ

ECSでもEC2と同様、IAMロールを作成し、コンテナのアプリケーションが使用する資格情報を提供できます。最小特権の原則に従って、各ECSタスク定義に個別のロールとして割り当てるのが最善です。

**46 B** ➔ 問題：484 ページ、本文解説：395 ページ

ElastiCache(Redis)では、リードレプリカを作成することで、マルチAZでの自動フェイルオーバーを構成することができます。クラスタモードはシャーディングするかどうかのオプションなので、この設問の要件には必ずしも必須ではありません。最小コストでの可用性確保を考えるのであれば、必要ありません。

**47 A** ➔ 問題：485 ページ、本文解説：132 ページ

Auroraには、プライマリインスタンスを指し示す「クラスタエンドポイント」とリードレプリカを指し示す「読み取りエンドポイント」があります。書き込み処理があるサービスは前者を、参照のみのサービスは後者を参照するよう構成できます。

**48 C** ➔ 問題：486 ページ、本文解説：68、408 ページ

S3に対してPutObjectが実行されるイベントを契機に、リサイズ処理を行うLambda関数を起動します。リサイズ処理がエラーになっても、複数回リトライ処理を行ったり、エラーが検知できたりするよう、SQSを経由させます。SQSからのLambda関数の実行は同期呼び出しになってしまふため、送信先の設定は利用できず、正常終了とエラー終了それぞれLambda関数内で管理者へ通知するよう実装する必要があります。なお、エラーの通知は、Lambda関数の実行エラーが規定回数を超えたのち、SQSのデッドレターキューにより設定されます。他の選択肢の不正解理由は、以下のとおりです。

- ・選択肢A、D：Lambda関数が監視する処理を実装する必要はなく、S3のイベントトリガーで起動を構成できます。なお、その場合、非同期処理で実行されることになります。
- ・選択肢B：Lambda関数は同期処理で呼び出されるため、送信先の設定は利用できません。

**49 C** ➔ 問題：487 ページ、本文解説：143 ページ

S3へのパブリックアクセスを行う場合には、バケットポリシーでAction要素に「s3:GetObject」の実行権限と、Principalに「\*」、Resourceに、パブリックアクセスを許可するオブジェクトを指定することが推奨されます。他の選択肢の不正解理由は、以下のとおりです。

- ・選択肢A：事前定義されたACLを利用するよりもバケットポリシーによるパブリックアクセスが推奨されます。
- ・選択肢B：Action要素の実行権限に読み取り以外を付与しているのは好ましくありません。
- ・選択肢D：FULL\_CONTROLを付与するのは好ましくありません。

**50 C** ➔ 問題：487 ページ、本文解説：31 ページ

通常S3は、インターネットを経由したアクセスがデフォルトになります。この問題では、ネットワークはインターネット接続が許されていないため、ゲートウェイ型のVPCエンドポイントをプライベートサブネットにアタッチする必要があります。インターフェース型はサポートされません。

**51 B** ➔ 問題：488 ページ、本文解説：146 ページ

Webサイトの有料会員に対し、S3に管理されている動画コンテンツの権限を割り当てる場合は、有効期限を設定した署名付きURLを利用する方法が最適です。ACLやIAMポリシーは認証済みのIAMユーザーに対するアクセス許可であるため、アプリケーションのユーザに対するアクセス制御には適していません。

**52 B** ➔ 問題：488 ページ、本文解説：404 ページ

設問の要件では、20[s]の間隔で実行されるロングポーリングが適しているため、「ReceiveMessageWaitTimeSeconds」を設定します。

**53 C** ➔ 問題：489 ページ、本文解説：409 ページ

プロデューサーアプリケーションからの2重送信などの重複処理防止に対処するには、メッセージ重複排除IDをプロデューサー側で指定して、意図的に一意のメッセージとして識別して送信するのが効果的です。メッセージ重複排除IDはFIFOキューを使用する必要があります。

**54 C** ➔ 問題：489 ページ、本文解説：68、408 ページ

SQSキューのメッセージをLambda関数が実行する場合、同期呼び出しになります。そのため、送信先やLambdaのデッドレターキュー設定は利用できません。SQSのデッドレターキューを用いるには、「Redriveポリシー」に最大受信回数やキューのARNなど配信ルールを定義します。

**55 A** ➔ 問題：490 ページ、本文解説：158 ページ

EC2インスタンスはKCLワーカーを複数実行することができますが、1つのシャードに対し、1つのKCLワーカーが対応するように構成します(1つのワーカーが複数のシャードのデータレコードを処理することはありません)。つまり、シャードの数を超えるEC2インスタンスを用意しても処理は分散できません。スループットを最大化するのであれば、シャードと同数となるようにEC2インスタンスおよびKCLワーカーを1:1で構成します。

**56****C**

➡ 問題：490 ページ、本文解説：424 ページ

CloudFrontでは、有効期限が切れる前にキャッシュされているコンテンツを無効化することもできます(Invalidation)。コンソール上やCLIからファイルを指定して、キャッシュを無効化します。ワイルドカードを使用して複数のファイルを同時に無効化することもできます。無効化後は再びオリジンから最新バージョンを取得します。他の選択肢の不正解理由は、以下のとおりです。

- ・選択肢A：すべてのキャッシュを一度削除し、新しくキャッシュを作り直すことに相当するので、過剰にコストがかかります。
- ・選択肢B：この方法はキャッシュを無効化する方法ではありません。
- ・選択肢D：一度ディストリビューションを無効化して、再度有効化しても特定のキャッシュを無効化することにはなりません。TTLの設定に従うだけです。

**57****C**

➡ 問題：491 ページ、本文解説：151 ページ

S3 Transfer Accelerationは、S3バケットに対するアップロード高速化に寄与します。その他の選択肢の不正解理由は以下のとおりです。

- ・選択肢A、D：API GatewayをS3との間に設定しても高速化に寄与しません。
- ・選択肢B：S3へのPUTリクエストにCloudFrontを介して高速化が見込めますが、一部のリージョンでは、署名バージョン4を利用したリクエスト認証やPOSTリクエストはサポートされません<sup>※1</sup>。

**58****A**

➡ 問題：492 ページ、本文解説：424 ページ

CloudFrontでは、有効期限が切れる前にキャッシュされているコンテンツを無効化することもできます(Invalidation)。コンソール上やCLIからファイルを指定して、キャッシュを無効化します。ワイルドカードを使用して複数のファイルを同時に無効化することもできます。無効化後は再びオリジンから最新バージョンを取得します。他の選択肢の不正解理由は、以下のとおりです。

- ・選択肢B：オリジンではなく、CloudFront上の古いキャッシュを無効にしなければいけません。
- ・選択肢C：一度ディストリビューションを無効化して、再度有効化しても特定のキャッシュを無効化することにはなりません。TTLの設定に従うだけです。
- ・選択肢D：Lambda@Edgeでファイルを更新することに意味はありません。CloudFront上の古いキャッシュを無効にしなければいけません。

**59****D**

➡ 問題：492 ページ、本文解説：431 ページ

ターゲット追跡スケーリングポリシーを用いて、定められたメトリクスを維持するようEC2インスタンス数を調整することができます。

**60****C**

➡ 問題：492 ページ、本文解説：82 ページ

DAXは、マルチアベイラビリティゾーン構成で自動フェイルオーバー機能を持つインメモリキャッシュです。DynamoDBを使用する場合、こちらのサービスがより設問の要件に合致します。

**61****B**

➡ 問題：493 ページ、本文解説：364 ページ

監視の中で重要な指標であるメトリクスがCloudWatch標準には含まれていない場合があります。そのようなケースでは、CloudWatchエージェントを使用するか、EC2上でターゲットとなるデータ収集のためのスクリプトを実行し、AWS CLIでカスタムメトリクスとして送信します。なお、CloudWatchエージェントはオンプレミスでのサーバでもインストールして利用できます。

※1 [https://docs.aws.amazon.com/ja\\_jp/AmazonCloudFront/latest/DeveloperGuide/private-content-restricting-access-to-s3.html#private-content-origin-access-identity-signature-version-4](https://docs.aws.amazon.com/ja_jp/AmazonCloudFront/latest/DeveloperGuide/private-content-restricting-access-to-s3.html#private-content-origin-access-identity-signature-version-4)

**62 D****→ 問題：493 ページ、本文解説：363 ページ**

サイトへのアクセス数はアプリケーションから収集する高度なメトリクスになります。これらはCloudWatchエージェントでStatsDやcollectdプロトコルを使用して収集できます<sup>※2</sup>。標準解像度ではメトリクスの送信は最短1分間隔なので、高解像度メトリクスを使用します。

**63 C****→ 問題：494 ページ、本文解説：378 ページ**

S3に保存されたオブジェクトに対する操作や、Lambda関数の実行、DynamoDBテーブルのデータ更新などはデータイベントに記録されます。デフォルトでは管理イベントのみが記録対象となるので、データイベントを記録するには、有効化しなければなりません。

**64 A****→ 問題：494 ページ、本文解説：392 ページ**

CloudWatch ServiceLensではX-Rayのデータを使用して、サービスマップやトレースリストなど、アプリケーション可視化を1箇所でまとめて確認することができます。

**65 B****→ 問題：494 ページ、本文解説：390 ページ**

注釈はフィルタ式で使用するためのインデックス化されたキーバリューペアです。例えばアプリケーションで、注釈に特定の値を設定し、フィルタ式のキーワードで、「annotations.key」を使用すると、該当のトレースデータだけをフィルタできます。

---

※2 [https://docs.aws.amazon.com/ja\\_jp/AmazonCloudWatch/latest/monitoring/CloudWatch-Agent-custom-metrics-statsd.html](https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch-Agent-custom-metrics-statsd.html)

## 索引

### A

Amazon OpsWorks	328
Amazon RDS	126
Amazon Route 53	94
Amazon S3	134
Amazon SageMaker	436
Amazon SNS	411
Amazon SQS	400
AMI	36
API	44
APIのキャッシュ	57
Application Auto Scaling	434
Application Recovery Controller Zonal Shift	100
appspec.yml	302
AppStream 2.0 フリート	434
AssumeRole	215
AssumeRole API	228
Aurora DB クラスタ	434
Aurora Serverless	133
Auto Scaling	429
AWS	20
AWS Amplify	165
AWS AppConfig	328
AWS App Runner	166
AWS AppSync	165
AWS CLI	23
AWS CloudFormation	306
AWS CloudTrail	377
AWS CodeArtifact	329
AWS CodeBuild	288
AWS CodeCommit	280
AWS CodeDeploy	299
AWS CodePipeline	293
AWS Copilot	329
ACL (Access Control List)	141
ALB	107
Amazon API Gateway	44
Amazon Athena	165, 380
Amazon Aurora	130
Amazon Certificate Manager	252
Amazon CloudFront	416
Amazon CloudWatch	360
Amazon Cognito	241
Amazon Comprehend	435
Amazon Detective	256
Amazon DynamoDB	71
Amazon EBS	40
Amazon EC2	36
Amazon ECR	286
Amazon ECS	115
Amazon ECS サービス	435
Amazon ECS マネジメントコンソール	119
Amazon EFS	41
Amazon ElastiCache	393
Amazon EMR (Elastic Map Reduce) クラスタ	435
Amazon GuardDuty	254
Amazon Inspector	254
Amazon Keyspaces テーブル	436
Amazon Kinesis	153
Amazon Machine Image	36
Amazon Macie	256
Amazon Memory DB for Redis	399
Amazon MSK (Managed Service for Kafka)	
クラスタストレージ	436
Amazon OpenSearch Service	166, 381

AWS Elastic Beanstalk	323
AWS IAM	206
AWS KMS (Key Management Service)	234
AWS Lambda	59
AWS SDK	23
AWS Secrets Manager	255
AWS Security Token Service (AWS STS)	212, 227
AWS Serverless Application Model	317
AWS Shield	253
AWS Step Functions	85
AWS Systems Manager	255
AWS X-Ray	383
AWSマネジメントコンソール	23

### D

DAX	82
DNS	94
DNS フォワーダ	101
Docker	115, 286
DynamoDB Accelerator	82
DynamoDB Streams	83
DynamoDB Trigger	83
DynamoDB テーブル	435

### B

BitBucket	289
buildspec.yml	290

### E

EC2 Auto Scaling	429
EC2 起動型	117
ECS サービス実行	122
ECS タスク	120
ElastiCache for Memcached	393
ElastiCache for Redis	395
ELB	103
Embedded Metrics Format (EMF)	368
Extensions	63
EXTERNAL	117

### C

CDK	235
CDN	22
CI	288
CLB	114
CloudFormation デザイナー	315
CloudFront Functions	428
CloudWatch Alarms	365
CloudWatch Dashboards	373
CloudWatch Events	373
CloudWatch Logs	367
CloudWatch Logs Insights	369, 382
CloudWatch Metrics	362
CMK	235
CodeBuild Local	292
Cognito オーソライザ	54

### F

FARGATE	117
FIFO キュー	401

### G

Gateway	44
Git	280

GitHub ..... 289  
 GLB ..... 113  
 Globals要素 ..... 317

**H**

Hosted Zone ..... 96  
 HTTP API ..... 50

**I**

IAM Roles Anywhere ..... 213  
 IAMアクセス権限 ..... 53  
 IAMユーザ ..... 208  
 IDプール ..... 246

**J**

Jenkins ..... 288  
 JWT ..... 247

**K**

Kibana ..... 166  
 Kinesis Client Library (KCL) ..... 158  
 Kinesis Data Analytics ..... 161  
 Kinesis Data Firehose ..... 160  
 Kinesis Data Streams ..... 154  
 Kinesis Video Streams ..... 163

**L**

Lambda@Edge ..... 427  
 Lambdaオーソライザ ..... 53  
 Launch Configuration ..... 429  
 Launch Template ..... 429  
 Layers ..... 63

**M**

Memcached ..... 393, 398  
 M out of N ..... 366

**N**

NATゲートウェイ ..... 29  
 NLB ..... 111  
 NoSQLデータベース ..... 71

**O**

OIDC (OpenID Connect) ..... 245  
 OIDCプロバイダ認証 ..... 110

**P**

PassRole ..... 219  
 Private Certificate Authority ..... 252  
 Provisioned Concurrency ..... 62  
 Publisher ..... 412

**R**

RCUs ..... 77  
 RDB ..... 76  
 Redis ..... 395, 398  
 REST API ..... 45  
 Route 53 Application Recovery Controller ..... 99  
 Route 53 Resolver ..... 100, 102

**S**

S3 Object Ownership ..... 146  
 S3 Storage Lens ..... 151  
 S3 Transfer Acceleration ..... 151  
 S3アナリティクス ..... 152

S3インベントリ ..... 152  
 SAM ..... 317  
 SAM CLI ..... 321  
 Scaling Plan ..... 429  
 SSE-KMS ..... 148  
 SSE-S3 ..... 147  
 SSL/TLS ..... 252  
 SSL/TLSターミネーション ..... 105  
 SSL/ TLS型通信暗号化 ..... 107  
 SSL/TLS通信 ..... 113  
 Step Functions Distribution Map ..... 93  
 Subscriber ..... 412

**T**

TimeToLive ..... 81  
 Topic ..... 412  
 TraceID ..... 386  
 TTL ..... 81

**V**

VPC ..... 27  
 VPCアクセス ..... 61  
 VPCエンドポイント ..... 30, 145  
 VPCピアリング接続 ..... 30

**W**

WAF ..... 252  
 WAF連携 ..... 57  
 WCUs ..... 77  
 WebRTC ..... 164  
 WebSocket API ..... 49

**X**

X-Rayデーモンの設定 ..... 387

**あ行**

アカウント ..... 207  
 アクション ..... 209  
 アクセスコントロール ..... 138  
 アクセス制御 ..... 33  
 アクセスポイント ..... 145  
 アナリティクス ..... 384  
 アベイラビリティゾーン ..... 21  
 暗号化 ..... 147, 234  
 イベント ..... 65  
 イベントソース ..... 374  
 イベントターゲット ..... 376  
 イベント通知機能 ..... 149  
 インサイト ..... 385  
 インサイトイベント ..... 378  
 インスタンス ..... 36  
 インターネットゲートウェイ ..... 29  
 ウオームスタート ..... 62  
 エリア ..... 64  
 エッジサーバ ..... 416  
 エッジ最適化APIエンドポイント ..... 46  
 エッジロケーション ..... 22, 417  
 エラーハンドリング ..... 68  
 エンベロープ暗号化 ..... 235  
 オーケストレーション ..... 115  
 オブジェクト ..... 135  
 オブジェクトストレージ ..... 134  
 オブジェクトロック ..... 151  
 オペレーション ..... 209  
 親キー ..... 74  
 オリジン ..... 420  
 オリジンフェイルオーバー ..... 427  
 オンデマンドキャパシティモード ..... 77

**か行**

- 外部IDフェデレーションユーザ ..... 222  
 外部ネットワーク型診断 ..... 255  
 外部フェデレーション ..... 246  
 可視性タイムアウト ..... 408  
 カスタマーゲートウェイ ..... 30  
 カスタマーデータキー ..... 235  
 カスタマーマスターキー ..... 235  
 カスタムランタイム ..... 62  
 カスタムリソース ..... 313  
 仮想プライベートゲートウェイ ..... 30  
 カナリアリリース ..... 57  
 環境データ ..... 209  
 関係演算子 ..... 74  
 監視連携 ..... 57  
 関数 ..... 59  
 管理イベント ..... 377  
 キー ..... 135  
 擬似パラメータ ..... 307  
 キャッシュコントロール ..... 420  
 キャパシティユニット ..... 77  
 キュー ..... 401  
 クエリ言語 ..... 371  
 組み込み関数 ..... 307  
 クラウドコンピューティングサービス ..... 20  
 クラスタ ..... 117  
 クラスタモード ..... 395, 397  
 グローバルセカンダリインデックス ..... 75, 435  
 グローバルテーブル ..... 84  
 クロスアカウントロール ..... 219  
 クロススタックリファレンス ..... 311  
 繼続的インテグレーション ..... 288  
 繼続的デリバリー ..... 293  
 結果整合性オプション ..... 73

- コールドスタート ..... 62  
 子キー ..... 74  
 コンシューマ ..... 155, 401  
 コンテナ ..... 115  
 コンテナレジストリ ..... 286  
 コンテンツ配信ネットワーク ..... 22  
 コントロールプレーン ..... 117

**さ行**

- クライアントサイド暗号化 ..... 240  
 サーバサイド暗号化 ..... 239  
 サーバログアクセス機能 ..... 152  
 サービスマップ ..... 384  
 サブスクリプションフィルタ ..... 368  
 サブセグメント ..... 386  
 サブネット ..... 27  
 シャード ..... 157  
 手動スケーリング ..... 431  
 条件付き書き込み ..... 78  
 ショートポーリング ..... 404  
 署名付きURL ..... 146  
 署名付きURL/Cookie ..... 425  
 スケーリング ..... 429  
 スケジューリング ..... 431  
 スケジュール起動 ..... 70  
 スタック ..... 306  
 スタックセット ..... 315  
 スタンダードキュー ..... 401  
 スティッキーセッション ..... 107  
 ステートマシン ..... 85  
 ストリームデータ ..... 153  
 ストレージクラス ..... 42, 136  
 スロットリング ..... 55  
 静的Webサイトホスティング ..... 148

- 静的コンテンツ ..... 418  
 セキュリティグループ ..... 34  
 セグメント ..... 386  
 セッションポリシー ..... 215  
 相互TLS認証 ..... 55  
 ソートキー ..... 74

**た行**

- ダイナミックリファレンス ..... 312  
 タスク配置制約 ..... 125  
 タスク配置戦略 ..... 123  
 地域制限 ..... 427  
 遅延キュー ..... 407  
 チェンジセット ..... 313  
 注釈 ..... 390  
 ディストリビューション ..... 419  
 ディメンション ..... 363  
 ディレクトリサービス ..... 241  
 データイベント ..... 378  
 データキー（対称キー） ..... 235  
 データキーペア（非対称キー） ..... 235  
 データストリーム ..... 156  
 データブレーン ..... 117  
 データレコード ..... 156  
 デッドレターキュー ..... 408  
 デプロイ ..... 299  
 デプロイ設定 ..... 301  
 デプロイパッケージ ..... 63  
 テンプレート ..... 306, 317  
 テンプレート検証・テストツール ..... 316  
 同期呼び出し ..... 67, 69  
 等値系演算子 ..... 74  
 動的スケーリング ..... 431  
 トラフィックフロー ..... 99  
 トラフィックルーティング ..... 98  
 トランジット ..... 21

**な行**

- 名前空間 ..... 363  
 認証情報 ..... 209, 227  
 ネステッドスタック ..... 310  
 ネットワークACL ..... 35

**は行**

- バージョニング ..... 64  
 バージョン ..... 135  
 パーティションキー ..... 74  
 パーミッションバウンダリー ..... 215  
 パイプライン ..... 293  
 パケット ..... 135  
 パケットポリシー ..... 139  
 ハッシュキー ..... 74  
 非同期呼び出し ..... 67, 69  
 ビヘイビア ..... 420  
 ファイル圧縮 ..... 426  
 フィルタ式 ..... 390  
 フィルタを使った読み込み ..... 78  
 フェデレーテッドユーザ ..... 208  
 プライベートAPIエンドポイント ..... 46  
 プライマリキー ..... 74  
 プリンシパル ..... 208  
 プレイスマントグループ ..... 38  
 ブロックパブリックアクセス ..... 144  
 プロデューサー ..... 155, 401  
 プロビジョンドモード ..... 77  
 ヘルスチェック ..... 104  
 ホスト型診断 ..... 254  
 ホストゾーン ..... 96

- ポリシー ..... 209  
ポリシーテンプレート ..... 319

**ま行**

- マクロ ..... 313  
マネジメントコンソール ..... 379  
マルチリージョン構成 ..... 21  
メタデータ ..... 135, 390  
メッセージタイマー ..... 407  
メッセージ重複排除 ..... 409  
メトリクス ..... 363, 383  
メトリクスフィルタ ..... 368

**や行**

- ユーザ ..... 208  
ユーザープール ..... 241  
ユーザポリシー ..... 138  
予測スケーリング ..... 431

**ら行**

- ランタイム ..... 59  
リージョン ..... 20  
リージョンAPIエンドポイント ..... 46  
リージョンエッジキャッシュ ..... 22  
リードレプリカ ..... 129  
リクエスタ支払い ..... 151  
リクエスト ..... 208  
リクエスト認証 ..... 212  
リクエストモニタリング・ログ ..... 104  
リソース ..... 209  
リソースデータ ..... 209  
リトライ処理 ..... 68, 414  
ルートテーブル ..... 33  
ルートユーザ ..... 208  
ルール ..... 374

**■著者**

川畠 光平（かわばた・こうへい）

- NTTデータ エグゼクティブ ITスペシャリスト ソフトウェアアーキテクト
- 金融機関システム業務アプリケーション開発・システム基盤担当、ソフトウェア開発自動化関連の研究開発を経て、デジタル技術関連の研究開発・推進に従事。

Red Hat Certified Engineer、Pivotal Certified Spring Professional、AWS Certified Solutions Architect Professionalなどの資格を持ち、アプリケーション基盤・クラウドなどさまざまな開発プロジェクト支援にも携わる。

AWS Top Engineers & Ambassadors選出。

**STAFF**

- |        |                  |
|--------|------------------|
| 編集     | 澤田 竹洋（浦辺制作所）     |
|        | 畠中 二四            |
| 校正協力   | 株式会社トップスタジオ      |
| DTP制作  | 関口 忠             |
| 表紙デザイン | 馬見塚意匠室           |
|        | 阿部 修（G-Co. Inc.） |

編集長 玉巻秀雄