# AWS でのビッグデータ分析オプション

*2016 年 1 月*

# 注意

本文書は、情報提供の目的のみのために提供されるものです。本書の発行時点における AWS の現行製品と慣行を表したものであり、それらは予告なく変更されることがあります。お客様は本文書の情報および AWS 製品の使用について独自に評価する責任を負うものとします。これらの情報は、明示または黙示を問わずいかなる保証も伴うことなく、「現状のまま」提供されるものです。本書のいかなる内容も、AWS、その関係者、サプライヤー、またはライセンサーからの保証、表明、契約的責任、条件や確約を意味するものではありません。AWS がそのお客様に対して負う責任と義務は AWS の契約によって管理され、本書は、AWS とそのお客様の間のいかなる契約にも属さず、そのような契約を変更するものでもありません。

# 目次


はじめに 1

AWS でビッグデータ分析を行う利点 2

- Amazon Kinesis Streams 4
- AWS Lambda 8
- Amazon EMR 13
- Amazon Machine Learning 21
- Amazon DynamoDB 25
- Amazon Redshift 31
- Amazon Elasticsearch Service 36
- Amazon QuickSight 42
- Amazon EC2 42

AWS でのビッグデータ問題の解決 46

- 例 1: エンタープライズデータウェアハウス 48
- 例 2: センサーデータのキャプチャと分析 52
- 例 3: ソーシャルメディアの感情分析 56

まとめ 59

寄稿者 60

その他の資料 60

ドキュメントの改訂 62

# 要約

本書では、アーキテクト、データサイエンティスト、および開発者のために AWS クラウドで利用できるビッグデータ分析オプションとしての各種サービスを紹介し、サービスごとに以下の各項目について説明します。

1. 最適な使用パターン
2. 料金モデル
3. パフォーマンス
4. 耐久性と可用性
5. スケーラビリティと伸縮性
6. インターフェイス
7. アンチパターン

本書の最後に分析オプションの導入事例と AWS でビッグデータ分析を開始する際に役立つその他の参考資料を記載しています。

# はじめに

社会のデジタル化が進むほど、データの作成や収集が活発化し、データ量は加速度的に増えていきます。従来の分析ツールでは、この増え続けるデータに対応しきれなくなります。データが生成される量とデータを効果的に分析できる量のギャップを埋めるには、革新が必要です。

ビッグデータ用のツールやテクノロジーを使用すると、データの効率的な分析が可能になり、顧客ニーズの理解を深め、市場での競争優位を確立し、ビジネスを成長させるためのチャンスが生まれます。また、課題も浮き彫りになります。データ管理アーキテクチャは、従来のデータウェアハウスモデルから、より複雑なアーキテクチャへと進化し、リアルタイムバッチ処理、構造化データ/非構造化データ、高速トランザクションなどの新しく追加された要件に対処するようになっています。

アマゾン ウェブ サービス (AWS) が提供するマネージド型サービスの広範なプラットフォームでは、エンドツーエンドのビッグデータアプリケーションの構築、セキュリティ保護、シームレスなスケールがすばやく簡単に実現できます。アプリケーションの要件がリアルタイムのストリーミングであるか、バッチデータ処理であるかを問わず、AWS が提供するインフラストラクチャとツールを使えば、今後のビッグデータプロジェクトに対応できるようになります。ハードウェアの調達は不要であり、インフラストラクチャの管理やスケールも不要です。ビッグデータの収集、保存、処理、分析のみに集中できます。AWS には、増え続けるデータに対応し、ビジネスへの洞察をもたらすために特化された分析ソリューションのエコシステムがあります。

# AWS でビッグデータ分析を行う利点

大規模なデータセットの分析には、入力データの量と分析の種類に応じてスケールできる強力なコンピューティング性能が要求されます。このようなビッグデータワークロードの特徴に最適なのは、需要に応じてアプリケーションを容易にスケールアップ/ダウンできる従量課金制のクラウドコンピューティングドモデルです。AWS では、要件の変動に応じて、環境を簡単に (縦横に) スケールしてニーズを満たすことができます。新しいハードウェアを追加する必要はなく、必要な容量をプロビジョニングするために過剰投資を強いられることもありません。

従来のインフラストラクチャのミッションクリティカルなアプリケーションでは、ビジネスニーズが増大してデータが急増した場合に備えてシステムは常に対応可能であることが要求されるため、システムデザイナーはオーバープロビジョニングせざるを得ません。それに対して、AWS では、わずか数分でプロビジョニングする容量を増やしてコンピューティングできます。つまり、ビッグデータアプリケーションは需要に応じて伸縮するため、システムはほぼ最適な効率で動作します。

さらに、AWS のグローバルインフラストラクチャの柔軟なコンピューティングでは、さまざまな地理的リージョンを利用できるだけでなく、他のスケーラブルなサービスと連携して高度なビッグデータアプリケーションを構築できます。[1] 連携できる他のサービスとしては、データを保存する Amazon Simple Storage Service (Amazon S3)、移動するジョブの調整とデータの変換を容易にする AWS Data Pipeline[2]、接続されたデバイスがクラウドアプリケーションやその他の接続されたデバイスとの間でやり取りできる AWS IoT などがあります。[3]

さらに、AWS にはクラウドにデータを移行するためのオプションとして、ペタバイト級のデータを高速転送する AWS Import/Export Snowball、ストリーミングデータをロードする Amazon Kinesis Firehose、スケーラブルでプライベートな接続を確立する AWS Direct Connect など、セキュリティで保護された多くのツールがあります。[4] モバイルでの利用の急増に対応するには、AWS Mobile Hub 内のサービスのスイートを使用して、アプリの使用状況やデータを収集・計測したり、データを別のサービスにエクスポートして詳しいカスタム分析を実施したりできます。[5]

このような AWS プラットフォームの機能は、ビッグデータ問題の解決に最適であり、多くのお客様が AWS でのビッグデータ分析ワークロードの実装に成功しています。導入事例の詳細については、「ビッグデータ活用における AWS 国内導入事例 Powered by the AWS クラウド」を参照してください。[6]

以下のサービスの順に、ビッグデータの収集、処理、保存、および分析について説明します。

- Amazon Kinesis Streams
- AWS Lambda
- Amazon Elastic MapReduce
- Amazon Machine Learning
- Amazon DynamoDB
- Amazon Redshift
- Amazon Elasticsearch Service
- Amazon QuickSight

さらに、Amazon EC2 インスタンスをセルフマネージド型のビッグデータアプリケーションとして使用することもできます。

## Amazon Kinesis Streams

Amazon Kinesis Streams を使用して、ストリーミングデータをリアルタイムに処理または分析するカスタムアプリケーションを構築できます。Amazon Kinesis Streams は、ウェブサイトのクリックストリーム、金融取引、ソーシャルメディアフィード、IT ログ、場所追跡イベントなど、数十万ものソースから毎時テラバイト単位で送られてくるデータを連続的に取得し保存します。[7]

Amazon Kinesis クライアントライブラリ (KCL) を使用すると、Amazon Kinesis アプリケーションを構築し、ストリーミングデータを使用してリアルタイムのダッシュボードの駆動、アラートの生成、および動的な価格設定と広告の実装を行うことができます。また、Amazon Kinesis Streams から他の AWS のサービス (Amazon Simple Storage Service (Amazon S3)、Amazon Redshift、Amazon Elastic MapReduce (Amazon EMR)、AWS Lambda など) にデータを発行することもできます。

AWS マネジメントコンソール、API[11]、または SDK を使用してデータストリームに必要な入力と出力のレベルを 1 秒あたり 1 メガバイト (MB/sec) のブロック単位でプロビジョニングします。[8] ストリームのサイズは、いつでも調整して増減できます。そのためにストリームを再起動する必要はなく、データソースからストリームへのデータのプッシュに影響を与えることもありません。ストリームに取り込まれたデータは、数秒以内で分析可能になります。

ストリームデータは、同一リージョン内の複数のアベイラビリティーゾーンに 24 時間保存されます。この間に、データの読み取り、再読み取り、バックフィル、および分析が可能であり、長期保存のために Amazon S3 や Amazon Redshift にデータを移動することもできます。KCL を使用すると、開発者は、ストリーミングデータの負荷分散、分散サービスの調整、および耐障害性のあるデータ処理に伴う手間のかかる作業から解放されて、ビジネスアプリケーションの作成に専念できます。

## 最適な使用パターン

Amazon Kinesis Streams は、プロデューサー (データソース) から迅速にデータを移動して継続的に処理する必要がある場合に役立ちます。たとえば、データを変換して別のデータストアに送信する、メトリクスや分析をリアルタイムで実行する、複数のストリームを取得してより複雑なストリームに集約する、下流処理を行うなどの場合に適しています。Amazon Kinesis Streams を使用した一般的な分析のシナリオは以下のとおりです。

- **リアルタイムデータ分析** – Amazon Kinesis Streams では、ウェブサイトのクリックストリームデータの分析や顧客エンゲージメント分析など、ストリーミングデータのリアルタイム分析ができます。
- **ログとデータフィードの取り込みと処理** – Amazon Kinesis Streams を使用すると、プロデューサーから Amazon Kinesis ストリームにデータを直接プッシュできます。たとえば、システムとアプリケーションのログを Amazon Kinesis Streams に送信し、そのストリームに数秒以内にアクセスして処理できます。これにより、フロントエンドサーバーやアプリケーションサーバーで障害が発生してもログデータが失われることはなく、ソースでローカルに保存されるログも減少します。Amazon Kinesis Streams でデータの取り込みが加速されるのは、取り込むデータがサーバーでバッチ処理されずに送信されるためです。
- **リアルタイムのメトリクスとレポート作成** – Amazon Kinesis Streams に取り込まれたデータからメトリクスを抽出し、KPI を生成することで、リアルタイムでレポートとダッシュボードを実行できます。これにより、データ処理アプリケーションのロジックでは、データのバッチが到着するのを待たずに、データがストリーミングされると同時に処理できます。

## 料金モデル

Amazon Kinesis Streams では、シンプルな従量制の課金システムを採用しており、前払いの義務や最低料金はなく、実際に利用したリソース分だけ支払います。Amazon Kinesis ストリームは 1 つ以上のシャードで構成され、シャードごとに読み取りトランザクションは 1 秒あたり 5 件、読み取られるデータ量は 1 秒あたり最大 2 MB です。シャードごとに、書き込みトランザクションは 1 秒あたり最大 1,000 件、書き込まれるデータ量は 1 秒あたり最大 1 MB です。

ストリームのデータ容量は、ストリームに指定したシャードの数によって決まります。ストリームの総容量は各シャードの容量の合計です。料金構成は、シャード単位の時間料金と百万件の PUT トランザクションごとの料金の 2 つのみです。詳細については、Amazon EC2 料金表を参照してください。[9] Amazon EC2 で実行して Amazon Kinesis ストリームを処理するアプリケーションには、Amazon EC2 の標準料金も適用されます。

## パフォーマンス

Amazon Kinesis Streams では、必要なスループット容量をシャード単位で選択できます。Amazon Kinesis ストリームのシャードごとに、1 秒あたり 1,000 件の書き込みトランザクションを 1 秒あたり 1 MB のデータまでキャプチャできます。Amazon Kinesis アプリケーションは、各シャードのデータを 1 秒あたり 2 MB まで読み取ることができます。必要な容量に応じて、いくつでも必要なだけのシャードをプロビジョニングできます。たとえば、1 GB/秒のデータストリームには 1,024 個のシャードが必要です。

## 耐久性と可用性

Amazon Kinesis Streams では、同じ AWS リージョン内の 3 つのアベイラビリティーゾーン間でデータが同期的にレプリケートされため、高い可用性とデータ耐久性が得られます。

さらに、DynamoDB にカーソルを保存して、Amazon Kinesis ストリームから読み取られた内容を永続的に追跡できます。ストリームからデータを読み取り中にアプリケーションで障害が発生した場合は、アプリケーションを再起動し、カーソルを使用して、障害で中断した正確な箇所から読み取りを再開できます。

## スケーラビリティと伸縮性

ストリームの容量は、ビジネスニーズや運用ニーズに応じていつでも増減できます。その際に、進行中のストリーム処理が中断されることはありません。API 呼び出しや開発ツールを使用することで、Amazon Kinesis Streams 環境を自動的にスケールして需要を満たし、使用分だけを支払うことができます。

## インターフェイス

Amazon Kinesis Streams には 2 つのインターフェイスがあります。Amazon Kinesis Streams にデータを送信するためにデータプロデューサーで使用する入力と、着信するデータを処理して分析するための出力です。プロデューサーでは、データの入力に Amazon Kinesis PUT API、AWS Software Development Kit (SDK) またはツールキット抽象、Amazon Kinesis Producer Library (KPL)、または Amazon Kinesis Agent を使用できます。[10]

Amazon Kinesis ストリームに入力されたデータを処理するには、ストリーミングデータをリアルタイムで処理するアプリケーションを構築および実行するためのクライアントライブラリが用意されています。KCL[17] は、Amazon Kinesis Streams と特定の処理ロジックを含むビジネスアプリケーションとの仲介として機能します。[11] また、Amazon Kinesis ストリームと Apache Storm を統合して前者から後者への読み取りを行う Amazon Kinesis Storm Spout もあります。[12]

### アンチパターン

Amazon Kinesis Streams には以下のアンチパターンがあります。

- **小規模な一貫したスループット** – Amazon Kinesis Streams は、200 KB/秒以下のストリーミングデータにも使用できますが、実際はそれより大きなデータのスループット用に設計および最適化されています。
- **長期のデータ保存と分析** – Amazon Kinesis Streams は、長期のデータ保存には適していません。デフォルトでは、データの保持期間は 24 時間であり、最大 7 日間まで保持期間を延長できます。7 日間より長く保存する必要があるデータは、別の永続的なストレージサービス (Amazon S3、Amazon Glacier、Amazon Redshift、DynamoDB など) に移動できます。

## AWS Lambda

AWS Lambda を使用すると、サーバーをプロビジョニングまたは管理しなくてもコードを実行できます。[13] 使用したコンピューティング時間に対してのみ課金され、コードが実行中でなければ料金はかかりません。Lambda では、実質的にすべてのタイプのアプリケーションやバックエンドサービスでコードを実行でき、一切の管理が不要です。コードをアップロードするだけで、コードの実行とスケールに必要な処理はすべて Lambda により自動的に実行され、高い

可用性が維持されます。コードは、AWS の他のサービスから自動的にトリガーするか、ウェブやモバイルアプリケーションから直接呼び出すように設定できます。

## 最適な使用パターン

Lambda では、データの変更、システム状態の変化、ユーザーによるアクションなどのトリガーに応じてコードを実行できます。Lambda は、Amazon S3、DynamoDB、Amazon Kinesis Streams、Amazon Simple Notification Service (Amazon SNS)、Amazon CloudWatch などの AWS のサービスから直接トリガーされるため、さまざまなリアルタイムデータ処理システムを構築できます。

- **ファイルのリアルタイム処理** – Amazon S3 にアップロードしたファイルや変更したファイルに対して、Lambda をトリガーして処理を呼び出すことができます。たとえば、Amazon S3 にアップロードしたイメージをカラーからグレースケールに変更する場合などに便利です。
- **ストリームのリアルタイム処理** – Amazon Kinesis Streams と Lambda を使用して、クリックストリーム分析、ログフィルタリング、およびソーシャルメディア分析のためのストリーミングデータを処理できます。
- **抽出、変換、およびロード** – Lambda を使用して、データリポジトリ間でデータを変換してロードするためのジョブを実行できます。
- **cron の代用** – EC2 インスタンスで cron を実行するよりも安価で可用性の高いソリューションとして、スケジュール式を使用して Lambda 関数を定期的に実行できます。
- **AWS イベントの処理** – Amazon S3 にログインし、S3 バケット通知を使用して Lambda 関数をトリガーするだけで、他の多くのサービス (AWS CloudTrail など) をイベントソースとして使用できます。

## 料金モデル

Lambda では、使用した分だけの料金を支払います。料金は、関数に対するリクエスト数とコードの実行時間に応じて請求されます。Lambda では、毎月、100 万件の無料リクエストおよび 400,000 GB/秒のコンピューティング時間が無料利用枠となっています。無料利用枠を使い切ると、100 万件のリクエストあたり 0.20 USD (0.0000002 USD/リクエスト) が課金されます。さらに、コードの実行時間に対しても、割り当てられたメモリに応じた料金がかかります。1 GB/秒の使用につき 0.00001667 USD の料金が発生します。詳細については、Amazon AWS Lambda 料金表を参照してください。[14]

## パフォーマンス

コードを Lambda に初めてデプロイすると、通常、アップロードの数秒以内に関数の呼び出しの準備が完了します。Lambda は、ミリ秒単位でイベントを処理するように設計されています。Lambda 関数を作成または更新した直後や、最近使用していなかった場合は、レイテンシーが高くなります。

## 耐久性と可用性

Lambda は、レプリケーションと冗長性を通じて、サービス自体とそれが操作する Lambda 関数の両方の高可用性を実現するよう設計されています。いずれに対しても、メンテナンスウィンドウや予定されたダウンタイムはありません。障害が発生すると、Lambda 関数の同期呼び出しでは応答として例外がスローされます。Lambda 関数の非同期呼び出しでは、最低 3 回の試行後にイベントは拒否される場合があります。

## スケーラビリティと伸縮性

実行できる Lambda 関数の数には制限がありません。ただし、安全策として同一リージョン内でのアカウントあたりの同時実行数はデフォルトで 100 に制限されています。この制限は、AWS サポートチームに連絡して引き上げることができます。

Lambda は、ユーザーに代わって自動的にスケールするように設計されています。関数のスケーリングに対する基本的な制限はありません。Lambda は、受信イベントのレートに合わせて動的に容量を割り当てます。

## インターフェイス

Lambda 関数は、さまざまな方法で管理できます。Lambda 関数は、Lambda コンソールのダッシュボードを使用して簡単に表示、削除、更新、およびモニタリングができます。Lambda 関数は、AWS CLI と AWS SDK で管理することもできます。

Lambda 関数は、Amazon S3 バケット通知、DynamoDB Streams、CloudWatch ログ、Amazon SES、Amazon Kinesis Streams、Amazon SNS、Amazon Cognito などの AWS イベントからトリガーできます。CloudTrail をサポートするサービスでは、CloudTrail 監査ログに応答することで、どの API 呼び出しでも Lambda のイベントとして処理できます。イベントソースの詳細については、「コアコンポーネント: AWS Lambda 関数とイベントソース」を参照してください。[15]

Lambda は、Java、Node.js、Python などのプログラミング言語をサポートしています。コードには、既存のライブラリだけでなく、ネイティブライブラリも含めることができます。Lambda 関数は、Bash、Go、Ruby など、Amazon Linux AMI でサポートされている言語を使用して簡単にプロセスを起動できます。[16]詳細については、Node.js、Python、および Java に関するドキュメントを参照してください。[17]

## アンチパターン

Lambda には以下のアンチパターンがあります。

- **長期実行アプリケーション** – 各 Lambda 関数は 300 秒以内に完了する必要があります。ジョブの実行時間が 5 分を超えるような長期実行アプリケーションには、Amazon EC2 をお勧めします。または、Lambda 関数のチェーンを作成して、関数 1 が関数 2 を呼び出し、関数 2 が関数 3 を呼び出すというようにしてプロセスを完了します。
- **動的なウェブサイト** – Lambda では、静的なウェブサイトは実行できますが、非常に動的で大容量のウェブサイトを実行すると、パフォーマンスが大幅に制限される場合があります。動的なウェブサイトには、Amazon EC2 や Amazon CloudFront を使用することをお勧めします。
- **ステートフルアプリケーション** – Lambda のコードは、「ステートレス」なスタイルで記述する必要があります。つまり、基盤となるコンピューティングインフラストラクチャとのアフィニティはないものと想定する必要があります。ローカルファイルシステムへのアクセス、子プロセス、および同様のアーティファクトはリクエストの有効期限を超えることができないため、すべての永続的な状態は Amazon S3、Amazon DynamoDB など、インターネットが利用可能なストレージサービスに保存する必要があります。

# Amazon EMR

Amazon EMR[25] は、費用対効果の高い方法でデータを簡単に処理して迅速に保存するための高分散コンピューティングフレームワークです。[18] Amazon EMR は、オープンソースフレームワークである Aacphe Hadoop を使用して、Amazon EC2 インスタンスの拡大縮小可能なクラスター全体にデータおよび処理を分散し、Hadoop の代表的なツールである Hive、Pig、Spark などを使用できるようにします。Hadoop はビッグデータの処理と分析を実行するフレームワークを提供し、Hadoop クラスターのインフラストラクチャとソフトウェアのプロビジョニング、管理、維持に伴うすべての手間のかかる作業は Amazon EMR が担当します。

## 最適な使用パターン

Amazon EMR の柔軟なフレームワークにより、データセットの小さなジョブへの分割、Hadoop クラスターの多数のコンピューティングノードへのジョブの分散など、大規模な処理の問題が軽減されます。この機能は、ビッグデータ分析のさまざまな使用パターンに役立ちます。数例を以下に示します。

- ログの処理と分析
- 移動するデータの大規模な抽出、変換、およびロード (ETL)
- リスクモデリングと脅威分析
- 広告ターゲティングとクリックストリーム分析
- ゲノミクス
- 予測分析
- アドホックデータマイニングと分析

詳細については、「Amazon EMR のベストプラクティス」ホワイトペーパーを参照してください。[19]

## 料金モデル

Amazon EMR では、無限に実行する永続的なクラスターを起動するか、分析の完了後に終了する一時的なクラスターを起動することができます。いずれの場合でも、料金はクラスターの実行時間分だけ発生します。

Amazon EMR は、多様な Amazon EC2 インスタンスタイプ (スタンダード、ハイ CPU、ハイメモリ、ハイ I/O など) とすべての Amazon EC2 料金オプション (オンデマンド、リザーブド、およびスポット) をサポートしています。Amazon EMR クラスター (「ジョブフロー」とも呼ばれます) を起動するときに、プロビジョニングする Amazon EC2 インスタンスの数とタイプを選択します。Amazon EMR の料金は、Amazon EC2 の料金が加算された額です。詳細については、Amazon EMR 料金表を参照してください。[20]

## パフォーマンス

Amazon EMR のパフォーマンスは、どのタイプの EC2 インスタンスを使用してクラスターを実行し、いくつの EC2 インスタンスを使用して分析を実行するかに依存します。処理の要件に適したインスタンスタイプを選択し、十分なメモリ、ストレージ、および処理能力を確保する必要があります。EC2 インスタンスの仕様の詳細については、「Amazon EC2 インスタンスタイプ」を参照してください。[21]

## 耐久性と可用性

デフォルトでは、Amazon EMR はコアノードのエラーに対して耐障害性があり、スレーブノードがダウンしてもジョブの実行を続行します。現時点では、スレーブが失敗した場合に Amazon EMR で代わりのノードが自動的にプロビジョニングされませんが、お客様はノードの状態をモニタリングして失敗したノードを CloudWatch で置き換えることができます。

マスターノードの障害はめったに発生しませんが、発生した場合に備えてデータを Amazon S3 などの永続的なストアにバックアップしておくことをお勧めします。必要に応じて、MapR ディストリビューションで Amazon EMR を実行することもできます。MapR は NameNode がないアーキテクチャーを提供します。このアーキテクチャーは、自動フェイルオーバーおよびフォールバックにより、複数の障害が同時に発生しても耐えることができます。[22] メタデータは、データと同じように分散およびレプリケートされます。NameNode がないアーキテクチャーでは、保存できるファイル数に実質的に制限がなく、外部のネットワーク接続ストレージにも依存しません。

## スケーラビリティと伸縮性

Amazon EMR では、実行中のクラスターのサイズ変更が簡単にできます。[23] Hadoop Distributed File System (HDFS) を保持するコアノードをいつでも追加して、処理能力を強化し、HDFS ストレージ容量 (およびスループット) を増やすことができます。さらに、Amazon S3 をネイティブで使用するか、EMFS をローカル HDFS と併用または代用することで、メモリとコンピューティングをストレージから分離し、柔軟性とコスト効率を高めることができます。

また、Hadoop ジョブを処理できるが HDFS を保持しないタスクノードは、いつでも追加または削除できます。一部のお客様は、バッチ処理の発生時に何百というインスタンスをクラスターに追加し、処理の完了時に余分なインスタンスを削除しています。たとえば、クラスターで今後 6 か月間のデータ処理量が予測できない場合や、処理ニーズが突発的に急増する場合があります。

Amazon EMR では、いつでも容量を簡単に追加または削除できるため、将来のニーズの予測やピーク需要に備えたプロビジョニングは必要ありません。

また、コンソールで数回クリックするか、プログラムで API を 1 回呼び出すだけで、さまざまなサイズの新しいクラスターをいつでも一括して追加または削除できます。[24]

## インターフェイス

Amazon EMR は、ビッグデータ分析に使用できる、Hadoop 上で動作する多数のツールをサポートしています。代表的なオプションを以下に簡単に紹介します。

***Hive***

Hive はオープンソースのデータウェアハウスであり、Hadoop 上で動作する分析パッケージです。Hive は SQL ベースの言語である Hive QL で運用され、データの構造化、集約、クエリを行うことができます。Hive QL は標準的な SQL を超えるものであり、map/reduce 関数や、JSON や Thrift といった複雑で拡張可能なユーザー定義のデータタイプに対して高度なサポートを提供します。この機能により、テキスト文書やログファイルなど、複雑で構造化されていないデータソースの処理が可能になります。

Hive では Java で書かれたユーザー定義の関数を使用したユーザー拡張が可能です。Amazon EMR では DynamoDB や Amazon S3 との直接統合など、Hive に多数の機能強化を行っています。たとえば、Amazon EMR では Amazon S3 から自動的にテーブルパーティションを読み込んだり、一時ファイルを使用せずに Amazon S3 のテーブルにデータを書き込んだり、カスタム map/reduce 操作のスクリプトや追加のライブラリといった Amazon S3 のリソースにアクセスしたりすることができます。詳細については、「Apache Hive」 (*Amazon EMR* リリースガイド) を参照してください。[25]

### *Pig*

Pig は Hadoop 上で実行されるオープンソースの分析パッケージです。Pig は SQL に類似した言語である Pig Latin で運用され、データの構造化、集約、クエリを行うことができます。Pig Latin では、SQL に類似した操作に加えて、map/reduce 関数および複雑で拡張可能なユーザー定義のデータタイプに対しても高度なサポートを提供します。この機能により、テキスト文書やログファイルなど、複雑で構造化されていないデータソースの処理が可能になります。

Pig では Java で書かれたユーザー定義の関数を使用したユーザー拡張が可能です。Amazon EMR では、複数のファイルシステムの使用 (Pig は通常 1 つのリモートファイルシステムにのみアクセス可能)、Amazon S3 からの顧客の JAR やスクリプトの読み込み (例: 「REGISTER s3:///my-bucket/piggybank.jar」)、文字列と日時の処理に関する追加機能など、Pig に多数の機能強化を行っています。詳細については、「Apache Pig」[33] (*Amazon EMR* リリースガイド) を参照してください。

### *Spark*

Spark は、Hadoop 上に構築されたオープンソースのデータ分析エンジンであり、インメモリ MapReduce の基本機能を備えています。Spark は、特定の分析を高速化します。また、Shark (SQL 駆動のデータウェアハウス)、Spark Streaming (ストリーミングアプリケーション)、GraphX (グラフシステム)、MLlib (機械学習) など、他のパワーツールの基盤でもあります。詳細については、ブログ記事の「Amazon EMR クラスターへの Apache Spark のインストール」を参照してください。[26]

### *HBase*

HBase は、Google の BigTable に基づいて設計されたオープンソースのリレーショナルでない分散データベースです。Apache Software Foundation の Hadoop プロジェクトの一部として開発され、Hadoop Distributed File System (HDFS) 上で動作して、BigTable に似た機能を Hadoop で実現します。HBase では、列ベースの圧縮および保存を使用することにより、障害に強く効率的な方法で大量のスパースデータを保存できます。さらに、データがディスクではなくメモリに格納されるので、データ検索が迅速に実行されます。

HBase は、シーケンシャル書き込み操作用に最適化されており、バッチ挿入、更新、および削除処理に効率的に対応します。 HBase は、Hadoop のファイルシステムを共有して、Hadoop ジョブに対する直接入力および出力として機能することで、Hadoop とシームレスに連携します。また、HBase テーブルに対する SQL のようなクエリ、Hive ベースのテーブルとの結合、および Java Database Connectivity (JDBC) を有効にすることで、Apache Hive とも統合されます。Amazon EMR では、HBase を Amazon S3 にバックアップ (完全/増分、手動/自動) したり、以前に作成したバックアップから復元したりすることができます。詳細については、「HBase と EMR」 (*Amazon EMR* 開発者ガイド) を参照してください。[27]

### *Impala*

Impala は SQL 構文を使用したインタラクティブなアドホッククエリを行う Hadoop エコシステムのオープンソースツールです。これは、従来のリレーショナルデータベース管理システム (RDBMS) にある超並列処理 (MPP) エンジンに類似した MPP を MapReduce の代わりに活用します。このアーキテクチャでは、HDFS または HBase テーブルのデータを非常にすばやくクエリできるほか、多様なデータ型を処理するだけでなく実行時にスキーマを提供する Hadoop の機能を活用することができます。このため、Impala はインタラクティブでレイテンシーが短い分析を行う最適なツールです。

また、Impala には Java および C++ のユーザー定義関数があり、ODBC ドライバーと JDBC ドライバーを通じて BI ツールに接続できます。Impala は Hive メタストアを使用して、パーティション名やデータ型などの入力データに関する情報を保管します。詳細については、「Impala と EMR」(*Amazon EMR* 開発者ガイド) を参照してください。[28]

***Hunk***

Hunk は、マシンデータにだれでもアクセスして利用し、そのデータから価値を引き出せるようにするために Splunk によって開発されました。Hunk を使用すると、Amazon EMR と Amazon S3 に保存されたデータをインタラクティブに探索、分析、および視覚化し、Hadoop で Splunk 分析を活用できます。詳細については、「Amazon EMR with Hunk: Splunk Analytics for Hadoop and NoSQL」を参照してください。[37]

***Presto***

Presto は、低レイテンシーでアドホックなデータ分析用に最適化されたオープンソースの分散 SQL クエリエンジンです。ANSI SQL 標準をサポートし、複雑なクエリ、集計、結合、ウィンドウ関数を実行できます。Presto を使用して、Hadoop Distributed File System (HDFS) や Amazon S3 など、複数のデータソースのデータを処理できます。

**その他のサードパーティー製ツール**

Amazon EMR では、R (統計)、Mahout (機械学習)、Ganglia (モニタリング)、Accumulo (安全な NoSQL データベース)、Hue (Hadoop データを分析するためのユーザーインターフェイス)、Sqoop (リレーショナルデータベースコネクタ)、HCatalog (テーブルおよびストレージ管理) といった他の人気のあるさまざまなアプリケーションやツールを Hadoop エコシステムでサポートしています。

また、ビジネスニーズに対応するために独自のソフトウェアを Amazon EMR 上にインストールすることもできます。AWS では、Amazon EMR の S3DistCp を使用して、大量のデータを Amazon S3 から HDFS へ、HDFS から Amazon S3 へ、および Amazon S3 バケット間で移動できます。S3DistCp は、MapReduce を使用して大量のデータを効率的に移動するオープンソースツールである DistCp の拡張です。[29]

必要に応じて、HDFS の実装である EMR File System (EMRFS) を使用し、Amazon EMR クラスターのデータを Amazon S3 に保存することもできます。EMRFS の整合性のあるビューに加えて、Amazon S3 サーバー側およびクライアント側の暗号化を有効にすることができます。EMRFS を使用すると、メタデータストアが DynamoDB に透過的に構築されて Amazon S3 とのやり取りが管理しやすくなり、複数の EMR クラスターで Amazon S3 の同じ EMRFS メタデータおよびストレージを使用できるようになります。

## アンチパターン

Amazon EMR には以下のアンチパターンがあります。

- **小さなデータセット** – Amazon EMR は大量の並列処理用に構築されています。データセットが小さくて 1 つのマシン、1 つのスレッドで迅速に実行できる場合があります。そのような小さなデータセットは、MapReduce ジョブに余分なオーバーヘッドを与えるまでもなく、1 つのシステムのメモリ内で簡単に処理できます。

- **ACID トランザクション要件** – Hadoop で ACID (アトミック性、整合性、分離、および堅牢性) または制限された ACID を達成する方法はいくつかありますが、要件が厳格なワークロードに対しては別のデータベース (Amazon RDS や、Amazon EC2 で実行するリレーショナルデータベースなど) を使うほうが適しています。

## Amazon Machine Learning

Amazon ML は、予測分析や機械学習テクノロジーをだれでも利用できるようにするサービスです。[30] Amazon ML は、複雑な機会学習 (ML) アルゴリズムやテクノロジーを習得することなく、ML モデルを作成するための可視化ツールとウィザードを提供します。モデルの準備ができたら、Amazon ML で API オペレーションを使用し、アプリケーションの予測を簡単に取得できます。カスタムの予測生成コードを実装したりインフラストラクチャを管理したりする必要はありません。

Amazon ML は、Amazon S3、Amazon Redshift、または Amazon RDS に保存されているデータに基づいて ML モデルを作成できます。組み込まれているウィザードの手順に従って、データの操作、ML モデルのトレーニング、モデルの品質の評価、ビジネスの目標に合わせた出力の調整をインタラクティブに行うことができます。モデルの準備が整ったら、バッチを使用するか、低レイテンシーのリアルタイム API を使用して予測をリクエストできます。

### 最適な使用パターン

Amazon ML では効率よくデータのパターンを検索し、これらのパターンに基づいて ML モデルを作成して、新しい未見のデータポイントに関する予測を生成できます。たとえば、以下のことができます。

- **疑わしいトランザクションにフラグを設定するアプリケーションの有効化** – 新しいトランザクションが正当または不正であるかを予測する ML モデルを構築します。
- **製品需要の予測** – 注文の履歴情報を入力し、将来の注文数量を予測します。
- **アプリケーションコンテンツのパーソナライズ** – ユーザーにとって最も関心が高い事項を予測し、これらの予測をアプリケーションからリアルタイムで取得します。
- **ユーザーアクティビティの予測** – ユーザーの行動を分析し、ウェブサイトをカスタマイズしてユーザーエクスエペリエンスを向上させます。
- **ソーシャルメディアからの情報の取り込み** – ビジネスの意思決定に影響を与える可能性があるソーシャルメディアフィードを取り込み、分析します。

## 料金モデル

Amazon ML では、使用した分のみ料金が発生します。最低料金や前払いの義務は発生しません。Amazon ML では、予測モデルを構築するためのコンピューティング所要時間に対する時間単位の料金と、アプリケーションのために生成された予測の数に応じた料金が請求されます。リアルタイムの予測に対しては、モデルを実行するためのメモリ所要量に基づく時間単位のリザーブドキャパシティー料金もかかります。

データ分析、モデルトレーニング、および評価の料金は、それらを実行するためのコンピューティング所要時間に基づき、入力データのサイズと属性数、および適用された変換の数と種類に応じて変動します。データの分析およびモデルの構築料金は 0.42 USD/時間です。予測料金はバッチおよびリアルタイムに

分かれます。バッチ予測は予測 1,000 件ごとに 0.10 USD で、端数は次の 1,000 件に切り上げられます。リアルタイム予測は予測 1 件ごとに 0.0001 USD で、端数は次のペニーに切り上げられます。リアルタイム予測に対しては、モデルのためにプロビジョニングされたメモリ 10 MB ごとに 0.001 USD/時間のリザーブドキャパシティー料金もかかります。

モデルの作成時に、コストを管理して予測パフォーマンスを制御するために、各モデルの最大メモリサイズを指定します。リザーブドキャパシティー料金がかかるのは、リアルタイム予測でモデルを有効にしている間のみです。Amazon S3、Amazon RDS、または Amazon Redshift に保存されたデータの料金は個別に請求されます。詳細については、Amazon Machine Learning 料金表を参照してください。[31]

## パフォーマンス

モデルを作成する所要時間や、これらのモデルからバッチ予測をリクエストする所要時間は、データレコードの数、これらのレコード内に含まれる属性の種類とディストリビューション、および指定するデータ処理の「レシピ」の複雑さに応じて異なります。

大部分のリアルタイム予測リクエストでは、レスポンスが 100 ミリ秒以内に返されるため、インタラクティブなウェブ、モバイル、またはデスクトップアプリケーションに申し分のない速さです。リアルタイム API で予測を生成する所要時間は、入力データレコードのサイズ、および予測を生成する ML モデルに関連付けられたデータ処理の「レシピ」の複雑さに応じて異なります。[32] リアルタイム予測のために有効になっている ML モデルごとに、デフォルトで最大 200 件のトランザクション/秒をリクエストできます。この件数は、カスタマーサポートに連絡して引き上げることができます。ML モデルからの予測リクエスト数は、CloudWatch メトリクスでモニタリングできます。

## 耐久性と可用性

Amazon ML は高可用性を実現するよう設計されています。メンテナンスウィンドウや予定されたダウンタイムはありません。サービスは Amazon の可用性の高い実証済みデータセンターで実行され、AWS リージョンごとに 3 つの施設間でのサービススタックレプリケーションが設定されるため、サーバーの障害やアベイラビリティーゾーンの機能停止が発生しても耐障害性が維持されます。

## スケーラビリティと伸縮性

ML モデルの作成やバッチ予測のリクエストでは、最大 100 GB までのデータセットを処理できます。大規模なバッチ予測の場合は、入力データレコードを複数のチャンクに分割して大量の予測データを処理できます。

デフォルトでは、最大 5 つのジョブを同時に実行できます。カスタマーセンターに連絡して、この制限を引き上げることもできます。Amazon ML はマネージド型サービスであるため、サーバーのプロビジョニングは不要です。アプリケーションの拡大に応じてスケールできるため、オーバープロビジョニングしたり、使用していないリソースに支払ったりする必要はありません。

## インターフェイス

データソースは簡単に作成できます。データを Amazon S3 に追加するだけです。または Amazon Redshift や、Amazon RDS が管理する MySQL データベースから直接データをプルできます。データソースを定義したら、コンソールを使用して Amazon ML を操作できます。Amazon ML へのプログラムによるアクセスを有効にするには、AWS SDK および Amazon ML API を使用します。[33] Windows、Mac、および Linux/UNIX システムで AWS CLI を使用して Amazon ML エンティティを作成し、管理することもできます。

### アンチパターン

Amazon ML には以下のアンチパターンがあります。

- **非常に大規模なデータセット** – Amazon ML では最大 100 GB のデータをサポートできますが、現時点ではテラバイトスケールのデータの取り込みはサポートされていません。そのようなユースケースに対しては、通常、Amazon EMR を使用して Spark の Machine Learning Library (MLlib) を実行します。
- **サポート対象外の学習タスク** – Amazon ML では、バイナリ分類 (2 つの選択肢から 1 を選択して測定指標を提供)、複数クラス分類 (選択肢を 2 つ以上のオプションに拡大)、または数値回帰 (数値を直接予測) を行う ML モデルを作成できます。シーケンス予測や管理されないクラスタリングなどのサポート対象外の ML タスクを処理するには、Amazon EMR を使用して Spark および MLlib を実行できます。

## Amazon DynamoDB

DynamoDB は、高速なフルマネージド型 NoSQL データベースサービスで、任意のデータ量の格納と取得、任意のレベルのリクエストトラフィックへの対応を簡素化し、コストを効率化します。[34] DynamoDB は、可用性が高い分散データベースクラスターの運用とスケーリングに伴う管理作業を肩代わりします。DynamoDB は、10 ミリ秒未満のレイテンシーと予測可能なパフォーマンスをシームレスなスループットとスケーラブルなストレージで提供することで、要求の厳しいアプリケーションのレイテンシーとスループットの要件を満たします。

DynamoDB は、インデックス化されたテーブルに構造化データを保存し、1 バイト～400 KB の項目に対する低レイテンシーの読み書きアクセスを許可します。DynamoDB は、データ型として数値、文字列、バイナリの 3 つをスカラーおよび多値セットの両方でサポートしています。これらのデータ型では JSON、XML、HTML などのドキュメントストアがサポートされます。テーブルには固定されたスキーマがないため、データ項目ごとに異なる数の属性を保持できます。プライマリキーは、単一属性ハッシュキーにすることも、複合ハッシュ範囲キーにすることもできます。

DynamoDB は、グローバルおよびローカルの両方のセカンダリインデックスを提供し、プライマリキー以外の属性に対するクエリの柔軟性を強化します。DynamoDB は、結果的に整合性のある読み込み (デフォルト) と強力な整合性のある読み込み (オプション) の両方、さらに項目の入力、更新、削除、条件付きオペレーション、およびインクリメント/デクリメントに対する項目レベルの暗黙的なトランザクションを提供します。

DynamoDB は、分析、データウェアハウス、データのインポート/エクスポート、バックアップ、およびアーカイブの目的で Amazon EMR、Amazon Redshift、AWS Data Pipeline、Amazon S3 などの他のサービスと統合されます。

## 最適な使用パターン

DynamoDB は、読み書きのレイテンシーが低く、コードの変更やダウンタイムを伴わずにストレージとスループットを必要に応じてスケールアップ/ダウンできる柔軟な NoSQL データベースを必要とする既存または新規のアプリケーションに最適です。

一般的ユースケース:

- モバイルアプリ
- ゲーム
- デジタル広告サービス
- ライブ投票
- ライブイベントへのオーディエンス参加
- センサーネットワーク
- ログ処理
- ウェブベースコンテンツへのアクセスコントロール
- Amazon S3 オブジェクトのメタデータストレージ
- e コマースショッピングカート
- ウェブセッション管理

これらのユースケースの多くでは、ダウンタイムやパフォーマンスの低下は組織のビジネスに即座に悪影響をもたらすため、可用性の高いスケーラブルなデータベースが必要です。

## 料金モデル

DynamoDB では、使用分だけの支払いとなり、最低料金はありません。DynamoDB の料金構成には、プロビジョンドスループット性能 (時間単位)、インデックス化データストレージ (月額料金/GB)、データ転送イン/アウト (月額料金/GB) の 3 つがあります。新規のお客様は、AWS 無料利用枠を使用して DynamoDB を無料で開始できます。[35] 詳細については、Amazon DynamoDB 料金表を参照してください。[36]

## パフォーマンス

SSD と、属性に対するインデックス作成の制限により、高スループットと低レイテンシーが実現され、読み書き操作のコストが大幅に削減されます。[37] データセットが拡大すると、ワークロードの低レイテンシーを維持するために予測可能なパフォーマンスが要求されます。この予測可能なパフォーマンスは、特定のテーブルに必要なプロビジョニングされたスループット容量を定義することで達成できます。

必要なスループットレートを達成するためのリソースのプロビジョニングはバックグラウンドで処理されます。アプリケーションのスループットレートに影響する可能性があるインスタンス、ハードウェア、メモリなどの要因をユーザーが気にする必要はありません。プロビジョンドスループット性能の予約は伸縮自在であり、オンデマンドで増減できます。

## 耐久性と可用性

DynamoDB には耐障害性機能が組み込まれています。データは同じリージョンの 3 つのアベイラビリティーゾーン間で自動的に同時にレプリケートされるため、高可用性が実現され、個別のマシンだけでなく施設全体で障害が発生してもデータが保護されます。

DynamoDB Streams は、テーブルで発生するすべてのデータアクティビティをキャプチャし、地理的リージョン間でのレプリケーション機能を設定して可用性を高めます。[38]

## スケーラビリティと伸縮性

DynamoDB はスケーラブルで伸縮自在です。DynamoDB テーブルに格納できるデータの量に制限はありません。DynamoDB の書き込み API オペレーションを使用して追加のデータを格納すると、追加のストレージが自動的に割り当てられます。データは自動的にパーティション分割され、必要に応じて再パーティション分割されます。SSD の使用により、データの規模を問わず、予測可能な低レイテンシーの応答時間が実現されます。また、ニーズの変動に応じてテーブルの読み書きキャパシティーを単に「ダイヤルアップ」または「ダイヤルダウン」できるため、サービスは伸縮自在です。[39]

## インターフェイス

DynamoDB には、低レベルの REST API に加えて、より高レベルの Java、ET、および PHP 用の SDK が用意されています。これらの SDK では、低レベルの REST API をラップし、いくつかのオブジェクトリレーショナルマッピング (ORM) 関数を提供します。これらの API は、DynamoDB に対して管理インターフェイスとデータインターフェイスの両方を提供します。API が現時点で提供するオペレーションでは、テーブル管理 (メタデータの作成、表示、削除、および取得) と属性の操作 (属性の取得、書き込み、および削除、インデックスを使用したクエリ、フルスキャン) が可能です。

標準 SQL は使用できませんが、DynamoDB の選択オペレーションを使用して SQL に似たクエリを作成し、指定した条件に基づいて属性のセットを取得できます。コンソールを使用して DynamoDB を操作することもできます。

## アンチパターン

DynamoDB には以下のアンチパターンがあります。

- **従来のリレーショナルデータベースに結び付けられている作成済みのアプリケーション** – 既存のアプリケーションを AWS クラウドに移行してリレーショナルデータベースの使用を継続する場合は、Amazon RDS (Amazon Aurora、MySQL、PostgreSQL、Oracle、または SQL Server) を使用するか、多くの事前設定された Amazon EC2 データベース AMI の 1 つを使用することができます。または、任意のデータベースソフトウェアを、管理している EC2 インスタンスにインストールすることもできます。

- **結合または複雑なトランザクション** – 多くのソリューションでは DynamoDB を利用してユーザーをサポートできますが、アプリケーションで結合や複雑なトランザクションが必要になる場合、またはその他従来のデータベースプラットフォームが提供するリレーショナルインフラストラクチャが必要になる場合があります。このような場合は、セルフマネージド型データベースを使用して Amazon Redshift、Amazon RDS、または Amazon EC2 を試すことができます。

- **バイナリラージオブジェクト (BLOB) データ** – デジタルビデオ、イメージ、音楽などの大きな (400 KB を超える) BLOB データを保存する場合は、Amazon S3 を使用することをお勧めします。ただし、このような場合でも、DynamoDB はバイナリオブジェクトに関するメタデータ (項目の名前、サイズ、作成日、所有者、場所など) を追跡するのに役立ちます。

- **I/O レートが低い大きなデータ** – DynamoDB は SSD ドライブを使用し、保存済みの GB あたりの I/O レートが高いワークロードに最適化されています。頻繁にはアクセスしない大量のデータを保存する場合は、Amazon S3 などの他のストレージオプションがより適している場合があります。

# Amazon Redshift

Amazon Redshift は、高速な、フルマネージド型のペタバイトスケールのデータウェアハウスサービスです。シンプルで費用対効果の高さが特長であり、すべてのデータを既存のビジネスインテリジェンスツールで効率的に分析できます。[40] 数百ギガバイトのデータから 1 ペタバイト以上の規模のデータセットに対して最適化されており、従来型の多くのデータウェアハウスソリューションの 10 分の 1 未満の費用で運用できるように設計されています。

Amazon Redshift では、列指向ストレージ技術、および複数ノード間でのクエリの並列化と分散化により、ほぼすべてのサイズのデータセットに対応する高速なクエリと I/O のパフォーマンスを実現しています。データウェアハウスのプロビジョニング、設定、モニタリング、バックアップ、保護に関連するほとんどの一般的な管理タスクが自動化されるので、低いコストで簡単に管理および保守できます。このオートメーションにより、ペタバイトスケールのデータウェアハウスを数分で構築できます。従来のオンプレミス実装のように数週間または数か月かかることはありません。

## 最適な使用パターン

Amazon Redshift は、既存のビジネスインテリジェンスツールを使用したオンライン分析処理 (OLAP) に適しています。組織は、以下の目的で Amazon Redshift を使用しています。

- 全世界における複数製品の販売データを分析する
- 過去の株式取引データを保存する
- 広告のインプレッション数やクリック数を分析する
- ゲームデータを集計する

- ソーシャルトレンドを分析する
- 医療における臨床的品質、運用効率、財務実績を測定する

## 料金モデル

Amazon Redshift のデータウェアハウスクラスターでは、長期契約や前払いが不要です。これにより、設備投資が不要になり、データウェアハウスのキャパシティーを前もって計画および購入することの複雑さから解放されます。料金は、クラスターのノードのサイズと数に基づいて決まります。

プロビジョニングされたストレージの 100% まではバックアップストレージに追加料金がかかりません。たとえば、使用中のクラスターに 2 つの XL ノードがあり、合計ストレージが 4 TB である場合、AWS は追加料金なしで最大 4 TB のバックアップストレージを Amazon S3 で提供します。プロビジョニングされたストレージサイズを超えるバックアップストレージと、クラスターの終了後に保存されたバックアップは、標準の Amazon S3 料金で課金されます。[41] Amazon S3 と Amazon Redshift の間の通信にデータ転送料金はかかりません。詳細については、Amazon Redshift 料金表をご覧ください。[42]

## パフォーマンス

Amazon Redshift では、さまざまな革新技術を駆使して、数百ギガバイト～1 ペタバイト以上のデータセットで高いパフォーマンスを実現しています。列指向ストレージ、データ圧縮、ゾーンのマッピングが使用されているため、クエリ実行に必要な I/O の量が削減されます。

Amazon Redshift には超並列処理 (MPP) アーキテクチャが採用されており、SQL オペレーションの並列化と分散化によって、すべての利用可能なリソースが活用されます。基盤となるハードウェアは、高パフォーマンスなデータ処理に合わせて設計されており、CPU とドライブの間のスループットを最大化するためにローカル接続ストレージを使用し、ノード間のスループットを最大化するために 10 GigE メッシュネットワークを使用しています。パフォーマンスは、データウェアハウスのニーズに基づいて調整できます。AWS には、SSD ドライブを使用する高密度コンピューティング (DC) オプションと高密度ストレージ (DS) オプションが用意されています。

## 耐久性と可用性

Amazon Redshift はデータウェアハウスクラスター内で障害が発生したノードを自動的に検出して置き換えます。代替ノードがプロビジョニングされて DB に追加されるまで、データウェアハウスクラスターは読み取り専用になりますが、通常はわずか数分しかかかりません。Amazon Redshift では、代替ノードはすぐに使用可能になり、最もよくアクセスするデータが Amazon S3 から最初にストリーミングされるため、データのクエリ実行を迅速に再開できます。

さらに、ドライブで障害が発生した場合でも、データウェアハウスクラスターは依然として使用可能です。Amazon Redshift は、クラスター全体でデータをミラーリングするため、別のノードのデータを使用して、障害が発生したノードを再構築します。Amazon Redshift クラスターは、1 つのアベイラビリティーゾーンに配置されます。ただし、Amazon Redshift のマルチ AZ セットアップを行う場合は、ミラーを設定してレプリケーションとフェイルオーバーを自己管理できます。[43]

## スケーラビリティと伸縮性

コンソールで数回クリックするか、API 呼び出しを 1 回使用するだけで、パフォーマンスまたはキャパシティーのニーズが変わったときにノードの数やタイプを簡単に変更できます。[44] Amazon Redshift では、160 GB の単一ノードから始め、多数のノードを使用するペタバイト以上の圧縮ユーザーデータまでスケールアップできます。詳細については、*Amazon Redshift* 管理ガイドの Amazon Redshift クラスターに関するトピックで、「クラスターおよびノード」セクションを参照してください。[45]

サイズを変更すると、Amazon Redshift により既存のクラスターが読み取り専用モードに変更され、選択したサイズの新しいクラスターがプロビジョニングされた後、古いクラスターから新しいクラスターにデータが並行してコピーされます。このプロセス中は、アクティブな Amazon Redshift クラスターの料金だけが発生します。新しいクラスターのプロビジョニング中は、古いクラスターに対してクエリを引き続き実行できます。データが新しいクラスターにコピーされると、Amazon Redshift によりクエリが新しいクラスターに自動的にリダイレクトされ、古いクラスターが削除されます。

## インターフェイス

Amazon Redshift には、カスタムの JDBC ドライバーと ODBC ドライバーが用意されており、コンソールの [Connect Client] タブからダウンロードできます。これにより、使い慣れたさまざまな SQL クライアントを使用できます。さらに、標準の PostgreSQL JDBC ドライバーと ODBC ドライバーを使用することもできます。Amazon Redshift ドライバーの詳細については、「Amazon Redshift および PostgreSQL」を参照してください。[46]

多くの一般的な BI および ETL ベンダーとの検証済みの統合の例も多く用意されています。[47]データウェアハウスクラスターにデータを取り込んだり、Amazon S3 や DynamoDB の内外にデータを移動したりするレートを最大化するために、各コンピューティングノードに対するロードとアンロードは同時に試行されます。Amazon Kinesis Firehose を使用してストリーミングデータを Amazon Redshift に簡単にロードできるため、現在使用している既存のビジネスインテリジェンスツールやダッシュボードで準リアルタイムの分析を行うことができます。コンピューティングの使用率、メモリの使用率、ストレージの使用率、および Amazon Redshift データウェアハウスクラスターに対する読み取り/書き込みトラフィックのメトリクスは、コンソールまたは CloudWatch API オペレーションを介して無料で使用できます。

## アンチパターン

Amazon Redshift には以下のアンチパターンがあります。

- **小規模なデータセット** – Amazon Redshift は、クラスター全体での並列処理用に構築されています。データセットが 100 GB 未満の場合、Amazon Redshift の機能は過分であるため、Amazon RDS の方が適していると言えます。
- **オンライントランザクション処理 (OLTP)** – Amazon Redshift は、非常に高速でコストの低い分析機能を実現するデータウェアハウスワークロードのために設計されています。高速なトランザクション型システムが必要な場合は、Amazon RDS 上に構築された従来型のリレーショナルデータベースシステムを使用するか、DynamoDB などの NoSQL データベースサービスを使用することができます。

- **非構造化データ** – Amazon Redshift のデータは、各行の任意のスキーマ構造をサポートするのではなく、定義済みのスキーマで構造化する必要があります。構造化されていないデータは、Amazon EMR で抽出、変換、およびロード (ETL) を実行することで、Amazon Redshift 内にロードできるようになります。
- **BLOB データ** – デジタルビデオ、イメージ、音楽など、大きなバイナリファイルを保存する場合は、データを Amazon S3 に保存して Amazon Redshift で保存先を参照することをお勧めします。このシナリオでは、バイナリオブジェクトに関するメタデータ (項目の名前、サイズ、作成日、所有者、場所など) は Amazon Redshift に記録されますが、ラージオブジェクト自体は Amazon S3 に保存されます。

## Amazon Elasticsearch Service

Amazon ES は、AWS クラウドで Elasticsearch を簡単にデプロイ、運用、スケールするためのマネージド型サービスです。[48] Elasticsearch は分散型のリアルタイム検索および分析エンジンです。これまでは実現できなかった速さと範囲でデータを探索できます。全文検索、構造化検索、分析、およびこれら 3 つすべての組み合わせが使用できます。

Amazon ES クラスターは、コンソールを使用して数分で設定できます。Amazon ES は、リクエストしたインフラストラクチャキャパシティーのプロビジョニングから Elasticsearch ソフトウェアの管理まで、ドメインのセットアップに伴う作業を管理します。

ドメインを実行すると、バックアップの実行、インスタンスのモニタリング、Amazon ES インスタンスを稼働するソフトウェアのパッチ適用など、一般的な管理作業が Amazon ES で自動化されます。Amazon ES では、障害が発生した Elasticsearch ノードが自動的に検出されて置き換えられるため、セルフマネージド型インフラストラクチャと Elasticsearch ソフトウェアに伴うオーバーヘッドが減少します。また、API を 1 回 呼び出すか、コンソールで数回クリックするだけで、クラスターを簡単にスケールできます。

Amazon ES では、Elasticsearch のオープンソース API に直接アクセスできるため、既存の Elasticsearch 環境で使用しているコードとアプリケーションがシームレスに連携します。Amazon ES では、ログなどのイベントデータの処理に役立つオープンソースのデータパイプラインである Logstash との統合がサポートされています。また、データの解明に役立つオープンソースの分析および可視化プラットフォームである Kibana が組み込みでサポートされています。

## 最適な使用パターン

Amazon ES は、大量のデータのクエリと検索に最適です。組織では、Amazon ES を使用して以下のことができます。

- 顧客向けのアプリケーションウェブサイトのログなど、アクティビティログの分析
- Elasticsearch による CloudWatch ログの分析
- さまざまなサービスやシステムからの製品使用状況データの分析
- ソーシャルメディアの感情や CRM データの分析と、ブランドや製品の動向の把握
- AWS の他のサービス (Amazon

- Kinesis Streams や DynamoDB など) からのデータストリーム更新の分析
- 検索とナビゲーションに関する充実した顧客エクスペリエンスの提供
- モバイルアプリケーションの使用状況のモニタリング

## 料金モデル

Amazon ES では、コンピューティングおよびストレージリソースの使用分だけ支払います。最低料金や前払いの義務はありません。Amazon ES インスタンスの時間、Amazon EBS ストレージ (このオプションを選択した場合)、および標準のデータ転送料金が請求対象になります。[49]

ストレージとして EBS ボリュームを使用する場合、Amazon ES ではボリュームタイプを選択できます。プロビジョンド IOPS (SSD) ストレージを選択すると、ストレージとプロビジョニングしたスループットが請求対象になります。[50] ただし、消費した I/O は請求対象になりません。ドメイン内のデータノードにアタッチされた EBS ボリュームの累積サイズに基づいて、追加のストレージに対してお支払いいただくオプションもあります。

Amazon ES は、Amazon ES ドメインごとに自動スナップショット用のストレージスペースを無料で提供します。手動スナップショットは、Amazon S3 のストレージ料金に従って請求されます。詳細については、Amazon Elasticsearch Service 料金表を参照してください。[51]

## パフォーマンス

Amazon ES のパフォーマンスは、インスタンスタイプ、ワークロード、インデックス、シャードの使用数、リードレプリカ、ストレージ設定 (インスタンスストレージまたは汎用 SSD などの EBS ストレージ) などの複数の要因に依存します。インデックスは、複数のアベイラビリティーゾーンの異なるインスタンスに分散可能なデータのシャードで構成されます。

シャードのリードレプリカは、ゾーン対応を有効にすると、Amazon ES によって別のアベイラビリティーゾーンで管理されます。Amazon ES では、インデックスの保存に高速の SSD インスタンスストレージを使用するか、複数の EBS ボリュームを使用することができます。検索エンジンはストレージデバイスを多用するため、ディスクを高速化すると、クエリと検索のパフォーマンスが高速化されます。

## 耐久性と可用性

ドメインの作成時にゾーン対応オプションを有効にするか、ライブドメインを変更することで、高可用性を実現するように Amazon ES ドメインを設定できます。ゾーン対応を有効にすると、Amazon ES はドメインをサポートするインスタンスを 2 つの異なるアベイラビリティーゾーンに分散します。次に、Elasticsearch でレプリカを有効にすると、インスタンスは、クロスゾーンのレプリケーションを実現するように自動的に分散されます。

Amazon ES ドメインのデータの耐久性は、自動スナップショットおよび手動スナップショットを通じて構築できます。スナップショットを使用して、データがプリロードされたドメインを復元するか、データがプリロードされた新しいドメインを作成できます。スナップショットは、耐久性と拡張性の高いセキュアなオブジェクトストレージである Amazon S3 に保存されます。デフォルト

では、Amazon ES は各ドメインのスナップショットを毎日自動的に作成します。さらに、Amazon ES のスナップショット API を使用して追加の手動スナップショットを作成することもできます。手動スナップショットは Amazon S3 に保存されます。手動スナップショットは、リージョン間の災害復旧と耐久性の強化に使用できます。

## スケーラビリティと伸縮性

インスタンスは追加または削除できます。また、データの増大に応じて Amazon EBS ボリュームを容易に変更できます。ドメインの状態を CloudWatch メトリクスでモニタリングするコードを数行記述し、Amazon ES API を呼び出して、設定したしきい値に基づいてドメインをスケールアップ/ダウンできます。サービスでは、ダウンタイムなしでスケーリングが実行されます。

Amazon ES は、クラスターに関連付けられたインスタンスごとに 1 つの EBS ボリューム (最大サイズは 512 GB) をサポートしています。Amazon ES クラスターごとに最大 10 個のインスタンスが許可されるため、約 5 TB のストレージを 1 つの Amazon ES ドメインに割り当てることができます。

## インターフェイス

Amazon ES では、Elasticsearch API がサポートされているため、既存の Elasticsearch 環境で使用しているコード、アプリケーション、および一般的なツールがシームレスに連携します。[52] AWS SDK は、すべての Amazon ES API オペレーションをサポートしているため、ユーザーは希望するテクノロジーを使用してドメインを簡単に管理および操作できます。ドメインの作成と管理には、AWS CLI またはコンソールを使用することもできます。

Amazon ES では、Amazon S3、Amazon Kinesis Streams、DynamoDB Streams からのデータストリーミングなど、AWS のいくつかのサービスとの統合がサポートされています。統合では、新しいデータに応答するイベントハンドラーとして Lambda 関数をクラウドで使用し、データを処理して Amazon ES ドメインにストリーミングします。Amazon ES では、Amazon ES ドメインのメトリクスをモニタリングするために CloudWatch との統合と、Amazon ES ドメインへの設定 API 呼び出しを監査するための CloudTrail との統合もサポートされています。

Amazon ES には、オープンソースの分析および視覚化プラットフォームである Kibana との統合が組み込まれており、ログなどのイベントデータの処理に役立つオープンソースのデータパイプラインである Logstash との統合もサポートされています。Logstash の実装に伴うすべてのログのバックエンドストアとして Amazon ES ドメインを設定し、さまざまなソースからの構造化/非構造化データを簡単に取り込むことができます。

## アンチパターン

Amazon ES には以下のアンチパターンがあります。

- **オンライントランザクション処理 (OLTP)** – Amazon ES は、分散型のリアルタイム検索および分析エンジンです。データ操作に関するトランザクションや処理はサポートされていません。高速なトランザクション型システムが必要な場合は、Amazon RDS 上に構築された従来型のリレーショナルデータベースシステムや、DynamoDB などの NoSQL データベースサービスがより適しています。

- **ペタバイトストレージ** – Amazon ES クラスターごとに最大 10 個のインスタンスが許可されるため、約 5 TB のストレージを 1 つの Amazon ES ドメインに割り当てることができます。それを超えるワークロードに対しては、Amazon EC2 でセルフマネージド型の Elasticsearch を使用することをお勧めします。

## Amazon QuickSight

2015 年 10 月、AWS は Amazon QuickSight のプレビューを導入しました。QuickSight は、クラウド駆動の高速なビジネスインテリジェンス (BI) サービスです。視覚化を構築して、アドホック分析を実行し、データからビジネス上の洞察を得ることがすばやく簡単にできます。

QuickSight は、新しい超高速、並列のインメモリ計算エンジン (SPICE) を使用して高度な計算を実行し、迅速に可視化します。QuickSight は、AWS データソースと自動的に統合されます。組織は数十万人までのユーザーにスケールでき、SPICE のクエリエンジンを介して迅速かつ応答性の高いクエリパフォーマンスを提供できます。QuickSight では、従来のソリューションと比べて 10 分の 1 の費用で、組織内の誰もが BI 機能を安価に利用できます。プレビューの詳細とサインアップ方法については、「Amazon QuickSight」を参照してください。[53]

## Amazon EC2

Amazon EC2 では、インスタンスが AWS 仮想マシンとして機能するため、AWS インフラストラクチャでセルフマネージド型のビッグデータ分析アプリケーションを運用する最適なプラットフォームとなります。[54] Linux または Windows 仮想化環境にインストールできるほぼすべてのソフトウェアは、Amazon EC2 で実行可能であり、従量制料金モデルを利用できます。実行できないのは、このホワイトペーパーで説明している、他のサービスに付随するア

プリケーションレベルのマネージド型サービスに限られます。セルフマネージド型のビッグデータ分析のオプションは数多くあります。いくつかの例を以下に示します。

- NoSQL サービス (MongoDB など)
- データウェアハウスまたは列指向ストア (Vertica など)
- Hadoop クラスター
- Apache Storm クラスター
- Apache Kafka 環境

## 最適な使用パターン

- **専用の環境** – カスタムアプリケーション、標準 Hadoop セットのバリエーション、または他のサービスのいずれでもカバーされないアプリケーションを実行する場合、Amazon EC2 はコンピューティングニーズに応じた柔軟性とスケーラビリティを提供します。
- **コンプライアンス要件** – コンプライアンス要件によっては、マネージド型サービスの代わりに自分でアプリケーションを EC2 で実行する必要があります。

## 料金モデル

Amazon EC2 には、複数のファミリー別 (スタンダード、ハイ CPU、ハイメモリ、ハイ I/O など) にさまざまなインスタンスタイプがあり、複数の料金オプション (オンデマンド、リザーブド、スポット) があります。アプリケーションの要件に応じて、Amazon EC2 と別のサービスを併用できます。たとえば、Amazon Elastic Block Store (Amazon EBS) を直接アタッチされた永続的なストレージとして併用したり、Amazon S3 を堅牢なオブジェクトストアとして併用

したりできます。併用するサービスごとに別の料金モデルが適用されます。ビッグデータアプリケーションを Amazon EC2 で実行すると、独自のデータセンターの場合と同様に、ライセンス料を負担することになります。AWS Marketplace は、さまざまなサードパーティー製のビッグデータソフトウェアパッケージを提供しており、ボタンを 1 回クリックするだけで起動するように事前設定されています。[55]

## パフォーマンス

Amazon EC2 のパフォーマンスは、ビッグデータプラットフォームとして選択したインスタンスタイプによって実現されます。インスタンスタイプごとに CPU、RAM、ストレージ、IOP の各サイズ、およびネットワーキング機能は異なります。アプリケーションの要件に応じて適切なパフォーマンスレベルを選択できます。

## 耐久性と可用性

重要なアプリケーションは、同じ AWS リージョン内の複数のアベイラビリティーゾーンをまたぐクラスター内で実行し、どのインスタンスやデータセンターで障害が発生してもアプリケーションのユーザーに影響が及ばないようにします。重要なアプリケーションは稼働中でないときに Amazon S3 にバックアップすることで、インスタンスまたはゾーンで障害が発生した場合に、リージョンの任意のアベイラビリティーゾーンに復元できます。実行しているアプリケーションや要件に応じて、他のオプション (アプリケーションのミラーリングなど) も利用できます。

## スケーラビリティと伸縮性

Auto Scaling は、定義した条件に応じて Amazon EC2 のキャパシティーを自動的にスケールアップ/ダウンできるサービスです。[56] Auto Scaling では、使用中の Amazon EC2 インスタンスの数を、需要が急上昇したときはシームレスにスケールアップしてパフォーマンスを維持し、需要が低下したときは自動的にスケールダウンしてコストを最小化できます。Auto Scaling は、特に使用量が時間、日、週ごとに変動するアプリケーションに最適です。Auto Scaling は CloudWatch で有効にします。CloudWatch の料金を超える追加料金は発生しません。

## インターフェイス

Amazon EC2 とは、API、SDK、またはコンソールを介してインターフェイスできます。コンピューティングの使用率、メモリの使用率、ストレージの使用率、ネットワーク消費量、およびインスタンスに対する読み取り/書き込みトラフィックのメトリクスは、コンソールまたは CloudWatch API オペレーションを介して無料で使用できます。

Amazon EC2 上で実行するビッグデータ分析ソフトウェアのインターフェイスは、選択したソフトウェアの特徴に基づいて異なります。

## アンチパターン

Amazon EC2 には以下のアンチパターンがあります。

- **マネージド型サービス** – マネージド型サービスを使用してビッグデータ分析からインフラストラクチャレイヤーや管理を抽出する必要がある場合、このモデルは「Do It Yourself」として Amazon EC2 上で独自の分析ソフトウェアを管理するものであるため、適切な選択肢とは言えません。

- **専門知識やリソースの不足** – 組織のシステムに高可用サービスをインストールして管理するためのリソースや専門知識が不足しているか、そのようなリソースや専門組織に投資することを望まない場合は、AWS の同等サービスである Amazon EMR、DynamoDB、Amazon Kinesis Streams、Amazon Redshift などを使用することを検討してください。

# AWS でのビッグデータ問題の解決

本書では、ビッグデータ分析に役立つ AWS のツールを取り上げています。この情報を参考にしてビッグデータアプリケーションの設計を開始できます。ただし、特定のユースケースに適したツールを選ぶには、ほかにも検討すべき点があります。通常、分析ワークロードごとに以下の特徴や要件は異なるため、それらに合わせたツールを使用する必要があります。

- 分析結果を取得するまでの所用時間 (リアルタイム、秒単位、または時間単位)
- これらの分析が組織にもたらす価値と予算上の制約
- データのサイズと増大率
- データの構造
- プロデューサーとコンシューマーが利用できる統合機能
- プロデューサーとコンシューマーの間で許容されるレイテンシー
- ダウンタイムに伴う損失、またはソリューションに要求される可用性と耐久性
- 分析ワークロードが一定しているか伸縮するか

以上の特徴や要件に沿って適切なツールを判断できます。常に同じ要件に基づいて特定のサービスを割り当てることができるビッグデータ分析ワークロードもありますが、ほとんどのビッグデータ分析ワークロードは、実際の状況に応じて、同じデータセットでも特徴や要件が異なる場合や相互に矛盾する場合さえもあります。

たとえば、必要な分析結果を得るために、一部の分析は毎日のバッチ処理で実行できても、一部の分析はユーザーがシステムとやり取りする際のリアルタイムの要件に基づく場合があります。このように同じデータセットに対する異なる要件は、分離して複数のツールで解決する必要があります。上記の例の両方を同じツールセットで解決しようとすると、オーバープロビジョニングして不要な応答時間に余分に支払うようになるか、ソリューションが遅くなってユーザーにリアルタイムで応答できなくなります。コンピューティングおよびストレージのリソースを最もコスト効率よく利用するには、分析問題のタイプ別に最適なツールを使い分けることです。

ビッグデータが「ビッグコスト」であるとは限りません。したがって、アプリケーションの設計では、コスト効率を最も重視します。他の案と比べてコスト効率が劣るような設計は最適なものとは言えません。別の一般的な誤解として、ビッグデータ問題の解決に複数のツールセットを使うのは、1 つの大規模なツールを使用するより、高価で管理しにくいというものがあります。同じ例 (同じデータセットに対する 2 つの異なる要件) で、リアルタイムリクエストは CPU に対しては低いが I/O に対しては高く、リアルタイムより遅い処理のリクエストではコンピューティングを集中的に行うという場合があります。分離することは、各ツールを正確な仕様で構築し、オーバープロビジョニングを避けることができるため、費用が大幅に低減して管理も容易になります。サービスモデルとしてのインフラストラクチャの使用分だけに支払うという AWS の従

量課金制では、バッチ分析を 1 時間実行したら、その 1 時間のコンピューティングリソースに対してのみ支払うため、価値が大幅に高まります。また、このアプローチのほうが、1 つのシステムですべての要件を満たすよりも管理しやすくなります。1 つのツールでさまざまな要件に対処することは、合わないもの (リアルタイムリクエストと大規模なデータウェアハウス) を無理に一緒にすることになります。

AWS プラットフォームでは、同じデータセットを複数の異なるツールで分析することで、アーキテクチャを簡単に分離できます。AWS のサービスには統合が組み込まれているため、並列化を使用して、データのサブセットをツール間で簡単にすばやく移動できます。ビッグデータ分析問題の実例をいくつか使って、AWS のアーキテクチャソリューションの使い方をステップごとに説明しましょう。

## 例 1: エンタープライズデータウェアハウス

ある多国籍衣料品メーカーには 1,000 以上の直営店があります。特定の衣料品についてはデパートやディスカウントストアでも販売し、さらにオンラインストアも運営しています。

現在、これら 3 つのチャネルは技術的な判断で独立して運用されています。管理者、POS システム、経理部がそれぞれ異なります。これらのデータセットのすべてを統合して CEO がビジネス全体への洞察を得られるようなシステムはありません。CEO は、会社のすべてのチャネルを統合した全体像と、必要に応じて実行できるアドホック分析を求めています。この会社に必要な分析の例:

- すべてのチャネルに共通するトレンド
- すべてのチャネルで業績のよい地理的リージョン

- 広告やクーポンの効果
- すべてのチャネルでの衣料品別のトレンド
- 販売に影響する可能性のある外部要因 (失業率や天候など)
- 店舗の属性が売上に与える影響 (従業員と管理者の在職期間、ショッピングモールの天井の有無、店舗内の商品の配置、プロモーション、特設コーナー、チラシ、店舗内ディスプレイなど)

エンタープライズデータウェアハウスは、この問題を解決する優れた方法です。データウェアハウスは、3 つの各チャネルのさまざまなシステムおよび一般的な天気や経済情報の記録からデータを収集します。データは、各データソースから毎日送信され、データウェアハウスで処理されます。各データソースの構造は異なる場合があるため、抽出、変換、およびロード (ETL) プロセスを実行してデータを一般的な構造にフォーマットし直します。これで、すべてのソースのデータに対して分析を同時に実行できます。そのためには、次の図に示すデータフローアーキテクチャを使用します。


エンタープライズデータウェアハウス

1. このプロセスの最初のステップでは、さまざまなソースから Amazon S3 にデータを取得します。Amazon S3 を使用するのは、耐久性が高く、コストが低いスケーラブルなストレージプラットフォームであり、さまざまなソースから非常に低コストで同時に書き込みを行うことができるためです。

2. Amazon EMR は、データをソースフォーマットからターゲットフォーマットに変換して最適化するために使用します。Amazon EMR には Amazon S3 との統合が組み込まれており、Amazon S3 との間で、クラスターを構成する各ノードのスループットの並列スレッドが可能になります。通常、データウェアハウスはさまざまなソースから毎晩新しいデータを取得します。これらの分析は深夜に行う必要がないため、この変換プロセスの唯一の要件は、CEO や他のビジネスユーザーが結果を必要とする朝までに完了することです。したがって、ここで Amazon EC2 スポット市場を使用して変換のコストをさらに下げることができます。[57] スポット戦略として、深夜に非常に低い価格で入札を始め、キャパシティーが得られるまで時間をかけて価格を徐々に上げていくことをお勧めします。スポット入札に成功しないまま期限間近になった場合は、オンデマンド価格に戻して完了期限の要件を満たすことができます。Amazon EMR ではソースごとに変換プロセスが異なる場合がありますが、AWS の従量料金制モデルに従って、可能な限りの最低価格ですべてのデータ変換ジョブを完了できるように、変換別に Amazon EMR クラスターを作成して最適な容量に調整できます。その際に、他のジョブのリソースに影響を与えることはありません。

3. 各変換ジョブにより、フォーマットおよび最適化されたデータが Amazon S3 に入力されます。ここで再度 Amazon S3 を使用するのは、Amazon Redshift では、このデータを各ノードから複数のスレッドで並行処理できるためです。ここで Amazon S3 を使うことは、履歴レコードとしても役立ち、システム間でフォーマットされた正しい情報源となります。時間が経過して追加の要件が生じた場合は、Amazon S3 のデータの分析に他のツールを使用できます。

4. Amazon Redshift は、データをテーブル内にロード、ソート、分散、および圧縮して、分析クエリを効率的に並行して実行できるようにします。Amazon Redshift はデータウェアハウスのワークロード用に構築されており、時間の経過とともにデータサイズやビジネスが拡大した場合は、別のノードを追加して簡単に拡張できます。

5. 分析を可視化するには、Amazon QuickSight を使用するか、Amazon Redshift への ODBC/JDBC 接続を介して、さまざまなパートナー可視化プラットフォームのいずれかを使用できます。この段階で、CEO およびそのスタッフがレポートやグラフを参照できるようになります。エグゼクティブが、このデータに基づいて会社のリソースに関する的確な判断を行うことで、最終的に収益の向上と株主への配当増がもたらされます。

このアーキテクチャは柔軟性が高く、ビジネスの拡大、新しいデータソースの導入、新しいチャネルの開設、または顧客専用のデータを反映したモバイルアプリケーションのリリースに伴って簡単に拡張できます。いつでも、Amazon Redshift クラスターのノード数を増やすことで、数回のクリックで追加のツールを統合したり、ウェアハウスをスケールしたりできます。

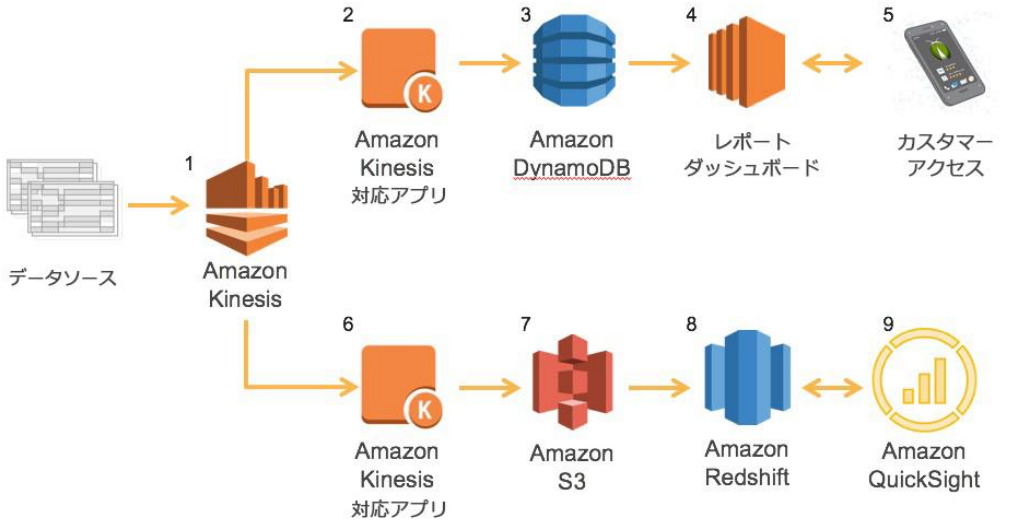
## 例 2: センサーデータのキャプチャと分析

ある国際的なエアコンメーカーは、さまざまな商業企業および産業企業に販売する大量のエアコンを保持しています。競合他社と差別化するために、エアコンユニットを販売するだけでなく、モバイルアプリケーションやウェブブラウザでリアルタイムのダッシュボードを表示できる追加サービスも提供しています。各ユニットからは、処理および分析のためのセンサー情報が送信されます。このデータは、メーカーおよび顧客が利用します。この機能により、メーカーはデータセットを視覚化し、トレンドを把握できます。

現在、メーカーには、この機能を搭載した数千の未販売のユニットがあります。これらのユニットは数か月以内に顧客に販売する予定であり、まもなく、このプラットフォームは何千というユニットで使用されるものと期待しています。成功すれば、このサービスを一般消費者にも広げ、大幅に供給量を増やして、マーケットシャアを拡大したいと考えています。ソリューションは、大量のデータを処理し、ビジネスの拡大に伴って中断なしにスケールできる必要があります。どうすれば、このようなシステムを設計できるでしょうか。まず、同じデータに基づく 2 つのワークストリームに分割します。

- 準リアルタイムの要件を反映したエアコンユニットの最新情報と、この情報を利用する大量の顧客。
- 内部利用向けのトレンドと分析を実行するための、エアコンユニットに関するすべての履歴情報。

このビッグデータ問題の解決方法を示すデータフローアーキテクチャは次のとおりです。




**センサーデータのキャプチャと分析**

1. このプロセスでは、まず各エアコンユニットから一定のデータストリームが Amazon Kinesis Streams に提供されます。これにより、ユニットが通信できる伸縮自在の耐久性が高いインターフェイスが提供され、このインターフェイスはエアコンユニットの販売数が増えてオンライン化されるに伴ってシームレスにスケールされます。

6. Amazon Kinesis Streams が提供する Kinesis Client Library や SDK などのツールを使用すると、Amazon EC2 上にシンプルなアプリケーションを構築して、Amazon Kinesis Streams に着信するデータを読み取って分析し、リアルタイムダッシュボードの更新に値するデータであるかどうかを判断できます。このアプリケーションで、システムオペレーションの変化、温度の変動、およびユニットで発生するエラーを確認します。

7. このデータフローは、ユニットの異変をできるだけ早く顧客やメンテナンスチームに知らせるために、準リアルタイムで実行する必要があります。ダッシュボードのデータには一部のトレンド情報が集約されますが、主に現在の状態やシステムエラーに関するものです。そのため、

ダッシュボードへの入力を要するデータは比較的少量です。さらに、このデータに対しては以下のソースからの大量のアクセスが予想されます。

- モバイルデバイスやブラウザを通じてシステムを確認する顧客
- 全ユニットのステータスを確認するメンテナンスチーム
- レポートプラットフォームのデータとインテリジェンスのアルゴリズムと分析 (建物の温度が下がらないままエアコンのファンが異常に長く稼働している場合など、このプラットフォームで把握されたトレンドはアラートとして送信されます)

ここで DynamoDB を使用するのは、可用性が高くスケーラブルであり、準リアルタイムでデータセットを保存できることによります。このデータのスループットは、プラットフォームが導入されて普及するに従って、簡単にスケールアップ/ダウンしてコンシューマーのニーズを満たすことができます。

8. レポートダッシュボードは、このデータセットに基づいて構築され、Amazon EC2 上で実行されるカスタムウェブアプリケーションです。システムのステータスとトレンドに基づいてコンテンツを提供し、ユニットで異変が発生した場合に顧客とメンテナンスチームに知らせます。
9. 顧客は、モバイルデバイスやウェブブラウザからデータにアクセスし、システムの最新状態を確認して、履歴トレンドを視覚化します。

前述のデータフロー (ステップ 2～5) は、準リアルタイムで情報をコンシューマー (人) に報告するために構築されています。低レイテンシーを実現する構築および設計になっており、非常に迅速にスケールして需要を満たすことができます。図の下半分に示したデータフロー (ステップ 6～9) については、スピー

ドとレイテンシーの要件がそれほど厳格ではありません。これにより、アーキテクトはより大量のデータをより小さいコスト (情報のバイトあたり) で保持する別のソリューションスタックを設計し、より安価なコンピューティングおよびストレージリソースを選択できます。

10. Amazon Kinesis ストリームから読み取るには、別の Amazon Kinesis 対応のアプリケーションがあり、これをより小さい、スケール速度の遅い EC2 インスタンスで実行します。このアプリケーションは上側のデータフローと同じデータセットを分析しますが、このデータの最終的な目的は、記録として長期保存し、データウェアハウスのデータセットをホストすることにあります。このデータセットでは、すべてのデータがシステムから送信され、準リアルタイムの要件を伴わない、より広範囲にわたる分析が実行されます。
11. Amazon Kinesis 対応のアプリケーションによって、データは長期保存に適したフォーマットに変換され、データウェアハウスにロードされて Amazon S3 に保存されます。Amazon S3 は、Amazon Redshift にデータを並行して取り込むポイントとして機能するだけでなく、このシステムを通過するすべてのデータを保持する耐久性の高いストレージです。また、唯一の正しい情報源となります。追加の要件が生じた場合は、これを使用して他の分析ツールをロードできます。データを長期のコストが低いコールドストレージへと循環させる必要がある場合のために、Amazon S3 には Amazon Glacier との統合も組み込まれています。
12. より大きなデータセットには、Amazon Redshift をデータウェアハウスとして再び使用します。データセットが増大した場合は、クラスターに別のノードを追加することで、簡単にスケールできます。

13. 分析を可視化するには、Amazon Redshift への ODBC/JDBC 接続を介して、さまざまなパートナー可視化プラットフォームのいずれかを使用できます。この段階で、データセットに対してレポート、グラフ、およびアドホック分析を実行して、エアコンユニットのパフォーマンス低下や故障につながるような特定の変数やトレンドを確認できます。

このアーキテクチャは、小規模で開始し、必要に応じて拡大できます。また、2 つの異なるワークストリームに分離することで、それぞれが前払いなしで必要に応じて独自のレートで拡大できるため、メーカーは大きな投資を行うことなくこの新しいサービスの成否を評価できます。この先の展開としては、Amazon ML などを追加して、エアコンユニットの耐用期間を正確に予測し、予測アルゴリズムに基づいてメンテナンスチームを前もって派遣することで、顧客にできるだけ最善のサービスとエクスペリエンスを提供できます。このようなサービスは、競合他社に対する差別化となり、将来の売上増につながります。

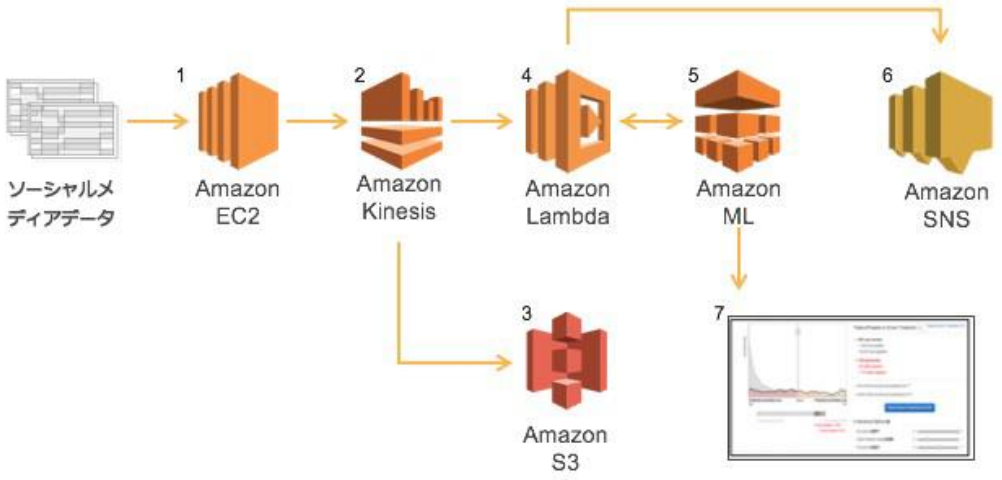
## 例 3: ソーシャルメディアの感情分析

あるオモチャメーカーが急成長中であり、製品ラインを拡大しています。新しいオモチャを発表するたびに、メーカーは消費者がどれぐらい歓迎し利用しているかを知りたいと思っています。また、自社製品に対する顧客満足度も確認したいと思っています。オモチャのエコシステムが拡大するに従って、メーカーの製品が依然として顧客にアピールするものであることを確認し、顧客からのフィードバックに基づいて将来のロードマップ商品を確実に企画できることを求めています。メーカーは、ソーシャルメディアから以下のための情報を得たいと考えています。

- 消費者がメーカーの製品をどのように利用しているか
- 顧客が満足していることの確認

- 将来のロードマップの計画

さまざまなソーシャルネットワークから情報を取り込むことは比較的簡単ですが、問題はインテリジェンスをプログラムで構築することです。メーカーは、取り込んだデータをコスト効率の高いプログラム的アプローチで分析して分類できることを望んでいます。そのためには、次の図に示すアーキテクチャを使用できます。




**ソーシャルメディアの感情分析**

最初に行うことは、対象とするソーシャルメディアを決めることです。次に、これらのサイトを対応する API でポーリングし、Amazon EC2 で実行するアプリケーションを作成します。

次に、Amazon Kinesis ストリームを作成します。これは、データソースが Twitter、Tumblr など複数になる可能性があるためです。これにより、新しく追加されるデータソースごとに新しいストリームを作成でき、既存のアプリケーションコードとアーキテクチャを使用できます。さらに、この例では、別の Amazon Kinesis ストリームを作成して raw データを Amazon S3 にコピーします。

アーカイブ、長期分析、および履歴参照の目的で、raw データは Amazon S3 に保存します。Amazon S3 内のデータから追加の Amazon ML バッチモデルを実行して予測分析を行い、消費者の購買トレンドを追跡できます。

アーキテクチャ図で示しているように、Lambda を使用してデータの処理と正規化を行い、Amazon ML に予測をリクエストします。Amazon ML から予測が返されたら、その予測に基づいて Lambda 関数でアクションを実行できます。たとえば、ソーシャルメディアの投稿をカスタマーサービスチームにルーティングして詳細に検討できるようにします。

Amazon ML で入力データに基づく予測を行います。たとえば、ML モデルを構築してソーシャルメディアのコメントを分析し、顧客が製品にネガティブな感情を持っているかどうかを判断できます。Amazon ML で正確な予測を得るには、まずトレーニングデータを試し、ML モデルが適切に動作することを確認します。初めて ML モデルを作成する場合は、「チュートリアル: Using Amazon ML to Predict Responses to a Marketing Offer」を参照してください。[58] 前述したように、複数のソーシャルネットワークを使用する場合は、予測の精度を高めるために、それぞれに別の ML モデルを作成することをお勧めします。

最後に Lambda から Amazon SNS に実施可能なデータが送信され、詳しい調査のためにテキストまたは E メールで適切なリソースに配信されます。

感情分析の一環として、正確な結果を得るには、Amazon ML モデルを作成して定期的に更新する必要があります。特定のモデルに関する追加のメトリクスとして、正確性、誤検出レート、精度、リコールなどをグラフで表示できます。詳細については、「Step 4: Review the ML Model Predictive Performance and Set a Cut-Off」を参照してください。[59]

Amazon Kinesis Streams、Lambda、Amazon ML、および Amazon SES を組み合わせて使用することで、スケーラブルで簡単にカスタマイズできるソーシャルリスニングプラットフォームを作成しました。この図は ML モデルを作成するアクションを示すものではないことに注意してください。このアクションは最低 1 回実行します。ただし、通常はモデルを最新状態に保つために定期的に実行します。新しいモデルを作成する頻度はワークロードによって異なります。また、変更があったときにモデルをより正確なものにする場合に限り、作成します。

# まとめ

データの生成量や収集量が増えるに従って、分析から洞察をタイムリーに得るには、スケーラビリティ、柔軟性、およびパフォーマンスに優れたツールが必要です。ただし、ビッグデータのエコシステムでは新しいツールが目まぐるしく入れ替わります。したがって、臨機応変に適切なツールを選択することは非常に難しくなります。

本書では、このような課題に対処するための最初のステップを提供しています。AWS プラットフォームには、ビッグデータを収集、処理、分析するためのマネージド型サービスが豊富に用意されています。これらのサービスを利用すると、ビッグデータアプリケーションの構築、デプロイ、スケールが容易になり、ツールの更新や管理に煩わされることなくビジネスの問題に専念できます。

AWS には、ビッグデータの分析要件に対応できるソリューションが数多く用意されています。ビッグデータアーキテクチャでは、通常、複数の AWS ツールを使用して完全なソリューションを構築します。これにより、パフォーマン

ス、耐障害性、コスト効率を可能な限り最大化して、厳しいビジネス要件を満たすことができます。その結果、AWS のグローバルなインフラストラクチャで、ビジネスの成長に合わせて柔軟にスケールできるビッグデータアーキテクチャが実現します。

# 寄稿者

本書の執筆に当たり、次の人物および組織が寄稿しました。

- Erik Swensson、ソリューションアーキテクチャマネージャー、アマゾン ウェブ サービス
- Erick Dame、ソリューションアーキテクト、アマゾン ウェブ サービス
- Shree Kenghe、ソリューションアーキテクト、アマゾン ウェブ サービス

# その他の資料

AWS でビッグデータ分析の実行を開始には、以下の資料が参考になります。

- Big Data on AWS[60]

  ビッグデータサービスの包括的なポートフォリオと、ビッグデータソリューションに関する AWS ビッグデータパートナー、チュートリアル、記事、AW Marketplace サービスなどの他のリソースへのリンクがあります。サポートが必要な場合は、こちらまでお問い合わせください。[61]

- AWS Big Data Blog をお読みください[62]

  このブログでは、ビッグデータの収集、保存、最適化、処理、および視覚化に役立つ最新の実例やアイデアが定期的に紹介されます。

- AWS ビッグデータ Test Drive ラボをお試しください[63]

  AWS を使用してビッグデータの課題に対処するために設計された製品の豊富なエコシステムをお試しください。Test Drive は AWS パートナーネットワーク (APN) のコンサルティングおよびテクノロジーパートナーによって開発されたものであり、教育、デモ、評価の目的で無償で提供されています。

- ビッグデータに関する AWS トレーニングコースを受講してください[64]

  Big Data on AWS コースでは、クラウドベースのビッグデータソリューションと Amazon EMR を紹介します。Pig や Hive といった Hadoop ツールの広範なエコシステムを使用して、Amazon EMR でデータを処理する方法について説明します。また、ビッグデータ環境の作成方法、Amazon DynamoDB と Amazon Redshift の連携方法、Amazon Kinesis Streams の利点、ベストプラクティスを活用してセキュリティとコスト効果に優れたデータ環境を設計する方法についても学習します。

- ビッグデータ活用における AWS 国内導入事例をご覧ください[65]

  AWS クラウドで強力なビッグデータプラットフォームの構築に成功している他のお客様の事例を参考にしてください。

# ドキュメントの改訂

| 改訂日 | 説明 |
| --- | --- |
| **2016 年 1 月** | Amazon Machine Learning、AWS Lambda、Amazon Elasticsearch Service に関する情報を追加し、全般的な更新を行いました |
| **2014 年 12 月** | 初版発行 |

# Notes

[1] http://aws.amazon.com/about-aws/globalinfrastructure/

[2] https://aws.amazon.com/s3/

http://aws.amazon.com/datapipeline/

[3] https://aws.amazon.com/iot/

[4] https://aws.amazon.com/importexport/

http://aws.amazon.com/kinesis/firehose

https://aws.amazon.com/directconnect/

[5] https://aws.amazon.com/iot/

[6] http://aws.amazon.com/solutions/case-studies/big-data/

[7] https://aws.amazon.com/kinesis/streams

[8] http://docs.aws.amazon.com/kinesis/latest/APIReference/Welcome.html

http://docs.aws.amazon.com/aws-sdk-php/v2/guide/service-kinesis.html

[9] http://aws.amazon.com/kinesis/pricing/

[10] http://aws.amazon.com/tools/

http://docs.aws.amazon.com/kinesis/latest/dev/developing-producers-with-kpl.html

http://docs.aws.amazon.com/kinesis/latest/dev/writing-with-agents.html

[11] https://github.com/awslabs/amazon-kinesis-client

[12] https://github.com/awslabs/kinesis-storm-spout

[13] https://aws.amazon.com/lambda/

[14] https://aws.amazon.com/lambda/pricing

[15] http://docs.aws.amazon.com/lambda/latest/dg/intro-core-components.html

[16] https://aws.amazon.com/amazon-linux-ami/

[17] http://docs.aws.amazon.com/lambda/latest/dg/nodejs-create-deployment-pkg.html

http://docs.aws.amazon.com/lambda/latest/dg/lambda-python-how-to-create-deployment-package.html

http://docs.aws.amazon.com/lambda/latest/dg/lambda-java-how-to-create-deployment-package.html

[18] http://aws.amazon.com/elasticmapreduce/

[19] https://media.amazonwebservices.com/AWS_Amazon_EMR_Best_Practices.pdf

[20] http://aws.amazon.com/elasticmapreduce/pricing/

[21] http://aws.amazon.com/ec2/instance-types/

[22] http://aws.amazon.com/elasticmapreduce/mapr/

[23] http://docs.aws.amazon.com/ElasticMapReduce/latest/DeveloperGuide/emr-manage-resize.html

[24] http://docs.aws.amazon.com/ElasticMapReduce/latest/API/Welcome.html

[25] http://docs.aws.amazon.com/ElasticMapReduce/latest/ReleaseGuide/emr-hive.html

[26] http://blogs.aws.amazon.com/bigdata/post/Tx15AY5C50K70RV/Installing-Apache-Spark-on-an-Amazon-EMR-Cluster

[27] http://docs.aws.amazon.com/ElasticMapReduce/latest/DeveloperGuide/emr-hbase.html

[28] http://docs.aws.amazon.com/ElasticMapReduce/latest/DeveloperGuide/emr-impala.html

[29] http://docs.aws.amazon.com/ElasticMapReduce/latest/DeveloperGuide/UsingEMR_s3distcp.html

[30] https://aws.amazon.com/machine-learning/

[31] https://aws.amazon.com/machine-learning/pricing/

[32] http://docs.aws.amazon.com/machine-learning/latest/dg/suggested-recipes.html

[33] http://docs.aws.amazon.com/machine-learning/latest/APIReference/Welcome.html

[34] https://aws.amazon.com/dynamodb

[35] http://aws.amazon.com/free/

[36] http://aws.amazon.com/dynamodb/pricing/

[37] 通常、サーバー側の平均応答時間は 1 桁台のミリ秒単位です。

[38] http://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Streams.html

[39] DynamoDB では、UpdateTable API オペレーションを 1 回呼び出すだけで、プロビジョニングされたスループットレベルを最大 100% まで変更できます。スループットを 100% を超えて増やすには、UpdateTable を再度呼び出します。プロビジョニングされたスループットは必要なだけ何回でも増やすことができますが、減らすのは 1 日 2 回までに制限されています。

[40] http://aws.amazon.com/redshift/

[41] http://aws.amazon.com/s3/pricing/

[42] http://aws.amazon.com/redshift/pricing/

[43] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-regions-availability-zones.html

[44] http://docs.aws.amazon.com/redshift/latest/APIReference/Welcome.html

[45] http://docs.aws.amazon.com/redshift/latest/mgmt/working-with-clusters.html#rs-about-clusters-and-nodes

[46] http://docs.aws.amazon.com/redshift/latest/dg/c_redshift-and-postgres-sql.html

[47] http://aws.amazon.com/redshift/partners/

[48] https://aws.amazon.com/elasticsearch-service/

[49] https://aws.amazon.com/ec2/pricing/

[50] https://aws.amazon.com/ebs/details/

[51] https://aws.amazon.com/elasticsearch-service/pricing/

[52] https://aws.amazon.com/elasticsearch-service/faqs/

[53] https://aws.amazon.com/quicksight

[54] https://aws.amazon.com/ec2/

[55] https://aws.amazon.com/marketplace

[56] http://aws.amazon.com/autoscaling/

[57] http://aws.amazon.com/ec2/spot/

[58] http://docs.aws.amazon.com/machine-learning/latest/dg/tutorial.html

[59] http://docs.aws.amazon.com/machine-learning/latest/dg/step-4-review-the-ml-model-predictive-performance-and-set-a-cut-off.html

[60] http://aws.amazon.com/big-data

[61] https://aws.amazon.com/marketplace

http://aws.amazon.com/big-data/contact-us/

[62] http://blogs.aws.amazon.com/bigdata/

[63] https://aws.amazon.com/testdrive/bigdata/

[64] http://aws.amazon.com/training/course-descriptions/bigdata/

[65] http://aws.amazon.com/solutions/case-studies/big-data/