ユーザーガイド

AWS IoT Analytics

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS IoT Analytics: ユーザーガイド

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon の商標およびトレードドレスはAmazon 以外の製品およびサービスに使用することはできません。また、お客様に誤解を与える可能性がある形式で、または Amazon の信用を損なう形式で使用することもできません。Amazon が所有していないその他のすべての商標は Amazon との提携、関連、支援関係の有無にかかわらず、それら該当する所有者の資産です。

Table of Contents

とは AWS IoT Analytics 1
の使用方法 AWS IoT Analytics 1
主な特徴 2
AWS IoT Analytics のコンポーネントと概念 4
アクセス AWS IoT Analytics 6
ユースケース 7
AWS IoT Analytics サポート終了 9
移行オプション 9
移行ガイド 12
ステップ 1: 進行中のデータ取り込みをリダイレクトする 13
ステップ 2: 以前に取り込んだデータをエクスポートする 15
両方のパターンに対してオンデマンドクエリを実行する 23
概要 23
使用開始 コンソール 25
AWS IoT Analytics コンソールにサインインする 26
チャネルの作成 26
データストアの作成 28
パイプラインを作成する 29
データセットの作成 31
を使用してメッセージデータを送信する AWS IoT 33
AWS IoT メッセージの進行状況を確認する 34
クエリ結果へのアクセス 35
データの探索 35
ノートブックテンプレート 38
入門 39
チャンネルの作成 39
データストアの作成 41
Amazon S3 ポリシー 41
ファイル形式 43
カスタムパーティション 46
パイプラインの作成 49
へのデータの取り込み AWS IoT Analytics 50
AWS IoT メッセージブローカーの使用 51
BatchPutMessage API の使用 55

取り込まれたデータのモニタリング 56
データセットの作成 58
データのクエリ 59
クエリされたデータへのアクセス 59
AWS IoT Analytics データの探索 35
Amazon S3 60
AWS IoT Events 61
QuickSight 61
Jupyter Notebook 62
複数のバージョンのデータセットを保持する 62
メッセージペイロード構文 63
AWS IoT SiteWise データの使用 63
データセットを作成する 64
データセットコンテンツへのアクセス 68
チュートリアル: AWS IoT SiteWise データのクエリ 69
パイプラインアクティビティ 77
チャネルアクティビティ 77
データストアアクティビティ 77
AWS Lambda アクティビティ 78
Lambda 関数の例 1 78
Lambda 関数の例 2 80
AddAttributes アクティビティ 81
RemoveAttributes アクティビティ 83
SelectAttributes アクティビティ 84
フィルターアクティビティ 85
DeviceRegistryEnrich アクティビティ 85
DeviceShadowEnrich アクティビティ 87
Math アクティビティ 89
Math アクティビティ演算子と関数 90
RunPipelineActivity 106
チャネルメッセージの再処理 108
パラメータ 108
チャネルメッセージの再処理 (コンソール) 109
チャネルメッセージの再処理 (API) 110
チャネルの再処理アクティビティのキャンセル 111
ワークフローの自動化 112

ユースケース 113
Docker コンテナの使用 114
カスタム Docker コンテナの入力/出力変数 117
アクセス許可 119
データセットを作成 (Java と AWS CLI) 121
例 1 -- SQL データセットを作成する java) 122
例 2 -- デルタウィンドウを使用して SQL データセットを作成する java) 122
例 3 -- 独自のスケジュールトリガーを使用してコンテナデータセットを作成する java) 124
例 4 -- SQL データセットをトリガーとして使用してコンテナデータセットを作成する java) 125
例 5 -- SQL データセットを作成する CLI) 126
例 6 -- デルタウィンドウを使用して SQL データセットを作成する CLI) 126
ノートブックをコンテナ化する 127
AWS IoT Analytics コンソールで作成されていないノートブックインスタンスのコンテナ化を有効にする 128
ノートブックのコンテナ化拡張機能を更新する 131
コンテナ化イメージを作成する 131
カスタムコンテナの使用 137
データの可視化 146
可視化 (コンソール) 146
可視化 (QuickSight) 147
Tagging 151
タグの基本 151
IAM ポリシーでのタグの使用 152
タグの制限 154
SQL 式 156
ポートされている SQL 機能 157
サポートされているデータ型 157
サポートされている関数 158
一般的な問題のトラブルシューティング 159
セキュリティ 160
AWS Identity and Access Management 160
対象者 160
アイデンティティを使用した認証 161
アクセスの管理 164
IAM の操作 166

サービス間の混乱した代理の防止 170
IAM ポリシーの例 176
ID とアクセスのトラブルシューティング 182
ログ記録とモニタリング 184
自動モニタリングツール 184
手動モニタリングツール 185
CloudWatch Logs によるモニタリング 185
CloudWatch Events を使用した のモニタリング 190
CloudTrail による API コールのログ記録 199
コンプライアンス検証 204
耐障害性 205
インフラストラクチャセキュリティ 205
クォータ 206
コマンド 207
AWS IoT Analytics アクション 207
AWS IoT Analytics データ 207
トラブルシューティング 208
どうすれば AWS IoT Analyticsにメッセージが取り込まれているかどうかを確認できますか? . 208
パイプラインからメッセージが欠落するのはなぜですか? 解決策は? 209
データストア内にデータがないのはなぜですか? 210
データセットに __dt が表示されるのはなぜですか? 210
データセットの完了に伴ってイベントを発生させるコードはどのように記述しますか? 211
AWS IoT Analyticsを使用するようにノートブックインスタンスを正しく設定するにはどうしますか? 211
インスタンスでノートブックを作成できないのはなぜですか? 212
QuickSight でデータセットが表示されないのはなぜですか ? 212
既存の Jupyter Notebook にコンテナ化ボタンが表示されないのはなぜですか? 213
コンテナ化プラグインのインストールが失敗する原因は何ですか? 213
なぜコンテナ化プラグインによってエラーが返されるのですか? 213
コンテナ化中に使用する変数が表示されません。 214
どのような変数を入力としてコンテナに追加できますか? 214
コンテナ出力をこの先の入力として設定するにはどうすればよいですか? 214
コンテナデータセットが失敗する原因は何ですか? 214
ドキュメント履歴 216
以前の更新 218
..... ccxix

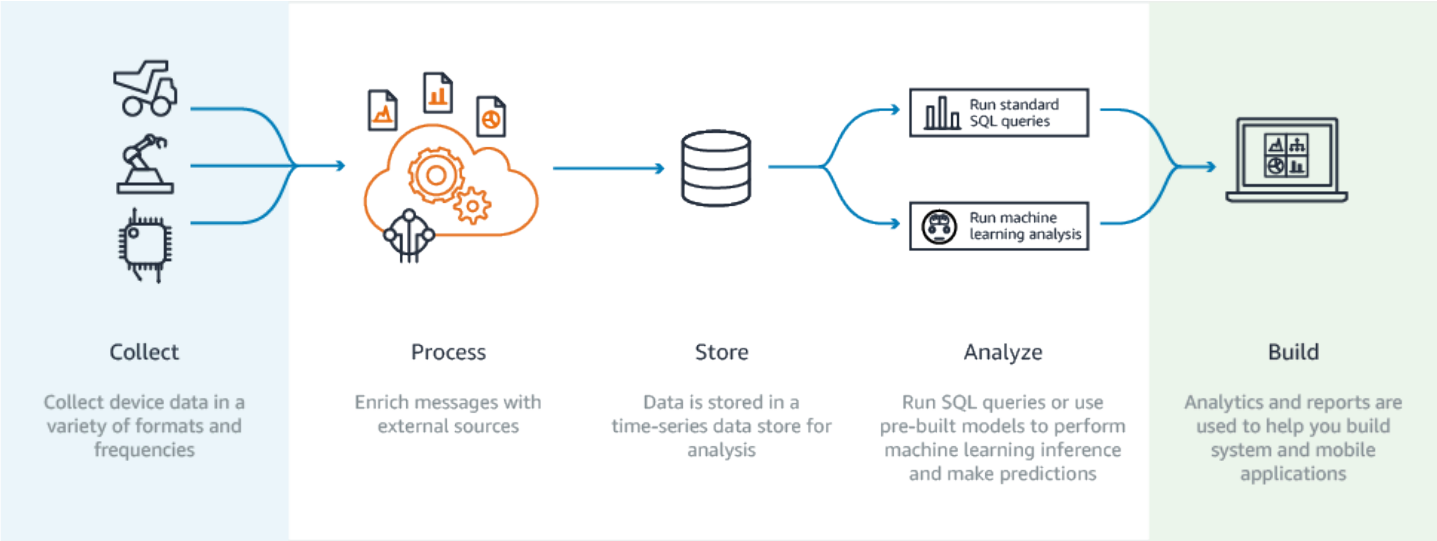
とは AWS IoT Analytics

AWS IoT Analytics は、IoT デバイスからのデータの分析に必要なステップを自動化します。 は、分析のために時系列データストアに保存する前に、IoT データを AWS IoT Analytics フィルタリング、変換、強化します。デバイスから必要なデータのみを収集して、数学的変換を適用してデータを処理し、処理されたデータを保存する前にデバイスの種類や場所などのデバイス固有のメタデータでデータを強化するサービスを設定できます。次に、組み込みの SQL クエリエンジンを使用してクエリを実行することでデータを分析したり、より複雑な分析や機械学習の推論を実行したりできます。 AWS IoT Analytics は、Jupyter Notebook との統合を通じて高度なデータ探索を可能にします。 AWS IoT Analytics また、 は QuickSight との統合を通じてデータの可視化を可能にします。QuickSight は、次のリージョンで使用できます。

従来の分析およびビジネスインテリジェンスツールは、構造化されたデータを処理するように設計されています。生の IoT データは、多くの場合、あまり構造化されていないデータ 温度、モーション、サウンドなどが記録されるデバイスから送信されます。この結果、これらのデバイスのデータには、大きな誤差やメッセージの破損、誤認識が含まれる場合があるため、分析を行う前にクリーンアップする必要があります。また、IoT データは、多くの場合、外部ソースからの他のデータのコンテキストでのみ意味があります。 AWS IoT Analytics は、これらの問題に対処し、大量のデバイスデータを収集し、メッセージを処理して保存します。その後、データをクエリして分析できます。には、一般的な IoT ユースケース向けに構築済みのモデル AWS IoT Analytics が含まれているため、どのデバイスが失敗するか、どの顧客がウェアラブルデバイスを放棄するリスクがあるかなどの質問に答えることができます。

の使用方法 AWS IoT Analytics

次の図は AWS IoT Analyticsの使用方法の概要を示しています。



主な特徴

収集

- と統合 AWS IoT Core—AWS IoT Analytics は と完全に統合 AWS IoT Core されているため、接続されたデバイスからストリーミング中にメッセージを受信できます。
- バッチ API を使用して任意のソースからデータを追加し、HTTP を介して任意のソースからデータを受信AWS IoT Analytics できます。つまり、インターネットに接続された任意のデバイスやサービスから AWS IoT Analyticsにデータを送信できます 詳細については、AWS IoT Analytics API リファレンスの BatchPutMessage を参照してください。
- 保存および分析するデータのみを収集する - コンソールを使用して AWS IoT Analytics 、さまざまな形式と頻度で MQTT トピックフィルターを介してデバイスからメッセージを受信する AWS IoT Analytics ように を設定できます。 は、データが定義してチャネルを作成する特定のパラメータ内にあることを AWS IoT Analytics 検証します。次に、これらのチャネルを適切なパイプラインにルーティングして、メッセージの処理、変換、強化を行います。

処理

- クリーンアップとフィルタリング —AWS IoT Analytics が欠落データ AWS IoT Analytics を検出したときにトリガーされる AWS Lambda 関数を定義できるため、コードを実行してギャップを推定して埋めることができます。また、最大フィルター、最小フィルター、パーセンタイルのしきい値を定義して、データ上の異常値を削除することもできます。
- 変換 — 定義した数学的または条件付きロジックを使用してメッセージをAWS IoT Analytics 変換できるため、摂氏などの一般的な計算を華氏に変換できます。
- エンリッチ — 天気予報などの外部データソースでデータを強化し、データを AWS IoT Analytics データストアにルーティングAWS IoT Analytics できます。

保存

- 時系列データストア - デバイスデータを最適化された時系列データストアに保存し、取得と分析を高速化AWS IoT Analytics します。アクセス許可の管理、データ保持ポリシーの実装、外部アクセスポイントへのデータのエクスポートを行うこともできます。
- 処理済みデータと未加工データの保存 - 処理済みデータAWS IoT Analytics を保存し、取り込まれた未加工データを自動的に保存して、後で処理できるようにします。

分析

- アドホック SQL クエリの実行 - は SQL クエリエンジンAWS IoT Analytics を提供するため、アドホッククエリを実行して結果をすばやく取得できます。また、このサービスでは、標準の SQL クエリを使用してデータストアからデータを抽出し、コネクテッド車両群の平均走行距離や、午後 7 時以降に閉鎖されるスマートビルディングの部屋数などの質問に回答することができます。これらのクエリは、接続されたデバイスやフリートのサイズ、分析要件が変更されても再利用することができます。
- 時系列分析 - は時系列分析AWS IoT Analytics をサポートしているため、デバイスのパフォーマンスを経時的に分析し、それらが使用されている方法と場所を理解し、デバイスデータを継続的にモニタリングしてメンテナンスの問題を予測するとともに、センサーをモニタリングして環境条件を予測して対応できます。
- 高度な分析と機械学習向けにホストされたノートブック –AWS IoT Analytics では、統計的分析と Machine Learning 向けにホストされたノートブックが Jupyter Notebook でサポートされます。このサービスには、 AWS認可された機械学習モデルとビジュアライゼーションを含む一連のノートブックテンプレートが含まれています。これらのテンプレートを使用すれば、デバイス故障プロファイリング、顧客が製品を廃棄する目安となる使用量の低下などの予測イベント、顧客の使用レベル 例: 使用頻度の高いユーザー、週末のみ使用するユーザーやデバイスのヘルス状態に基づくデバイスのセグメント化に関する IoT ユースケースを開始できます。ノートブックを作成すると、指定したスケジュールに基づいてこのノートブックをコンテナ化して実行できます。詳細については、「ワークフローの自動化」を参照してください。
- 予測 — ロジスティック回帰と呼ばれる方法で、統計的な分類を行うことができます。また、長短期記憶 LSTMを使用することもできます。LSTM は、時間の経過とともに変化するプロセスの出力や状態を予測する強力なニューラルネットワーク技術です。あらかじめ作成されているノートブックテンプレートは、デバイスをセグメント化する k 平均法アルゴリズムをサポートしています。これにより、デバイスは、デバイスのグループにクラスタ化されます。これらのテンプレートは通常、チョコレート工場の HVAC ユニットや風力タービンの羽根の摩耗や破損といったデバイスの健全性や状態のプロファイリングに使用されます。ここでも、これらのノートブックテンプレートをコンテナ化し、スケジュールに従って実行できます。

構築および可視化

- QuickSight 統合 — QuickSight ダッシュボードでデータセットを視覚化できるように、QuickSight にコネクタAWS IoT Analytics を提供します。
- コンソール統合 — コンソールの埋め込み Jupyter Notebook で結果またはアドホック分析を視覚化することもできます AWS IoT Analytics。

AWS IoT Analytics のコンポーネントと概念

Channel

チャネルは、MQTT トピックからデータを収集し、未処理の raw メッセージをアーカイブしてから、データをパイプラインに発行します。BatchPutMessage API を使用して、チャンネルにメッセージを直接送信することもできます。未処理のメッセージは、ユーザーまたは AWS IoT Analytics 管理する Amazon Simple Storage Service (Amazon S3) バケットに保存されます。

パイプライン

パイプラインではチャネルからのメッセージが消費されます。パイプラインを使用すれば、メッセージをデータストアに保存する前にメッセージを処理することができます。このステップはアクティビティと呼ばれ パイプラインアクティビティ)、メッセージで変換を実行します。これには、メッセージ属性の削除/名前変更/追加、属性値に基づくメッセージのフィルタ処理、高度な処理のためのメッセージでの Lambda 関数の呼び出し、数値変換の実行によるデバイスデータの正規化などが含まれます。

データストア

パイプラインは、処理されたメッセージをデータストアに保存します。データストアはデータベースではなく、メッセージのスケーラブルでクエリ可能なリポジトリです。異なるデバイスや場所から送信されたメッセージに対して複数のデータストアを使用したり、パイプラインの設定と要件によってメッセージ属性でフィルタしたりできます。未処理のチャネルメッセージと同様に、データストアで処理されたメッセージは、ユーザーまたは AWS IoT Analytics 管理する Amazon S3 バケットに保存されます。

データセット

データストアからデータを取得するには、データセットを作成します。 AWS IoT Analytics を使用すると、SQL データセットまたはコンテナデータセットを作成できます。

データセットを作成したら、QuickSight を使用して統合することで、データを探索してインサイトを得ることができます。また、Jupyter Notebook と連携してより高度な分析機能を実行するこ

ともできます。Jupyter Notebook は、機械学習と様々な統計的分析を実行できる強力なデータサイエンスツールを備えています。詳細については、「ノートブックテンプレート」を参照してください。

データセットのコンテンツを Amazon S3 バケットに送信して、既存のデータレイクとの統合、または社内アプリケーションと可視化ツールからのアクセスができます。デバイスやプロセスのオペレーション時の障害や変化をモニタリングし、このようなイベントが発生した時に追加のアクションをトリガーするサービスである AWS IoT Events への入力としてデータセットコンテンツを送信できます。

SQL データセット

SQL データセットは、SQL データベースの具体化されたビューに似ています。SQL アクションを適用して SQL データセットを作成することができます。SQL データセットは、トリガーを指定して、定期的なスケジュールで自動的に生成できます。

コンテナデータセット

コンテナデータセットでは、自動で分析ツールを実行して結果を生成できます。詳細については、「ワークフローの自動化」を参照してください。コンテナデータセットでは、入力としての SQL データ、分析ツールと必要なライブラリファイルが含まれた Docker コンテナ、入力と出力の変数、およびオプションのスケジュールトリガーが 1 つにまとめられます。入力と出力の変数では、データを取得して結果を保存する先を実行可能イメージに指示します。トリガーによる分析は、SQL データセットでコンテンツの作成が終了した時点で実行されるか、時間のスケジュール式に従って実行されます。コンテナデータセットは、分析ツールの実行、結果の生成と保存を自動的に行います。

Trigger トリガー)

トリガーを指定して、データセットを自動的に作成できます。トリガーは、時間間隔 たとえば、このデータセットを 2 時間ごとに作成するなど)、あるいは別のデータセットのコンテンツの作成時 たとえば、`myOtherDataset` がコンテンツの作成を終了したときにこのデータセットを作成する、などに指定できます。あるいは、CreateDatasetContent API を使用してデータセットコンテンツを手動で生成することもできます。

Docker コンテナ

独自の Docker コンテナを作成して、分析ツールをパッケージ化したり、SageMaker AI が提供するオプションを使用したりできます。詳細については「Docker コンテナ」を参照してください。独自の Docker コンテナを作成して、分析ツールをパッケージ化したり、SageMaker AI が提供するオプションを使用したりできます。指定する Amazon ECR レジストリにコンテナを保存でき

るため、使用するプラットフォームでインストールできます。Docker コンテナでは、Matlab、オクターブ、Wise.io、SPSS、R、Fortran、Python、Scala、Java、C++ などで使用できるように準備されたカスタムの分析コードを実行できます。詳細については「ノートブックのコンテナ化」を参照してください。

デルタウィンドウ

デルタウィンドウは、重複することなく連続する時間間隔であり、ユーザーが定義します。デルタウィンドウを使用すると、前回の分析以降にデータストアに到着した新しいデータを使用してデータセットコンテンツを作成し、これらのデータに対して分析を実行できます。デルタウィンドウを作成するには、データセットの `queryAction` の `filters` 部分に `deltaTime` を設定します。詳細については、`CreateDataset` API を参照してください。通常は、時間間隔トリガー `triggers:schedule:expression`も設定して、データセットコンテンツを自動的に作成します。この操作で特定の時間ウィンドウに到着したメッセージをフィルタリングし、以前の時間ウィンドウのメッセージに含まれたデータが重複してカウントされないようにします。詳細については「例 6 -- デルタウィンドウを使用して SQL データセットを作成する CLI)」を参照してください。

アクセス AWS IoT Analytics

の一部として AWS IoT、 は、デバイスがデータを生成し、アプリケーションが生成されたデータとやり取りできるように、次のインターフェイス AWS IoT Analytics を提供します。

AWS Command Line Interface (AWS CLI)

Windows、OS X、Linux AWS IoT Analytics で のコマンドを実行します。これらのコマンドで、モノ、証明書、ルール、およびポリシーを作成し、管理することができます。使用を開始する方法については『AWS Command Line Interface ユーザーガイド』を参照してください。のコマンドの詳細については AWS IoT、「 AWS Command Line Interface リファレンス」の「iot」を参照してください。

> **Important**
>
> `aws iotanalytics` コマンドを使用して を操作します AWS IoT Analytics。IoT システムの他の部分を操作するには、`aws iot` コマンドを使用します。

AWS IoT API

HTTP または HTTPS リクエストを使用して IoT アプリケーションを構築します。これらの API アクションで、モノ、証明書、ルール、およびポリシーを作成し、管理することができます。詳細については、AWS IoT API リファレンスガイドの「アクション」を参照してください。

AWS SDKs

言語固有の APIs を使用して AWS IoT Analytics アプリケーションを構築します。これらの SDK により HTTP と HTTPS API がラップされ、サポートされている言語でプログラミングを実行できます。詳細については、AWS の SDK およびツールを参照してください。

AWS IoT デバイス SDKs

メッセージを送信するデバイスで実行されるアプリケーションを構築します AWS IoT Analytics。詳細については、AWS IoT SDK を参照してください。

AWS IoT Analytics コンソール

コンポーネントを構築すれば AWS IoT Analytics コンソールで結果を可視化できます。

ユースケース

予知保全

AWS IoT Analytics には、予測メンテナンスモデルを構築し、デバイスに適用するためのテンプレートが用意されています。例えば、 AWS IoT Analytics を使用して、接続された貨物車両で冷暖房システムが故障する可能性が高いタイミングを予測し、車両を再ルーティングして出荷の損傷を防ぐことができます。自動車メーカーでは、お客様の摩耗したブレーキパッドを検知し、車両のメンテナンスを勧めるアラートを提供できます。

サプライ品の事前補充

AWS IoT Analytics では、インベントリをリアルタイムでモニタリングできる IoT アプリケーションを構築できます。たとえば、食品飲料会社は、フード自動販売機のデータを分析し、サプライ品が不足すると事前に商品を再注文することができます。

プロセス効率のスコアリング

を使用すると AWS IoT Analytics、さまざまなプロセスの効率を常にモニタリングし、プロセスを改善するためのアクションを実行する IoT アプリケーションを構築できます。たとえば、鉱業会社では、1 回で輸送できる量を最大化して、鉱石トラックの効率を高めることができます。を使用すると AWS IoT Analytics、時間の経過とともに場所やトラックの最も効率的な負荷を特定

し、ターゲット負荷からの偏差をリアルタイムで比較し、効率を向上させるためのガイドラインをより適切に計画できます。

スマートアグリカルチャー

AWS IoT Analytics は、 AWS IoT レジストリデータまたはパブリックデータソースを使用して IoT デバイスデータをコンテキストメタデータで強化できるため、分析が時間、場所、温度、高度、その他の環境条件の要因になります。この分析を使用して、お使いのデバイス用に推奨されるアクションを出力してフィールドに取り込むモデルを作成できます。たとえば、かんがいシステムでは、湿度センサーのデータを降雨データで強化することで注水時期を決定し、水を効率的に使用できます。

AWS IoT Analytics サポート終了

慎重に検討した結果、2025 年 12 月 15 日 AWS IoT Analyticsをもってサポートを終了することにしました。 AWS IoT Analytics は 2024 年 7 月 24 日以降、新規顧客を受け入れなくなります。2024 年 7 月 23 日より前にサービスにサインアップしたアカウントを持つ既存のお客様は、引き続き AWS IoT Analytics 機能を使用できます。2025 年 12 月 15 日以降、 を使用することはできません AWS IoT Analytics。

AWS IoT Analytics が 2025 年 12 月 15 日にend-of-serviceと、お客様は移行オプションを理解することが重要です。このページでは、 の主な機能の概要 AWS IoT Analytics を示し、機能をレプリケートするために使用される代替 AWS サービスにマッピングします。これらの代替サービスの機能を理解することで、お客様はスムーズな移行を計画して実行し、 AWS IoT データ分析ワークフローを中断することなく継続できます。

トピック

- 移行オプション
- 移行ガイド

移行オプション

からの移行を検討するときは AWS IoT Analytics、この移行の利点と理由を理解することが重要です。次の表は、代替オプションと既存の AWS IoT Analytics 機能へのマッピングを示しています。

アクション	AWS IoT Analytics	代替サービス	理由
収集	AWS IoT Analytics を使用すると、`BatchPutMessage` API を使用して AWS IoT Core または他のソースから直接データを簡単に取り込むことができます。この統合により、デバイスから分析プラットフォームへのデー	• Amazon Kinesis Data Streams • Amazon Data Firehose	Amazon Kinesis Data Streams は堅牢なソリューションを提供します。Kinesis はデータをリアルタイムでストリーミングし、即時の処理と分析を可能にします。これは、リアルタイムのインサイトと異常検出を必要とする

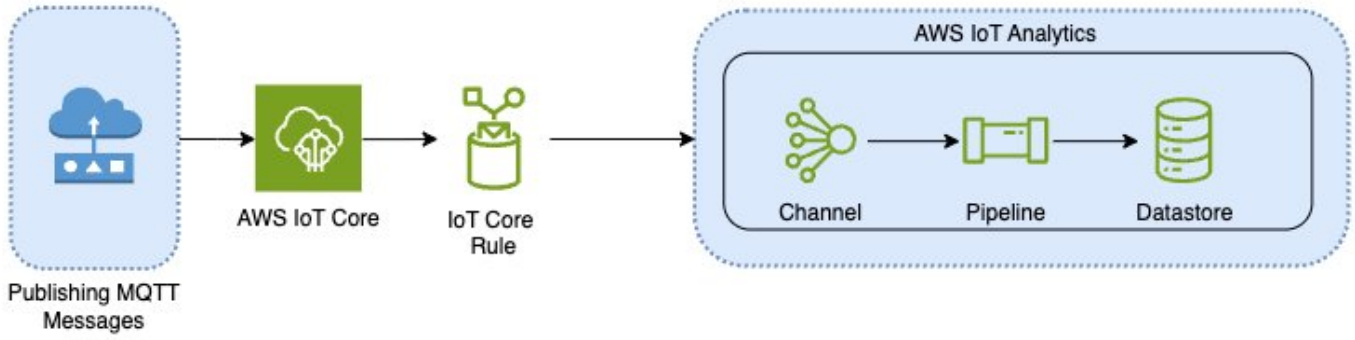
アクション	AWS IoT Analytics	代替サービス	理由
	タのシームレスなフローが保証されます。		アプリケーションにとって不可欠です。 Amazon Data Firehose は、Amazon S3 に到達する前にストリーミングデータをキャプチャおよび変換するプロセスを簡素化し、データスループットに合わせて自動的にスケーリングします。
プロセス	でデータを処理する AWS IoT Analytics には、外部ソースによるデータのクレンジング、フィルタリング、変換、強化が必要です。	• Amazon Managed Service for Apache Flink • Amazon Data Firehose	Amazon Managed Service for Apache Flink は、高度な AWS IoT Analytics シナリオに不可欠なパターンマッチングや集約などの複雑なイベント処理をサポートしています。 Amazon Data Firehose は、より単純な変換を処理し、カスタム処理のために AWS Lambda 関数を呼び出すことができるため、Flink の複雑さを伴わずに柔軟性を提供します。

アクション	AWS IoT Analytics	代替サービス	理由
保存	AWS IoT Analytics は、 AWS IoT データに最適化された時系列データストアを使用します。これには、データ保持ポリシーやアクセス管理などの機能が含まれます。	• Amazon S3 • Amazon Timestream	Amazon S3 は、スケーラブルで耐久性があり、費用対効果の高いストレージソリューションを提供します。Amazon S3 と他の AWS サービスとの統合により、大規模なデータセットの長期保存と分析に最適です。 Amazon Timestream は、専用の時系列データベースです。Amazon S3 からデータをバッチロードできます。

アクション	AWS IoT Analytics	代替サービス	理由
分析	AWS IoT Analytics は、組み込みの SQL クエリ機能、時系列分析、ホストされた Jupyter Notebooks のサポートを提供するため、高度な分析と機械学習を簡単に実行できます。	• AWS Glue • Amazon Athena	AWS Glue は ETL プロセスを簡素化し、データの抽出、変換、ロードを容易にすると同時に、Athena と統合してクエリを容易にするデータカタログを提供します。 Amazon Athena は、インフラストラクチャを管理することなくAmazon S3 に保存されているデータに対して SQL クエリを直接実行できるようにすることで、このステップをさらに進めます。
可視化	AWS IoT Analytics は QuickSight と統合され、豊富なビジュアライゼーションとダッシュボードを作成できます。	• Amazon QuickSight	Amazon S3 など、使用する代替データストアに応じて、QuickSight を引き続き使用します。Amazon S3

移行ガイド

現在のアーキテクチャでは、 AWS IoT データは AWS IoT Core ルール AWS IoT Analytics を介してから AWS IoT Core に流れます。 AWS IoT Analytics は取り込み、変換、ストレージを処理します。



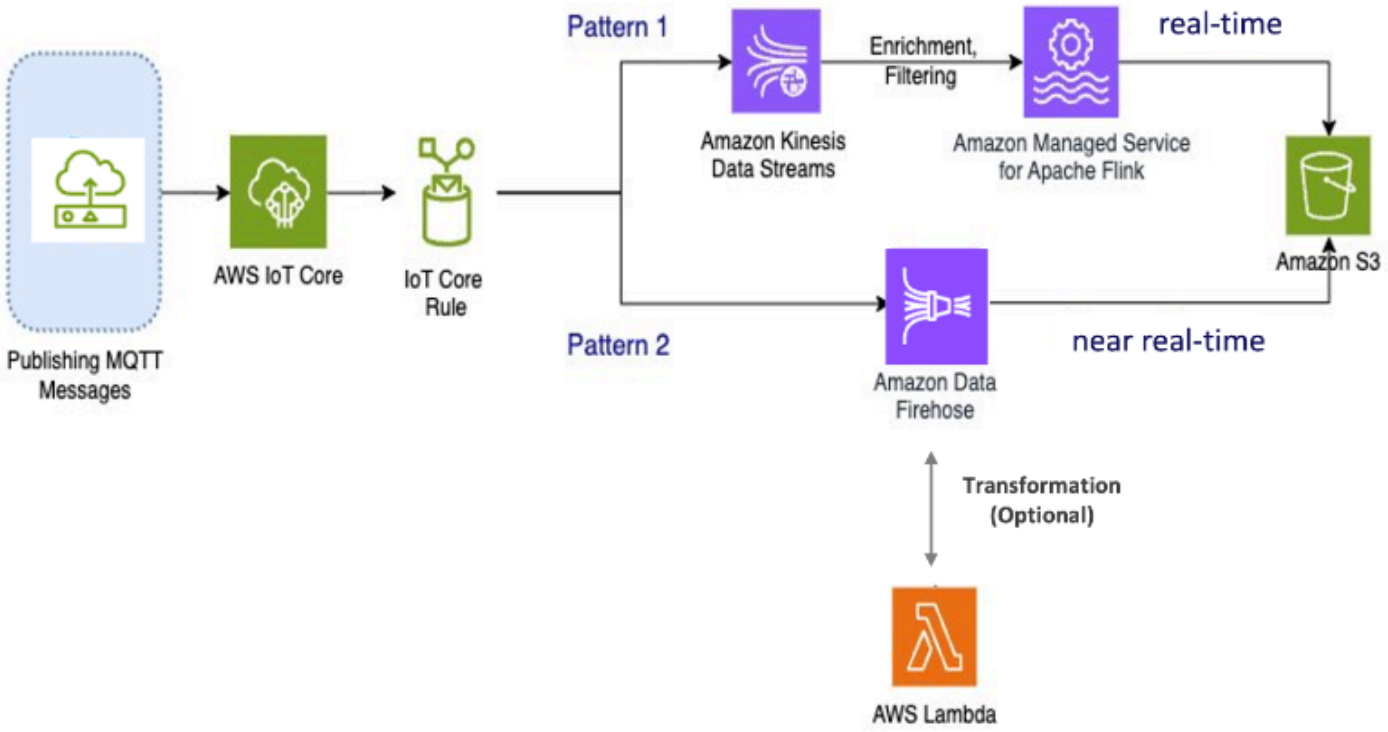
移行を完了するには、次の 2 つのステップに従います。

トピック

- ステップ 1: 進行中のデータ取り込みをリダイレクトする
- ステップ 2: 以前に取り込んだデータをエクスポートする
- 両方のパターンに対してオンデマンドクエリを実行する
- 概要

ステップ 1: 進行中のデータ取り込みをリダイレクトする

移行の最初のステップは、進行中のデータ取り込みを新しいサービスにリダイレクトすることです。特定のユースケースに基づいて 2 つのパターンをお勧めします。



パターン 1: Amazon Managed Service for Apache Flink を使用した Amazon Kinesis Data Streams

このパターンでは、まず、 が Amazon Kinesis Data Streams と統合 AWS IoT Core されているデータを公開し、大量のデータをリアルタイムで収集、処理、分析できるようにします。

メトリクスと分析

1. Data: AWS IoT data の取り込みは、Amazon Kinesis Data Streams にリアルタイムで取り込まれます。Amazon Kinesis Data Streams は、数百万の AWS IoT デバイスからのデータの高スループットを処理できるため、リアルタイムの分析と異常検出が可能になります。
2. データの処理: Amazon Managed Service for Apache Flink を使用して、Amazon Kinesis Data Streams からデータを処理、強化、フィルタリングします。Flink は、集計、結合、時間操作など、複雑なイベント処理のための堅牢な機能を提供します。
3. データの保存: Flink は、処理されたデータを Amazon S3 に出力して、保存とさらなる分析を行います。このデータは、Amazon Athena を使用してクエリすることも、他の AWS 分析サービスと統合することもできます。

アプリケーションで高帯域幅ストリーミングデータが必要で、パターンマッチングやウィンドウ処理などの高度な処理が必要な場合は、このパターンを使用します。このパターンが最適です。

パターン 2: Amazon Data Firehose を使用する

このパターンでは、データは に発行され AWS IoT Core、Amazon Data Firehose と統合されるため、データを Amazon S3 に直接保存できます。このパターンは、 を使用した基本的な変換もサポートしています AWS Lambda。

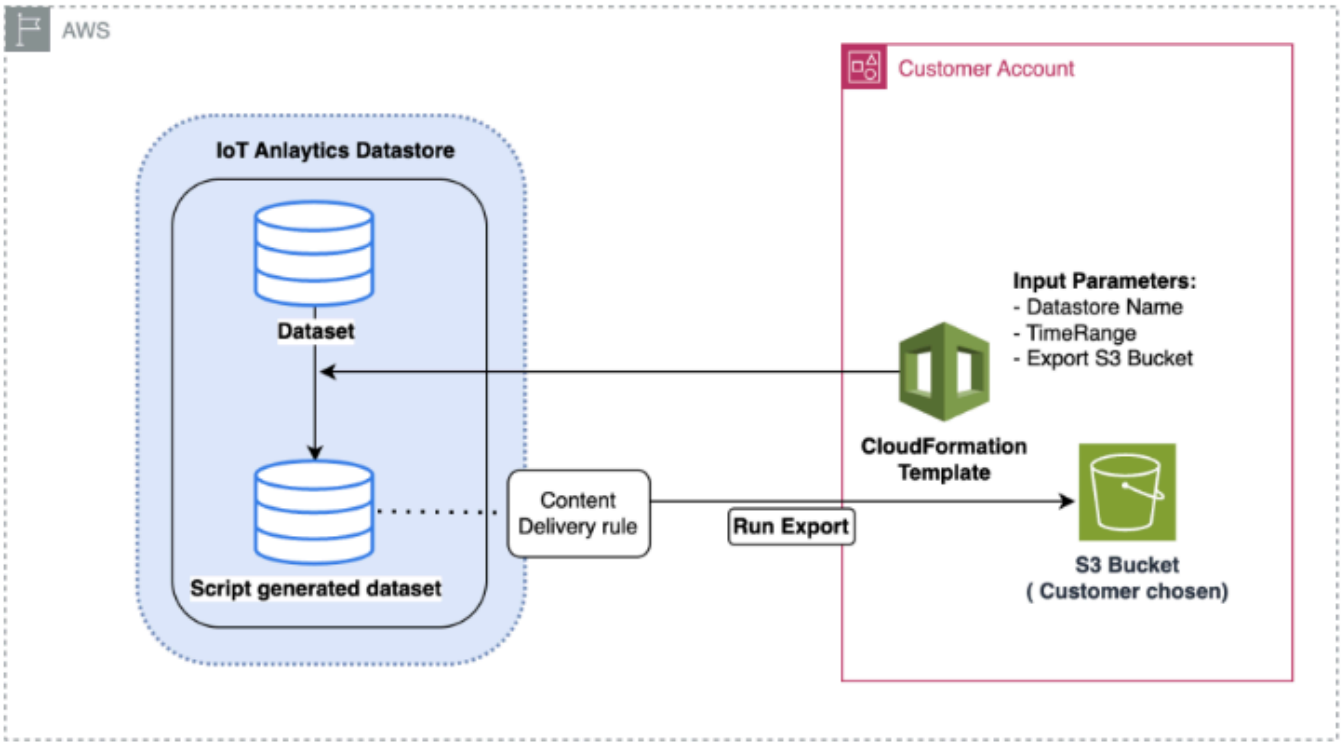
メトリクスと分析

1. Data: AWS IoT data の取り込みは、デバイスから、または Amazon Data Firehose AWS IoT Core に直接取り込まれます。
2. データの処理: Amazon Data Firehose は、形式変換やエンリッチメントなど、データに対して基本的な変換と処理を実行します。Firehose データ変換を有効にするには、 AWS Lambda 関数を呼び出して受信ソースデータを変換してから送信先に配信するように設定します。
3. データの保存: 処理されたデータは、ほぼリアルタイムで Amazon S3 に配信されます。Amazon Data Firehose は、受信データのスループットに合わせて自動的にスケーリングされるため、信頼性が高く効率的なデータ配信が可能になります。

このパターンは、基本的な変換と処理を必要とするワークロードに使用します。さらに、Amazon Data Firehose は、Amazon S3 に保存されているデータのデータバッファリングと動的パーティショニング機能を提供することで、プロセスを簡素化します。

ステップ 2: 以前に取り込んだデータをエクスポートする

以前に取り込んで保存したデータについては AWS IoT Analytics、Amazon S3 にエクスポートする必要があります。このプロセスを簡素化するために、テンプレートを使用して AWS CloudFormation データエクスポートワークフロー全体を自動化できます。スクリプトは、部分的な (時間範囲ベースの) データ抽出に使用できます。



AWS CloudFormation Amazon S3 にデータをエクスポートするための テンプレート

上記の図は、 AWS CloudFormation テンプレートを使用して同じ AWS IoT Analytics データストア内にデータセットを作成し、タイムスタンプに基づいて選択できるようにするプロセスを示しています。これにより、ユーザーは目的の期間内に特定のデータポイントを取得できます。さらに、Amazon S3 バケットにデータをエクスポートするためのコンテンツ配信ルールが作成されます。

以下の手順は、ステップを示しています。

1. AWS CloudFormation テンプレートを準備し、YAML ファイルとして保存します。例えば、`migrate-datasource.yaml`。

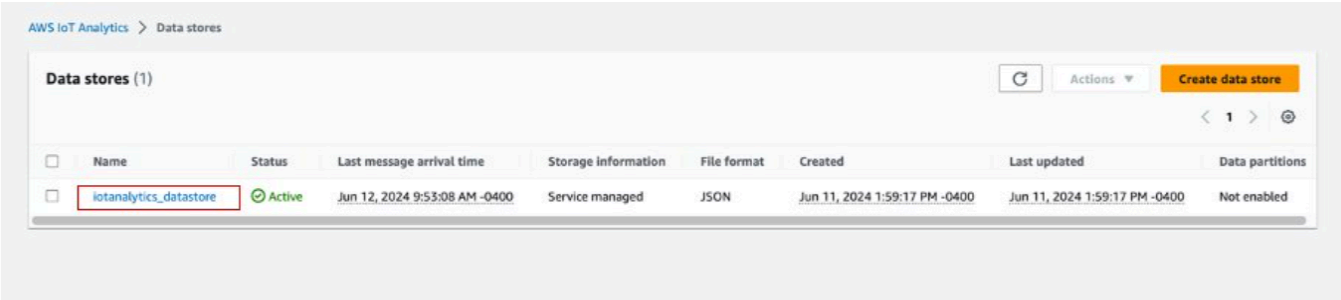
```
# Cloudformation Template to migrate an AWS IoT Analytics datastore to an external
 dataset
AWSTemplateFormatVersion: 2010-09-09
Description: Migrate an AWS IoT Analytics datastore to an external dataset
Parameters:
  DatastoreName:
    Type: String
```

```
    Description: The name of the datastore to migrate.
    AllowedPattern: ^[a-zA-Z0-9_]+$
  TimeRange:
    Type: String
    Description: |
      This is an optional argument to split the source data into multiple files.
      The value should follow the SQL syntax of WHERE clause.
      E.g. WHERE DATE(Item_TimeStamp) BETWEEN '09/16/2010 05:00:00' and '09/21/2010
 09:00:00'.
    Default: ''
  MigrationS3Bucket:
    Type: String
    Description: The S3 Bucket where the datastore will be migrated to.
    AllowedPattern: (?!(^xn--|.+-s3alias$))^[a-z0-9][a-z0-9-]{1,61}[a-z0-9]$
  MigrationS3BucketPrefix:
    Type: String
    Description: The prefix of the S3 Bucket where the datastore will be migrated
 to.
    Default: ''
    AllowedPattern: (^([a-zA-Z0-9.\-_]*\/)*$)|(^$)
Resources:
  # IAM Role to be assumed by the AWS IoT Analytics service to access the external
 dataset
  DatastoreMigrationRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          - Effect: Allow
            Principal:
              Service: iotanalytics.amazonaws.com
            Action: sts:AssumeRole
      Policies:
        - PolicyName: AllowAccessToExternalDataset
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              - Effect: Allow
                Action:
                  - s3:GetBucketLocation
                  - s3:GetObject
                  - s3:ListBucket
                  - s3:ListBucketMultipartUploads
```

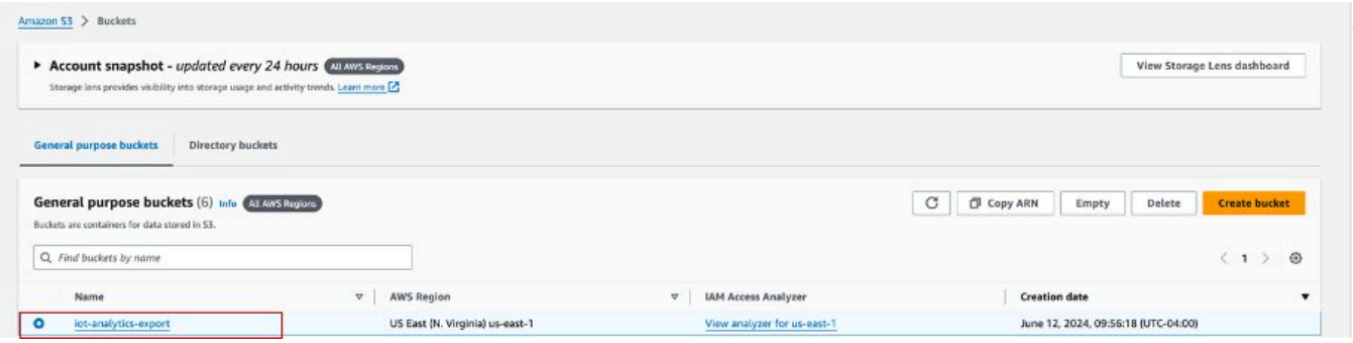
```
                  - s3:ListMultipartUploadParts
                  - s3:AbortMultipartUpload
                  - s3:PutObject
                  - s3:DeleteObject
                Resource:
                  - !Sub arn:aws:s3:::${MigrationS3Bucket}
                  - !Sub arn:aws:s3:::${MigrationS3Bucket}/
${MigrationS3BucketPrefix}*

  # This dataset that will be created in the external S3 Export
  MigratedDataset:
    Type: AWS::IoTAnalytics::Dataset
    Properties:
      DatasetName: !Sub ${DatastoreName}_generated
      Actions:
        - ActionName: SqlAction
          QueryAction:
            SqlQuery: !Sub SELECT * FROM ${DatastoreName} ${TimeRange}
      ContentDeliveryRules:
        - Destination:
            S3DestinationConfiguration:
              Bucket: !Ref MigrationS3Bucket
              Key: !Sub ${MigrationS3BucketPrefix}${DatastoreName}/!
{iotanalytics:scheduleTime}/!{iotanalytics:versionId}.csv
              RoleArn: !GetAtt DatastoreMigrationRole.Arn
      RetentionPeriod:
        Unlimited: true
      VersioningConfiguration:
        Unlimited: true
```

2. データをエクスポートする必要がある AWS IoT Analytics データストアを決定します。このガイドでは、 という名前のサンプルデータストアを使用します`iot_analytics_datastore`。

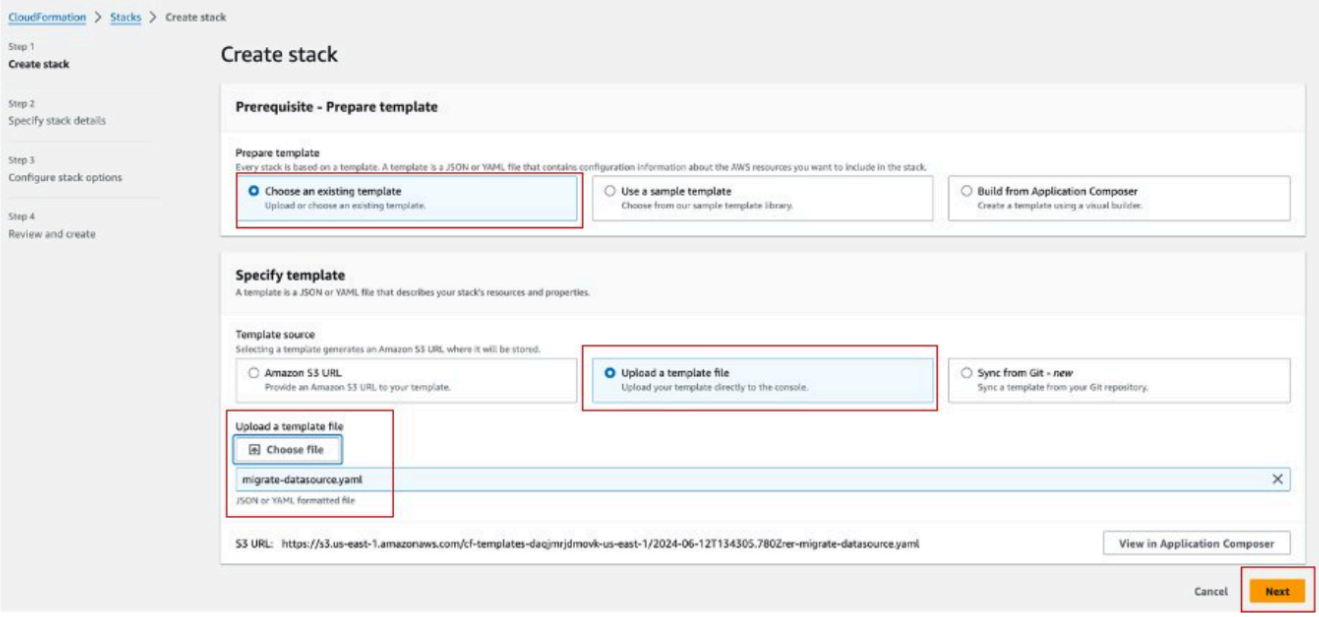


3. データがエクスポートされる Amazon S3 バケットを作成または識別します。このガイドでは、`iot-analytics-export`バケットを使用します。



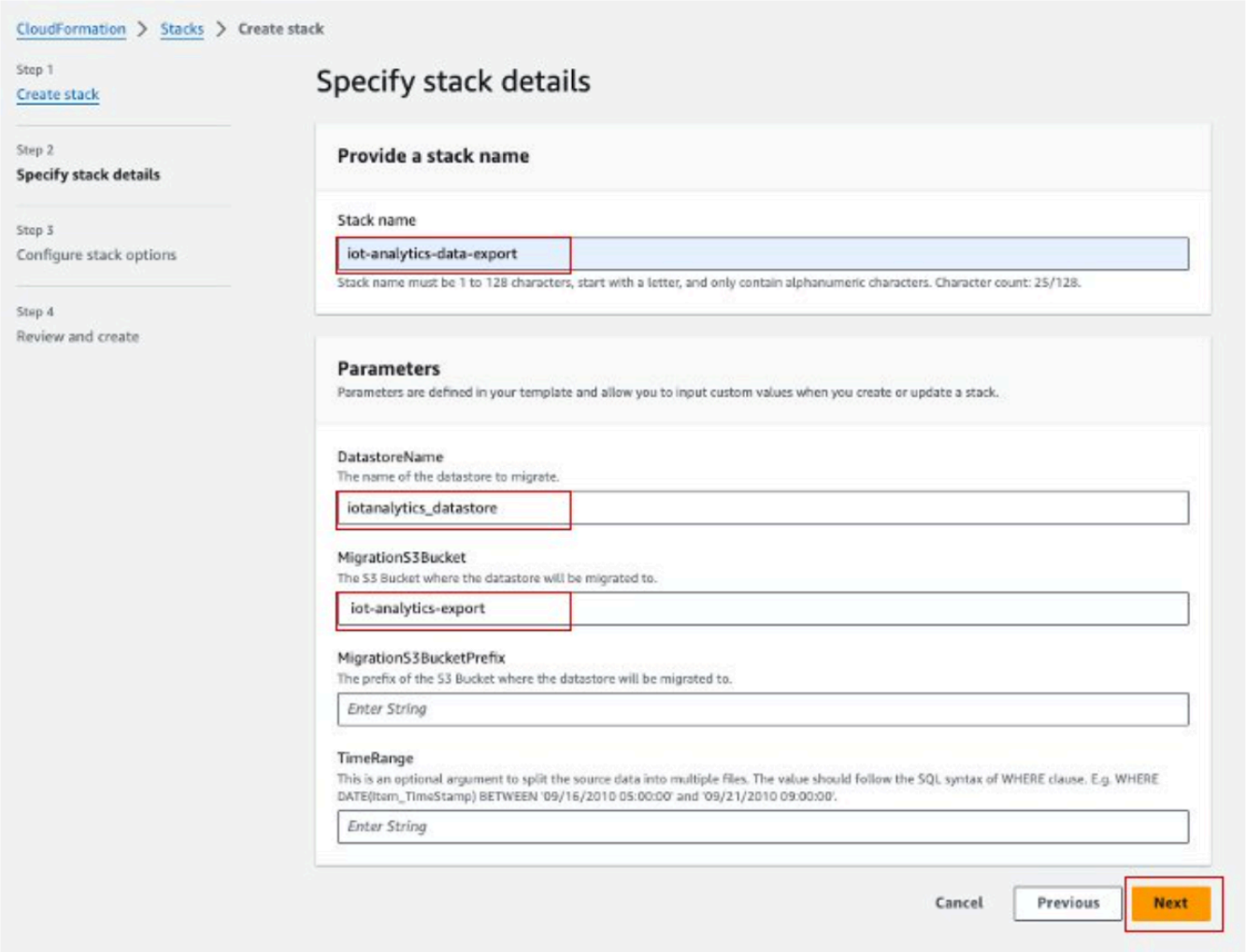
4. AWS CloudFormation スタックを作成します。

- https://console.aws.amazon.com/cloudformation に移動します。
- スタックの作成をクリックし、新しいリソース (標準) で を選択します。
- `migrate-datasource.yaml` ファイルをアップロードします。

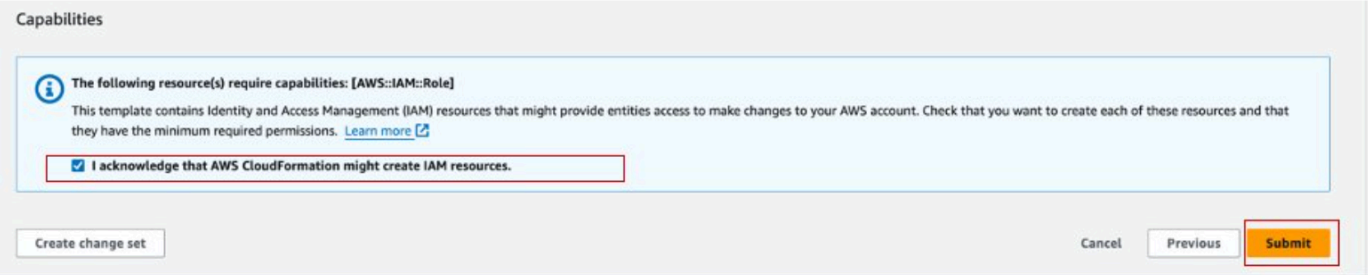


5. スタック名を入力し、次のパラメータを指定します。

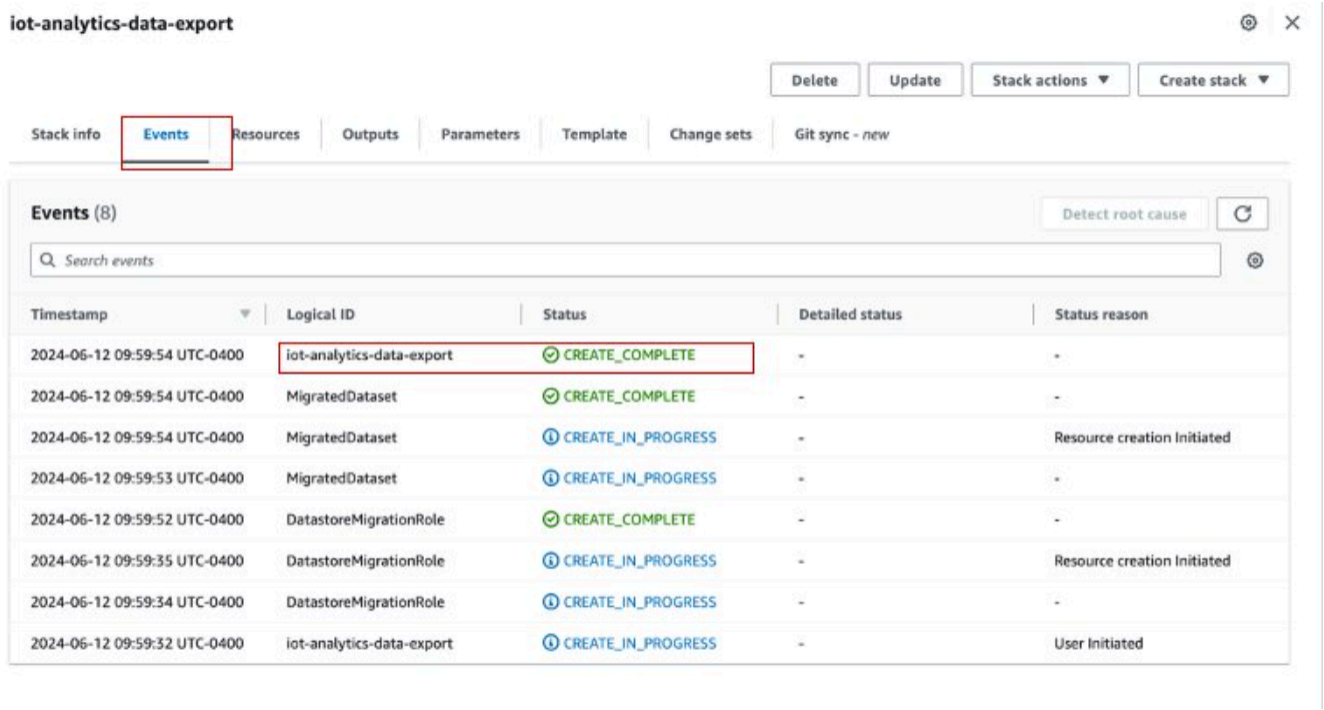
- DatastoreName: 移行する AWS IoT Analytics データストアの名前。
- MigrationS3Bucket: 移行されたデータが保存されている Amazon S3 バケット。
- MigrationS3BucketPrefix (オプション) : Amazon S3 バケットのプレフィックス。
- TimeRange (オプション) : エクスポートするデータをフィルタリングするSQL `WHERE`句。指定した時間範囲に基づいてソースデータを複数のファイルに分割できます。



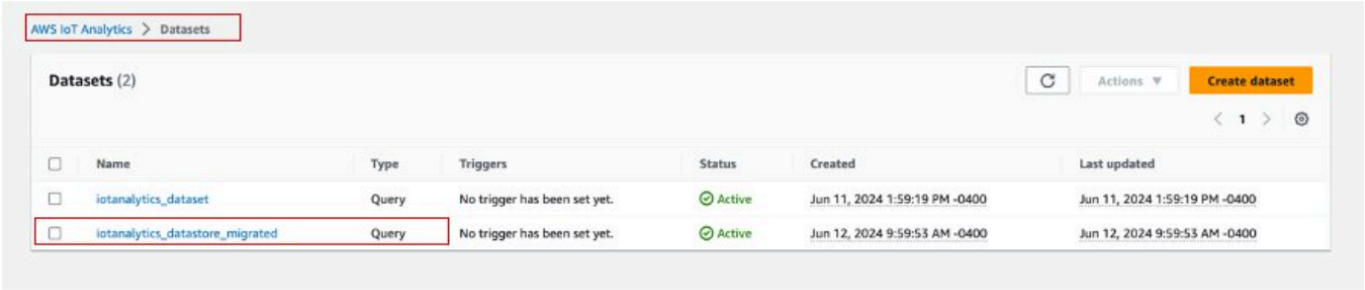
6. スタックオプションの設定画面の次へをクリックします。
7. チェックボックスを選択して IAM リソースの作成を確認し、送信をクリックします。



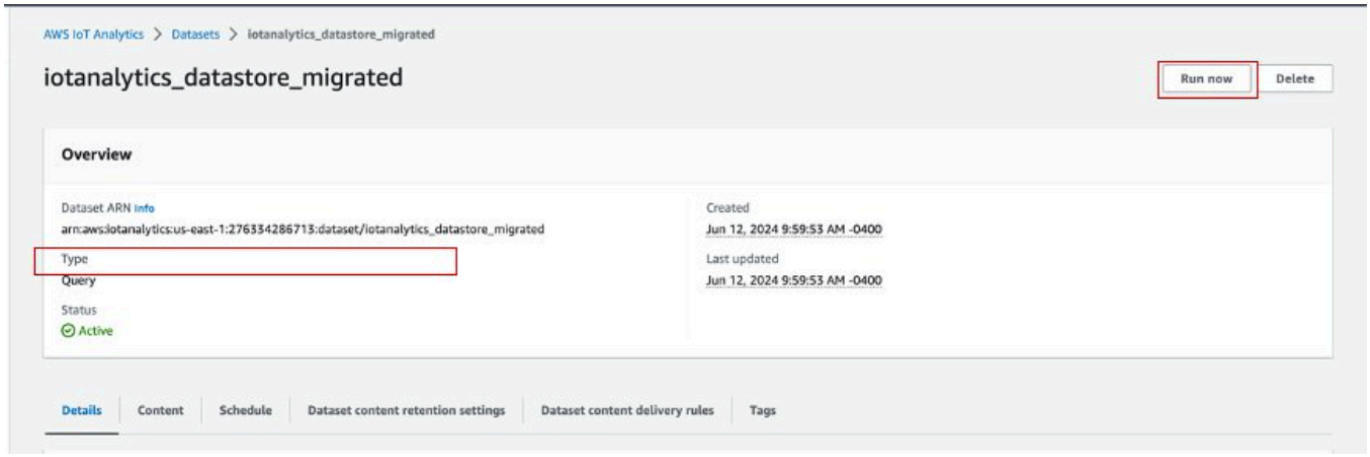
8. イベントタブのスタック作成の完了を確認します。



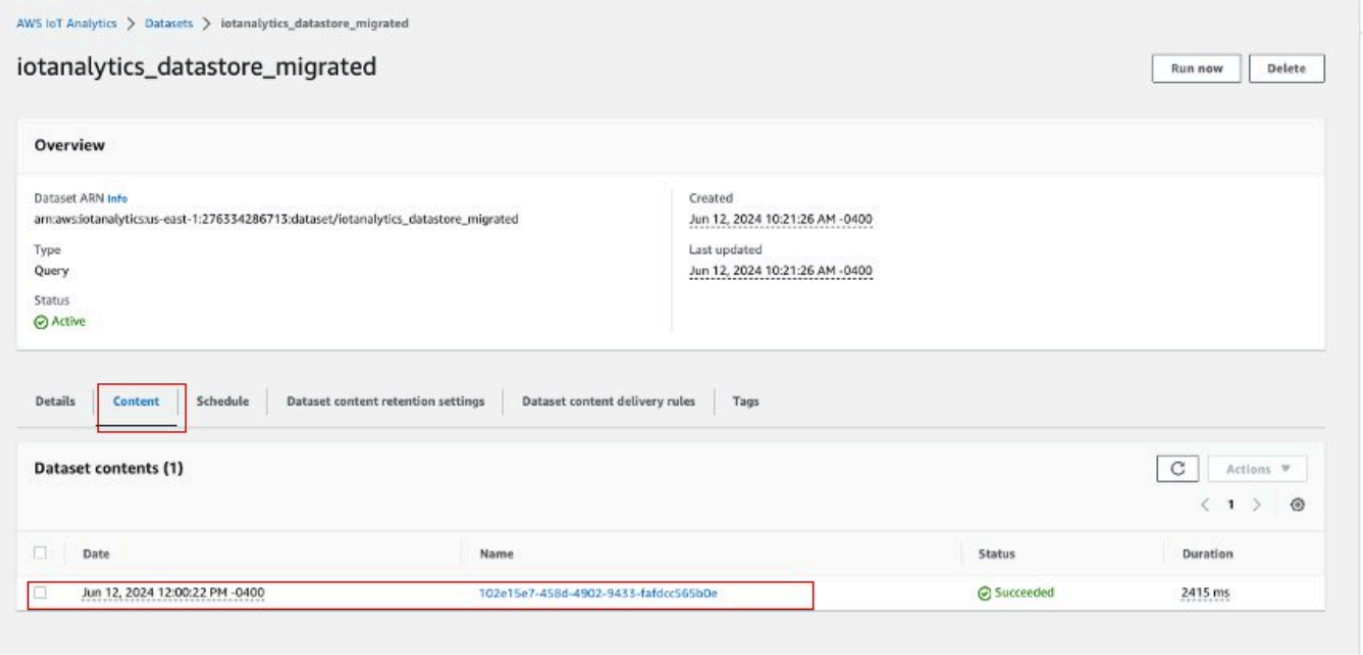
9. スタックが正常に完了したら、 AWS IoT Analytics → Datasets に移動して、移行されたデータセットを表示します。



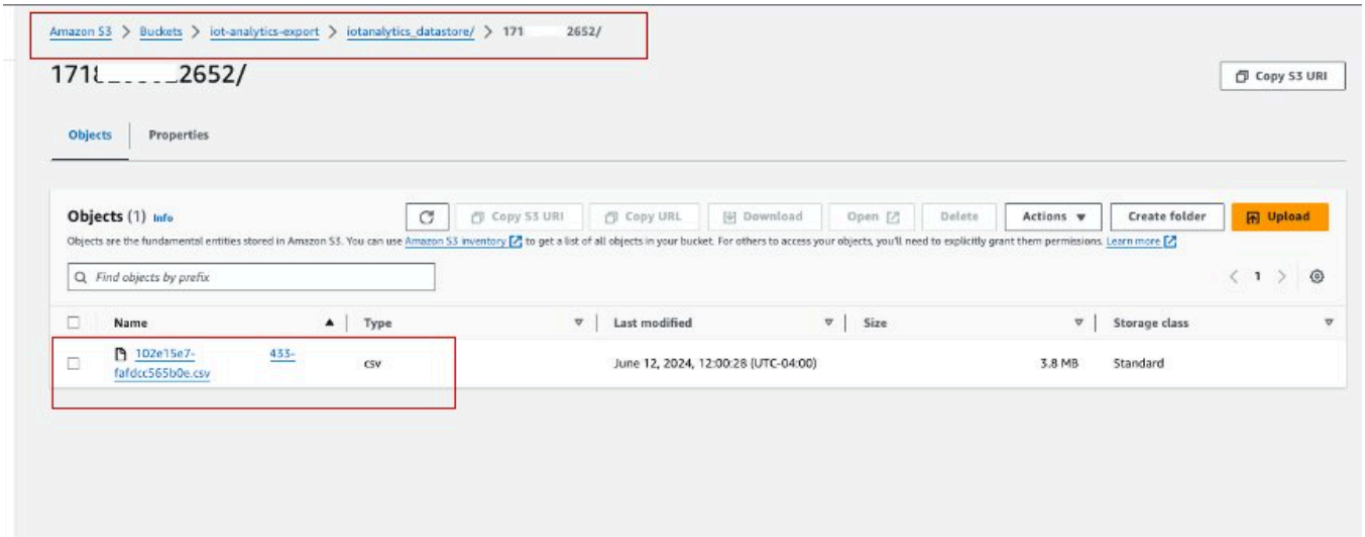
10. 生成されたデータセットを選択し、今すぐ実行をクリックしてデータセットをエクスポートします。



11. コンテンツは、データセットのコンテンツタブで表示できます。

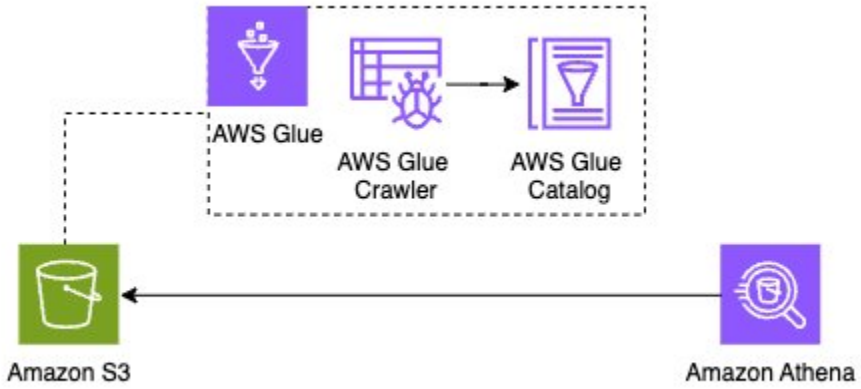


12. 最後に、Amazon S3 コンソールで iot-analytics-export バケットを開いて、エクスポートされたコンテンツを確認します。



両方のパターンに対してオンデマンドクエリを実行する

AWS IoT Analytics ワークロードを Amazon Kinesis Data Streams または Amazon Data Firehose に移行すると、 AWS Glue と Amazon Athena を活用することで、データ分析プロセスをさらに合理化できます。Amazon Athena はデータの準備と変換 AWS Glue を簡素化しAmazon Athena はデータのサーバーレスクエリをすばやく実行できます。これらを組み合わせることで、 AWS IoT データを分析するための強力でスケーラブル、費用対効果の高いソリューションを提供します。



概要

AWS IoT Analytics ワークロードを から AWS IoT Analytics Amazon Kinesis Data Streams、Amazon S3、 に移行すると、大規模で複雑な AWS IoT データを処理できるようになります。このアーキテ

クチャは、スケーラブルで耐久性のあるストレージと強力な分析機能を提供し、IoT データからリアルタイムでより深いインサイトを得ることができます。

を使用して作成されたリソースをクリーンアップ AWS CloudFormation することは、移行が完了した後の予期しないコストを避けるために不可欠です。

データ移行に関連するコストについては、 AWS IoT Analytics 料金表ページを参照してください。不要な費用が発生しないように、完了したら新しく作成したデータセットを削除することを検討してください。

データセットの完全エクスポート: 時間ベースの分割なしでデータセット全体をエクスポートするには、 AWS IoT Analytics コンソールを使用して、それに応じてコンテンツ配信ルールを設定することもできます。

移行ガイドに従うことで、データの取り込みと処理パイプラインをシームレスに移行し、継続的で信頼性の高いデータフローを確保できます。 AWS Glue と Amazon Athena を活用すると、データの準備とクエリがさらに簡素化され、インフラストラクチャを管理せずに高度な分析を実行できます。

このアプローチにより、 AWS IoT Analytics 労力を効果的にスケールできるため、ビジネスの要求の増大に適応し、 AWS IoT データから最大価値を引き出すことが容易になります。

の開始方法 AWS IoT Analytics（コンソール）

このチュートリアルでは、IoT デバイスデータに関する有用なインサイトを発見するために必要な AWS IoT Analytics リソース (コンポーネントとも呼ばれます) を作成します。

> **メモ**
>
> - 次のチュートリアルで大文字を入力すると、 AWS IoT Analytics によって自動的に小文字に変更されます。
> - AWS IoT Analytics コンソールには、チャネル、パイプライン、データストア、データセットを作成するためのワンクリックの開始方法機能があります。この機能は、 AWS IoT Analytics コンソールにサインインするときに確認できます。
> - このチュートリアルでは、 AWS IoT Analytics リソースを作成するための各ステップについて説明します。

AWS IoT Analytics チャネル、パイプライン、データストア、データセットを作成するには、次の手順に従います。このチュートリアルでは、 AWS IoT Core コンソールを使用して、取り込まれるメッセージを送信する方法も示します AWS IoT Analytics。

トピック

- AWS IoT Analytics コンソールにサインインする
- チャネルの作成
- データストアの作成
- パイプラインを作成する
- データセットの作成
- を使用してメッセージデータを送信する AWS IoT
- AWS IoT メッセージの進行状況を確認する
- クエリ結果へのアクセス
- データの探索
- ノートブックテンプレート

AWS IoT Analytics コンソールにサインインする

開始するには、 AWS アカウントが必要です。すでに AWS アカウントをお持ちの場合は、https://console.aws.amazon.com/iotanalytics/ に移動します。

AWS アカウントがない場合は、次のステップに従ってアカウントを作成します。

AWS アカウントを作成するには

1. https://portal.aws.amazon.com/billing/signup を開きます。
2. オンラインの手順に従います。

   サインアップ手順の一環として、電話またはテキストメッセージを受け取り、電話キーパッドで検証コードを入力します。

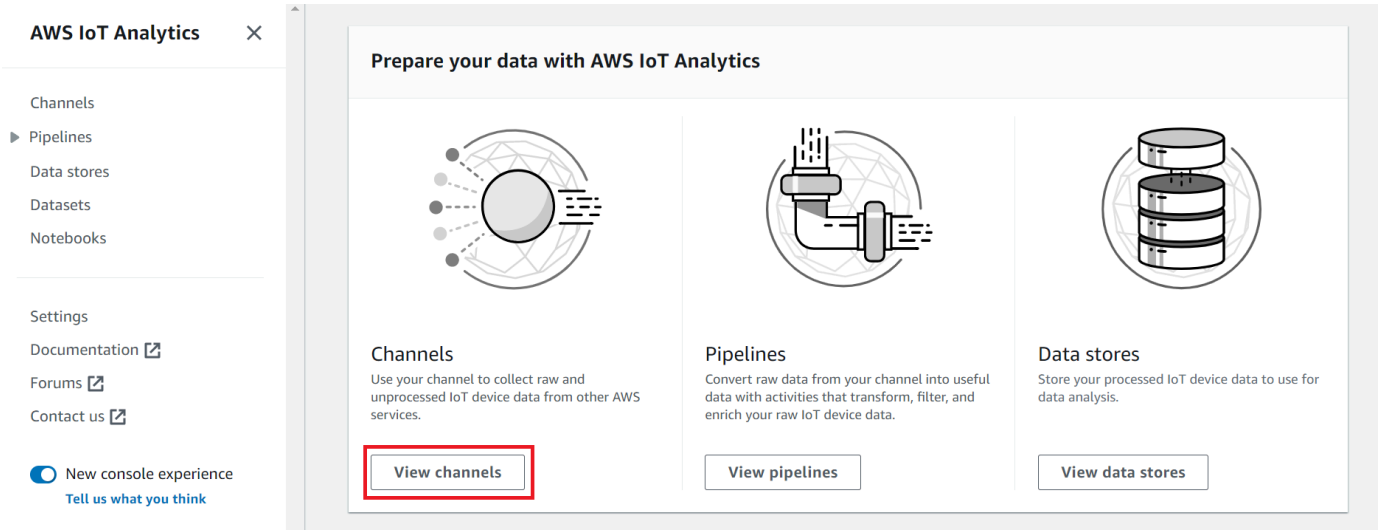
   にサインアップすると AWS アカウント、 AWS アカウントのルートユーザー が作成されます。ルートユーザーには、アカウントのすべての AWS のサービス とリソースへのアクセス権があります。セキュリティベストプラクティスとして、ユーザーに管理アクセス権を割り当て、ルートユーザーアクセスが必要なタスクの実行にはルートユーザーのみを使用するようにしてください。
3. にサインイン AWS Management Console し、https://console.aws.amazon.com/iotanalytics/ に移動します。

チャネルの作成

チャネルは、生データ、未処理、および非構造化の IoT デバイスデータを収集してアーカイブします。チャネルを作成するには次の手順に従います。

チャネルを作成するには

1. https://console.aws.amazon.com/iotanalytics/ の とデータの準備 AWS IoT Analyticsセクションで、チャネルの表示 を選択します。



> **Tip**
>
> ナビゲーションペインから チャネル を選択することもできます。

2. チャネル ページで、チャネルの作成 を選択します。
3. チャネルの詳細の指定 ページで、チャネルの詳細情報を入力します。
   a. チャネル名は一意で識別しやすいものを入力してください。
   b. オプション Tags タグ で、1 つまたは複数のカスタムタグ キーと値のペア をチャネルに追加します。タグを使用すると、 AWS IoT Analyticsのために作成するリソースを識別することができます。
   c. [次へ] を選択します。
4. AWS IoT Analytics は、未処理の未加工の IoT デバイスデータを Amazon Simple Storage Service (Amazon S3) バケットに保存します。独自の Amazon S3 バケットを選択できます。このバケットにアクセスして管理 AWS IoT Analytics することも、Amazon S3 バケットを自分で管理することもできます。
   a. このチュートリアルでは、ストレージタイプ で、 サービス管理ストレージ を選択します。
   b. 生データ保存期間の選択 で 無期限 を選択します。
   c. [次へ] を選択します。
5. ソースの設定ページで、 がメッセージデータを収集 AWS IoT Analytics するための情報を入力します AWS IoT Core。

   a. AWS IoT Core トピックフィルターを入力します。例: `update/environment/dht1`。このチュートリアルでは、後でこのトピックフィルターを使用してメッセージデータをチャネルに送信します。
   b. IAM ロール名 エリアで 新規作成 を選択します。新しいロールの作成 ウィンドウでロールの 名前 を入力し、次に ロールの作成 を選択します。これで、適切なポリシーがアタッチされた新しいロールが自動的に作成されます。
   c. Next 次へ をクリックします。
6. 選択内容を確認し、チャネルの作成 を選択します。
7. 新しいチャネルが チャネルページに表示されていることを確認します。

データストアの作成

データストアはメッセージデータの受信と保存を行いいます。データストアはデータベースではありません。データストアは、 Amazon S3 バケット内にあるスケーラブルでクエリ可能なリポジトリです。異なるデバイスまたは場所から取得されたメッセージに対して、複数のデータストアを使用できます。または、パイプラインの設定と要件に応じて、メッセージデータをフィルタリングできます。

データストアを作成するには以下の手順を実行します。

データストアを作成する方法

1. https://console.aws.amazon.com/iotanalytics/ の AWS IoT Analyticsとデータの準備セクションで、データストアの表示 を選択します。
2. データストア ページで、データストアの作成を選択します。
3. データストア詳細の指定ページで、データストアに関する基本情報を入力します。
   a. データストア ID に、一意のデータストア ID を入力します。この ID は作成後には変更できません。
   b. オプショナルタグ で 新しいタグを追加を選択すれば、データストアに 1 つまたは複数のカスタムタグ キーと値のペア を追加できます。タグを使用すると、 AWS IoT Analyticsのために作成するリソースを識別することができます。
   c. [次へ] を選択します。
4. ストレージタイプの設定 ページで、データの保存方法を指定します。
   a. ストレージタイプ で、サービス管理ストレージ を選択します。

b. 処理されたデータを保持する期間を設定します 処理データの保存期間の設定 で、無期限 を選択します。

c. [次へ] を選択します。

5. AWS IoT Analytics データストアは、JSON ファイル形式と Parquet ファイル形式をサポートしています。データストアのデータ形式で、JSON または Parquet を選択します。ファイル形式でサポートされるファイルタイプについては、「 AWS IoT Analytics 」を参照してください。

   [次へ] を選択します。

6. (オプション) はデータストア内のカスタムパーティション AWS IoT Analytics をサポートしているため、プルーニングされたデータをクエリしてレイテンシーを改善できます。サポートされるカスタムパーティションの詳細については、「カスタムパーティション」を参照してください。

   Next 次へ をクリックします。

7. 選択内容を確認してから、データストアの作成を選択します。

8. データストアページに新しいデータストアが表示されていることを確認します。

パイプラインを作成する

データストアにチャネルを接続するには、パイプラインを作成する必要があります。ベーシックなパイプラインでは、データの収集とメッセージの送信先となるデータストアの識別を行うチャネルが指定されるだけです。詳細については、「パイプラインアクティビティ」を参照してください。

このチュートリアルでは、データストアにチャネルを接続するだけのパイプラインを作成します。後で、パイプラインアクティビティを追加すればこのデータを処理できます。

パイプラインを作成するには以下の手順を実行します。

パイプラインを作成するには

1. https://console.aws.amazon.com/iotanalytics/ の AWS IoT Analyticsセクションで、データの準備パイプラインの表示を選択します。

   > **Tip**
   > ナビゲーションペインで パイプラインを選択することもできます。

2. パイプライン ページ で、Cパイプラインの作成 をクリックします。
3. パイプラインの詳細情報を入力します。
   a. パイプライン ID とソースのセットアップにパイプライン名を入力します。
   b. パイプラインのソースを選択します。これは、パイプラインがメッセージを読み取る AWS IoT Analytics チャネルです。
   c. パイプラインの出力を指定します。これは、処理されたメッセージデータの保存先となるデータストアです。
   d. オプション タグ で、1 つまたは複数のカスタムタグ キーと値のペア をパイプラインに追加します。
   e. メッセージ属性の推測 ページで、属性名と値の例を入力し、リストからデータタイプを選択して、属性の追加 を選択します。
   f. 必要な属性の数だけ前の手順を繰り返したら、次へを選択します。
   g. 現時点ではパイプラインアクティビティは追加しません。メッセージの強化、変換、フィルタリング ページで 次へ を選択します。
4. 選択内容を確認し、パイプラインの作成 選択します。
5. 新しいパイプラインが パイプライン ページに表示されていることを確認します。

> **Note**
>
> AWS IoT Analytics リソースを作成して、以下を実行できるようにしました。
>
> - チャネルを使って生で未処理の IoT デバイスメッセージデータを収集する。
> - IoT デバイスメッセージデータをデータストアに保存する。
> - パイプラインを使って、データのクリーニング、フィルタリング、変換、および強化を行う。
>
> 次に、 AWS IoT Analytics SQL データセットを作成して、IoT デバイスに関する有用なインサイトを見つけます。

データセットの作成

> **Note**
> データセットは、通常、表形式で編成される場合とそうでない場合があるデータの集合です。対照的に、 は SQL クエリをデータストアのデータに適用してデータセット AWS IoT Analytics を作成します。

これで、チャネルでパイプラインに生のメッセージデータをルーティングし、データストアにデータを保存してクエリを実行できます。データに対してクエリを実行するには、データセットを作成します。データセットには、データストアのクエリに使用する SQL のステートメントと式、および、指定した日時にクエリを繰り返すオプションのスケジュールが含まれます。オプションのスケジュールを作成するには、Amazon CloudWatch のスケジュール式のような式を使用できます。

データセットを作成する方法

1. https://console.aws.amazon.com/iotanalytics/ で、左側のナビゲーションペインの データセット を選択します。
2. データセットの作成 ページで SQL の作成 を選択します。
3. データセットの詳細を指定 でデータセットの詳細を指定します。
   a. データセットの名前を入力します。
   b. データストアのソース で、前の段階で作成したデータストアを識別する一意の ID を選択します。
   c. オプション タグ で、1 つまたは複数のカスタムタグ キーと値のペア をデータセットに追加します。
4. SQL 式を使用して、データをクエリし、分析的な質問に答えます。クエリの結果はこのデータセットに保存されます。
   a. クエリの作成 フィールドに、ワイルドカードを使用してデータを最大で 5 行表示する SQL クエリを入力します。

      ```
      SELECT * FROM my_data_store LIMIT 5
      ```

      でサポートされている SQL 機能の詳細については AWS IoT Analytics、「」を参照してください の SQL 式 AWS IoT Analytics。

b. クエリのテスト を選択すれば、入力の正誤を確認でき、さらに、クエリの後に結果をテーブルに表示できます。

> **Note**
>
> - チュートリアルのこの時点では、データストアが空である可能性があります。空のデータストアで SQL クエリを実行しても結果は返されないため、`__dt` しか表示されないかもしれません。
> - Athena により、実行できるクエリ数が制限されるため、長期間にわたって実行されることがないように、慎重に SQL クエリを合理的なサイズに制限する必要があります。このため、慎重に SQL クエリを合理的なサイズに制限する必要があります。
>
> テスト中にクエリで `LIMIT` 句を使用することをお勧めします。テストが成功したら、この句を削除できます。

5. オプション 指定した時間枠のデータを使用してデータセットコンテンツを作成する場合、一部のデータが処理に間に合わない可能性があります。オフセット、またはデルタを指定すれば遅延を許可できます。詳細については、「Amazon CloudWatch Events を通じた遅延データ通知の取得」を参照してください。

   この時点ではデータ選択フィルターを設定しません。データ選択フィルターの設定ページで 次へ を選択します。

6. オプション このクエリを定期的に実行してデータセットを更新するようにスケジュールできます。データセットスケジュールの作成と編集はいつでも行うことができます。

   この時点では、クエリを繰り返す実行はスケジュールしないため、クエリスケジュールの設定ページでは、次へ を選択します。

7. AWS IoT Analytics は、このデータセットコンテンツのバージョンを作成し、指定された期間の分析結果を保存します。保存期間は 90 日をお勧めしますが、カスタムの保存ポリシーを設定することもできます。また、保存されるデータセットコンテンツバージョンの数を制限することもできます。

   デフォルトのデータセット保存期間を 無期限 として使用し、バージョニング を無効のままにしておくことができます。分析の結果の設定 ページで 次へを選択します。

8. オプション データセット結果の配信ルールを、 AWS IoT Eventsなどの特定の宛先に設定できます。

このチュートリアルの他の段階では結果を提供しないので、データセットコンテンツ配信ルールページで 次へ を選択します。

9. 選択内容を確認してから、データセットの作成 選択します。
10. データセット ページに新しいデータセットが表示されていることを確認します。

を使用してメッセージデータを送信する AWS IoT

クエリ可能なデータストアにデータを保存するパイプラインにデータをルーティングするチャネルがある場合、メッセージデータを AWS IoT Analyticsに送信する準備ができたことになります。次のオプションを使用して AWS IoT Analytics 、 にデータを送信できます。

- AWS IoT メッセージブローカーを使用します。
- AWS IoT Analytics BatchPutMessage API オペレーションを使用します。

次の手順では、 がこのデータ AWS IoT Analytics を取り込むことができるように、 AWS IoT Core コンソールのメッセージブローカーから AWS IoT メッセージデータを送信します。

> **Note**
>
> メッセージのトピック名を作成する場合は、次のことに注意してください。
>
> - トピック名では大文字と小文字が区別されません。同じペイロード内に `example` と `EXAMPLE` という名前のフィールドがある場合は重複と見なされます。
> - トピック名は $ 文字から始めることはできません。$ で始まるトピック名は予約済みのトピックで、 AWS IoTによってのみ使用されます。
> - トピック名に個人を特定できる情報を含めないでください。この情報は暗号化されていない通信やレポートに表示される可能性があります。
> - AWS IoT Core はアカウント AWS または AWS リージョン間でメッセージを送信できません。

を使用してメッセージデータを送信するには AWS IoT

1. AWS IoT コンソール にサインインします。

2. ナビゲーションペインで、テスト を選択し、次に MQTT test client MQTT テストクライアント を選択します。
3. MQTT テストクライアント ページで、トピックへの発行 を選択します。
4. トピック名 で、チャネルの作成時に入力したトピックフィルターに一致する名前を入力します。この例では `update/environment/dht1` を使用します。
5. メッセージペイロード で、次の JSON コンテンツを入力します。

```
{
  "thingid": "dht1",
  "temperature": 26,
  "humidity": 29,
  "datetime": "2018-01-26T07:06:01"
}
```

6. オプション 追加のメッセージプロトコルオプションを使用する場合は 設定を追加 を選択します。
7. 発行 を選択します。

   これにより、チャンネルによってキャプチャされたメッセージが発行されます。その後、パイプラインによりメッセージがデータストアにルーティングされます。

AWS IoT メッセージの進行状況を確認する

次の手順を実行すれば、チャネルにメッセージが取り込まれたことを確認できます。

AWS IoT メッセージの進行状況を確認するには

1. https://console.aws.amazon.com/iotanalytics/ にサインインします。
2. ナビゲーションペインで、チャネル を選択し、前の段階で作成したチャネル名を選択します。
3. データセットを作成 ページで、下にスクロールして モニタリング セクションに到達したら、表示される時間枠 1h 3h 12h 1d 3d 1w を調整します。先週のデータを表示する場合は 1w 1 週間 などのように、値を選択します。

同様の機能を使用して、Pipeline's details パイプラインの詳細 ページでパイプラインアクティビティのランタイムとエラーを監視できます。このチュートリアルでは、アクティビティをパイプラインの一部として指定していないため、ランタイムエラーは表示されません。

パイプラインのアクティビティを監視する方法

1. ナビゲーションペインで、パイプライン を選択し、次に、前に作成したパイプラインの名前を選択します。
2. パイプラインの詳細 ページで、下にスクロールして モニタリング セクションに到達したら、時間枠インジケータ 1h 3h 12h 1d 3d 1w のうちの 1 つを選択することで、表示される時間枠を調整します。

クエリ結果へのアクセス

データセットのコンテンツは、クエリの結果が含まれている CSV 形式のファイルです。

1. https://console.aws.amazon.com/iotanalytics/ で、左側のナビゲーションペインの Datesets データセット を選択します。
2. Datasets データセット ページで、前に作成したデータセットの名前を選択します。
3. データセット情報ページの右上で、今すぐ実行 を選択します。
4. データセットの準備ができているかどうかを確認するには、そのデータセットの下にデータセットのクエリが正常に開始されましたというようなメッセージが表示されているか確認します。Dataset content データセットコンテンツ タブにはクエリ結果が含まれており、Succeeded 成功 が表示されます。
5. 成功したクエリの結果のプレビューを表示するには、データセットコンテンツ タブでそのクエリ名を選択します。クエリ結果を含む CSV ファイルを表示または保存するには、ダウンロード を選択します。

> **Note**
>
> AWS IoT Analytics は、データセットの内容ページに Jupyter Notebook の HTML 部分を埋め込むことができます。詳細については、「コンソールを使用した AWS IoT Analytics データの視覚化」を参照してください。

データの探索

データの保存、分析および可視化についてはいくつかのオプションがあります。

Amazon Simple Storage Service

データセットのコンテンツを Amazon S3 バケットに送信して、既存のデータレイクとの統合、または社内アプリケーションと可視化ツールからのアクセスができます。CreateDataset オペレーションの `contentDeliveryRules::destination::s3DestinationConfiguration` フィールドを確認します。

AWS IoT Events

データセットコンテンツを への入力として送信できます。このサービスでは AWS IoT Events、デバイスやプロセスをモニタリングしてオペレーションの失敗や変更がないか確認し、そのようなイベントが発生したときに追加のアクションを開始できます。

これを行うには、CreateDataset オペレーションを使用してデータセットを作成し、フィールド で AWS IoT Events 入力を指定します`contentDeliveryRules :: destination :: iotEventsDestinationConfiguration :: inputName`。また、 を実行する AWS IoT Analytics アクセス許可を付与するロール`roleArn`の も指定する必要があります`iotevents:BatchPutMessage`。データセットコンテンツが作成されるたびに、 AWS IoT Analytics は各データセットコンテンツエントリをメッセージとして指定された AWS IoT Events 入力に送信します。たとえば、データセットに次のコンテンツが含まれている場合を考えてみましょう。

```
"what","who","dt"
"overflow","sensor01","2019-09-16 09:04:00.000"
"overflow","sensor02","2019-09-16 09:07:00.000"
"underflow","sensor01","2019-09-16 11:09:00.000"
...
```

次に AWS IoT Analytics 、次のようなフィールドを含むメッセージを送信します。

```
{ "what": "overflow", "who": "sensor01", "dt": "2019-09-16 09:04:00.000" }
```

```
{ "what": "overflow", "who": "sensor02", "dt": "2019-09-16 09:07:00.000" }
```

関心のあるフィールド (、`what`、 の 1 つ以上`dt`) を認識する AWS IoT Events 入力を作成し`who`、イベントでこれらの入力フィールドを使用してアクションをトリガーするか、内部変数を設定するディテクターモデルを作成します AWS IoT Events 。

Jupyter Notebook

Jupyter Notebook は、アドホックデータ検索と高度な分析を実行するためのたオープンソースのソリューションです。IoT デバイスデータに関して、より複雑な分析を深く掘り下げて適用でき、予測のための k 平均クラスタリングモデルや回帰モデルなどの機械学習手法を使用できます。

AWS IoT Analytics は、Amazon SageMaker AI ノートブックインスタンスを使用して Jupyter Notebooks をホストします。ノートブックインスタンスを作成する前に、 AWS IoT Analytics と Amazon SageMaker AI の関係を作成する必要があります。

1. SageMaker AI コンソールに移動し、ノートブックインスタンスを作成します。
   a. 詳細を入力し、新しいロールの作成 を選択します。ロールの ARN を書き留めておきます。
   b. ノートブックインスタンスを作成します。
2. IAM コンソールに移動し、SageMaker AI ロールを変更します。
   a. このロールを開きます。管理ポリシーが 1 つあります。
   b. インラインポリシーの追加を選択し、サービス で IotAnalytics を選択します。アクションの選択 を選択し、検索ボックスに **`GetDatasetContent`** を入力して選択します。ポリシーのレビュー を選択します。
   c. ポリシーが正しいことを確認し、名前を入力したら、ポリシーの作成 を選択します。

これにより、新しく作成されたロールにデータセットを読み取るアクセス許可が付与されます AWS IoT Analytics。

1. https://console.aws.amazon.com/iotanalytics/ に戻り、左側のナビゲーションペインの ノートブック を選択します。ノートブック ページで、ノートブックの作成 を選択します。
2. テンプレートの選択 ページで IoTA ブランクテンプレート を選択します。
3. ノートブックのセットアップ ページで、ノートブックの名前を入力します。データセットソースの選択 で、前に作成したデータセットを選択します。「ノートブックインスタンスの選択」で、SageMaker AI で作成したノートブックインスタンスを選択します。
4. 選択内容を確認したら、ノートブックの作成 を選択します。
5. ノートブックページで、ノートブックインスタンスが Amazon SageMaker AI コンソールで開きます。

ノートブックテンプレート

AWS IoT Analytics ノートブックテンプレートには AWS 、 AWS IoT Analytics ユースケースの開始に役立つ、作成された機械学習モデルとビジュアライゼーションが含まれています。これらのノートブックテンプレートを使えば詳細を調べることができ、再利用すればIoTデバイスデータへの適合と即値の獲得を実現できます。

AWS IoT Analytics コンソールには、次のノートブックテンプレートがあります。

- コンテキスト異常の検出 – ポアソン指数加重移動平均 PEWMA モデルを用いて風速測定値のコンテキスト異常を検出するアプリケーションです。
- ソーラーパネルの出力予測 – ソーラーパネルの出力を予測するための、区分時系列モデル、季節時系列モデル、線形時系列モデルのアプリケーションです。
- ジェットエンジンの予知保守 – ジェットエンジンの故障を予測するための多変量長期短期記憶 LSTM ニューラルネットワークおよびロジスティック回帰のアプリケーションです。
- スマートホームの顧客の区分 – スマートホーム使用状況データにおける様々な顧客セグメントを検出するための k-平均および PCA 分析 PCA のアプリケーションです。
- スマートシティの混雑予測 – 都市高速道路の利用率を予測するための LSTM のアプリケーションです。
- スマートシティの大気汚染予測 – 都市中央部の大気汚染を予測するための LSTM のアプリケーションです。

の開始方法 AWS IoT Analytics

このセクションでは、 を使用してデバイスデータを収集、保存、処理、クエリするために使用する基本的なコマンドについて説明します AWS IoT Analytics。ここに示す例では、 AWS Command Line Interface () を使用していますAWS CLI。の詳細については AWS CLI、AWS Command Line Interface 「 ユーザーガイド」を参照してください。で使用できる CLI コマンドの詳細については AWS IoT、 AWS Command Line Interface リファレンスの「iot」を参照してください。

> **Important**
>
> `aws iotanalytics` コマンドを使用して、 AWS IoT Analytics を使用して を操作します AWS CLI。 AWS CLIを使用して IoT システムの他の部分を操作するには、`aws iot` コマンドを使用します。

> **Note**
>
> 以下の例に AWS IoT Analytics エンティティの名前 (チャネル、データセット、データストア、パイプライン) を入力すると、使用する大文字はシステムによって自動的に小文字に変更されることに注意してください。エンティティの名前は小文字で始まり、小文字、下線および数字のみを含める必要があります。

チャンネルの作成

チャネルは、このデータをパイプラインに公開する前に、raw、未処理のメッセージデータを収集してアーカイブします。受信メッセージはチャネルに送信されるため、最初のステップとして、データ用のチャネルを作成します。

```
aws iotanalytics create-channel --channel-name mychannel
```

AWS IoT メッセージを取り込みたい場合は AWS IoT Analytics、このチャネルにメッセージを送信する AWS IoT ルールエンジンルールを作成できます。これについては後ほど へのデータの取り込み AWS IoT Analytics で説明します。のデータをチャネルに取得するもう 1 つの方法は、 AWS IoT Analytics コマンド を使用することです`BatchPutMessage`。

作成済みのチャネルを一覧表示する方法:

```
aws iotanalytics list-channels
```

チャネルに関する追加情報を取得する方法。

```
aws iotanalytics describe-channel --channel-name mychannel
```

未処理のチャネルメッセージは、 が管理する Amazon S3 バケット AWS IoT Analytics、またはユーザーが管理するバケットに保存されます。`channelStorage` パラメータを使用して、どちらかを指定します。デフォルトは、サービスにより管理されている Amazon S3 バケットです。管理する Amazon S3 バケットにチャネルメッセージを保存する場合は、ユーザーに代わって Amazon S3 バケットでこれらのアクションを実行する AWS IoT Analytics アクセス許可を付与する必要があります。 `s3:GetBucketLocation` (バケットの場所の検証) (ストア) `s3:PutObject`、 (`s3:GetObject`読み取り) `s3:ListBucket`、 (再処理) 。

Example

```
{
    "Version": "2012-10-17",
    "Id": "MyPolicyID",
    "Statement": [
        {
            "Sid": "MyStatementSid",
            "Effect": "Allow",
            "Principal": {
                "Service": "iotanalytics.amazonaws.com"
            },
            "Action": [
                "s3:GetObject",
                "s3:GetBucketLocation",
                "s3:ListBucket",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::my-iot-analytics-bucket",
                "arn:aws:s3:::my-iot-analytics-bucket/*"
            ]
        }
    ]
}
```

カスタマー管理のチャネルストレージのオプションまたはアクセス許可を変更する場合、以前に取り込まれたデータがデータセットコンテンツに含まれるようにチャネルデータを再処理しなければならない可能性があります。「チャネルデータの再処理」を参照してください。

データストアの作成

データストアは、メッセージを受信して保存します。データストアはデータベースではなく、メッセージのスケーラブルでクエリ可能なリポジトリです。複数のデータストアを作成して、異なるデバイスや場所からのメッセージを保存したり、単一のデータストアを使用してすべての AWS IoT メッセージを受信したりできます。

```
aws iotanalytics create-datastore --datastore-name mydatastore
```

作成済みのデータストアを一覧表示する方法。

```
aws iotanalytics list-datastores
```

データストアに関する追加情報を取得する方法。

```
aws iotanalytics describe-datastore --datastore-name mydatastore
```

AWS IoT Analytics リソースの Amazon S3 ポリシー

処理されたデータストアメッセージは、 によって管理される Amazon S3 バケット、 AWS IoT Analytics またはユーザーが管理するバケットに保存できます。データストアを作成するときは、`datastoreStorage` API パラメータを使用して必要な Amazon S3 バケットを選択します。デフォルトは、サービスにより管理されている Amazon S3 バケットです。

管理する Amazon S3 バケットにデータストアメッセージを保存する場合は、Amazon S3 バケットでこれらのアクションを実行する AWS IoT Analytics アクセス許可を付与する必要があります。

- `s3:GetBucketLocation`
- `s3:PutObject`
- `s3:DeleteObject`

データストアを SQL クエリデータセットのソースとして使用する場合は、バケットのコンテンツで Amazon Athena クエリを呼び出す AWS IoT Analytics アクセス許可を付与する Amazon S3 バケットポリシーを設定します。 Amazon Athena

> **Note**
>
> 混乱した代理のセキュリティ上の問題を防止するために、バケットポリシーで aws:SourceArn を指定することをお勧めします。これにより、指定したアカウントからのリクエストのみを許可するようにアクセスを制限されます。混乱した代理に関する問題の詳細については、「the section called “サービス間の混乱した代理の防止”」を参照してください。

これらの必要なアクセス許可を付与するバケットポリシーの例を次に示します。

```
{
    "Version": "2012-10-17",
    "Id": "MyPolicyID",
    "Statement": [
        {
            "Sid": "MyStatementSid",
            "Effect": "Allow",
            "Principal": {
                "Service": "iotanalytics.amazonaws.com"
            },
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads",
                "s3:ListMultipartUploadParts",
                "s3:AbortMultipartUpload",
                "s3:PutObject",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ],
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": [
                        "arn:aws:iotanalytics:us-east-1:123456789012:dataset/DOC-
EXAMPLE-DATASET",
                        "arn:aws:iotanalytics:us-east-1:123456789012:datastore/DOC-
EXAMPLE-DATASTORE"
```

```
                    ]
                }
            }
        }
    ]
}
```

詳細については、Amazon Athena ユーザーガイドの「クロスアカウントアクセス」を参照してください。

> **Note**
>
> カスタマー管理のチャネルストレージのオプションまたはアクセス許可を変更する場合、以前に取り込まれたデータがデータセットコンテンツに含まれるようにチャネルデータを再処理しなければならない可能性があります。詳しくは、「チャンネルデータの再処理」を参照してください。

ファイル形式

AWS IoT Analytics データストアは現在、JSON ファイル形式と Parquet ファイル形式をサポートしています。デフォルトのファイル形式は JSON です。

- JSON (JavaScript Object Notation)-名前と値のペアと順序付けられた値リストをサポートするテキスト形式。
- Apache Parquet - 大量のデータを効率的に格納およびクエリするために使用する列指向ストレージ形式。

AWS IoT Analytics データストアのファイル形式を設定するには、データストアの作成時に `FileFormatConfiguration` オブジェクトを使用できます。

`fileFormatConfiguration`

ファイル形式の設定情報が含まれています。 AWS IoT Analytics データストアは JSON と Parquet をサポートしています。

デフォルトのファイル形式は JSON です。指定できるフォーマットは 1 つだけです。データストアを作成した後は、ファイル形式を変更することはできません。

`jsonConfiguration`

JSON 形式の構成情報が含まれます。

`parquetConfiguration`

Parquet 形式の構成情報が含まれます。

`schemaDefinition`

スキーマを定義するために必要な情報。

`columns`

データを保存する 1 つまたは複数の列を指定します。

各スキーマには、最大 100 個の列を設定できます。各列には、最大 100 個のネストされたタイプを含めることができます

`name`

列の名前。

長さの制約: 1 ~ 255 文字。

`type`

データのタイプ。サポートされているデータ型の詳細については、AWS Glue デベロッパーガイドの「共通のデータ型」を参照してください。

長さの制約: 1 ~ 131072 文字。

AWS IoT Analytics は、Amazon Athena のデータ型ページにリストされているすべてのデータ型をサポートします。ただし、 `DECIMAL(precision, scale)` - は除きます`precision`。

データストアを作成する (コンソール)

次の手順は、データが Parquet 形式で保存されるデータストアを作成する方法です。

データストアを作成する方法

1. https://console.aws.amazon.com/iotanalytics/ にサインインします。
2. ナビゲーションペインで、[Data stores] (データストア) を選択します。

3. [Data stores] (データストア) ページで、[Create data store] (データストアの作成) を選択します。
4. データストア詳細の指定ページで、データストアに関する基本情報を入力します。
   a. データストア ID に、一意のデータストア ID を入力します。この ID は作成後には変更できません。
   b. オプショナルタグ で 新しいタグを追加を選択すれば、データストアに 1 つまたは複数のカスタムタグ キーと値のペア を追加できます。タグを使用すると、 AWS IoT Analyticsのために作成するリソースを識別することができます。
   c. [Next (次へ)] を選択します。
5. ストレージタイプの設定 ページで、データの保存方法を指定します。
   a. ストレージタイプ で、サービス管理ストレージ を選択します。
   b. 処理されたデータを保持する期間を設定します 処理データの保存期間の設定 で、無期限 を選択します。
   c. [Next (次へ)] を選択します。
6. [Configure data format] (データ形式の設定) ページで、データレコードの構成と形式を定義します。
   a. [Classification] (分類) で、[Parquet] を選択します。データストアを作成した後でこのファイル形式を変更することはできません。
   b. [Inference source] (推論ソース) で、自分のデータストアに対して [JSON string] (JSON 文字列) を選択します。
   c. [String] (文字列) で、スキーマを JSON 形式で入力します。以下はその例です。

```
{
    "device_id": "0001",
    "temperature": 26,
    "humidity": 29,
    "datetime": "2018-01-26T07:06:01"
}
```

   d. [Infer schema] (スキーマの推測) を選択します。
   e. [Configure Parquet schema] (Parquet スキーマの設定) で、その形式が JSON 例に一致しているか確認します。形式が一致しない場合は、Parquet スキーマを手動で更新します。
      - スキーマでより多くの列を表示する場合は、[Add new column] (新しい列の追加) を選択して列名を入力し、データ型を選択します。

> **Note**
> デフォルトでは、スキーマに 100 列を設定できます。詳細については、AWS IoT Analytics クォータを参照してください。

- 既存の列のデータ型は変更が可能です。サポートされているデータ型の詳細については、AWS Glue デベロッパーガイドの「共通のデータ型」を参照してください。

> **Note**
> データストアを作成した後で、既存の列のデータ型を変更することはできません。

- 既存の列を削除するには、[Remove column] (列の削除) を選択します。

f. [Next (次へ)] を選択します。

7. (オプション) はデータストア内のカスタムパーティション AWS IoT Analytics をサポートしているため、プルーニングされたデータをクエリしてレイテンシーを改善できます。サポートされるカスタムパーティションの詳細については、「カスタムパーティション」を参照してください。

   [Next (次へ)] を選択します。

8. [Review and create] (確認して作成) ページで、選択内容を認し、[Create data store] (データストアの作成) を選択します。

> **Important**
> データストアを作成した後で、列のデータストア ID、ファイル形式、データ型を変更することはできません。

9. [Data stores] (データストア) ページに新しいデータストアが表示されていることを確認します。

カスタムパーティション

AWS IoT Analytics はデータのパーティション化をサポートしているため、データストア内のデータを整理できます。データのパーティション化を使用してデータを整理すると、プルーニングされた

データをクエリできます。これにより、クエリごとにスキャンされるデータの量が減少し、レイテンシーが改善されます。

メッセージデータ属性、またはパイプラインアクティビティを通じて追加された属性に従って、データをパーティション化することができます。

開始するには、データストアでデータのパーティション化を有効にします。1 つ以上のデータパーティションディメンションを指定し、パーティション化されたデータストアを AWS IoT Analytics パイプラインに接続します。次に、`WHERE` 句を使用するクエリを書き込んでパフォーマンスを最適化します。

データストアを作成する (コンソール)

次の手順は、カスタムパーティションを使用してデータストアを作成する方法を示します。

データストアを作成する方法

1. AWS IoT Analytics コンソール にサインインします。
2. ナビゲーションペインで、[Data stores] (データストア) を選択します。
3. [Data stores] (データストア) ページで、[Create data store] (データストアの作成) を選択します。
4. データストア詳細の指定ページで、データストアに関する基本情報を入力します。
   a. データストア ID に、一意のデータストア ID を入力します。この ID は作成後には変更できません。
   b. オプショナルタグ で 新しいタグを追加を選択すれば、データストアに 1 つまたは複数のカスタムタグ キーと値のペア を追加できます。タグは、作成するリソースを識別するのに役立ちます AWS IoT Analytics。
   c. [Next (次へ)] を選択します。
5. ストレージタイプの設定 ページで、データの保存方法を指定します。
   a. ストレージタイプ で、サービス管理ストレージ を選択します。
   b. 処理されたデータを保持する期間を設定します 処理データの保存期間の設定 で、無期限 を選択します。
   c. [Next (次へ)] を選択します。
6. [Configure data format] (データ形式の設定) ページで、データレコードの構成と形式を定義します。

a. データストアのデータ形式 [Classification] (分類) で、[JSON] または [Parquet] を選択します。 AWS IoT Analytics サポートされているファイルタイプの詳細については、「」を参照してください[ファイル形式](#)。

> **Note**
> データストアを作成した後でこのファイル形式を変更することはできません。

b. [Next (次へ)] を選択します。

7. このデータストアのカスタムパーティションを作成します。

a. [Add data partitions] (データパーティションの追加) で [Enable] (有効) を選択します。

b. [Data partition source] (データパーティションのソース) で、パーティションのソースに関する基本情報を指定します。

サンプルソースを選択し、このデータストアのメッセージを収集する AWS IoT Analytics チャネルを選択します。

c. [Message sample attributes] (メッセージサンプル属性)で、データストアのパーティションに使用するメッセージ属性を選択します。次に、[Actions] (アクション) で、選択項目を属性パーティションディメンションまたはタイムスタンプパーティションディメンションとして追加します。

> **Note**
> データストアに追加できるタイムスタンプパーティションは 1 つだけです。

d. [Custom data store partition dimensions] (カスタムのデータストアパーティションディメンション) で、パーティションディメンションに関する基本情報を定義します。前のステップで選択した各メッセージサンプル属性は、パーティションのディメンションになります。次のオプションを使用して、各ディメンションをカスタマイズします。

- [Partition type] (パーティションタイプ) - このパーティションディメンションのパーティションタイプが [Attribute] (属性) であるか、または [Timestamp] (タイムスタンプ) であるかを指定します。
- 属性名とディメンション名 - デフォルトでは、 AWS IoT Analytics は属性パーティションディメンションの識別子として選択したメッセージサンプル属性の名前を使用します。属

性名を編集して、パーティションディメンションの名前をカスタマイズします。WHERE 句のディメンション名を使用してクエリのパフォーマンスを最適化することができます。

- パーティション属性ディメンションの名前には接頭辞 __partition_ が付きます。
- タイムスタンプパーティションタイプの場合、 は、、__year、、 という名前の次の __month4 つのディメンション AWS IoT Analytics を作成します__day__hour。

- [Ordering] (順序指定) - パーティションディメンションを再配置してクエリのレイテンシーを改善します。

[Timestamp format] (タイムスタンプ形式) では、メッセージデータのタイムスタンプに合わせることで、タイムスタンプパーティションの形式を指定します。 AWS IoT Analytics リストされている形式オプションのいずれかを選択するか、データの形式に一致する形式を指定できます。Date Time Formatter の指定の詳細を確認してください。

メッセージ属性ではない新しいディメンションを追加するには、[Add new partitions] (新しいパーティションの追加) を選択します。

e. [Next (次へ)] を選択します。

8. [Review and create] (確認して作成) ページで、選択内容を認し、[Create data store] (データストアの作成) を選択します。

> **Important**
>
> - データストアを作成した後は、データストア ID を変更することはできません。
> - 既存のパーティションを編集するには、別のデータストアを作成し、パイプラインを介してデータを再処理する必要があります。

9. [Data stores] (データストア) ページに新しいデータストアが表示されていることを確認します。

パイプラインの作成

パイプラインは、チャネルからのメッセージを消費し、メッセージをデータストアに保存する前にメッセージを処理、およびフィルタリングすることができます。データストアにチャネルを接続するには、パイプラインを作成します。最も簡単なパイプラインには、データを収集するチャネルを指定するアクティビティと、メッセージの送信先のデータストアを識別するアクティビティのみが含まれます。より複雑なパイプラインの詳細については、「パイプラインアクティビティ」を参照してください。

最初に、データストアにチャネルを接続するだけのパイプラインを作成することをお勧めします。次に、raw データがデータストアに流れていることを確認した後、このデータを処理する追加のパイプラインアクティビティを導入できます。

次のコマンドを実行してパイプラインを作成します。

```
aws iotanalytics create-pipeline --cli-input-json file://mypipeline.json
```

`mypipeline.json` ファイルには次のコンテンツが含まれます。

```
{
    "pipelineName": "mypipeline",
    "pipelineActivities": [
        {
            "channel": {
                "name": "mychannelactivity",
                "channelName": "mychannel",
                "next": "mystoreactivity"
            }
        },
        {
            "datastore": {
                "name": "mystoreactivity",
                "datastoreName": "mydatastore"
            }
        }
    ]
}
```

既存のパイプラインを一覧表示するには次のコマンドを実行します。

```
aws iotanalytics list-pipelines
```

個別のパイプラインの設定を表示するには次のコマンドを実行します。

```
aws iotanalytics describe-pipeline  --pipeline-name mypipeline
```

へのデータの取り込み AWS IoT Analytics

クエリ可能なデータストアにデータを保存するパイプラインにデータをルーティングするチャネルがある場合、メッセージデータを AWS IoT Analyticsに送信する準備ができたことになります。ここで

は、 AWS IoT Analyticsにデータを取得する 2 つの方法を示します。メッセージブローカー AWS IoT Analytics または `BatchPutMessage` API を使用して AWS IoT メッセージを送信できます。

トピック

- AWS IoT メッセージブローカーの使用
- BatchPutMessage API の使用

AWS IoT メッセージブローカーの使用

AWS IoT メッセージブローカーを使用するには、ルールエンジンを使用して AWS IoT ルールを作成します。このルールは、特定のトピックを含むメッセージを にルーティングします AWS IoT Analytics。ただし、このルールでは、まず必要なアクセス許可を付与するロールを作成する必要があります。

IAM ロールの作成

AWS IoT メッセージを AWS IoT Analytics チャネルにルーティングするには、ルールを設定します。ただし、まず、メッセージデータを AWS IoT Analytics チャネルに送信するアクセス許可をそのルールに付与する IAM ロールを作成する必要があります。

次のコマンドを実行してロールを作成します。

```
aws iam create-role --role-name myAnalyticsRole --assume-role-policy-document file://
arpd.json
```

`arpd.json` ファイルの内容は次のようになります。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "iot.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

次に、ポリシードキュメントをロールにアタッチします。

```
aws iam put-role-policy --role-name myAnalyticsRole --policy-name myAnalyticsPolicy --
policy-document file://pd.json
```

`pd.json` ファイルの内容は次のようになります。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iotanalytics:BatchPutMessage",
            "Resource": [
               "arn:aws:iotanalytics:us-west-2:your-account-number:channel/mychannel"
            ]
        }
    ]
}
```

AWS IoT ルールの作成

チャネルにメッセージを送信する AWS IoT ルールを作成します。

```
aws iot create-topic-rule --rule-name analyticsTestRule --topic-rule-payload file://
rule.json
```

`rule.json` ファイルの内容は次のようになります。

```
{
    "sql": "SELECT * FROM 'iot/test'",
    "ruleDisabled": false,
    "awsIotSqlVersion": "2016-03-23",
    "actions": [ {
        "iotAnalytics": {
            "channelName": "mychannel",
            "roleArn": "arn:aws:iam::your-account-number:role/myAnalyticsRole"
        }
    } ]
}
```

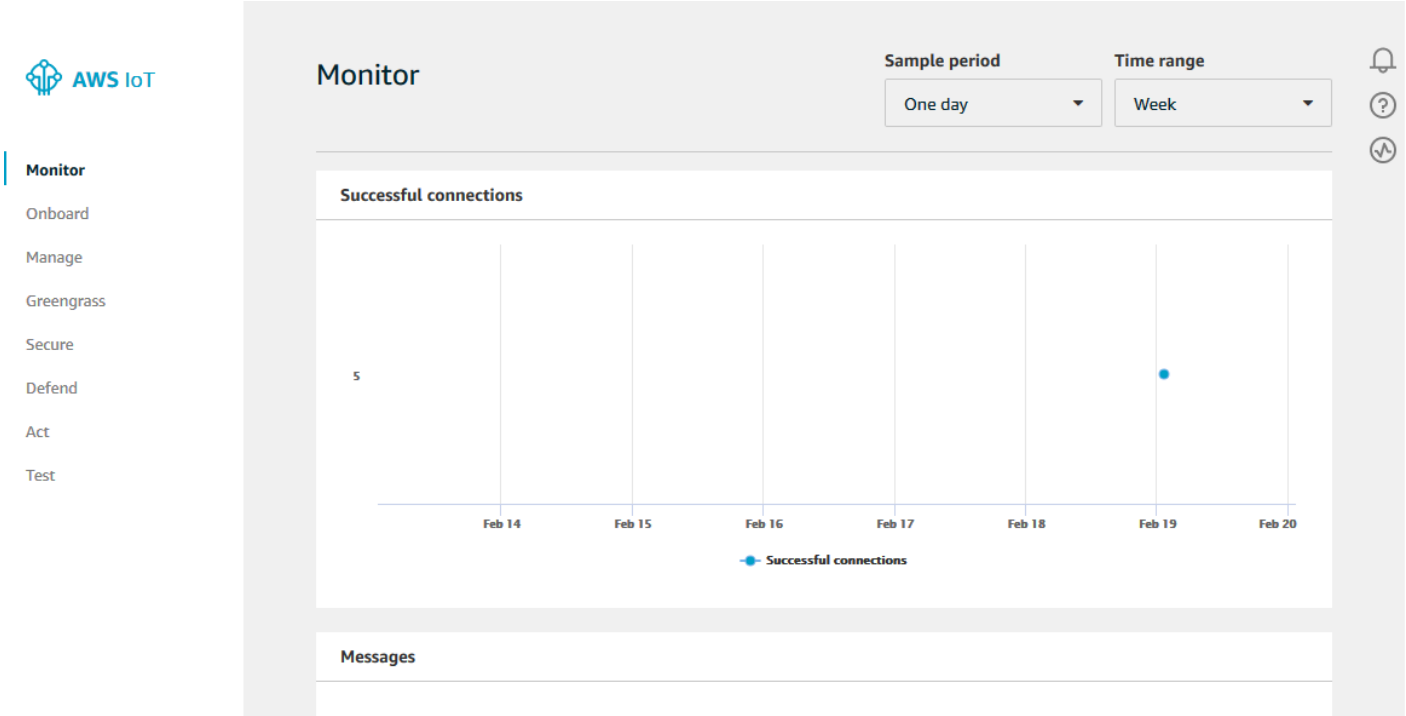
`iot/test` をルーティングする必要があるメッセージの MQTT トピックに置き換えます。チャネル名とロールを前のセクションで作成したものに置き換えます。

への MQTT メッセージの送信 AWS IoT Analytics

ルールをチャネル、チャネルをパイプライン、パイプラインをデータストアに結合すると、ルールに一致するすべてのデータがクエリ可能なデータストア AWS IoT Analytics に流れるようになりました。これをテストするには、 AWS IoT コンソールを使用してメッセージを送信します。

> **Note**
>
> 送信先のメッセージペイロード (データ) のフィールド名 AWS IoT Analytics。
>
> - 英数字およびアンダースコア (_) のみを使用することができます。他の特殊文字を使用することはできません。
> - 先頭は、英字または 1 つの下線 (_) にする必要があります。
> - ハイフン (-) を含めることはできません。
> - 正規表現では次の通りです: `"^[A-Za-z_]([A-Za-z0-9]*|[A-Za-z0-9][A-Za-z0-9_]*)$"`
> - 255 文字を超えることはできません。
> - 大文字と小文字は区別されます。同じペイロード内に `foo` と `FOO` という名前のフィールドがある場合は重複と見なされます。
>
> たとえば、メッセージペイロードでは `{"temp_01": 29}` や `{"_temp_01": 29}` は有効ですが、`{"temp-01": 29}`、`{"01_temp": 29}` や `{"__temp_01": 29}` は無効です。

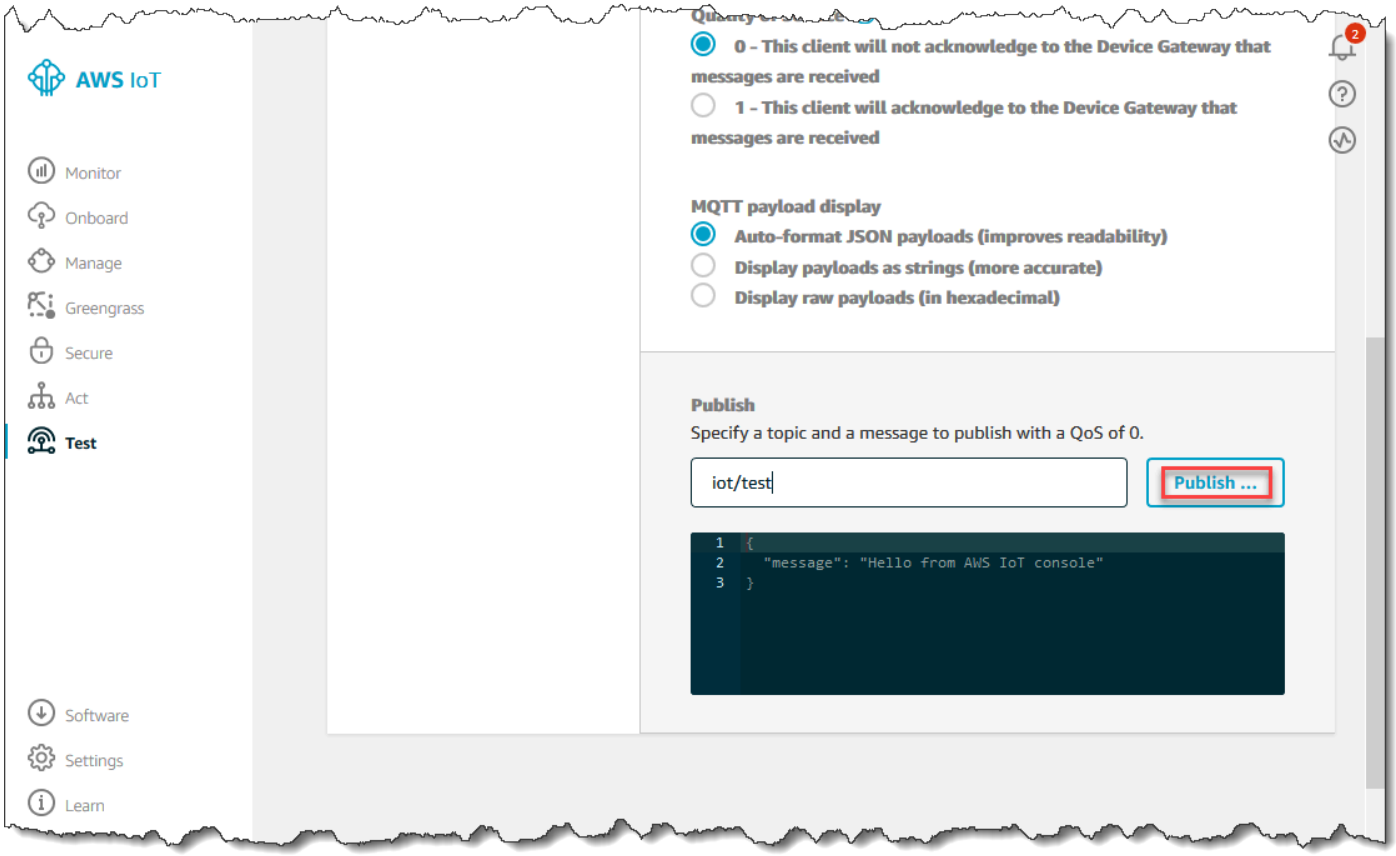
1. AWS IoT コンソールの、左のナビゲーションペインで、[Test (テスト)] を選択します。



2. MQTT クライアントのページで、[Specify a topic (トピックの指定)] の [Publish (発行)] セクションに「**`iot/test`**」と入力します。メッセージペイロードセクションで、次の JSON の内容が存在しているか確認し、存在しない場合は入力します。

```
{
    "message": "Hello from the IoT console"
}
```

3. [Publish to topic] (トピックに公開) を選択します。



これでメッセージが発行され、前に作成したデータストアにルーティングされます。

BatchPutMessage API の使用

メッセージデータを取得するもう 1 つの方法は、 `BatchPutMessage` API コマンドを使用すること AWS IoT Analytics です。この方法では、特定のトピックを持つメッセージをチャネルにルーティングする AWS IoT ルールを設定する必要はありません。ただし、データ/メッセージをチャネルに送信するデバイスが AWS SDK で作成されたソフトウェアを実行できること、または を使用して AWS CLI を呼び出すことができる必要があります`BatchPutMessage`。

1. 送信するメッセージが含まれているファイル `messages.json` を作成します (この例では、1 つのメッセージだけが送信されます)。

```
[
    { "messageId": "message01", "payload": "{ \"message\": \"Hello from the CLI
\" }" }
]
```

2. `batch-put-message` コマンドを実行します。

```
aws iotanalytics batch-put-message  --channel-name mychannel  --messages file://
messages.json --cli-binary-format raw-in-base64-out
```

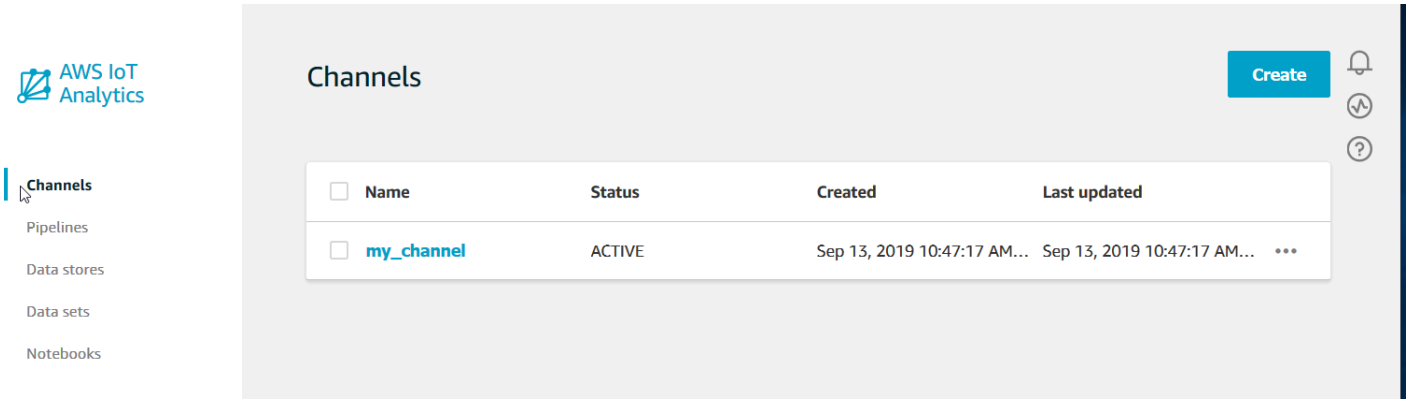
エラーがなければ次の出力が表示されます。

```
{
    "batchPutMessageErrorEntries": []
}
```

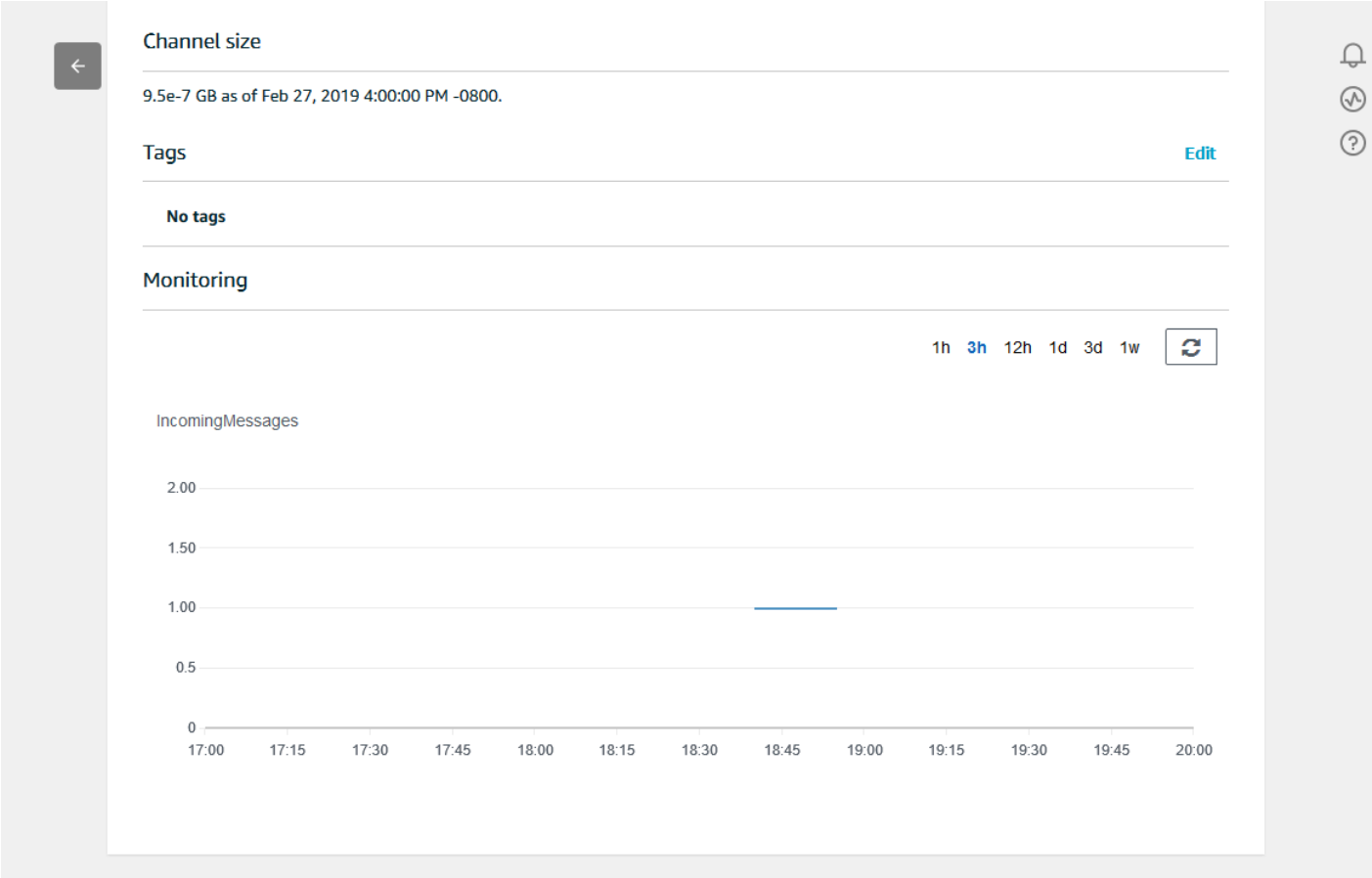
取り込まれたデータのモニタリング

AWS IoT Analytics コンソールを使用して、送信したメッセージがチャネルに取り込まれていることを確認できます。

1. AWS IoT Analytics コンソールの左のナビゲーションペインで、[Prepare] (準備) を選択し、(必要に応じて) [Channel] (チャネル) を選択し、前に作成したチャネルの名前を選択します。

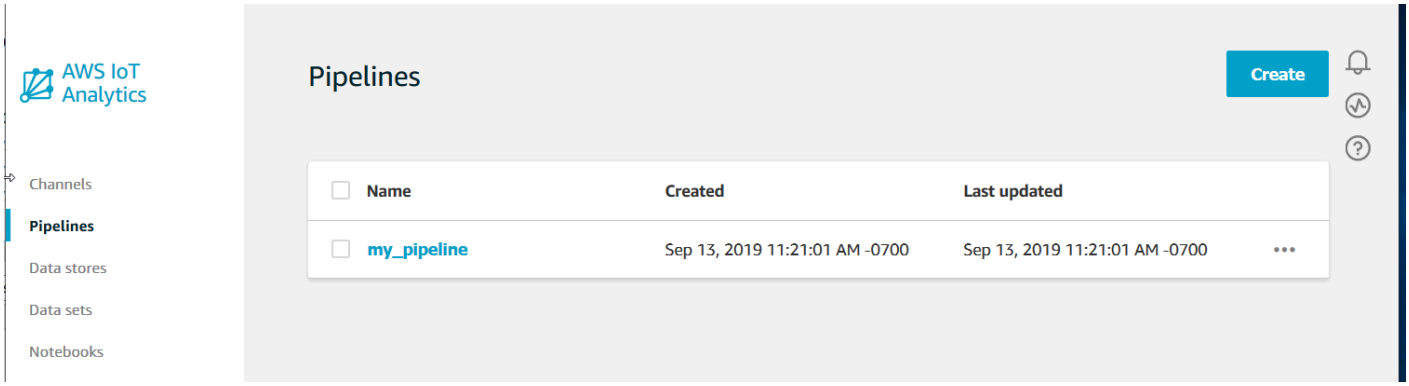


2. チャネルの詳細ページで、[Monitoring (モニタリング)] セクションまで下にスクロールします。必要に応じて、時間ウィンドウインジケータ [1h 3h 12h 1d 3d 1w] (1 時間 3 時間 12 時間 1 日 3 日 1 週間) のいずれかを選択することで表示される時間枠ウィンドウを調整します。指定された時間ウィンドウ内にチャネルに取り込まれたメッセージの数を示す線グラフが表示されます。



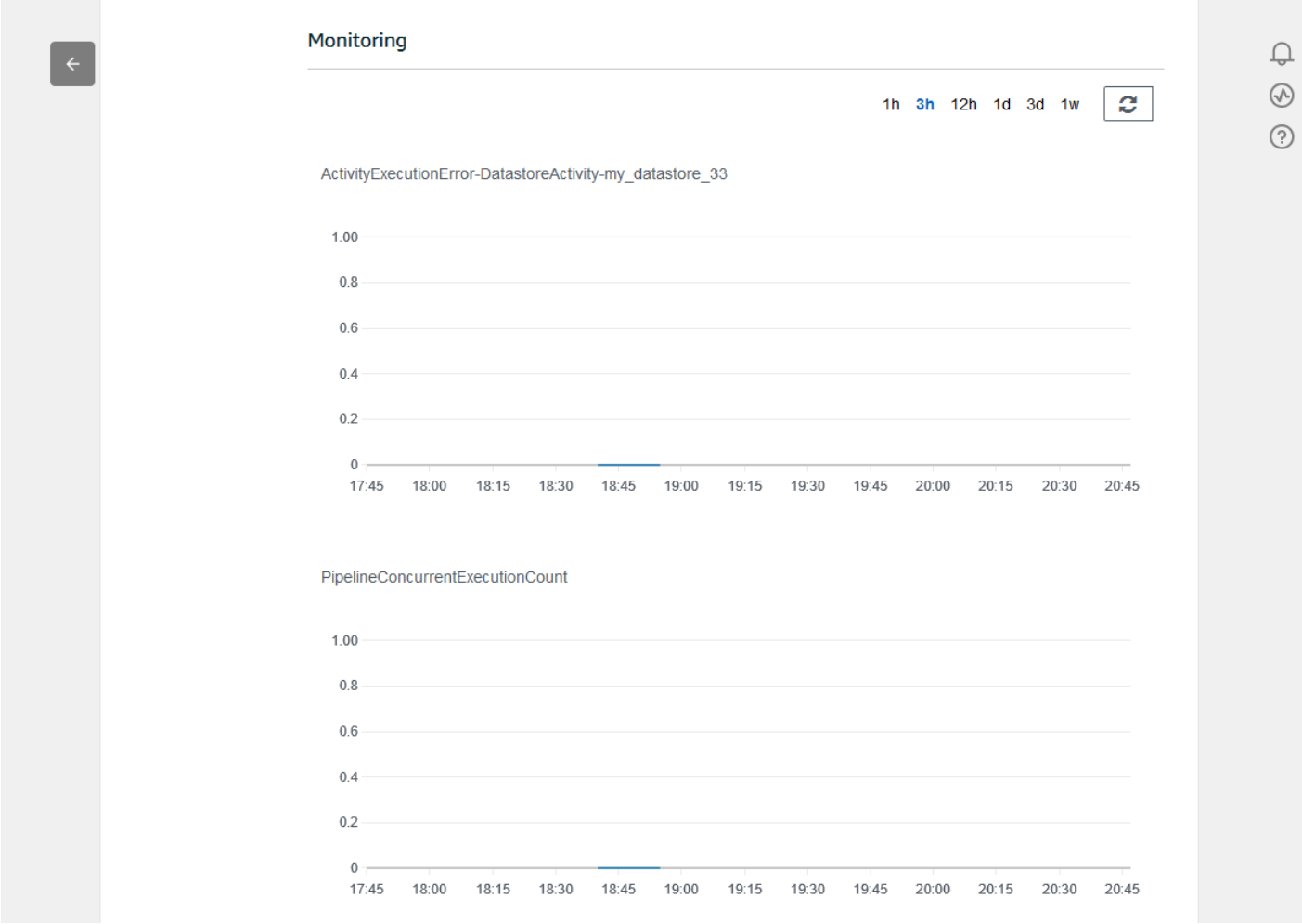
パイプラインのアクティビティの実行を確認する同様のモニタリング機能があります。パイプラインの詳細ページでアクティビティ実行エラーをモニタリングできます。アクティビティをパイプラインの一部として指定していない場合、実行エラー 0 が表示されます。

1. AWS IoT Analytics コンソールの左のナビゲーションペインで、[Prepare] (準備) を選択し、次に [Pipelines] (パイプライン) を選択し、前に作成したパイプラインの名前を選択します。



2. パイプラインの詳細ページで、[Monitoring (モニタリング)] セクションまで下にスクロールします。必要に応じて、時間ウィンドウインジケータ [1h 3h 12h 1d 3d 1w] (1 時間 3 時間 12 時間 1

日 3 日 1 週間) のいずれかを選択することで表示される時間ウィンドウを調整します。指定された時間ウィンドウ内のパイプラインのアクティビティ実行エラーの数を示す線グラフが表示されます。



データセットの作成

データストアからデータを取得するには、SQL データセットまたはコンテナデータセットを作成します。 AWS IoT Analytics はデータをクエリして分析の質問に回答できます。データストアはデータベースではありませんが、SQL 式を使用してデータにクエリを実行すると、生成された結果はデータセットに保存されます。

トピック

- データのクエリ
- クエリされたデータへのアクセス

データのクエリ

データに対してクエリを実行するには、データセットを作成します。データセットには、データストアにクエリを実行するために使用する SQL と、選択した日時にクエリを繰り返すオプションのスケジュールが含まれます。オプションのスケジュールを作成するには、Amazon CloudWatch のスケジュール式に似た式を使用します。

次のコマンドを実行してデータセットを作成します。

```
aws iotanalytics create-dataset --cli-input-json file://mydataset.json
```

`mydataset.json` ファイルには次のコンテンツが含まれます。

```
{
    "datasetName": "mydataset",
    "actions": [
        {
            "actionName":"myaction",
            "queryAction": {
                "sqlQuery": "select * from mydatastore"
            }
        }
    ]
}
```

クエリの実行によりデータセットのコンテンツを作成するには、次のコマンドを実行します。

```
aws iotanalytics create-dataset-content --dataset-name mydataset
```

データセットのコンテンツが作成されるまで数分待ってから、作業を続行してください。

クエリされたデータへのアクセス

クエリ結果は、CSV 形式のファイルとして保存されたデータセットコンテンツです。このファイルは Amazon S3 経由で用意されています。以下の例では、結果の準備が完了してファイルをダウンロードできることを確認する方法を示しています。

次の `get-dataset-content` コマンドを実行します。

```
aws iotanalytics get-dataset-content --dataset-name mydataset
```

データセットにデータが含まれている場合、get-dataset-content の出力は [status] フィールドに「"state": "SUCCEEDED"」と表示されます。以下はその例です。

```
{
    "timestamp": 1508189965.746,
    "entries": [
        {
          "entryName": "someEntry",
          "dataURI": "https://aws-iot-analytics-datasets-f7253800-859a-472c-aa33-
e23998b31261.s3.amazonaws.com/results/f881f855-c873-49ce-abd9-b50e9611b71f.csv?X-Amz-"

        }
    ],
    "status": {
      "state": "SUCCEEDED",
      "reason": "A useful comment."
    }
}
```

dataURI は、出力結果の署名付き URL です。有効期間は短時間 (数時間) です。このコマンドを呼び出すことで新しい署名付き URL が生成されるため、ワークフローによっては、コンテンツにアクセスする前に常に get-dataset-content を呼び出すようにします。

AWS IoT Analytics データの探索

データの保存、分析、視覚化には、いくつかのオプションがあります AWS IoT Analytics 。

このページのトピック

- Amazon S3
- AWS IoT Events
- QuickSight
- Jupyter Notebook

Amazon S3

データセットのコンテンツを Amazon Simple Storage Service (Amazon S3) バケットに送信して、既存のデータレイクとの統合、または社内アプリケーションと可視化ツールからのアクセスができます。CreateDataset のフィールド contentDeliveryRules::destination::s3DestinationConfiguration を確認します。

AWS IoT Events

データセットコンテンツを への入力として送信できます。このサービスでは AWS IoT Events、デバイスやプロセスをモニタリングしてオペレーションの失敗や変更がないか確認し、そのようなイベントが発生したときに追加のアクションをトリガーできます。

これを行うには、CreateDataset を使用してデータセットを作成し、 フィールドで AWS IoT Events 入力を指定しますcontentDeliveryRules :: destination :: iotEventsDestinationConfiguration :: inputName。また、「iotevents:BatchPutMessage」を実行する AWS IoT Analytics アクセス許可を付与するロールroleArnの を指定する必要があります。データセットのコンテンツが作成されるたびに、 は各データセットコンテンツエントリをメッセージとして指定された AWS IoT Events 入力 AWS IoT Analytics に送信します。たとえば、データセットには次のものが含まれている場合:

```
"what","who","dt"
"overflow","sensor01","2019-09-16 09:04:00.000"
"overflow","sensor02","2019-09-16 09:07:00.000"
"underflow","sensor01","2019-09-16 11:09:00.000"
...
```

は、次のようなフィールドを含むメッセージを送信 AWS IoT Analytics します。

```
{ "what": "overflow", "who": "sensor01", "dt": "2019-09-16 09:04:00.000" }
```

```
{ "what": "overflow", "who": "sensor02", "dt": "2019-09-16 09:07:00.000" }
```

と では、関心のあるフィールド (、what、 の 1 つ以上dt) を認識する AWS IoT Events 入力を作成しwho、イベントでこれらの入力フィールドを使用してアクションをトリガーするか、内部変数を設定する AWS IoT Events ディテクターモデルを作成します。

QuickSight

AWS IoT Analytics は QuickSight と直接統合できます。QuickSight は、視覚化の構築、アドホック分析の実行、データからのビジネスインサイトの迅速な取得に使用できる高速ビジネス分析サービスです。QuickSight を使用すると、組織は数十万人のユーザーにスケールし、堅牢なインメモリエンジン (SPICE) を使用して応答性の高いパフォーマンスを実現できます。QuickSight は、これらのリージョンで使用できます。

Jupyter Notebook

AWS IoT Analytics データセットは、高度な分析とデータ探索を実行するために Jupyter Notebook によって直接使用することもできます。Jupyter Notebook はオープンソースのソリューションです。http://jupyter.org/install.html からダウンロードしてインストールできます。Amazon がホストするノートブックソリューションである SageMaker AI との追加統合も利用できます。

複数のバージョンのデータセットを保持する

CreateDataset と UpdateDataset の API を呼び出すときにデータセット `retentionPeriod and versioningConfiguration` フィールドの値を指定することで、保持するデータセットコンテンツのバージョンの数と保持期間を選択できます。

```
...
"retentionPeriod": {
  "unlimited": "boolean",
  "numberOfDays": "integer"
},
"versioningConfiguration": {
  "unlimited": "boolean",
  "maxVersions": "integer"
},
...
```

これらの 2 つのパラメータの設定は連携しており、保持するデータセットコンテンツのバージョン数と期間を次の方法で決定します。

	retentionPeriod [指定されていません]	retentionPeriod: 無制限 = TRUE、numberOfDays = 未設定	retentionPeriod: 無制限 = FALSE、numberOfDays = X
versioningConfiguration: [指定されていません]	最終バージョンと最後に成功したバージョン (異なる場合) のみが、90 日間保持されます。	最終バージョンと最後に成功したバージョン (異なる場合) のみが、無期限に保持されます。	最終バージョンと最後に成功したバージョン (異なる場合) のみが、X 日間保持されます。

versioningConfiguration: 無制限 = TRUE、maxVersions 未設定	数に関係なく、過去 90 日間のすべてのバージョンが保持されます。	保持されるバージョンの数に制限はありません。	数に関係なく、過去 X 日間のすべてのバージョンが保持されます。
versioningConfiguration: 無制限 = FALSE、maxVersions = Y	最大で過去 90 日間の Y バージョンが保持されます。	バージョンの経過日数に関係なく、最大 Y バージョンが保持されます。	最大で過去 X 日間の Y バージョンが保持されます。

メッセージペイロード構文

送信先のメッセージペイロード (データ) のフィールド名 AWS IoT Analytics：

- 英数字およびアンダースコア (_) のみを使用することができます。他の特殊文字を使用することはできません。
- 先頭は、英字または 1 つの下線 (_) にする必要があります。
- ハイフン (-) を含めることはできません。
- 正規表現では次の通りです: `"^[A-Za-z_]([A-Za-z0-9]*|[A-Za-z0-9][A-Za-z0-9_]*)$"`
- 255 文字を超えることはできません。
- 大文字と小文字は区別されます。同一ペイロードにおいてフィールド名の "foo" と "FOO" は重複と見なされます。

たとえば、メッセージペイロードでは `{"temp_01": 29}` や `{"_temp_01": 29}` は有効ですが、`{"temp-01": 29}`、`{"01_temp": 29}` や `{"__temp_01": 29}` は無効です。

AWS IoT SiteWise データの使用

AWS IoT SiteWise は、産業機器から大規模にデータを収集、モデル化、分析、視覚化するために使用できるマネージドサービスです。このサービスでは、産業用のデバイス、プロセス、および施設の表現を構築するためのアセットモデリングフレームワークを提供します。

AWS IoT SiteWise アセットモデルを使用すると、消費する産業機器データと、データを複雑なメトリクスに処理する方法を定義できます。 AWS クラウドでデータを収集して処理するようにアセットモデルを設定できます。詳細については、AWS IoT SiteWise ユーザーガイドを参照してください。

AWS IoT Analytics は と統合 AWS IoT SiteWise されているため、 AWS IoT SiteWise データに対して SQL クエリを実行およびスケジュールできます。 AWS IoT SiteWise データのクエリを開始するには、AWS IoT SiteWise 「 ユーザーガイド」の「ストレージ設定の構成」の手順に従ってデータストアを作成します。次に、 AWS IoT SiteWise データを使用してデータセットを作成する (コンソール) または AWS IoT SiteWise データを含むデータセットを作成する (AWS CLI) の手順に従って AWS IoT Analytics データセットを作成し、産業データに対して SQL クエリを実行します。

トピック

- AWS IoT SiteWise データを使用して AWS IoT Analytics データセットを作成する
- データセットコンテンツへのアクセス
- チュートリアル: で AWS IoT SiteWise データをクエリする AWS IoT Analytics

AWS IoT SiteWise データを使用して AWS IoT Analytics データセットを作成する

AWS IoT Analytics データセットには、データストア内のデータのクエリに使用する SQL ステートメントと式と、指定した日時にクエリを繰り返すオプションのスケジュールが含まれています。オプションのスケジュールを作成するには、Amazon CloudWatch のスケジュール式のような式を使用できます。

> **Note**
>
> データセットは、通常、表形式で編成される場合とそうでない場合があるデータの集合です。対照的に、 は、データストア内のデータに SQL クエリを適用してデータセット AWS IoT Analytics を作成します。

AWS IoT SiteWise データのデータセットの作成を開始するには、次の手順に従います。

トピック

- AWS IoT SiteWise データを使用してデータセットを作成する (コンソール)
- AWS IoT SiteWise データを含むデータセットを作成する (AWS CLI)

AWS IoT SiteWise データを使用してデータセットを作成する (コンソール)

以下のステップを使用して、 AWS IoT Analytics コンソールで AWS IoT SiteWise データのデータセットを作成します。

データセットを作成するには

1. https://console.aws.amazon.com/iotanalytics/ で、左側のナビゲーションペインの [Datesets] (データセット) を選択します。
2. [Create dataset] (データセットの作成) ページで [Create SQL] (SQL の作成) を選択します。
3. データセットの詳細を指定 でデータセットの詳細を指定します。
   a. データセットの名前を入力します。
   b. データストアソースで、 AWS IoT SiteWise データストアを識別する一意の ID を選択します。
   c. オプション タグ で、1 つまたは複数のカスタムタグ キーと値のペア をデータセットに追加します。
4. SQL 式を使用して、データをクエリし、分析的な質問に答えます。
   a. [Author query] (クエリの作成) フィールドに、ワイルドカードを使用してデータを最大で 5 行表示する SQL クエリを入力します。

      ```
      SELECT * FROM my_iotsitewise_datastore.asset_metadata LIMIT 5
      ```

      でサポートされている SQL 機能の詳細については AWS IoT Analytics、「」を参照してください の SQL 式 AWS IoT Analytics。または、チュートリアル: で AWS IoT SiteWise データをクエリする AWS IoT Analytics で統計クエリの例を参照すると、データに対するインサイトが得られます。
   b. [Test query] (クエリのテスト) を選択すれば、入力の正誤を確認でき、さらに、クエリの後に結果をテーブルに表示できます。

      > **Note**
      > Amazon Athena は実行中のクエリの最大数を制限するため、SQL クエリを妥当なサイズに制限して、長期間実行しないようにする必要があります。

5. オプション 指定した時間枠のデータを使用してデータセットコンテンツを作成する場合、一部のデータが処理に間に合わない可能性があります。オフセット、またはデルタを指定すれば遅延を許可できます。詳細については、「Amazon CloudWatch Events を通じた遅延データ通知の取得」を参照してください。

   [Configure data selection filter] (データ選択フィルターの設定) ページでデータ選択フィルターを設定した後、[Next] (次へ) を選択します。

6. (オプション) [Set query schedule] (クエリスケジュールの設定) ページでは、このクエリを定期的に実行してデータセットを更新するようにスケジュールできます。データセットスケジュールの作成と編集はいつでも行うことができます。

   > **Note**
   >
   > は 6 AWS IoT SiteWise 時間 AWS IoT Analytics ごとに にデータを取り込みます。6 時間以上の頻度を選択することをお勧めします。

   [Frequency] (頻度) のオプションを選択したら、[Next] (次へ) を選択します。

7. AWS IoT Analytics は、このデータセットコンテンツのバージョンを作成し、指定された期間の分析結果を保存します。保存期間は 90 日をお勧めしますが、カスタムの保存ポリシーを設定することもできます。また、保存されるデータセットコンテンツバージョンの数を制限することもできます。

   [Configure the results of your dataset] (データセットの結果の設定) ページでオプションを選択したら、[Next] (次へ) を選択します。

8. (オプション) データセット結果の配信ルールを、 AWS IoT Eventsなどの特定の宛先に設定できます。

   [Configure dataset content delivery rules] (データセットコンテンツ配信の設定) ページでオプションを選択したら、[Next] (次へ) を選択します。

9. 選択内容を確認してから、[Create dataset] (データセットの作成) を選択します。

10. データセット ページに新しいデータセットが表示されていることを確認します。

AWS IoT SiteWise データを含むデータセットを作成する (AWS CLI)

次の AWS CLI コマンドを実行して、 AWS IoT SiteWise データのクエリを開始します。

ここに示す例では、 AWS Command Line Interface () を使用していますAWS CLI。の詳細については AWS CLI、「 AWS Command Line Interface ユーザーガイド」を参照してください。で使用できる CLI コマンドの詳細については AWS IoT Analytics、 AWS Command Line Interface リファレンスの「iotanalytics」を参照してください。

データセットを作成するには

1. 次の `create-dataset` コマンドを実行してデータセットを作成します。

```
aws iotanalytics create-dataset --cli-input-json file://my_dataset.json
```

`my_dataset.json` ファイルには次のコンテンツが含まれます。

```
{
    "datasetName": "my_dataset",
    "actions": [
        {
            "actionName":"my_action",
            "queryAction": {
                "sqlQuery": "SELECT * FROM my_iotsitewise_datastore.asset_metadata
 LIMIT 5"
            }
        }
    ]
}
```

でサポートされている SQL 機能の詳細については AWS IoT Analytics、「」を参照してください の SQL 式 AWS IoT Analytics。または、チュートリアル: で AWS IoT SiteWise データをクエリする AWS IoT Analytics で統計クエリの例を参照すると、データに対するインサイトが得られます。

2. 次の `create-dataset-content` コマンドを実行して、クエリの実行によりデータセットコンテンツを作成します。

```
aws iotanalytics create-dataset-content --dataset-name my_dataset
```

データセットコンテンツへのアクセス

SQL クエリの結果は、CSV 形式のファイルとして保存されたデータセットコンテンツです。このファイルは Amazon S3 経由で用意されています。以下の手順は、結果の準備が完了してファイルをダウンロードできることを確認する方法を示しています。

トピック

- でデータセットコンテンツにアクセスする AWS IoT Analytics（コンソール）
- AWS IoT Analytics（AWS CLI) でデータセットコンテンツにアクセスする

でデータセットコンテンツにアクセスする AWS IoT Analytics（コンソール）

データセットにデータが含まれている場合は、 AWS IoT Analytics コンソールで SQL クエリ結果をプレビューおよびダウンロードできます。

AWS IoT Analytics データセットの結果にアクセスするには

1. コンソールの [Datasets] (データセット) ページで、アクセスするデータセットの名前を選択します。
2. データセット概要ページで、[Content] (コンテンツ) タブを選択します。
3. [Dataset contents] (データセットのコンテンツ) テーブルで、結果のプレビューまたは結果の csv ファイルのダウンロードを実行するクエリの名前を選択します。

AWS IoT Analytics（AWS CLI) でデータセットコンテンツにアクセスする

データセットにデータが含まれる場合、SQL クエリの結果のプレビューとダウンロードが可能です。

ここに示す例では、 AWS Command Line Interface () を使用していますAWS CLI。の詳細については AWS CLI、「AWS Command Line Interface ユーザーガイド」を参照してください。で使用できる CLI コマンドの詳細については AWS IoT Analytics、 AWS Command Line Interface リファレンスの「iotanalytics」を参照してください。

AWS IoT Analytics データセットの結果にアクセスするには (AWS CLI)

1. クエリの結果を表示するには以下の `get-dataset-content` コマンドを実行します。

```
aws iotanalytics get-dataset-content --dataset-name my_iotsitewise_dataset
```

2. データセットにデータが含まれている場合、`get-dataset-content` の出力は [status] フィールドに「`"state": "SUCCEEDED"`」と表示されます。以下はその例です。

```
{
    "timestamp": 1508189965.746,
    "entries": [
        {
          "entryName": "my_entry_name",
          "dataURI": "https://aws-iot-analytics-datasets-f7253800-859a-472c-aa33-
e23998b31261.s3.amazonaws.com/results/f881f855-c873-49ce-abd9-b50e9611b71f.csv?X-
Amz-"

        }
    ],
    "status": {
      "state": "SUCCEEDED",
      "reason": "A useful comment."
    }
}
```

3. `get-dataset-content` からの出力には `dataURI` が含まれますが、これは出力結果の署名付き URL です。有効期間は短時間 (数時間) です。`dataURI` URL に進んで SQL クエリ結果にアクセスします。

> **Note**
>
> このコマンドを呼び出すことで新しい署名付き URL が生成されるため、ワークフローによっては、コンテンツにアクセスする前に常に `get-dataset-content` を呼び出すようにします。

チュートリアル: で AWS IoT SiteWise データをクエリする AWS IoT Analytics

このチュートリアルでは、 で AWS IoT SiteWise データをクエリする方法を示します AWS IoT Analytics。このチュートリアルでは、風力発電所のデータセットのサンプル AWS IoT SiteWise を提供する のデモのデータを使用します。

> **Important**
> このデモで作成および消費するリソースは有料です。

トピック

- 前提条件
- データのロードおよび検証
- データ探索
- 統計クエリの実行
- チュートリアルリソースのクリーンアップ

前提条件

このチュートリアルでは、以下のリソースが必要になります。

- AWS IoT SiteWise および の使用を開始するには、 AWS アカウントが必要です AWS IoT Analytics。このアカウントをお持ちでない場合は、「AWS アカウントを作成する方法」の手順を実行してください。
- Windows、macOS、Linux、または Unix を実行して、 AWS Management Consoleにアクセスする開発用コンピュータ。詳細については、「AWS Management Consoleの開始方法」を参照してください。
- AWS IoT SiteWise AWS IoT SiteWise モデルとアセットを定義し、風力発電所機器からデータを表すデータをストリーミングするデータ。データを作成するには、「 AWS IoT SiteWise ユーザーガイド」の AWS IoT SiteWise 「デモの作成」の手順に従います。
- 管理する既存のデータストアの AWS IoT SiteWise デモ風力発電所の機器データ。 AWS IoT SiteWise データのデータストアを作成する方法の詳細については、「 AWS IoT SiteWise ユーザーガイド」の「ストレージ設定の構成」を参照してください。

> **Note**
> AWS IoT SiteWise メタデータは作成後すぐに AWS IoT SiteWise データストアに表示されますが、raw データが表示されるまでに最大 6 時間かかる場合があります。その間、AWS IoT Analytics データセットを作成し、メタデータに対してクエリを実行できます。

次のステップ

データのロードおよび検証

データのロードおよび検証

このチュートリアルでクエリするデータは、風力発電所の風力エンジンタービンをモデル化する AWS IoT SiteWise データセットのサンプルです。

> **Note**
>
> このチュートリアルでは、データストア内の 3 つのテーブルをクエリします。
>
> - `raw` - 各アセットの未処理の生データが含まれます。
> - `asset_metadata` - 各アセットに関する一般的な情報が含まれます。
> - `asset_hierarchy_metadata` - アセット間の関係性に関する情報が含まれます。

このチュートリアルで SQL クエリを実行する方法

1. AWS IoT SiteWise データを使用してデータセットを作成する (コンソール) または の手順に従って、 AWS IoT SiteWise データの AWS IoT Analytics データセットAWS IoT SiteWise データを含むデータセットを作成する (AWS CLI) を作成します。
2. このチュートリアル全体を通してデータセットクエリを更新するには、次の手順を実行します。
   a. AWS IoT Analytics コンソールのデータセットページで、前のページで作成したデータセットの名前を選択します。
   b. データセット概要ページで、[Edit] (編集) を選択して SQL クエリを編集します。
   c. クエリの後に結果をテーブルに表示するには、[Test query] (クエリのテスト) を選択します。

   あるいは、次の `update-dataset` コマンドを実行して AWS CLIで SQL クエリを修正することができます。

   ```
   aws iotanalytics update-dataset --cli-input-json file://update-query.json
   ```

   `update-query.json` の内容:

```
{
    "datasetName": "my_dataset",
    "actions": [
        {
            "actionName": "myDatasetUpdateAction",
            "queryAction": {
                "sqlQuery": "SELECT * FROM my_iotsitewise_datastore.asset_metadata
 LIMIT 3"
            }
        }
    ]
}
```

3. AWS IoT Analytics コンソールまたは で AWS CLI、データに対して次のクエリを実行して、`asset_metadata`テーブルが正常にロードされたことを確認します。

```
SELECT COUNT(*) FROM my_iotsitewise_datastore.asset_metadata
```

同様に、`asset_hierarchy_metadata` テーブルと `raw` テーブルが空ではないことを確認できます。

次のステップ

データ探索

データ探索

データ AWS IoT SiteWise を作成してデータストアにロードしたら、 で AWS IoT Analytics データセットを作成し、SQL クエリを実行して AWS IoT Analytics 、アセットに関するインサイトを検出できます。次のクエリは、統計クエリを実行する前にデータを調べる方法を示しています。

SQL クエリでデータを調べる方法

1. 生テーブルなどの各テーブルに列と値のサンプルを表示します。

```
SELECT * FROM my_iotsitewise_datastore.raw LIMIT 5
```

2. `SELECT DISTINCT` を使用して`asset_metadata`テーブルをクエリし、 AWS IoT SiteWise アセットの (一意の) 名を一覧表示します。

```
SELECT DISTINCT assetname FROM my_iotsitewise_datastore.asset_metadata ORDER BY
 assetname
```

3. 特定の AWS IoT SiteWise アセットのプロパティに関する情報を一覧表示するには、 WHERE句を使用します。

```
SELECT assetpropertyname,
     assetpropertyunit,
     assetpropertydatatype
FROM my_iotsitewise_datastore.asset_metadata
WHERE assetname = 'Demo Turbine Asset 2'
```

4. を使用すると AWS IoT Analytics、次の例のように、データストア内の 2 つ以上のテーブルのデータを結合できます。

```
SELECT * FROM my_iotsitewise_datastore.raw AS raw
JOIN my_iotsitewise_datastore.asset_metadata AS asset_metadata
ON raw.seriesId = asset_metadata.timeseriesId
```

アセット間のすべての関係を表示するには、以下のクエリの JOIN 機能を使用します。

```
SELECT DISTINCT parent.assetName as "Parent name",
    child.assetName AS "Child name"
FROM (
    SELECT sourceAssetId AS parent,
            targetAssetId AS child
    FROM my_iotsitewise_datastore.asset_hierarchy_metadata
    WHERE associationType = 'CHILD'
)
AS relations
JOIN my_iotsitewise_datastore.asset_metadata AS child
    ON relations.child = child.assetId
JOIN my_iotsitewise_datastore.asset_metadata AS parent
    ON relations.parent = parent.assetId
```

次のステップ

統計クエリの実行

統計クエリの実行

データを調べたので AWS IoT SiteWise 、産業機器に貴重なインサイトを提供する統計クエリを実行できます。次のクエリは、取得できる情報の一部を示しています。

AWS IoT SiteWise デモ風力発電所データに対して統計クエリを実行するには

1. 次の SQL コマンドを実行して、特定のアセット (Demo Turbine Asset 4) の数値を含むすべてのプロパティの最新の値を検索します。

```
SELECT assetName,
    assetPropertyName,
    assetPropertyUnit,
    max_by(value, timeInSeconds) AS Latest
FROM (
    SELECT *,
        CASE assetPropertyDataType
        WHEN 'DOUBLE' THEN
        cast(doubleValue AS varchar)
        WHEN 'INTEGER' THEN
        cast(integerValue AS varchar)
        WHEN 'STRING' THEN
        stringValue
        WHEN 'BOOLEAN' THEN
        cast(booleanValue AS varchar)
        ELSE NULL
        END AS value
    FROM my_iotsitewise_datastore.asset_metadata AS asset_metadata
    JOIN my_iotsitewise_datastore.raw AS raw
        ON raw.seriesId = asset_metadata.timeSeriesId
    WHERE startYear=2021
        AND startMonth=7
        AND startDay=8
        AND assetName='Demo Turbine Asset 4'
)
GROUP BY assetName, assetPropertyName, assetPropertyUnit
```

2. 両方のメタデータテーブルと生テーブルを結合して、親アセットに加え、全アセットの最大風速プロパティを特定します。

```
SELECT child_assets_data_set.parentAssetId,
        child_assets_data_set.childAssetId,
        asset_metadata.assetPropertyId,
```

```
        asset_metadata.assetPropertyName,
        asset_metadata.timeSeriesId,
        raw_data_set.max_speed
FROM (
   SELECT sourceAssetId AS parentAssetId,
        targetAssetId AS childAssetId
    FROM my_iotsitewise_datastore.asset_hierarchy_metadata
    WHERE associationType = 'CHILD'
)
AS child_assets_data_set
JOIN mls_demo.asset_metadata AS asset_metadata
    ON asset_metadata.assetId = child_assets_data_set.childAssetId
JOIN (
    SELECT seriesId, MAX(doubleValue) AS max_speed
    FROM my_iotsitewise_datastore.raw
    GROUP BY seriesId
)
AS raw_data_set
ON raw_data_set.seriesId = asset_metadata.timeseriesid
WHERE assetPropertyName = 'Wind Speed'
ORDER BY max_speed DESC
```

3. アセット (Demo Turbine Asset 2) の特定のプロパティ (Wind Speed) の平均値を調べるには、次の SQL コマンドを実行します。`my_bucket_id` をバケットの ID に置き換える必要があります。

```
SELECT AVG(doubleValue) as "Average wind speed"
FROM my_iotsitewise_datastore.raw
WHERE seriesId =
    (SELECT timeseriesId
    FROM my_iotsitewise_datastore.asset_metadata as asset_metadata
    WHERE asset_metadata.assetname = 'Demo Turbine Asset 2'
            AND asset_metadata.assetpropertyname = 'Wind Speed')
```

次のステップ

チュートリアルリソースのクリーンアップ

チュートリアルリソースのクリーンアップ

チュートリアルを完了したら、料金が発生しないようにリソースをクリーンアップします。

AWS IoT SiteWise デモを削除するには

AWS IoT SiteWise デモは 1 週間後にそれ自体を削除します。デモリソースの使用が完了したら、早めにデモを消去してください。デモを手動で消去するには、次の手順を実行します。

1. AWS CloudFormation コンソールに移動します。
2. [スタック] のリストから [IoTSiteWiseDemoAssets] を選択します。
3. [Delete (削除)] を選択します。スタックを消去すると、デモ用に作成されたすべてのリソースが消去されます。
4. 確認ダイアログで、[Delete] (消去) を選択します。

   スタックの消去には約 15 分かかります。デモの消去に失敗した場合は、右上隅の [消去] を再度選択します。デモを再度削除できない場合は、 AWS CloudFormation コンソールの手順に従って削除に失敗したリソースをスキップし、もう一度試してください。

データストアを消去する方法

- マネージドデータストアを消去するには、CLI コマンド `delete-datastore` を実行します。以下に例を示します。

```
aws iotanalytics delete-datastore --datastore-name my_IotSiteWise_datastore
```

AWS IoT Analytics データセットを削除するには

- データセットを消去するには、CLI コマンド `delete-dataset` を実行します。以下に例を示します。このオペレーションの実行前にデータセットのコンテンツを消去する必要はありません。

```
aws iotanalytics delete-dataset --dataset-name my_dataset
```

> **Note**
>
> このコマンドでは何も出力されません。

パイプラインアクティビティ

最もシンプルな機能のパイプラインは、チャネルをデータストアに接続します。これにより、パイプラインに 2 つのアクティビティ channel アクティビティと datastore アクティビティが追加されます。さらに他のアクティビティをパイプラインに追加することで、より強力なメッセージ処理が可能です。

RunPipelineActivity 演算を使用すると、指定したメッセージペイロードに対してパイプラインアクティビティを実行した場合の結果をシミュレートできます。これは、パイプラインアクティビティを作成してデバッグするときに役立ちます。「RunPipelineActivity の例」でこの使用方法を実演します。

チャネルアクティビティ

パイプラインの最初のアクティビティは、処理するメッセージのソースを決定する channel アクティビティです。

```
{
    "channel": {
        "name": "MyChannelActivity",
        "channelName": "mychannel",
        "next": "MyLambdaActivity"
    }
}
```

データストアアクティビティ

datastore アクティビティは、最後のアクティビティであり、処理したデータの保存先を指定します。

```
{
    "datastore": {
        "name": "MyDatastoreActivity",
        "datastoreName": "mydatastore"
    }
}
```

AWS Lambda アクティビティ

lambdaアクティビティを使用すれば、メッセージでより複雑な処理を実行できます。たとえば、外部 API の出力からのデータによるメッセージの強化や、Amazon DynamoDB のロジックに基づくメッセージのフィルターがあります。ただし、データストアに入る前に、このパイプラインアクティビティを使用してメッセージを追加したり、既存のメッセージを削除したりすることはできません。

lambda アクティビティで使用される AWS Lambda 関数は、JSON オブジェクトの配列を受信して返す必要があります。例については、the section called “Lambda 関数の例 1”を参照してください。

Lambda 関数を呼び出す AWS IoT Analytics アクセス許可を付与するには、ポリシーを追加する必要があります。例えば、次の CLI コマンドを実行し、*exampleFunctionName* を Lambda 関数の名前に置き換え、*123456789012*「」を AWS アカウント ID に置き換え、指定された Lambda 関数を呼び出すパイプラインの Amazon リソースネーム (ARN) を使用します。

```
aws lambda add-permission --function-name exampleFunctionName --
action lambda:InvokeFunction --statement-id iotanalytics --principal
 iotanalytics.amazonaws.com --source-account 123456789012 --source-arn
 arn:aws:iotanalytics:us-east-1:123456789012:pipeline/examplePipeline
```

このコマンドは以下を返します。

```
{
    "Statement": "{\"Sid\":\"iotanalyticsa\",\"Effect\":\"Allow\",
\"Principal\":{\"Service\":\"iotanalytics.amazonaws.com\"},\"Action\":
\"lambda:InvokeFunction\",\"Resource\":\"arn:aws:lambda:aws-region:aws-
account:function:exampleFunctionName\",\"Condition\":{\"StringEquals\":
{\"AWS:SourceAccount\":\"123456789012\"},\"ArnLike\":{\"AWS:SourceArn\":
\"arn:aws:iotanalytics:us-east-1:123456789012:pipeline/examplePipeline\"}}}"
}
```

詳細については、AWS Lambda デベロッパーガイドの「AWS Lambdaのリソースベースのポリシーを使用する」を参照してください。

Lambda 関数の例 1

この例では、Lambda 関数は、元のメッセージのデータに基づいて、追加情報を追加します。デ次の例のようなペイロードを含むメッセージがデバイスから発行されます。

```
{
```

```
  "thingid": "00001234abcd",
  "temperature": 26,
  "humidity": 29,
  "location": {
    "lat": 52.4332935,
    "lon": 13.231694
  },
  "ip": "192.168.178.54",
  "datetime": "2018-02-15T07:06:01"
}
```

そのデバイスには次のパイプライン定義があります。

```
{
    "pipeline": {
        "activities": [
            {
                "channel": {
                    "channelName": "foobar_channel",
                    "name": "foobar_channel_activity",
                    "next": "lambda_foobar_activity"
                }
            },
            {
                "lambda": {
                    "lambdaName": "MyAnalyticsLambdaFunction",
                    "batchSize": 5,
                    "name": "lambda_foobar_activity",
                    "next": "foobar_store_activity"
                }
            },
            {
                "datastore": {
                    "datastoreName": "foobar_datastore",
                    "name": "foobar_store_activity"
                }
            }
        ],
        "name": "foobar_pipeline",
        "arn": "arn:aws:iotanalytics:eu-west-1:123456789012:pipeline/foobar_pipeline"
    }
}
```

次の Lambda Python 関数 `MyAnalyticsLambdaFunction`により GMaps URL と華氏の温度がメッセージに追加されます。

```
import logging
import sys

# Configure logging
logger = logging.getLogger()
logger.setLevel(logging.INFO)
streamHandler = logging.StreamHandler(stream=sys.stdout)
formatter = logging.Formatter('%(asctime)s - %(name)s - %(levelname)s - %(message)s')
streamHandler.setFormatter(formatter)
logger.addHandler(streamHandler)

def c_to_f(c):
    return 9.0/5.0 * c + 32

def lambda_handler(event, context):
    logger.info("event before processing: {}".format(event))
    maps_url = 'N/A'

    for e in event:
        #e['foo'] = 'addedByLambda'
        if 'location' in e:
            lat = e['location']['lat']
            lon = e['location']['lon']
            maps_url = "http://maps.google.com/maps?q={},{}".format(lat,lon)

        if 'temperature' in e:
            e['temperature_f'] = c_to_f(e['temperature'])

        logger.info("maps_url: {}".format(maps_url))
        e['maps_url'] = maps_url

    logger.info("event after processing: {}".format(event))

    return event
```

Lambda 関数の例 2

メッセージペイロードを圧縮およびシリアル化して、トランスポートおよびストレージコストを削減するのが便利です。この 2 番目の例では、Lambda 関数はメッセージペイロードが圧縮されて、文

字列として base64 エンコード シリアル化された元の JSON を表していると想定しています。元の JSON が返されます。

```
import base64
import gzip
import json
import logging
import sys

# Configure logging
logger = logging.getLogger()
logger.setLevel(logging.INFO)
streamHandler = logging.StreamHandler(stream=sys.stdout)
formatter = logging.Formatter('%(asctime)s - %(name)s - %(levelname)s - %(message)s')
streamHandler.setFormatter(formatter)
logger.addHandler(streamHandler)

def decode_to_bytes(e):
    return base64.b64decode(e)

def decompress_to_string(binary_data):
    return gzip.decompress(binary_data).decode('utf-8')

def lambda_handler(event, context):
    logger.info("event before processing: {}".format(event))

    decompressed_data = []

    for e in event:
        binary_data = decode_to_bytes(e)
        decompressed_string = decompress_to_string(binary_data)

        decompressed_data.append(json.loads(decompressed_string))

    logger.info("event after processing: {}".format(decompressed_data))

    return decompressed_data
```

AddAttributes アクティビティ

`addAttributes` アクティビティでは、メッセージの既存の属性に基づいて属性が追加されます。これにより、メッセージを保存する前にメッセージのシェイプを変更できます。たとえ

ば、addAttributes を使用して、様々な世代のデバイスファームウェアからのデータを正規化できます。

次の入力メッセージについて考えてみましょう。

```
{
    "device": {
        "id": "device-123",
        "coord": [ 47.6152543, -122.3354883 ]
    }
}
```

addAttributes アクティビティは次のようになります。

```
{
    "addAttributes": {
        "name": "MyAddAttributesActivity",
        "attributes": {
            "device.id": "id",
            "device.coord[0]": "lat",
            "device.coord[1]": "lon"
        },
        "next": "MyRemoveAttributesActivity"
    }
}
```

このアクティビティでは、デバイス ID をルートレベルに移動し、coord 配列の値を抽出して、lat および lon と呼ばれる最上位レベルの属性に昇格します。このアクティビティの結果として、入力メッセージは次のように変換されます。

```
{
    "device": {
        "id": "device-123",
        "coord": [ 47.6, -122.3 ]
    },
    "id": "device-123",
    "lat": 47.6,
    "lon": -122.3
}
```

元のデバイスの属性はまだ存在しています。これを削除するには、removeAttributes アクティビティを使用できます。

RemoveAttributes アクティビティ

removeAttributes アクティビティはメッセージから属性を削除します。たとえば、addAttributes アクティビティの結果、次のメッセージが発生したとします。

```
{
    "device": {
        "id": "device-123",
        "coord": [ 47.6, -122.3 ]
    },
    "id": "device-123",
    "lat": 47.6,
    "lon": -122.3
}
```

このメッセージを正規化し、ルートレベルに必須のデータのみを含むようにするには、次の removeAttributes アクティビティを使用します。

```
{
    "removeAttributes": {
        "name": "MyRemoveAttributesActivity",
        "attributes": [
            "device"
        ],
        "next": "MyDatastoreActivity"
    }
}
```

これにより、パイプラインに沿って次のメッセージが流れます。

```
{
    "id": "device-123",
    "lat": 47.6,
    "lon": -122.3
}
```

SelectAttributes アクティビティ

selectAttributes アクティビティは、元のメッセージから指定された属性のみを使用して、新しいメッセージを作成します。他のすべての属性は削除されます。selectAttributes はメッセージのルートのみに新しい属性を作成します。そのため、次のメッセージがあるとします。

```
{
    "device": {
        "id": "device-123",
        "coord": [ 47.6152543, -122.3354883 ],
        "temp": 50,
        "hum": 40
    },
    "light": 90
}
```

さらに、次のアクティビティがあるとします。

```
{
    "selectAttributes": {
        "name": "MySelectAttributesActivity",
        "attributes": [
            "device.temp",
            "device.hum",
            "light"
        ],
        "next": "MyDatastoreActivity"
    }
}
```

この結果、次のメッセージがパイプラインを通じて流れます。

```
{
    "temp": 50,
    "hum": 40,
    "light": 90
}
```

ここでも、selectAttributes はルートレベルのオブジェクトのみを作成できます。

フィルターアクティビティ

`filter` アクティビティは、属性に基づいてメッセージをフィルタリングします。このアクティビティで使用されている式は、Boolean を返す必要がある SQL `WHERE` 句に似ています。

```
{
    "filter": {
        "name": "MyFilterActivity",
        "filter": "temp > 40 AND hum < 20",
        "next": "MyDatastoreActivity"
    }
}
```

DeviceRegistryEnrich アクティビティ

`deviceRegistryEnrich` アクティビティにより、 AWS IoT デバイスレジストリからメッセージペイロードにデータを追加できます。たとえば、次のメッセージの場合:

```
{
    "temp": 50,
    "hum": 40,
    "device" {
        "thingName": "my-thing"
    }
}
```

そして、次のような `deviceRegistryEnrich` アクティビティを考えてみます。

```
{
    "deviceRegistryEnrich": {
        "name": "MyDeviceRegistryEnrichActivity",
        "attribute": "metadata",
        "thingName": "device.thingName",
        "roleArn": "arn:aws:iam::<your-account-number>:role:MyEnrichRole",
        "next": "MyDatastoreActivity"
    }
}
```

出力メッセージはこの例のようになります。

```
{
    "temp" : 50,
    "hum" : 40,
    "device" {
        "thingName" : "my-thing"
    },
    "metadata" : {
        "defaultClientId": "my-thing",
        "thingTypeName": "my-thing",
        "thingArn": "arn:aws:iot:us-east-1:<your-account-number>:thing/my-thing",
        "version": 1,
        "thingName": "my-thing",
        "attributes": {},
        "thingId": "aaabbbccc-dddeeef-gghh-jjkk-llmmnnoopp"
    }
}
```

適切なアクセス権限がアタッチされているアクティビティ定義の `roleArn` フィールド内でロールを使用する必要があります。このロールには、次の例のようなアクセス許可ポリシーが必要です。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iot:DescribeThing"
            ],
            "Resource": [
                "arn:aws:iot:<region>:<account-id>:thing/<thing-name>"
            ]
        }
    ]
}
```

また、信頼ポリシーは次のようになります。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
```

```
            "Effect": "Allow",
            "Principal": {
                "Service": "iotanalytics.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole"
            ]
        }
    ]
}
```

DeviceShadowEnrich アクティビティ

deviceShadowEnrich アクティビティは、 AWS IoT Device Shadow サービスからの情報をメッセージに追加します。たとえば、以下のメッセージがあるとします。

```
{
    "temp": 50,
    "hum": 40,
    "device": { "thingName": "my-thing" }
}
```

そして、次の deviceShadowEnrich アクティビティがあるとします。

```
{
    "deviceShadowEnrich": {
        "name": "MyDeviceShadowEnrichActivity",
        "attribute": "shadow",
        "thingName": "device.thingName",
        "roleArn": "arn:aws:iam::<your-account-number>:role:MyEnrichRole",
        "next": "MyDatastoreActivity"
    }
}
```

この結果は、次の例ようなメッセージです。

```
{
    "temp": 50,
    "hum": 40,
    "device": {
        "thingName": "my-thing"
```

```
    },
    "shadow": {
        "state": {
            "desired": {
                "attributeX": valueX, ...
            },
            "reported": {
                "attributeX": valueX, ...
            },
            "delta": {
                "attributeX": valueX, ...
            }
        },
        "metadata": {
            "desired": {
                "attribute1": {
                    "timestamp": timestamp
                }, ...
            },
            "reported": ": {
                "attribute1": {
                    "timestamp": timestamp
                }, ...
            }
        },
        "timestamp": timestamp,
        "clientToken": "token",
        "version": version
    }
}
```

適切なアクセス権限がアタッチされているアクティビティ定義の `roleArn` フィールド内でロールを使用する必要があります。このロールには、次のようなアクセス許可ポリシーが必要です。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iot:GetThingShadow"
            ],
            "Resource": [
```

```
                "arn:aws:iot:<region>:<account-id>:thing/<thing-name>"
            ]
        }
    ]
}
```

また、信頼ポリシーは次のようになります。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": "iotanalytics.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole"
            ]
        }
    ]
}
```

Math アクティビティ

math アクティビティは、メッセージの属性を使用して演算式を計算します。この式は数値を返す必要があります。たとえば、以下の入力メッセージがあるとします。

```
{
    "tempF": 50,
}
```

次の math アクティビティによる処理の後は次のようになります。

```
{
    "math": {
        "name": "MyMathActivity",
        "math": "(tempF - 32) / 2",
        "attribute": "tempC",
        "next": "MyDatastoreActivity"
```

```
    }
}
```

結果のメッセージは次のようになります。

```
{
    "tempF" : 50,
    "tempC": 9
}
```

Math アクティビティ演算子と関数

`math` アクティビティでは、次の演算子を使用できます。

+	追加
-	減算
*	乗算
/	除算
%	モジュロ

`math` アクティビティでは、以下の関数を使用できます。

- abs(10進数)
- acos(10進数)
- asin(10進数)
- atan(10進数)
- atan2(10進数, 10進数)
- ceil10 進数)
- cos10 進数)
- cosh(10進)
- exp(10進数)

- ln(10進数)
- ログ(10進数)
- mod(10進数, 10進数)
- べき乗(10進数, 10進数)
- 丸(10進)
- 符号(10進)
- sin(10進数)
- sinh(10進)
- sqrt(10進数)
- tan(10進数)
- tanh(10進数)
- trunc(10 進, 整数)

abs(10進数)

数値の絶対値を返します。

例: `abs(-5)` は 5 を返します。

引数の型	結果
`Int`	`Int`、引数の絶対値。
`Decimal`	`Decimal`、引数の絶対値。
`Boolean`	`Undefined` .
`String`	`Decimal`。結果は、引数の絶対値です。文字列が変換できない場合、結果は `Undefined` です。
配列	`Undefined` .
オブジェクト	`Undefined` .
Null	`Undefined` .

引数の型	結果
未定義	Undefined .

acos(10進数)

数値の逆コサインをラジアンで返します。Decimal 引数は関数適用の前に倍精度に丸められます。

例: acos(0) = 1.5707963267948966

引数の型	結果
Int	Decimal 「倍精度で」、引数の逆コサイン。仮想上の結果は、Undefined として返されます。
Decimal	Decimal 「倍精度で」、引数の逆コサイン。仮想上の結果は、Undefined として返されます。
Boolean	Undefined .
String	Decimal 「倍精度で」、引数の逆コサイン。文字列が変換できない場合、結果は Undefined です。仮想上の結果は、Undefined として返されます。
配列	Undefined .
オブジェクト	Undefined .
Null	Undefined .
未定義	Undefined .

asin(10進数)

数値の逆サインをラジアンで返します。Decimal 引数は関数適用の前に倍精度に丸められます。

例: `asin(0)` = 0.0

引数の型	結果
`Int`	`Decimal` 「倍精度」、引数の逆サイン。仮想上の結果は、`Undefined` として返されます。
`Decimal`	`Decimal` 「倍精度」、引数の逆サイン。仮想上の結果は、`Undefined` として返されます。
`Boolean`	`Undefined` .
`String`	`Decimal` 「倍精度」、引数の逆サイン。文字列が変換できない場合、結果は `Undefined` です。仮想上の結果は、`Undefined` として返されます。
配列	`Undefined` .
オブジェクト	`Undefined` .
Null	`Undefined` .
未定義	`Undefined` .

atan(10進数)

数値の逆タンジェントをラジアンで返します。`Decimal` 引数は関数適用の前に倍精度に丸められます。

例: `atan(0)` = 0.0

引数の型	結果
`Int`	`Decimal` 「倍精度で」、引数の逆タンジェント。仮想上の結果は、`Undefined` として返されます。

引数の型	結果
Decimal	Decimal 「倍精度で」、引数の逆タンジェント。仮想上の結果は、Undefined として返されます。
Boolean	Undefined .
String	Decimal 「倍精度で」、引数の逆タンジェント。文字列が変換できない場合、結果は Undefined です。仮想上の結果は、Undefined として返されます。
配列	Undefined .
オブジェクト	Undefined .
Null	Undefined .
未定義	Undefined .

atan2(10進数, 10進数)

正の X 軸と 2 つの引数で定義された点 x, yの間の角度をラジアンで返します。反時計回りの角度では正で 上半面、y > 0)、時計回りの角度では負です。Decimal 引数は関数適用の前に倍精度に丸められます。

例: atan(1, 0) = 1.5707963267948966

引数の型	引数の型	結果
Int / Decimal	Int / Decimal	Decimal 「倍精度で」、X 軸と指定した点 x,yの間の角度。
Int / Decimal / String	Int / Decimal / String	Decimal、示された点の逆タンジェント。文字列が変換できない場合、結果は Undefined です。

引数の型	引数の型	結果
その他の値	その他の値	Undefined .

ceil10 進数)

指定の Decimal を最も近い Int に切り上げます。

例:

ceil(1.2) = 2

ceil(11.2) = -1

引数の型	結果
Int	Int、引数値。
Decimal	Int、文字列は Decimal に変換され、最も近い Int へ切り上げられます。文字列が Decimal に変換できない場合、結果は Undefined です。
その他の値	Undefined .

cos10 進数)

数値のコサインをラジアンで返します。Decimal 引数は関数適用の前に倍精度に丸められます。

例: cos(0) = 1

引数の型	結果
Int	Decimal 「倍精度で」、引数のコサイン。仮想上の結果は、Undefined として返されます。

引数の型	結果
Decimal	Decimal 「倍精度で」、引数のコサイン。仮想上の結果は、Undefined として返されます。
Boolean	Undefined .
String	Decimal 「倍精度で」、引数のコサイン。文字列が Decimal に変換できない場合、結果は Undefined です。仮想上の結果は、Undefined として返されます。
配列	Undefined .
オブジェクト	Undefined .
Null	Undefined .
未定義	Undefined .

cosh(10進)

数値の双曲線コサインをラジアンで返します。Decimal 引数は関数適用の前に倍精度に丸められます。

例: cosh(2.3) = 5.037220649268761

引数の型	結果
Int	Decimal 「倍精度で」、引数の双曲線コサイン。仮想上の結果は、Undefined として返されます。
Decimal	Decimal 「倍精度で」、引数の双曲線コサイン。仮想上の結果は、Undefined として返されます。
Boolean	Undefined .

引数の型	結果
String	Decimal 「倍精度で」、引数の双曲線コサイン。文字列が Decimal に変換できない場合、結果は Undefined　です。仮想上の結果は、Undefined　として返されます。
配列	Undefined .
オブジェクト	Undefined .
Null	Undefined .
未定義	Undefined .

exp(10進数)

e を e 引数の累乗にして返します。Decimal 引数は関数適用の前に倍精度に丸められます。

例: exp(1) = 1

引数の型	結果
Int	Decimal 「倍精度で」、e^引数。
Decimal	Decimal 「倍精度で」、e^引数。
String	Decimal 「倍精度で」、e^引数。String が Decimal に変換できない場合、結果は Undefined　です。
その他の値	Undefined .

ln(10進数)

引数の自然対数を返します。Decimal 引数は関数適用の前に倍精度に丸められます。

例: ln(e) = 1

引数の型	結果
Int	Decimal (倍精度で)、引数の自然対数。
Decimal	Decimal 「倍精度で」、引数の自然対数。
Boolean	Undefined .
String	Decimal 「倍精度で」、引数の自然対数。文字列が Decimal に変換できない場合、結果は Undefined です。
配列	Undefined .
オブジェクト	Undefined .
Null	Undefined .
未定義	Undefined .

ログ(10進数)

引数の 10 を底とする対数を返します。Decimal 引数は関数適用の前に倍精度に丸められます。

例: log(100) = 2.0

引数の型	結果
Int	Decimal 「倍精度で」、引数の 10 を底とした対数。
Decimal	Decimal 「倍精度で」、引数の 10 を底とした対数。
Boolean	Undefined .
String	Decimal 「倍精度で」、引数の 10 を底とした対数。String が Decimal に変換できない場合、結果は Undefined です。

引数の型	結果
配列	`Undefined .`
オブジェクト	`Undefined .`
Null	`Undefined .`
未定義	`Undefined .`

mod(10進数, 10進数)

最初の引数を 2 番目の引数で割ったときの剰余を返します。同じモジュロ機能に挿入演算子として % も使用できます。

例: `mod(8, 3)` = 2

左のオペランド	右のオペランド	Output
`Int`	`Int`	`Int`、最初の引数を 2 番目の引数で割ったときの剰余。
`Int` / `Decimal`	`Int` / `Decimal`	`Decimal`、最初の引数を 2 番目の引数で割ったときの剰余。
`String` / `Int` / `Decimal`	`String` / `Int` / `Decimal`	すべての文字列を `Decimals` に変換した場合、結果は、最初の引数を 2 番目の引数で割ったときの剰余です。そうでない場合は、`Undefined` です。
その他の値	その他の値	`Undefined .`

べき乗(10進数, 10進数)

最初の引数を 2 番目の引数の累乗にして返します。Decimal 引数は関数適用の前に倍精度に丸められます。

例: power(2, 5) = 32.0

引数の型 1	引数の型 2	出力
Int / Decimal	Int / Decimal	Decimal「倍精度で」、最初の引数を 2 番目の引数の累乗にします。
Int / Decimal / String	Int / Decimal / String	Decimal「倍精度で」、最初の引数を 2 番目の引数の累乗にします。すべての文字列は Decimals に変換されます。いずれかの String が Decimal への変換に失敗したら、結果は Undefined です。
その他の値	その他の値	Undefined .

丸(10進)

指定の Decimal を最も近い Int に丸めます。Decimal が 2 つの Int 値と等距離である場合 例: 0.5)、Decimal は丸められます。

例:

Round(1.2) = 1

Round(1.5) = 2

Round(1.7) = 2

Round(-1.1) = -1

Round(-1.5) = -2

引数の型	結果
Int	引数
Decimal	Decimal は、最も近い Int に切り下げられます。
String	Decimal は、最も近い Int に切り下げられます。文字列が Decimal に変換できない場合、結果は Undefined です。
その他の値	Undefined .

符号(10進)

指定された数値の符号を返します。引数の符号が正の場合、1 を返します。引数の符号が負の場合、-1 を返します。引数が 0 の場合、0 を返します。

例:

sign(-7) = -1

sign(0) = 0.

sign(13) = 1

引数の型	結果
Int	Int、Int の値の符号。
Decimal	Int、Decimal の値の符号。
String	Int、Decimal の値の符号。文字列は Decimal の値に変換され、Decimal の値の符号が返されます。String が Decimal に変換できない場合、結果は Undefined です。
その他の値	Undefined .

sin(10進数)

数値のサインをラジアンで返します。Decimal 引数は関数適用の前に倍精度に丸められます。

例: sin(0) = 0.0

引数の型	結果
Int	Decimal 「倍精度で」、引数のサイン。
Decimal	Decimal (倍精度で)、引数のサイン。
Boolean	Undefined .
String	Decimal、引数のサイン。文字列が Decimal に変換できない場合、結果は Undefined です。
Array	Undefined .
Object	Undefined .
Null	Undefined .
Undefined	Undefined .

sinh(10進)

数値の双曲線サインを返します。Decimal 値は関数適用の前に倍精度に丸められます。結果は倍精度の Decimal 値です。

例: sinh(2.3) = 4.936961805545957

引数の型	結果
Int	Decimal (倍精度で)、引数のハイパーボリックサイン。
Decimal	Decimal 「倍精度で」、引数の双曲線サイン。

引数の型	結果
Boolean	Undefined .
String	Decimal、引数の双曲線サイン。文字列が Decimal に変換できない場合、結果は Undefined です。
Array	Undefined .
Object	Undefined .
Null	Undefined .
Undefined	Undefined .

sqrt(10進数)

数値の平方根を返します。Decimal 引数は関数適用の前に倍精度に丸められます。

例: sqrt(9) = 3.0

引数の型	結果
Int	引数の平方根。
Decimal	引数の平方根。
Boolean	Undefined .
String	引数の平方根。文字列が Decimal に変換できない場合、結果は Undefined です。
Array	Undefined .
Object	Undefined .
Null	Undefined .
Undefined	Undefined .

tan(10進数)

数値のタンジェントをラジアンで返します。Decimal 値は関数適用の前に倍精度に丸められます。

例: tan(3) = -0.1425465430742778

引数の型	結果
Int	Decimal 「倍精度で」、引数のタンジェント。
Decimal	Decimal 「倍精度で」、引数のタンジェント。
Boolean	Undefined .
String	Decimal 「倍精度で」、引数のタンジェント。文字列が Decimal に変換できない場合、結果は Undefined です。
Array	Undefined .
Object	Undefined .
Null	Undefined .
Undefined	Undefined .

tanh(10進数)

数値の双曲線タンジェントをラジアンで返します。Decimal 値は関数適用の前に倍精度に丸められます。

例: tanh(2.3) = 0.9800963962661914

引数の型	結果
Int	Decimal 「倍精度で」、引数の双曲線タンジェント。

引数の型	結果
Decimal	Decimal 「倍精度で」、引数の双曲線タンジェント。
Boolean	Undefined .
String	Decimal 「倍精度で」、引数の双曲線タンジェント。文字列が Decimal に変換できない場合、結果は Undefined です。
Array	Undefined .
Object	Undefined .
Null	Undefined .
Undefined	Undefined .

trunc(10 進, 整数)

2 番目の引数で指定された Decimal の場所の数で最初の引数を切り捨てます。2 番目の引数がゼロよりより少ない場合は、ゼロに設定されます。2 番目の引数が 34 より大きい場合は、34 に設定されます。末尾のゼロは結果から省かれます。

例:

trunc(2.3, 0) = 2

trunc(2.3123, 2) = 2.31

trunc(2.888, 2) = 2.88

trunc(2.00, 5) = 2

引数の型 1	引数の型 2	結果
Int	Int	ソース値。
Int / Decimal / String	Int / Decimal	最初の引数は 2 番目の引数で説明された長さに切り捨てら

引数の型 1	引数の型 2	結果
		れます。2 番目の引数は、Int でなければ、最も近い Int に切り下げられます。文字列は Decimal に変換されます。文字列変換が失敗した場合、結果は Undefined です。
その他の値		未定義

RunPipelineActivity

パイプラインアクティビティをテストするために、RunPipelineActivity コマンドを使用する方法の例を以下に示します。この例では、算術アクティビティをテストします。

1. テストするパイプラインアクティビティの定義が含まれている maths.json ファイルを作成します。

```
{
    "math": {
        "name": "MyMathActivity",
        "math": "((temp - 32) * 5.0) / 9.0",
        "attribute": "tempC"
    }
}
```

2. パイプラインアクティビティをテストするために使用されるサンプルペイロードが含まれている payloads.json ファイルを作成します。

```
[
    "{\"humidity\": 52, \"temp\": 68 }",
    "{\"humidity\": 52, \"temp\": 32 }"
]
```

3. コマンドラインから RunPipelineActivities 演算を呼び出します。

```
aws iotanalytics run-pipeline-activity  --pipeline-activity file://maths.json  --
payloads file://payloads.json --cli-binary-format raw-in-base64-out
```

これを実行すると、次の結果になります。

```
{
    "logResult": "",
    "payloads": [
        "eyJodW1pZGl0eSI6NTIsInRlbXAiOjY4LCJ0ZW1wQyI6MjB9",
        "eyJodW1pZGl0eSI6NTIsInRlbXAiOjMyLCJ0ZW1wQyI6MH0="
    ]
}
```

結果に記載されるペイロードは、Base64 でエンコードされた文字列です。これらの文字列がデコードされると、次のような結果が得られます。

```
{"humidity":52,"temp":68,"tempC":20}
{"humidity":52,"temp":32,"tempC":0}
```

チャネルメッセージの再処理

AWS IoT Analytics を使用すると、チャネルデータを再処理できます。これは次のような場合に便利です。

- 最初からやり直さずに、既存の取り込まれたデータを再生する。
- パイプラインを更新し、既存のデータに変更を反映して最新の状態にする。
- カスタマー管理型ストレージオプション、チャネルに対する許可、またはデータストアに変更を加える前に取り込まれたデータを含める。

パラメータ

を使用してパイプラインを介してチャネルメッセージを再処理する場合は AWS IoT Analytics、次の情報を指定する必要があります。

`StartPipelineReprocessing`

パイプラインによるチャネルメッセージの再処理を開始します。

`ChannelMessages`

再処理する 1 つ以上のチャネルメッセージセットを指定します。

`channelMessages` オブジェクトを使用する場合は `startTime` と `endTime` に対して値を指定してはいけません。

`s3Paths`

チャネルメッセージを保存する Amazon Simple Storage Service (Amazon S3) オブジェクトを識別するために 1 つ以上のキーを指定します。このキーにはフルパスを使用する必要があります。

パスの例: `00:00:00/1582940490000_1582940520000_123456789012_mychannel_0_2118.0.json`

タイプ: 文字列の配列

配列メンバーの制約: 1 ~ 100 項目

長さの制約: 1 ~ 1024 文字

`endTime`

再処理されるチャネルデータの終了時間 (その時間を含まない)。

`endTime` パラメータに対して値を指定する場合は、`channelMessages` オブジェクトを使用してはいけません。

タイプ: タイムスタンプ

`startTime`

再処理される未加工メッセージデータの開始時間 (その時間を含む)。

`startTime` パラメータに対して値を指定する場合は、`channelMessages` オブジェクトを使用してはいけません。

タイプ: タイムスタンプ

`pipelineName`

再処理を開始するパイプラインの名前。

タイプ: 文字列

長さの制約: 1 ~ 128 文字

チャネルメッセージの再処理 (コンソール)

このチュートリアルでは、 AWS IoT Analytics コンソールの指定された Amazon S3 オブジェクトに保存されているチャネルデータを再処理する方法を示します。

開始する前に、再処理するチャネルメッセージがカスタマー管理型の Amazon S3 バケットに保存されていることを確認してください。

1. AWS IoT Analytics コンソール にサインインします。
2. ナビゲーションペインで [Pipelines] (パイプライン) を選択します。
3. ターゲットのパイプラインを選択します。
4. [Actions] (アクション) から [Reprocess messages] (メッセージの再処理) を選択します。

5. [Pipeline reprocessing] (パイプラインの再処理) ページで、[Reprocess messages] (メッセージの再処理) に対して [S3 objects] (S3 オブジェクト) を選択します。

   AWS IoT Analytics コンソールには、次のオプションもあります。

   - [All available range] (すべての利用可能な範囲) - チャネル内の有効なデータをすべて再処理します。
   - [Last 120 days] (過去 120 日間) - 過去 120 日以内に到着したデータを再処理します。
   - [Last 90 days] (過去 90 日間) - 過去 90 日以内に到着したデータを再処理します。
   - [Last 30 days] (過去 30 日間) - 過去 30 日以内に到着したデータを再処理します。
   - [Custom range] (カスタムレンジ) - 指定された時間内に到着したデータを再処理します。どの時間範囲でも選択できます。
6. チャンネルメッセージを保存する Amazon S3 オブジェクトのキーを入力します。

   キーを見つけるには以下の作業を行います。

   a. Amazon S3 コンソールに移動します。
   b. ターゲットの Amazon S3 オブジェクトを選択します。
   c. [Properties] (プロパティ) の [Object overview] (オブジェクトの概要) セクションで、キーをコピーします。
7. [Start reprocessing] (再処理の開始) を選択します。

チャネルメッセージの再処理 (API)

`StartPipelineReprocessing` API を使用する場合は、以下に留意してください。

- `startTime` パラメータと `endTime` パラメータは、生データが取り込まれた時間を指定しますが、これらはおおよその推定時間です。最も近い時間に切り上げることができます。`startTime` は包括的ですが、`endTime` は排他的です。
- コマンドは再処理を非同期的に起動し、ただちに返ります。
- 再処理されるメッセージは、最初の処理順で処理される保証はありません。ほぼ同じですが、正確ではありません。
- パイプラインを通じて同じチャネルメッセージを再処理するために 24 時間ごとに実行できる `StartPipelineReprocessing` API リクエストは最高で 1000 件です。
- 生データの再処理には追加コストが発生します。

詳細については、AWS IoT Analytics API リファレンスの StartPipelineReprocessing を参照してください。

チャネルの再処理アクティビティのキャンセル

パイプラインの再処理アクティビティをキャンセルするには、CancelPipelineReprocessing API を使用するか、 AWS IoT Analytics コンソールのアクティビティページで再処理をキャンセルを選択します。再処理をキャンセルすると、残りのデータは再処理されません。別の再処理リクエストを開始する必要があります。

再処理のステータスを確認するには DescribePipeline API を使用します。レスポンスの `reprocessingSummaries` フィールドを確認します。

ワークフローの自動化

AWS IoT Analytics は、 の高度なデータ分析を提供します AWS IoT。自動的に IoT データを収集、処理、保存でき、データ分析と機械学習ツールを使用して分析できます。独自のカスタム分析コードまたは Jupyter Notebook をホストするコンテナを実行するか、サードパーティーのカスタムコードコンテナを使用することで、既存の分析ツールを再作成する必要がなくなります。データストアからデータ入力を取得して、自動化ワークフローにフィードするために、次の機能を使用できます。

定期的なスケジュールでデータセットコンテンツを作成します。

データセットコンテンツの自動作成をスケジュールするには、`CreateDataset triggers:schedule:expression`を呼び出すときにトリガーを指定します。データストアに入ったデータは、データセットコンテンツを作成するために使用されます。SQL クエリ`actions:queryAction:sqlQuery`を使用して希望するフィールドを選択します。

新しいデータセットコンテンツが前回以降に到着したデータのみを含むように、重複しない継続した時間間隔を定義します。`actions:queryAction:filters:deltaTime` および `:offsetSeconds` フィールドを使用して、デルタ時間間隔を指定します。次に、時間間隔が経過した時点データセットコンテンツを作成するためのトリガーを指定します。「the section called “例 6 -- デルタウィンドウを使用して SQL データセットを作成する CLI)”」を参照してください。

別のデータセットの完了時にデータセットコンテンツを作成します。

別のデータセットコンテンツの作成が完了したときに、新しいデータセットコンテンツの作成をトリガーします `triggers:dataset:name`)。

分析アプリケーションを自動的に実行します。

独自のカスタムデータ分析アプリケーションをコンテナ化し、別のデータセットのコンテンツが作成されたときにそれらを実行するようにトリガーします。このようにして、繰り返しスケジュールで作成されたデータセットのコンテンツからのデータをアプリケーションに提供できます。アプリケーション内から分析結果に対して自動的にアクションを実行できます `actions:containerAction`)。

別のデータセットの完了時にデータセットコンテンツを作成します。

別のデータセットコンテンツの作成が完了したときに、新しいデータセットコンテンツの作成をトリガーします `triggers:dataset:name`)。

分析アプリケーションを自動的に実行します。

独自のカスタムデータ分析アプリケーションをコンテナ化し、別のデータセットのコンテンツが作成されたときにそれらを実行するようにトリガーします。このようにして、繰り返しスケジュールで作成されたデータセットのコンテンツからのデータをアプリケーションに提供できます。アプリケーション内から分析結果に対して自動的にアクションを実行できます `actions:containerAction`)。

ユースケース

OpEx を削減する製品の品質測定の自動化

圧力、湿度や温度を測定するスマートバルブを使用したシステムを導入しています。システムは定期的に、またバルブの開閉時など、特定のイベントが発生したときにもイベントを照合します。を使用すると AWS IoT Analytics、これらの定期的なウィンドウから重複しないデータを集約し、最終製品の品質に関する KPI レポートを作成する分析を自動化できます。各バッチを処理後に全体的な製品の品質を測定し、最大化された実行量を使用することで運用支出を軽減できます。

多数のデバイスの分析を自動化

デバイスで 100 秒ごとに生成されたデータで 15 分ごとに分析 アルゴリズム、データサイエンス、または ML for KPIを実行します。次回の分析の実行用に各分析サイクルは生成され、状態は保存されます。各分析で、指定された時間ウィンドウ内で受信されたデータを使用したい場合があります。 AWS IoT Analytics を使用すると、分析をオーケストレーションし、実行ごとに KPI とレポートを作成し、将来の分析のためにデータを保存できます。

異常検出の自動化

AWS IoT Analytics を使用すると、データストアに到着した新しいデータに対して 15 分ごとに手動で実行する必要がある異常検出ワークフローを自動化できます。また、指定する期間内のデバイスの使用度と上部ユーザーを示すダッシュボードを自動化することもできます。

産業プロセス成果の予測

インダストリアルプロダクションラインがあります。利用可能なプロセス測定値を含む AWS IoT Analytics、 に送信されたデータを使用して、分析ワークフローを運用してプロセスの結果を予測できます。モデル用のデータは M x N マトリックスで整理でき、この各行にはラボのサンプルが取得された複数の時系列からのデータが含まれています。 AWS IoT Analytics は、デルタウィンドウを作成し、KPI を作成するデータサイエンスツールを使用して分析のワークフローを運用可能にし、測定デバイスの状態を保存するために役立ちます。

Docker コンテナの使用

このセクションでは、独自の Docker コンテナを構築する方法について説明します。サードパーティー製の Docker コンテナを再利用するとセキュリティ上のリスクがあります。これらのコンテナはユーザーの許可を得て任意のコードを実行できます。使用する前に、サードパーティー製のコンテナの著者を信頼できることを確認します。

最後に分析が実行されてから到着したデータの定期的なデータ分析を設定するための手順を以下に示します。

1. データアプリケーションに加えて必要なすべてのライブラリあるいは他の依存関係を含む Docker コンテナを作成します。

   IotAnalytics Jupyter 拡張機能には、コンテナ化プロセスを支援するためのコンテナ化 API が用意されています。また、必要なデータ分析や計算を実行するためのアプリケーションツールセットを作成または組み立てた独自の作成のイメージを実行することもできます。 AWS IoT Analytics では、入力データの送信元を定義し、変数によってアプリケーションと Docker コンテナの出力データの送信先をコンテナ化することができます。カスタム Docker コンテナ入力/出力変数では、カスタムコンテナで変数を使用する詳細を参照できます。)
2. コンテナを Amazon ECR レジストリにアップロードします。
3. データストアを作成して、デバイスからメッセージ データを受信して保存します `iotanalytics: CreateDatastore`)。
4. そのメッセージの送信先となるチャネルを作成します `iotanalytics: CreateChannel`)。
5. チャネルをデータストアに接続するパイプラインを作成します `iotanalytics: CreatePipeline`)。
6. AWS IoT Analytics チャネルにメッセージデータを送信するアクセス許可を付与する IAM ロールを作成する (`iam: CreateRole.`)
7. チャネルをメッセージデータの送信元に接続するために SQL クエリを使用する IoT を作成します `iot: CreateTopicRule` フィールド `topicRulePayload:actions:iotAnalytics`)。デバイスが適切なトピックを MQTT を介してメッセージを送信すると、使用するチャネルにルーティングされます。または、 `iotanalytics: BatchPutMessage`を使用して、 AWS SDK または を使用できるデバイスからチャネルに直接メッセージを送信することもできます AWS CLI。
8. タイムスケジュールによって作成がトリガーされる SQL データセットを作成します `iotanalytics: CreateDataset,` フィールド `actions: queryAction:sqlQuery`)。

また、メッセージデータに適用する事前フィルタを指定して、アクションが最後に実行されてから到着したメッセージに限定することもできます。フィールド `actions:queryAction:filters:deltaTime:timeExpression` はメッセージの時間が判断される式を提供し、フィールド `actions:queryAction:filters:deltaTime:offsetSeconds` はメッセージの到着で発生する可能性があるレイテンシーを表します。)

トリガーのスケジュールに加えて、事前フィルタによってデルタウィンドウが決まります。それぞれの新しい SQL データセットは、SQL データセットが最後に作成されてから受信したメッセージを使用して作成されます。SQL データセットが初めて作成された場合はどうなるのでしょう。データセットが前回作成されたと予測される時期は、スケジュールおよび事前フィルタに基づいて決定されます。)

9. 最初のデータセットフィールド `trigger:dataset`を作成するとトリガーされる別のデータセットを作成します。このデータセットでは、最初のステップで作成した Docker コンテナ示して実行に必要な情報を提供する「コンテナアクション」フィールド `actions:containerAction`を指定します。また、以下も指定します。

   - アカウントに保存された Docker コンテナの ARN `image`)。
   - コンテナアクション `executionRoleArn`を実行するために、システムが必要なリソースにアクセスできるためにアクセス許可を付与するロールの ARN。
   - コンテナアクションを実行するリソースの設定 `resourceConfiguration`)。
   - コンテナアクションを実行するために使用される計算リソースのタイプ 使用できる値がある `computeType`: `ACU_1 [vCPU=4, memory=16GiB] or ACU_2 [vCPU=8, memory=32GiB]`。
   - コンテナアクションを実行するために使用されるリソースインスタンスで利用できる永続的ストレージのサイズ GB`volumeSizeInGB`。
   - アプリケーションの実行のコンテキストで使用される変数の値 基本的には、アプリケーションにわたされるパラメータ`variables`)。

     これらの変数は、コンテナが実行されるときに置き換えられます。これにより、データセットコンテンツの作成時に提供された別の変数 パラメータを使用して、同じコンテナを実行できます。IotAnalytics Jupyter 拡張機能は、ノートブックでこの変数を自動的に認識し、コンテナ化プロセスの一部として利用可能にすることで、このプロセスを簡略化します。認識された変数を選択することも、独自のカスタム値を追加することもできます。コンテナを実行する前に、システムによってこの各変数は実行時の現在値で置き換えられます。

- 変数の 1 つは、最後のコンテンツがアプリケーションへの入力として使用されるデータセットの名前です 前のステップで作成したデータセットの名前になりますdatasetContentVersionValue:datasetName)。

SQL クエリと差分ウィンドウを使用してデータセットを生成し、アプリケーションでコンテナを使用すると、差分ウィンドウのデータに対して指定した間隔で実行されるスケジュールされた本番稼働用データセット AWS IoT Analytics を作成し、必要な出力を生成して通知を送信します。

必要に応じていつでも本番稼働用データセットアプリケーションを一時停止して、再開できます。本番データセットアプリケーションを再開すると、デフォルトでは AWS IoT Analytics、 は前回の実行以降に到着したが、まだ分析されていないすべてのデータをキャッチアップします。また、一連の連続実行を行うことで本番稼働用データセットジョブ ウィンドウ長を再開する方法を設定することもできます。また、デルタウィンドウで指定したサイズ内に収まるような新しく到着したデータのみを取得して、本番稼働用データセットアプリケーションを再開することもできます。

別のデータセットの作成によりトリガーされるデータセットを作成/定義する際には、以下の制限に注意してください。

- コンテナデータセットのみ、SQL データセットからトリガーできます。
- SQL データセットにより、最大で 10 個のコンテナデータセットをトリガーできます。

SQL データセットからトリガーされるコンテナデータセットを作成するときに、次のエラーが返される場合があります。

- 「データセットのトリガーはコンテナデータセットのみに追加できます」
- 「1 つのトリガーデータセットのみ可能です」

  このエラーは、2 つの異なる SQL データセットからトリガーされたコンテナデータセットを定義しようとする場合に発生します。
- 「トリガーするデータセット <dataset-name> がコンテナデータセットからトリガーできません」

  このエラーは、別のコンテナデータセットからトリガーされた別のコンテナデータセットを定義しようとする場合に発生します。
- 「<N> データセットはすでに <dataset-name> データセットに依存しています」

  このエラーは、すでに 10 個のコンテナデータセットをトリガーした SQL データセットからトリガーされる別のコンテナデータセットを定義しようとする場合に発生します。

- 「厳密に 1 つのトリガータイプを指定する必要があります。」

  このエラーは、スケジュールされたトリガーとデータセットトリガーの両方からトリガーされたデータセットを定義しようとする場合に発生します。

カスタム Docker コンテナの入力/出力変数

このセクションでは、カスタム Docker イメージを実行するプログラムが入力変数を読み取り、出力をアップロードする方法を示しています。

Params ファイル

入力変数と出力のアップロード先は、Docker イメージを実行するインスタンスの JSON ファイル `/opt/ml/input/data/iotanalytics/params`に保存されています。このファイルの内容の例を示します。

```
{
   "Context": {
       "OutputUris": {
           "html": "s3://aws-iot-analytics-dataset-xxxxxxx/notebook/results/
iotanalytics-xxxxxxx/output.html",
           "ipynb": "s3://aws-iot-analytics-dataset-xxxxxxx/notebook/results/
iotanalytics-xxxxxxx/output.ipynb"
       }
   },
   "Variables": {
       "source_dataset_name": "mydataset",
       "source_dataset_version_id": "xxxx",
       "example_var": "hello world!",
       "custom_output": "s3://aws-iot-analytics/dataset-xxxxxxx/notebook/results/
iotanalytics-xxxxxxx/output.txt"
   }
}
```

データセットの名前とバージョン ID に加えて、`Variables` セクションには `iotanalytics:CreateDataset` 呼び出しで指定される変数が含まれています。この例では、変数 `example_var` には 値 `hello world!` が与えられています。また、`custom_output` 変数ではカスタム出力 URI も提供されています。`OutputUris` フィールドには、コンテナがその出力をアップロードできるデフォルトの場所が含まれています。この例では、デフォルトの出力 URI は ipynb 出力および html 出力の両方に提供されています。

入力変数

Docker イメージによって起動されるプログラムは、`params` ファイルから変数を読み取ることができます。以下に示しているのは、`params` ファイルを開き、それを解析して、`example_var` 変数の値を印刷するプログラムの例を以下に示します。

```
import json

with open("/opt/ml/input/data/iotanalytics/params") as param_file:
    params = json.loads(param_file.read())
example_var = params["Variables"]["example_var"]
print(example_var)
```

出力のアップロード

Docker イメージから起動されたプログラムでは、Amazon S3 の場所に出力が保存される可能性もあります。この出力は、"bucket-owner-full-control" アクセスコントロールリストを使用してロードする必要があります。アクセスリストは、アップロードされた出力に対する AWS IoT Analytics サービスコントロールを許可します。この例では、前回のアクセス許可を拡張して、`params` ファイルの `custom_output` で定義される Amazon S3 の場所に `example_var` のコンテンツをアップロードします。

```
import boto3
import json
from urllib.parse import urlparse

ACCESS_CONTROL_LIST = "bucket-owner-full-control"

with open("/opt/ml/input/data/iotanalytics/params") as param_file:
    params = json.loads(param_file.read())
example_var = params["Variables"]["example_var"]

outputUri = params["Variables"]["custom_output"]
# break the S3 path into a bucket and key
bucket = urlparse(outputUri).netloc
key = urlparse(outputUri).path.lstrip("/")

s3_client = boto3.client("s3")
s3_client.put_object(Bucket=bucket, Key=key, Body=example_var, ACL=ACCESS_CONTROL_LIST)
```

アクセス許可

2 つの ロールを作成する必要があります。1 つのロールは、ノートブックをコンテナ化するために SageMaker AI インスタンスを起動するアクセス許可を付与します。もう 1 つのロールはコンテナを実行するために必要となります。

最初のロールは、自動的に作成するか、または手動で作成できます。 AWS IoT Analytics コンソールで新しい SageMaker AI インスタンスを作成する場合、SageMaker AI インスタンスの実行とノートブックのコンテナ化に必要なすべての権限を付与する新しいロールを自動的に作成できます。または、手動でこれらの権限を持つロールを作成できます。これを行うには、`AmazonSageMakerFullAccess` ポリシーがアタッチされたロールを作成し、次のポリシーを追加します。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecr:BatchDeleteImage",
        "ecr:BatchGetImage",
        "ecr:CompleteLayerUpload",
        "ecr:CreateRepository",
        "ecr:DescribeRepositories",
        "ecr:GetAuthorizationToken",
        "ecr:InitiateLayerUpload",
        "ecr:PutImage",
        "ecr:UploadLayerPart"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": "arn:aws:s3:::iotanalytics-notebook-containers/*"
    }
  ]
}
```

コンテナを実行する権限を付与する 2 番目のロールを手動で作成する必要があります。 AWS IoT Analytics コンソールを使用して最初のロールを自動的に作成した場合でも、これを行う必要があります。次のポリシーと信頼ポリシーがアタッチされたロールを作成します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:PutObject",
                "s3:GetObject",
                "s3:PutObjectAcl"
            ],
            "Resource": "arn:aws:s3:::aws-*-dataset-*/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iotanalytics:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:GetAuthorizationToken",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage",
                "ecr:BatchCheckLayerAvailability",
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:GetLogEvents",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
```

```
                "s3:ListBucket",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        }
    ]
}
```

以下に示しているのは、信頼ポリシーの例です。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": ["sagemaker.amazonaws.com", "iotanalytics.amazonaws.com"]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Java と AWS CLI経由での CreateDataset API の使用

データセットを作成します。データセットにより、`queryAction` SQL クエリまたは `containerAction` コンテナ化されたアプリケーションの実行を適用してデータストアから取得されたデータを保存します。このオペレーションによりデータセットのスケルトンが作成されます。データセットは、`CreateDatasetContent` を呼び出して手動で入力するか、指定した `trigger` に従って自動的に入力できます。詳細については、「CreateDataset」および「`CreateDatasetContent`」を参照してください。

トピック

- 例 1 -- SQL データセットを作成する java)
- 例 2 -- デルタウィンドウを使用して SQL データセットを作成する java)
- 例 3 -- 独自のスケジュールトリガーを使用してコンテナデータセットを作成する java)
- 例 4 -- SQL データセットをトリガーとして使用してコンテナデータセットを作成する java)
- 例 5 -- SQL データセットを作成する CLI)

- 例 6 -- デルタウィンドウを使用して SQL データセットを作成する CLI)

例 1 -- SQL データセットを作成する java)

```
CreateDatasetRequest request = new CreateDatasetRequest();
request.setDatasetName(dataSetName);
DatasetAction action = new DatasetAction();

//Create Action
action.setActionName("SQLAction1");
action.setQueryAction(new SqlQueryDatasetAction().withSqlQuery("select * from
 DataStoreName"));

// Add Action to Actions List
List<DatasetAction> actions = new ArrayList<DatasetAction>();
actions.add(action);

//Create Trigger
DatasetTrigger trigger = new DatasetTrigger();
trigger.setSchedule(new Schedule().withExpression("cron(0 12 * * ? *)"));

//Add Trigger to Triggers List
List<DatasetTrigger> triggers = new ArrayList<DatasetTrigger>();
triggers.add(trigger);

// Add Triggers and Actions to CreateDatasetRequest object
request.setActions(actions);
request.setTriggers(triggers);

// Add RetentionPeriod to CreateDatasetRequest object
request.setRetentionPeriod(new RetentionPeriod().withNumberOfDays(10));
final CreateDatasetResult result = iot.createDataset(request);
```

成功した出力:

```
{DatasetName: <datatsetName>, DatasetArn: <datatsetARN>, RetentionPeriod: {unlimited:
 true} or {numberOfDays: 10, unlimited: false}}
```

例 2 -- デルタウィンドウを使用して SQL データセットを作成する java)

```
CreateDatasetRequest request = new CreateDatasetRequest();
```

```
request.setDatasetName(dataSetName);
DatasetAction action = new DatasetAction();

//Create Filter for DeltaTime
QueryFilter deltaTimeFilter = new QueryFilter();
deltaTimeFilter.withDeltaTime(
                new DeltaTime()
                .withOffsetSeconds(-1 * EstimatedDataDelayInSeconds)
                .withTimeExpression("from_unixtime(timestamp)"));

//Create Action
action.setActionName("SQLActionWithDeltaTime");
action.setQueryAction(new SqlQueryDatasetAction()
                .withSqlQuery("SELECT * from DataStoreName")
                .withFilters(deltaTimeFilter));

// Add Action to Actions List
List<DatasetAction> actions = new ArrayList<DatasetAction>();
actions.add(action);

//Create Trigger
DatasetTrigger trigger = new DatasetTrigger();
trigger.setSchedule(new Schedule().withExpression("cron(0 12 * * ? *)"));

//Add Trigger to Triggers List
List<DatasetTrigger> triggers = new ArrayList<DatasetTrigger>();
triggers.add(trigger);

// Add Triggers and Actions to CreateDatasetRequest object
request.setActions(actions);
request.setTriggers(triggers);

// Add RetentionPeriod to CreateDatasetRequest object
request.setRetentionPeriod(new RetentionPeriod().withNumberOfDays(10));
final CreateDatasetResult result = iot.createDataset(request);
```

成功した出力:

```
{DatasetName: <datatsetName>, DatasetArn: <datatsetARN>, RetentionPeriod: {unlimited:
 true} or {numberOfDays: 10, unlimited: false}}
```

例 3 -- 独自のスケジュールトリガーを使用してコンテナデータセットを作成する java)

```
CreateDatasetRequest request = new CreateDatasetRequest();
request.setDatasetName(dataSetName);
DatasetAction action = new DatasetAction();

//Create Action
action.setActionName("ContainerActionDataset");
action.setContainerAction(new ContainerDatasetAction()
        .withImage(ImageURI)
        .withExecutionRoleArn(ExecutionRoleArn)
        .withResourceConfiguration(
                new ResourceConfiguration()
                .withComputeType(new ComputeType().withAcu(1))
                .withVolumeSizeInGB(1))
        .withVariables(new Variable()
        .withName("VariableName")
        .withStringValue("VariableValue"));

// Add Action to Actions List
List<DatasetAction> actions = new ArrayList<DatasetAction>();
actions.add(action);

//Create Trigger
DatasetTrigger trigger = new DatasetTrigger();
trigger.setSchedule(new Schedule().withExpression("cron(0 12 * * ? *)"));

//Add Trigger to Triggers List
List<DatasetTrigger> triggers = new ArrayList<DatasetTrigger>();
triggers.add(trigger);

// Add Triggers and Actions to CreateDatasetRequest object
request.setActions(actions);
request.setTriggers(triggers);

// Add RetentionPeriod to CreateDatasetRequest object
request.setRetentionPeriod(new RetentionPeriod().withNumberOfDays(10));
final CreateDatasetResult result = iot.createDataset(request);
```

成功した出力:

```
{DatasetName: <datatsetName>, DatasetArn: <datatsetARN>, RetentionPeriod: {unlimited:
 true} or {numberOfDays: 10, unlimited: false}}
```

例 4 -- SQL データセットをトリガーとして使用してコンテナデータセットを作成する java)

```
CreateDatasetRequest request = new CreateDatasetRequest();
request.setDatasetName(dataSetName);
DatasetAction action = new DatasetAction();

//Create Action
action.setActionName("ContainerActionDataset");
action.setContainerAction(new ContainerDatasetAction()
        .withImage(ImageURI)
        .withExecutionRoleArn(ExecutionRoleArn)
        .withResourceConfiguration(
                new ResourceConfiguration()
                .withComputeType(new ComputeType().withAcu(1))
                .withVolumeSizeInGB(1))
        .withVariables(new Variable()
        .withName("VariableName")
        .withStringValue("VariableValue"));

// Add Action to Actions List
List<DatasetAction> actions = new ArrayList<DatasetAction>();
actions.add(action);

//Create Trigger
DatasetTrigger trigger = new DatasetTrigger()
        .withDataset(new TriggeringDataset()
                .withName(TriggeringSQLDataSetName));

//Add Trigger to Triggers List
List<DatasetTrigger> triggers = new ArrayList<DatasetTrigger>();
triggers.add(trigger);

// Add Triggers and Actions to CreateDatasetRequest object
request.setActions(actions);
request.setTriggers(triggers);
final CreateDatasetResult result = iot.createDataset(request);
```

成功した出力:

```
{DatasetName: <datatsetName>, DatasetArn: <datatsetARN>}
```

例 5 -- SQL データセットを作成する CLI)

```
aws iotanalytics --endpoint <EndPoint>  --region <Region> create-dataset --dataset-
name="<dataSetName>" --actions="[{\"actionName\":\"<ActionName>\", \"queryAction\":
{\"sqlQuery\":\"<SQLQuery>\"}}]" --retentionPeriod numberOfDays=10
```

成功した出力:

```
{
    "datasetName": "<datasetName>",
    "datasetArn": "<datatsetARN>",
    "retentionPeriod": {unlimited: true} or {numberOfDays: 10, unlimited: false}
}
```

例 6 -- デルタウィンドウを使用して SQL データセットを作成する CLI)

デルタウィンドウは、重複することなく連続する時間間隔であり、ユーザーが定義します。デルタウィンドウを使用すると、前回の分析以降にデータストアに到着した新しいデータを使用してデータセットコンテンツを作成し、これらのデータに対して分析を実行できます。デルタウィンドウを作成するには、データセットの queryAction の filters 部分に deltaTime を設定します (データセットを作成)。通常は、間隔トリガーも設定して、データセットコンテンツを自動的に作成します triggers:schedule:expression)。基本的には、この操作で特定の時間ウィンドウに到着したメッセージをフィルタリングし、以前の時間ウィンドウのメッセージに含まれたデータが重複してカウントされないようにします。

この例では、前回以降に到着したデータのみを使用して 15 分ごとに新しいデータセットコンテンツを自動的に作成する新しいデータセットを作成します。メッセージが指定されたデータストアに到着するまでに 3 分間の遅延を許可する 3 分 180 秒deltaTime オフセットを指定します。そのため、データセットコンテンツが午前 10 時 30 分に作成された場合、使用されるデータ データセットコンテンツに含まれるは、タイムスタンプが午前 10 時 12 分から午前 10 時 27 分の間になります つまり、午前 10 時 30 分 - 15 分 - 3 分から午前 10 時 30 分 - 3 分)。

```
aws iotanalytics --endpoint <EndPoint>  --region <Region> create-dataset --cli-input-
json file://delta-window.json
```

ファイル delta-window.json には、次のものが含まれています。

```
{
  "datasetName": "delta_window_example",
  "actions": [
    {
      "actionName": "delta_window_action",
      "queryAction": {
        "sqlQuery": "SELECT temperature, humidity, timestamp FROM my_datastore",
        "filters": [
          {
            "deltaTime": {
              "offsetSeconds": -180,
              "timeExpression": "from_unixtime(timestamp)"
            }
          }
        ]
      }
    }
  ],
  "triggers": [
    {
      "schedule": {
        "expression": "cron(0/15 * * * ? *)"
      }
    }
  ]
}
```

成功した出力:

```
{
    "datasetName": "<datasetName>",
    "datasetArn": "<datatsetARN>",
}
```

ノートブックをコンテナ化する

このセクションでは、Jupyter ノートブックを使用して独自の Docker コンテナを構築する方法について説明します。サードパーティー製のノートブックを再利用するとセキュリティ上のリスクがあります。含まれるコンテナはユーザーの許可を得て任意のコードを実行できます。さらに、ノートブッ

クによって生成された HTML を AWS IoT Analytics コンソールに表示することができ、HTML を表示するコンピュータに潜在的な攻撃ベクトルを提供します。使用する前に、サードパーティー製のノートブックの著者を信頼できることを確認します。

高度な分析機能を実行する 1 つの方法として、Jupyter Notebook を使用します。Jupyter Notebook は、機械学習と様々な統計的分析を実行できる強力なデータサイエンスツールを備えています。詳細については、「ノートブックテンプレート」を参照してください。現在 JupyterLab 内でのコンテナ化はサポートされていないことに注意してください。) Jupyter Notebook とライブラリをコンテナにパッケージ化して、定義した差分時間枠 AWS IoT Analytics 中に が受信した新しいデータのバッチで定期的に実行できます。コンテナと指定された時間ウィンドウでキャプチャされるセグメント化された新しいデータを使用する分析ジョブをスケジュールし、この先にスケジュールされた分析用にこのジョブの出力を保存できます。

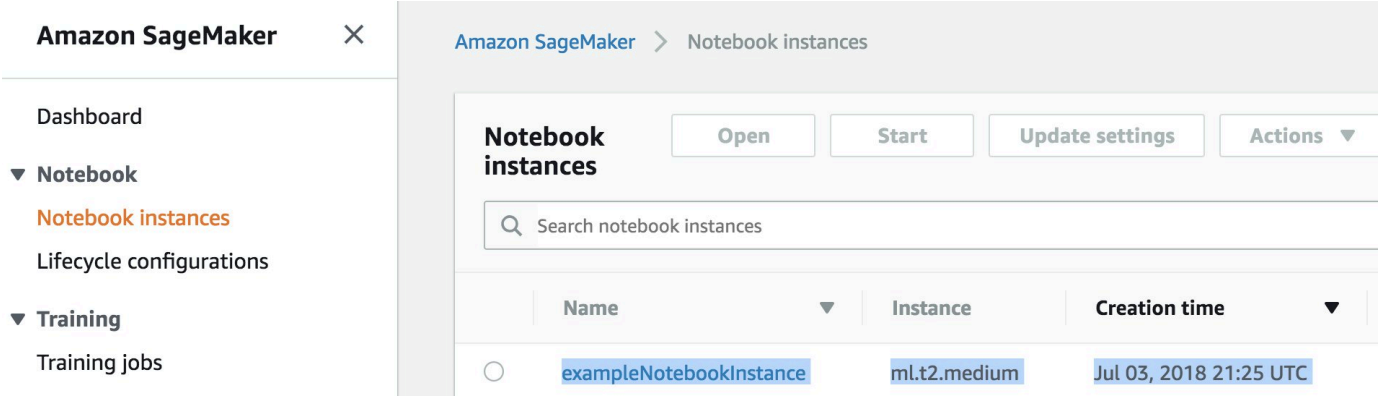
2018 年 8 月 23 日以降に AWS IoT Analytics コンソールを使用して SageMaker AI インスタンスを作成した場合、コンテナ化拡張機能のインストールが自動的に完了し、コンテナ化されたイメージの作成を開始できます。それ以外の場合は、このセクションに記載されている手順に従って、SageMaker AI インスタンスでノートブックのコンテナ化を有効にします。以下では、Amazon EC2 にコンテナイメージをアップロードできるように SageMaker AI 実行ロールを変更し、コンテナ化拡張機能をインストールします。

AWS IoT Analytics コンソールで作成されていないノートブックインスタンスのコンテナ化を有効にする

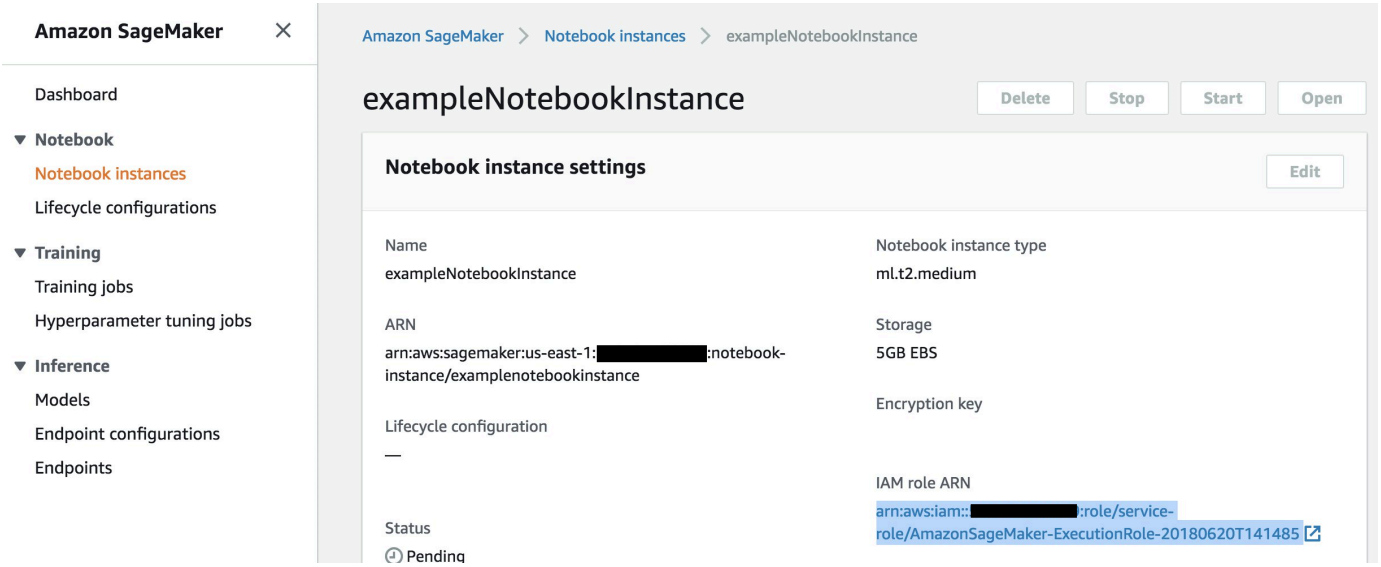
以下の手順を実行する代わりに、 AWS IoT Analytics コンソールを使用して新しい SageMaker AI インスタンスを作成することをお勧めします。新しいインスタンスでは、自動的にコンテナ化がサポートされます。

ここに示すようにコンテナ化を有効にした後に SageMaker AI インスタンスを再起動する場合、IAM ロールとポリシーを再追加する必要はありませんが、最後のステップに示すように、拡張機能を再インストールする必要があります。

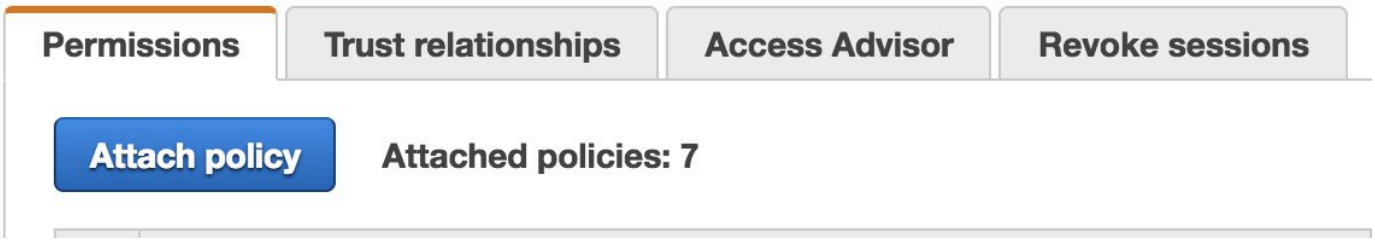
1. ノートブックインスタンスに Amazon ECS へのアクセスを許可するには、SageMaker AI ページで SageMaker AI インスタンスを選択します。



2. IAM ロール ARN で、SageMaker AI 実行ロールを選択します。

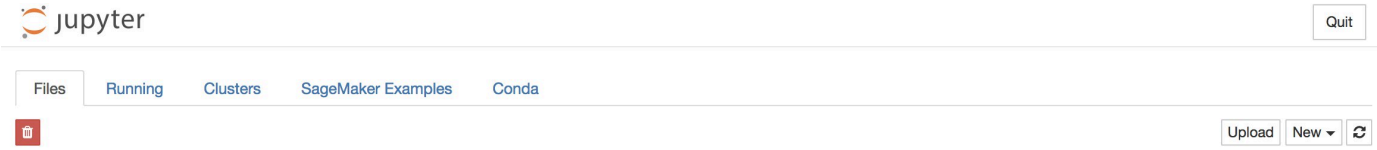


3. Attach Policy ポリシーのアタッチを選択したら、「Permissions アクセス許可)」に示すようにポリシーを定義してアタッチします。`AmazonSageMakerFullAccess` ポリシーがまだアタッチされていない場合は、ポリシーをアタッチします。



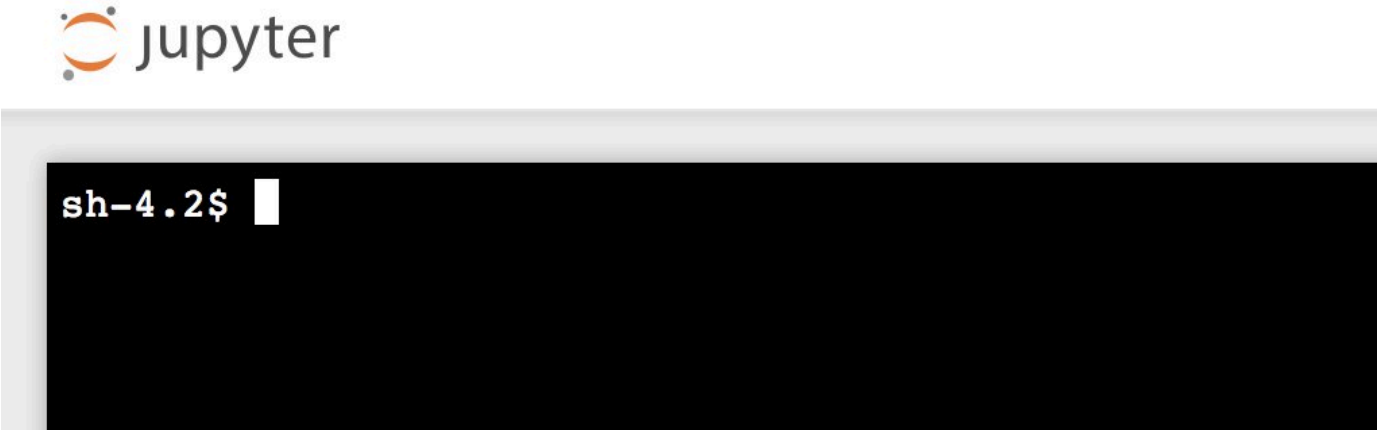
また、Amazon S3 からコンテナ化コードをダウンロードしてノートブックインスタンスにインストールする必要があります。最初のステップは、SageMaker AI インスタンスのターミナルにアクセスすることです。

1. Jupyter 内で、新規を選択します。



2. 表示されるメニューで、ターミナルを選択します。

3. ターミナル内で、コードをダウンロードするための次のコマンドを入力し、解凍してインストールします。これらのコマンドは、この SageMaker AI インスタンスでノートブックによって実行されるすべてのプロセスを強制終了することに注意してください。




```
cd /tmp

aws s3 cp s3://iotanalytics-notebook-containers/iota_notebook_containers.zip /tmp

unzip iota_notebook_containers.zip

cd iota_notebook_containers

chmod u+x install.sh
```

```
./install.sh
```

拡張機能が有効になりインストールされるまで、1 ~ 2 分ほど待ちます。

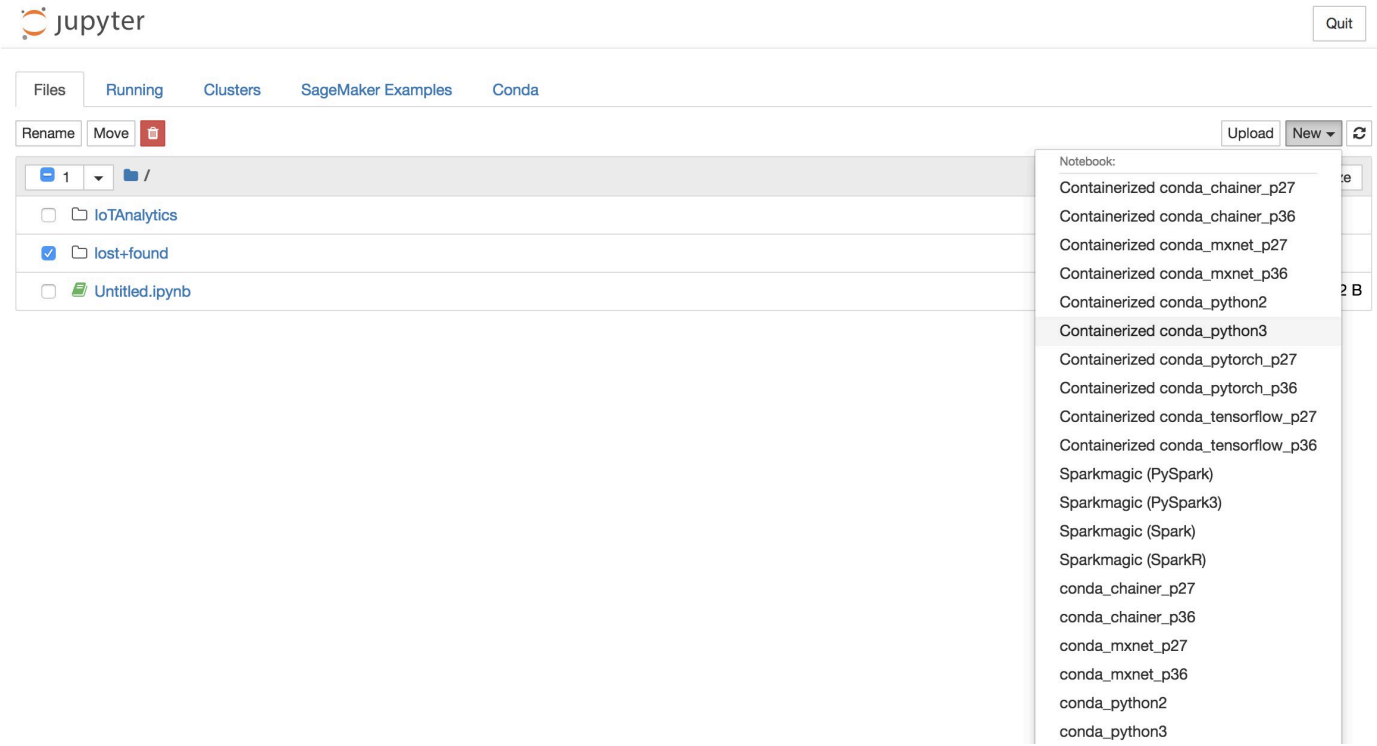
ノートブックのコンテナ化拡張機能を更新する

2018 年 8 月 23 日以降に AWS IoT Analytics コンソールで SageMaker AI インスタンスを作成した場合、コンテナ化拡張機能は自動的にインストールされました。SageMaker AI コンソールからインスタンスを再起動することで、拡張機能を更新できます。拡張機能を手動でインストールした場合は、AWS IoT Analytics 「コンソールで作成されていないノートブックインスタンスのコンテナ化を有効にする」に記載されているターミナルコマンドを再実行して拡張機能を更新できます。

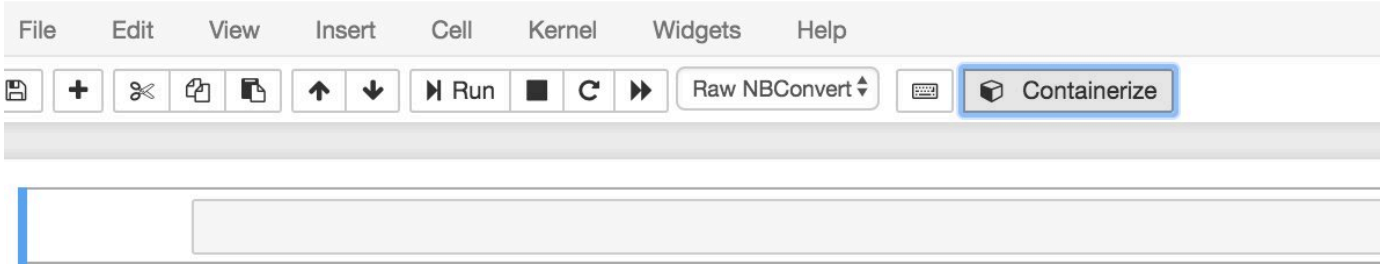
コンテナ化イメージを作成する

このセクションでは、ノートブックをコンテナ化するために必要な手順を示します。開始するには、Jupyter Notebook にアクセスして、コンテナ化カーネルを使用してノートブックを作成します。

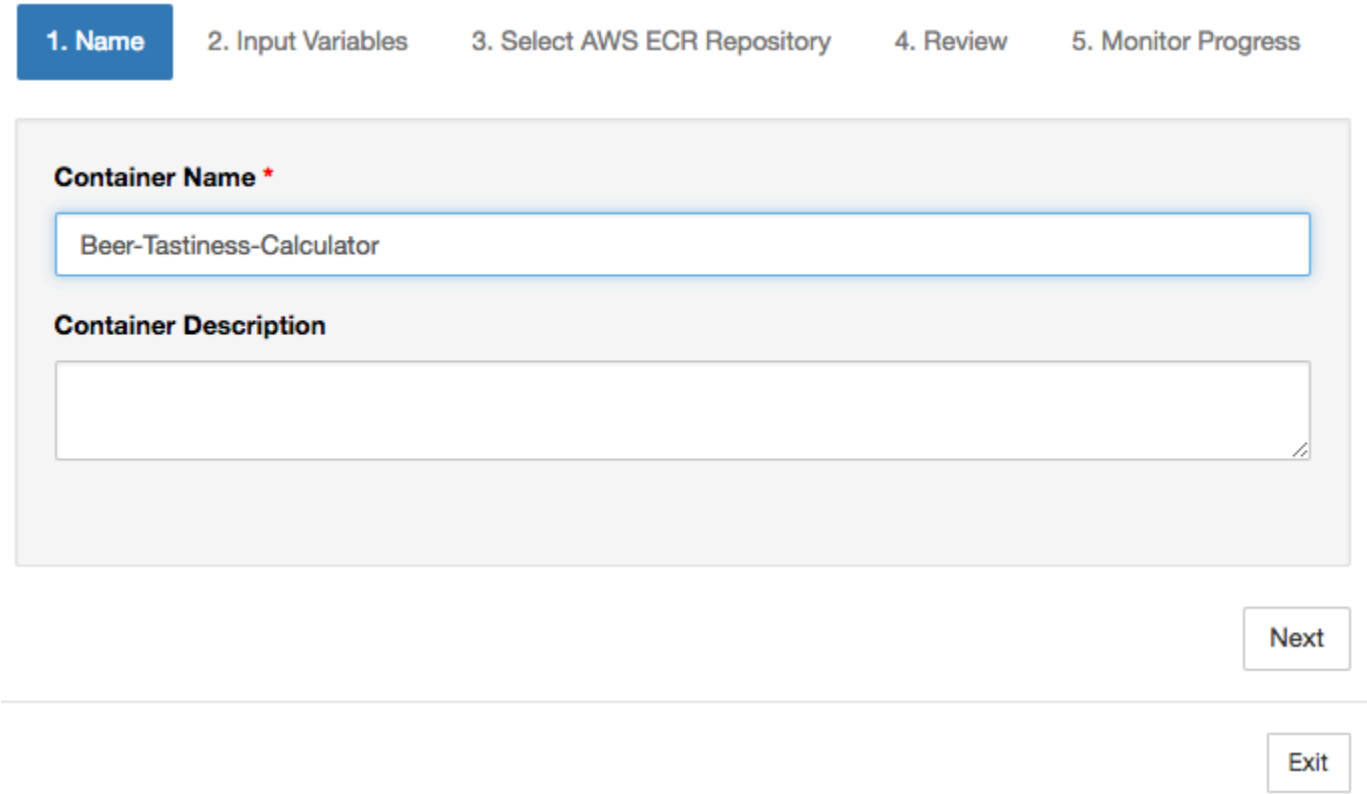
1. Jupyter Notebook で New 新規を選択し、ドロップダウンリストからカーネルのタイプを選択します。カーネルのタイプは、最初を「コンテナ化」に、最後を選択すべきカーネルにする必要があります。 たとえば、単純な Python 3.0 環境が必要な場合 conda_python3 など)、「Containerized conda_python3」を選択します)。



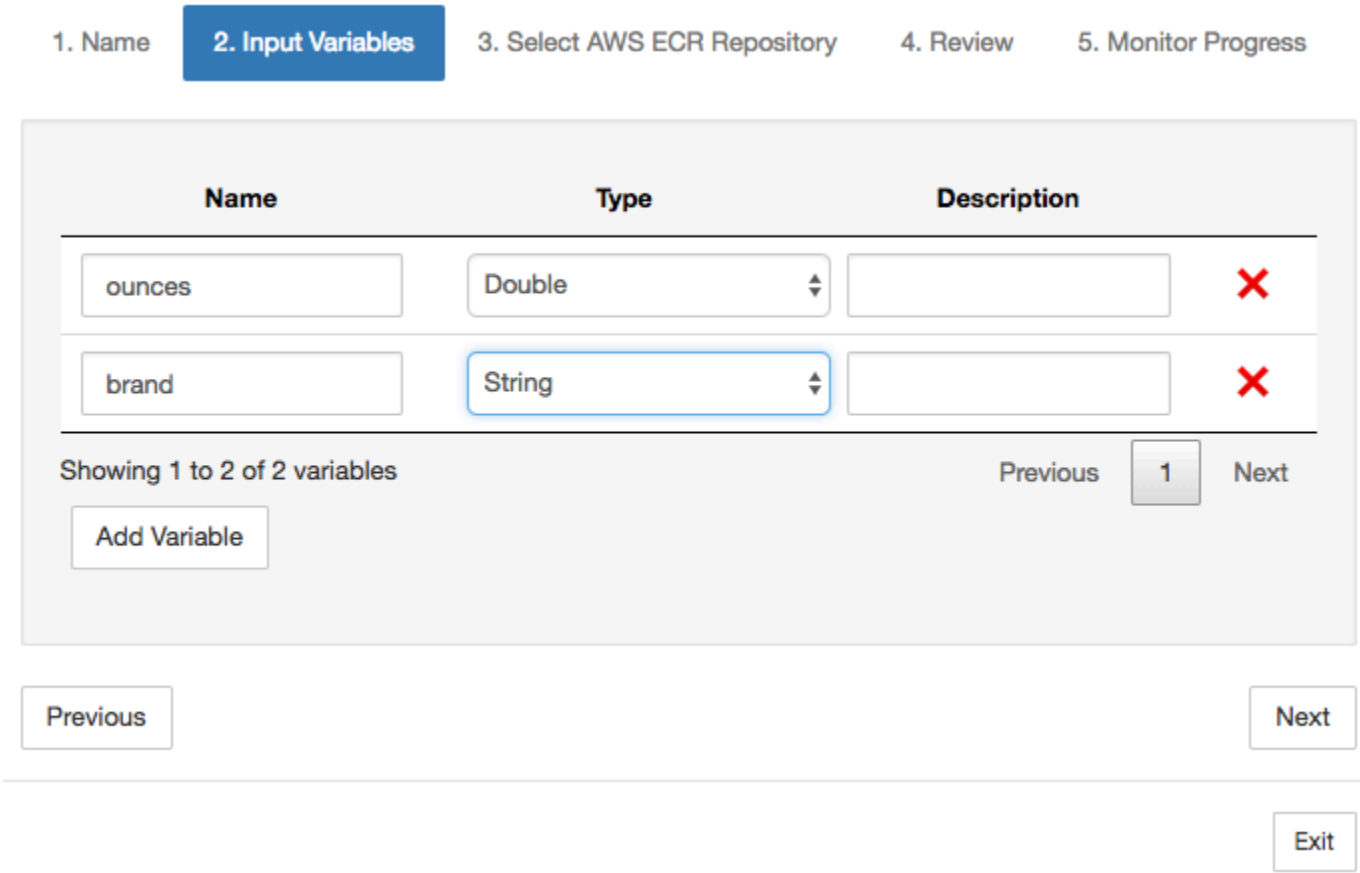
2. ノートブックで作業が完了し、これをコンテナ化するには、コンテナ化を選択します。



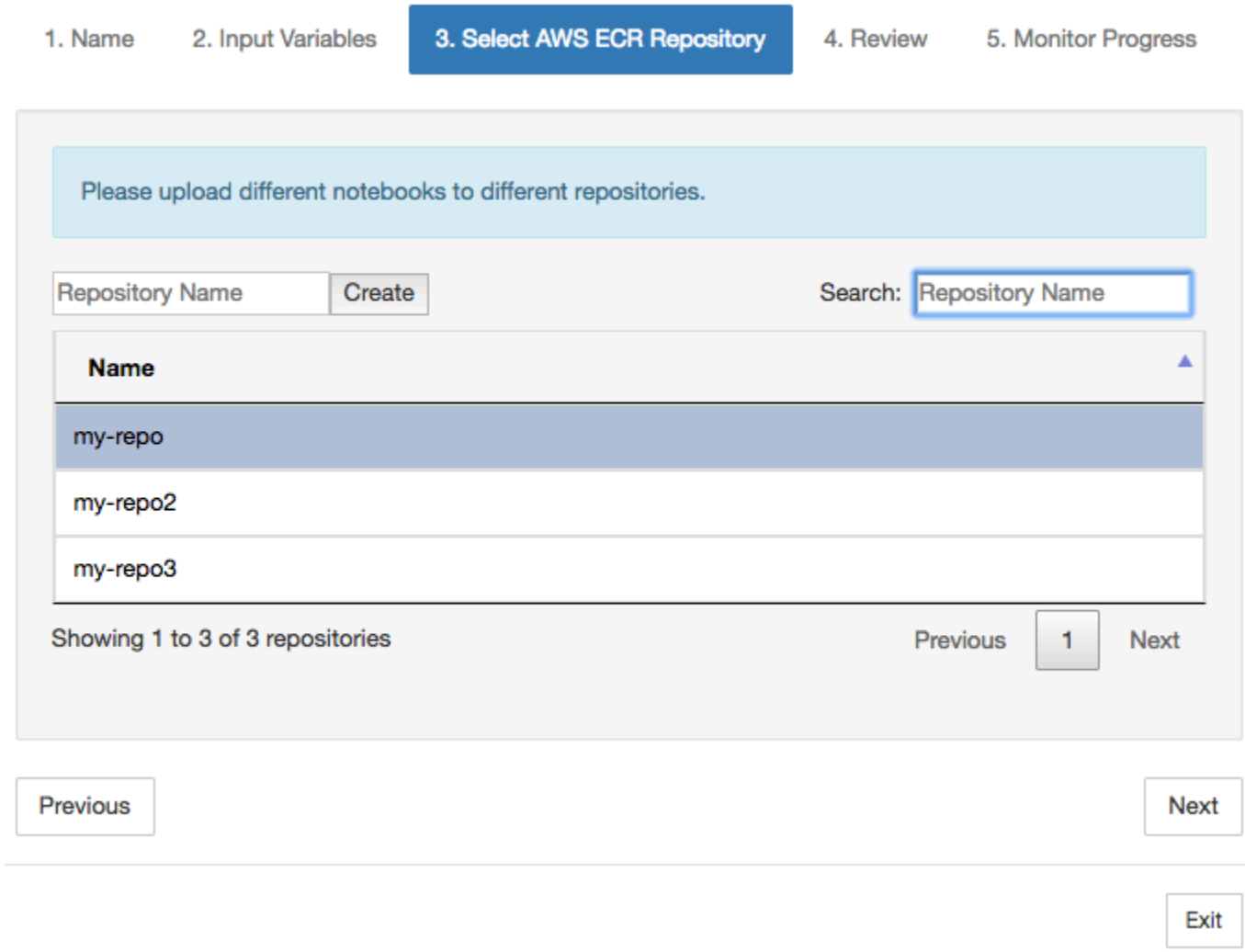
3. コンテナ化のノートブックの名前を入力します。オプションとして説明を入力することもできます。



4. ノートブックを呼び出す 入力変数 パラメータを指定します。ノートブックから自動的に検出される入力変数を選択することも、カスタム変数を定義することもできます。以前にノートブックを実行したことがある場合にのみ、入力変数が検出されることに注意してください。) 各入力変数でタイプを選択します。また、オプションで入力変数の説明も入力できます。

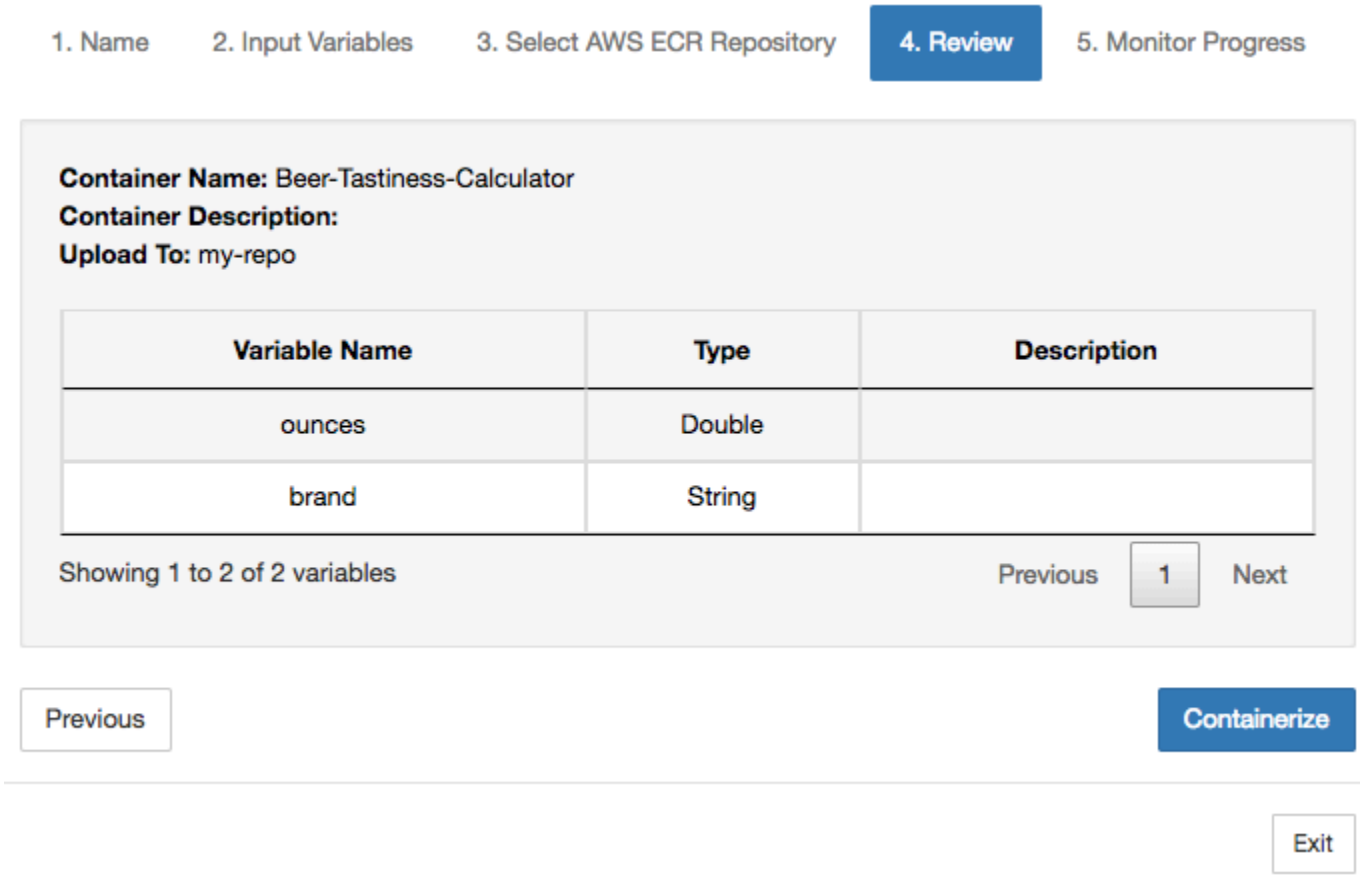


5. ノートブックから作成されたイメージがアップロードされる先の Amazon ECR リポジトリを選択します。

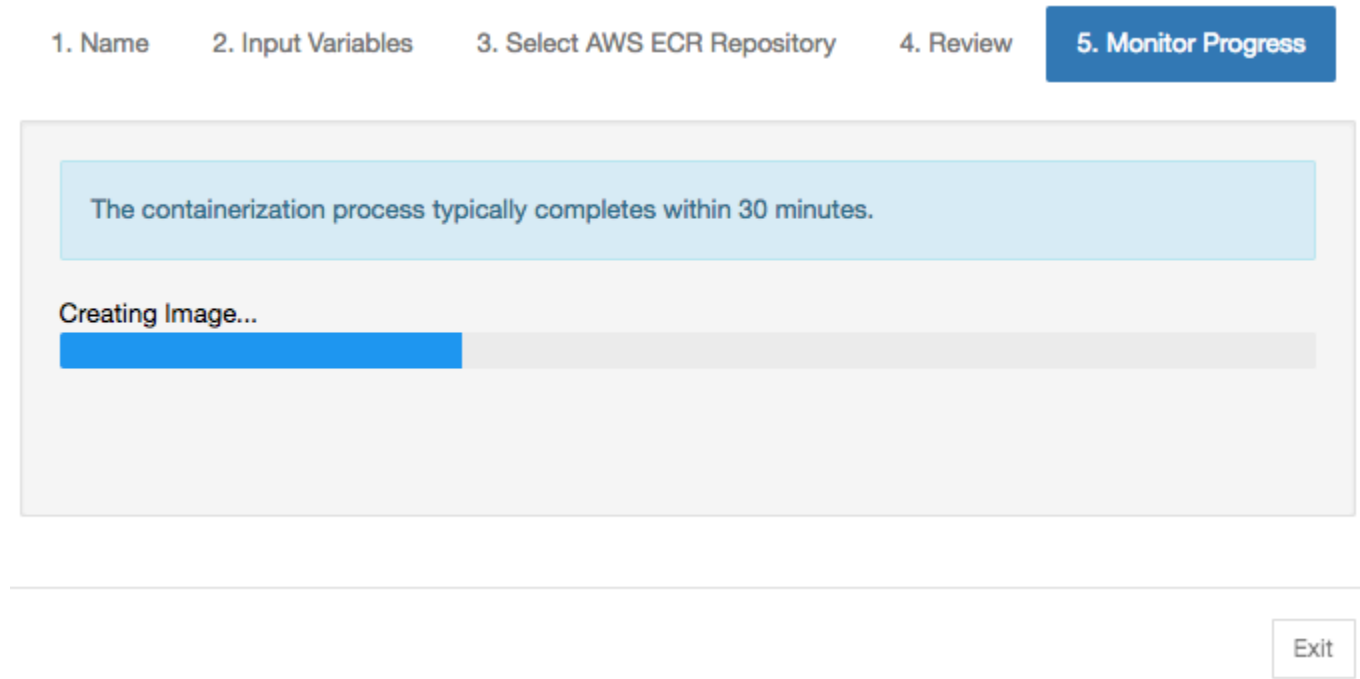


6. コンテナ化を選択してプロセスを開始します。

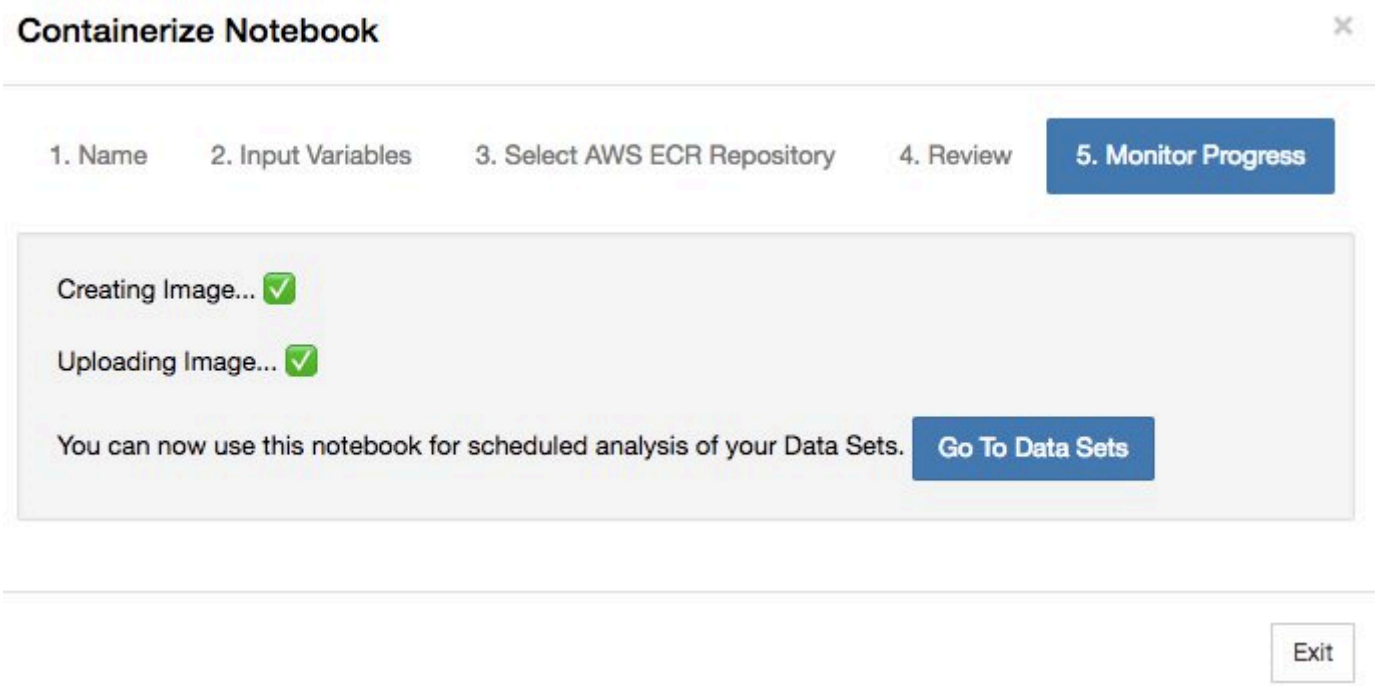
   入力をまとめた概要が表示されます。プロセスは開始してしまうとキャンセルできなくなるので注意してください。このプロセスには最長で 1 時間程度かかることがあります。



7. 次のページに進行状況が表示されます。



8. 誤ってブラウザを閉じた場合には、 AWS IoT Analytics コンソールの ノートブック セクションからコンテナ化プロセスのステータスをモニタリングできます。
9. プロセスが完了すると、コンテナ化されたイメージは Amazon ECR に保存され、使用できるようになります。



分析用カスタムコンテナの使用

このセクションでは、Jupyter ノートブックを使用して独自の Docker コンテナを構築する方法について説明します。サードパーティー製のノートブックを再利用するとセキュリティ上のリスクがあります。含まれるコンテナはユーザーの許可を得て任意のコードを実行できます。さらに、ノートブックによって生成された HTML を AWS IoT Analytics コンソールに表示することができ、HTML を表示するコンピュータに潜在的な攻撃ベクトルを提供します。使用する前に、サードパーティー製のノートブックの著者を信頼できることを確認します。

独自のカスタムコンテナを作成し、 AWS IoT Analytics サービスで実行できます。これを行うには、Docker イメージをセットアップし、Amazon ECR にアップロードしてから、コンテナアクションを実行するデータセットを設定します。このセクションでは、オクターブ を使用したプロセスの例を示します。

このチュートリアルでは、以下を前提としています。

- オクターブ がローカルコンピュータにインストールされていること

- ローカルコンピュータでセットアップされた Docker アカウント
- Amazon ECR または AWS IoT Analytics アクセスを持つ AWS アカウント

ステップ 1: Docker イメージをセットアップする

このチュートリアルには 3 つの主なファイルが必要となります。この名前と内容は以下のとおりです。

- `Dockerfile` – Docker のコンテナ化プロセスの初期セットアップ。

```
FROM ubuntu:16.04

# Get required set of software
RUN apt-get update
RUN apt-get install -y software-properties-common
RUN apt-get install -y octave
RUN apt-get install -y python3-pip

# Get boto3 for S3 and other libraries
RUN pip3 install --upgrade pip
RUN pip3 install boto3
RUN pip3 install urllib3

# Move scripts over
ADD moment moment
ADD run-octave.py run-octave.py

# Start python script
ENTRYPOINT ["python3", "run-octave.py"]
```

- `run-octave.py` – から JSON を解析し AWS IoT Analytics、Octave スクリプトを実行し、アーティファクトを Amazon S3 にアップロードします。

```
import boto3
import json
import os
import sys
from urllib.parse import urlparse

# Parse the JSON from IoT Analytics
with open('/opt/ml/input/data/iotanalytics/params') as params_file:
    params = json.load(params_file)
```

```
variables = params['Variables']

order = variables['order']
input_s3_bucket = variables['inputDataS3BucketName']
input_s3_key = variables['inputDataS3Key']
output_s3_uri = variables['octaveResultS3URI']

local_input_filename = "input.txt"
local_output_filename = "output.mat"

# Pull input data from S3...
s3 = boto3.resource('s3')
s3.Bucket(input_s3_bucket).download_file(input_s3_key, local_input_filename)

# Run Octave Script
os.system("octave moment {} {} {}".format(local_input_filename,
 local_output_filename, order))

# # Upload the artifacts to S3
output_s3_url = urlparse(output_s3_uri)
output_s3_bucket = output_s3_url.netloc
output_s3_key = output_s3_url.path[1:]

s3.Object(output_s3_bucket, output_s3_key).put(Body=open(local_output_filename,
 'rb'), ACL='bucket-owner-full-control')
```

- `moment` – 入力ファイルまたは出力ファイルと指定した順序に基づいてタイミングを計算するシンプルな オクターブ スクリプト。

```
#!/usr/bin/octave -qf

arg_list = argv ();
input_filename = arg_list{1};
output_filename = arg_list{2};
order = str2num(arg_list{3});

[D,delimiterOut]=importdata(input_filename)
M = moment(D, order)

save(output_filename,'M')
```

1. 各ファイルの内容をダウンロードします。新規のディレクトリを作成し、その中にすべてのファイルを配置して、cd をそのディレクトリに移動させます。
2. 以下のコマンドを実行してください。

```
docker build -t octave-moment .
```

3. Docker リポジトリに新しいイメージが表示されます。次のコマンドを使用してインストールします。

```
docker image ls | grep octave-moment
```

ステップ 2: Docker イメージを Amazon ECR リポジトリにアップロードする

1. Amazon ECR でリポジトリを作成します。

```
aws ecr create-repository --repository-name octave-moment
```

2. Docker 環境へのログインを取得します。

```
aws ecr get-login
```

3. 出力をコピーして実行します。出力は次のようになります。

```
docker login -u AWS -p password -e none https://your-aws-account-
id.dkr.ecr..amazonaws.com
```

4. Amazon ECR リポジトリタグを使用して、作成したイメージをタグ付けします。

```
docker tag your-image-id  your-aws-account-id.dkr.ecr.region.amazonaws.com/octave-
moment
```

5. Amazon ECR にイメージをプッシュします。

```
docker push your-aws-account-id.dkr.ecr.region.amazonaws.com/octave-moment
```

ステップ 3: サンプルデータを Amazon S3 バケットにアップロードする

1. ファイル 「`input.txt`」 に以下をダウンロードします。

```
0.857549  -0.987565  -0.467288  -0.252233  -2.298007
 0.030077  -1.243324  -0.692745   0.563276   0.772901
-0.508862  -0.404303  -1.363477  -1.812281  -0.296744
-0.203897   0.746533   0.048276   0.075284   0.125395
 0.829358   1.246402  -1.310275  -2.737117   0.024629
 1.206120   0.895101   1.075549   1.897416   1.383577
```

2. 「`octave-sample-data-`*`your-aws-account-id`*」という Amazon S3 バケットを作成します。
3. 作成した Amazon S3 バケットにファイル「`input.txt`」をアップロードします。これで、ファイル「`input.txt`」が含まれている `octave-sample-data-`*`your-aws-account-id`* という名前のバケットができました。

ステップ 4: コンテナの実行ロールを作成する

1. 次の JSON を `role1.json` という名前のファイルにコピーします。*`your-aws-account-id`* を AWS アカウント ID に置き換え、*`aws-region`* を AWS リソースの AWS リージョンに置き換えます。

> **Note**
>
> この例には、「混乱した代理」問題から保護するためのグローバル条件コンテキストキーが含まれています。詳細については、「the section called “サービス間の混乱した代理の防止”」を参照してください。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "sagemaker.amazonaws.com",
                    "iotanalytics.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole",
            "Condition": {
```

```
                "StringEquals": {
                  "aws:SourceAccount": "your-aws-account-id"
                },
                "ArnLike": {
                  "aws:SourceArn": "arn:aws:iotanalytics:aws-region:your-aws-account-
id:dataset/DOC-EXAMPLE-DATASET"
                }
              }
    ]
}
```

2. ダウンロードした ファイルを使用して AWS IoT Analytics、SageMaker AI および へのアクセス許可を付与`role1.json`するロールを作成します。

```
aws iam create-role --role-name container-execution-role --assume-role-policy-
document file://role1.json
```

3. 「`policy1.json`」という名前のファイルに以下をダウンロードし、*your-account-id* を自分のアカウント ID に置き換えます `Statement:Resource` の下の 2 つ目の ARN を参照)。

```
{
 "Version": "2012-10-17",
 "Statement": [
   {
     "Effect": "Allow",
     "Action": [
       "s3:GetBucketLocation",
       "s3:PutObject",
       "s3:GetObject",
       "s3:PutObjectAcl"
     ],
     "Resource": [
       "arn:aws:s3:::*-dataset-*/*",
       "arn:aws:s3:::octave-sample-data-your-account-id/*"
   },
   {
     "Effect": "Allow",
     "Action": [
       "iotanalytics:*"
     ],
     "Resource": "*"
   },
   {
```

```
      "Effect": "Allow",
      "Action": [
        "ecr:GetAuthorizationToken",
        "ecr:GetDownloadUrlForLayer",
        "ecr:BatchGetImage",
        "ecr:BatchCheckLayerAvailability",
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:DescribeLogStreams",
        "logs:GetLogEvents",
        "logs:PutLogEvents"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketLocation",
        "s3:ListBucket",
        "s3:ListAllMyBuckets"
      ],
      "Resource" : "*"
    }
  ]
}
```

4. ダウンロードした「`policy.json`」ファイルを使用して、IAM ポリシーを作成します。

```
aws iam create-policy --policy-name ContainerExecutionPolicy --policy-document
 file://policy1.json
```

5. ロールへのポリシーの付与

```
aws iam attach-role-policy --role-name container-execution-role --policy-arn
 arn:aws:iam::your-account-id:policy/ContainerExecutionPolicy
```

ステップ 5: コンテナアクションを使用してデータセットを作成する

1. 「`cli-input.json`」という名前のファイルに以下をダウンロードし、*`your-account-id`* と *`region`* のすべてのインスタンスを適切な値に置き換えます。

```
{
```

```
    "datasetName": "octave_dataset",
    "actions": [
        {
            "actionName": "octave",
            "containerAction": {
                "image": "your-account-id.dkr.ecr.region.amazonaws.com/octave-
moment",
                "executionRoleArn": "arn:aws:iam::your-account-id:role/container-
execution-role",
                "resourceConfiguration": {
                    "computeType": "ACU_1",
                    "volumeSizeInGB": 1
                },
                "variables": [
                    {
                        "name": "octaveResultS3URI",
                        "outputFileUriValue": {
                            "fileName": "output.mat"
                        }
                    },
                    {
                        "name": "inputDataS3BucketName",
                        "stringValue": "octave-sample-data-your-account-id"
                    },
                    {
                        "name": "inputDataS3Key",
                        "stringValue": "input.txt"
                    },
                    {
                        "name": "order",
                        "stringValue": "3"
                    }
                ]
            }
        }
    ]
}
```

2. ダウンロードして編集した「cli-input.json」ファイルを使用して、データセットを作成します。

```
aws iotanalytics create-dataset –cli-input-json file://cli-input.json
```

ステップ 6: データセットコンテンツの生成を呼び出す

1. 以下のコマンドを実行してください。

```
aws iotanalytics create-dataset-content --dataset-name octave-dataset
```

ステップ 7: データセットコンテンツを取得する

1. 以下のコマンドを実行してください。

```
aws iotanalytics get-dataset-content --dataset-name octave-dataset --version-id \
$LATEST
```

2. `DatasetContentState` が `SUCCEEDED` になるまで数分間かかる場合があります。

ステップ 8: オクターブ で出力を印刷する

1. オクターブ シェルを使用し、以下のコマンドを実行してコンテナからの出力を印刷します。

```
bash> octave
octave> load output.mat
octave> disp(M)
-0.016393 -0.098061 0.380311 -0.564377 -1.318744
```

AWS IoT Analytics データの視覚化

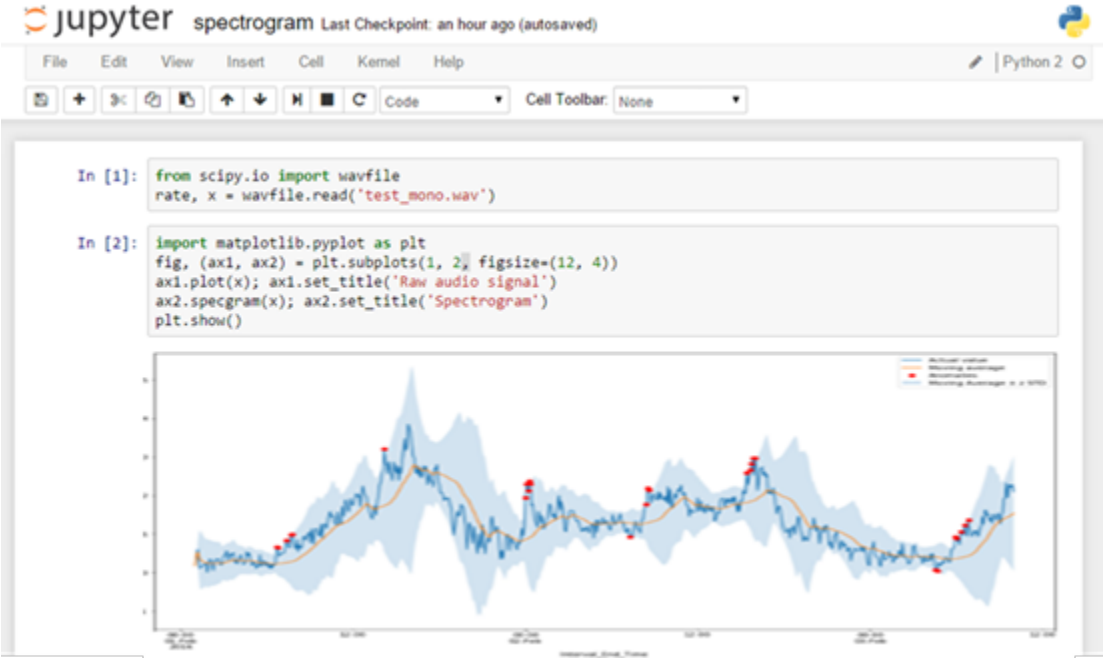
AWS IoT Analytics データを視覚化するには、 AWS IoT Analytics コンソールまたは QuickSight を使用できます。

トピック

- コンソールを使用した AWS IoT Analytics データの視覚化
- QuickSight を使用した AWS IoT Analytics データの視覚化

コンソールを使用した AWS IoT Analytics データの視覚化

AWS IoT Analytics は、コンテナデータセットの HTML 出力 (ファイル にありますoutput.html) を AWS IoT Analytics コンソールのコンテナデータセットコンテンツページに埋め込むことができます。たとえば、Jupyter ノートブックを実行するコンテナデータセットを定義し、Jupyter ノートブックで可視化を作成すると、データセットは次のようになる可能性があります。



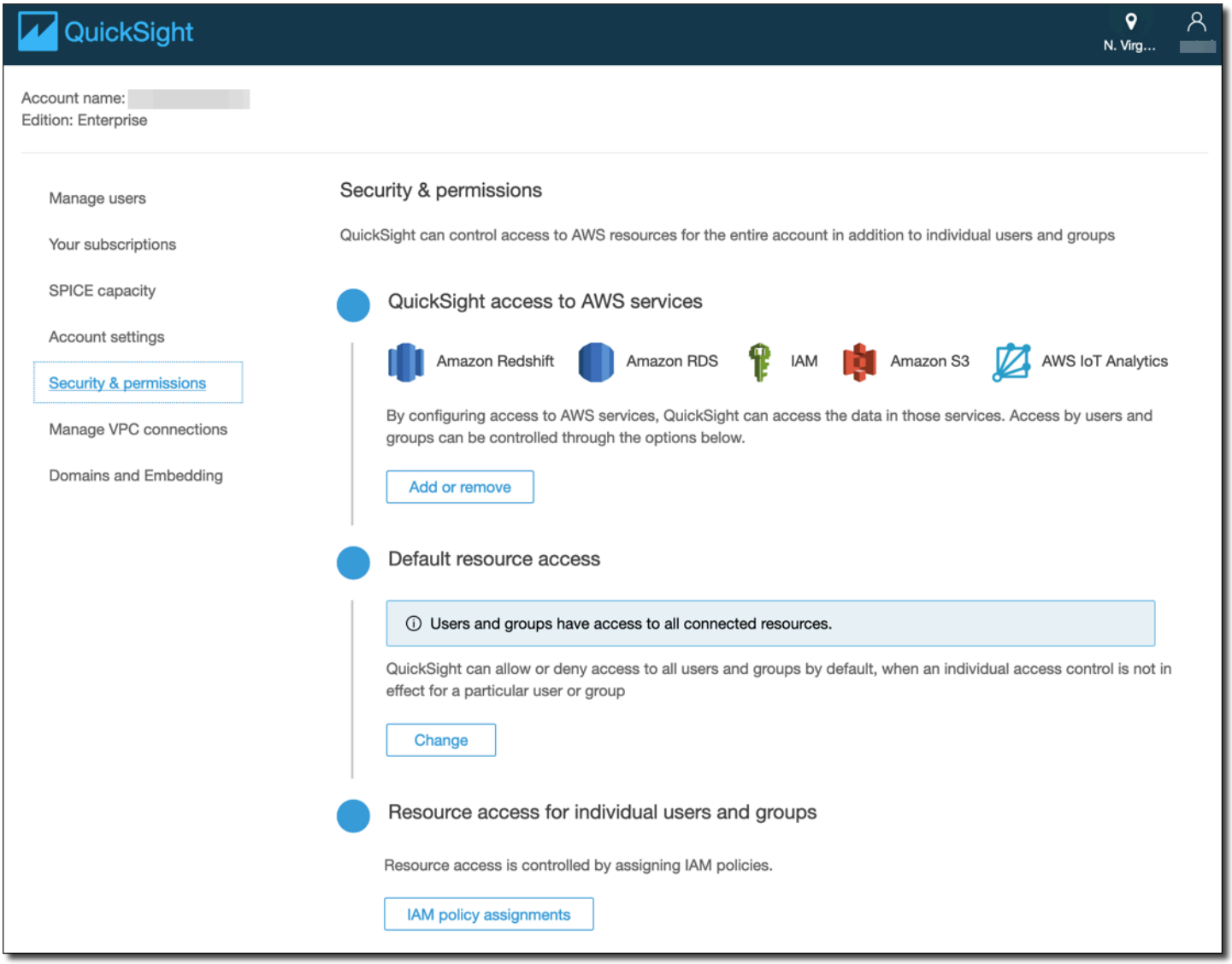
コンテナデータセットコンテンツが作成された後、コンソールの [Data Set] (データセット) コンテンツページでこの可視化を表示できます。

Jupyter ノートブックを実行するコンテナデータセットの作成の詳細については、「ワークフローの自動化」を参照してください。

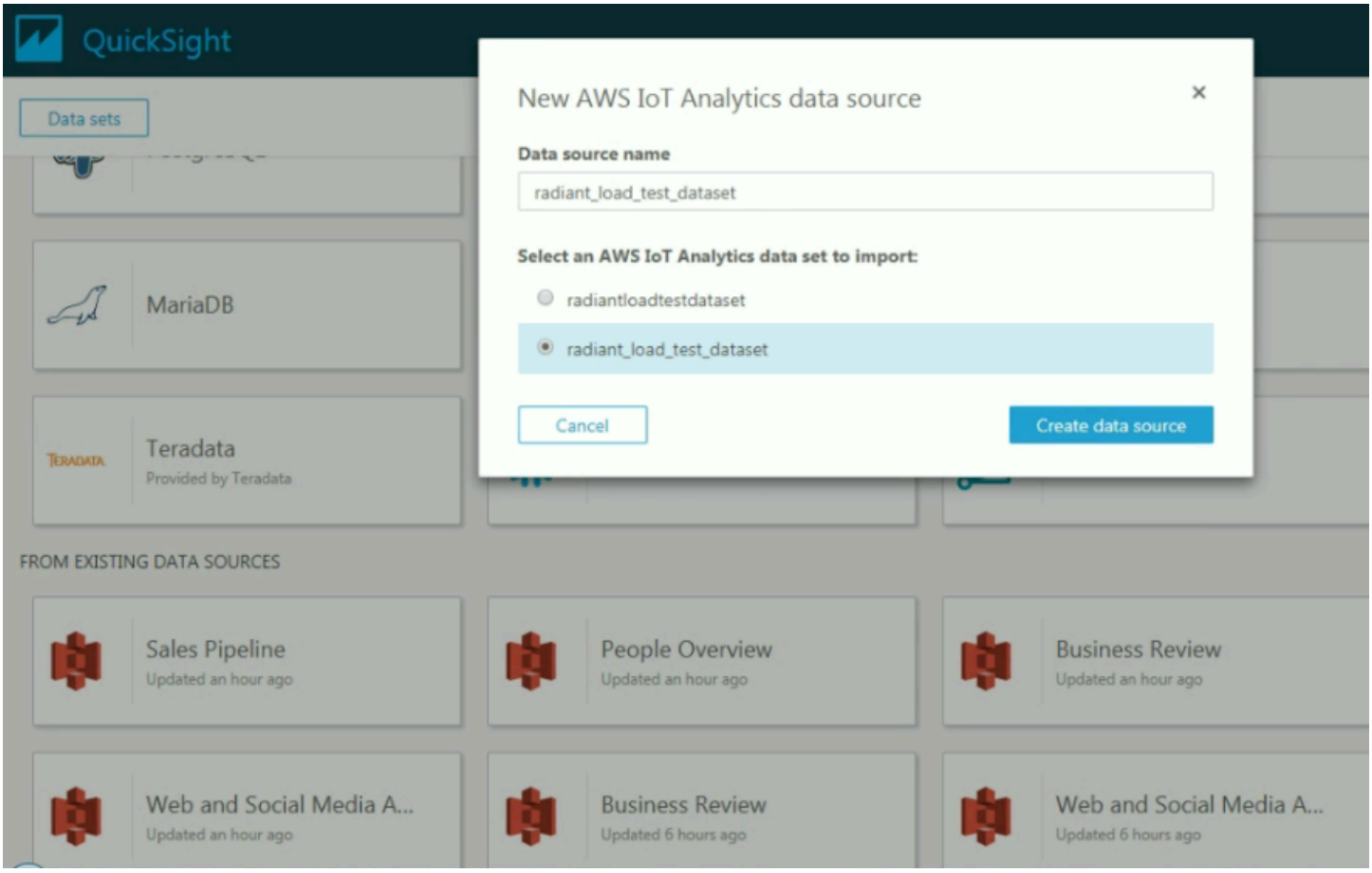
QuickSight を使用した AWS IoT Analytics データの視覚化

AWS IoT Analytics は QuickSight と直接統合できます。QuickSight は、視覚化の構築、アドホック分析の実行、データからのビジネスインサイトの迅速な取得に使用できる高速ビジネス分析サービスです。QuickSight を使用すると、組織は数十万人のユーザーにスケールし、堅牢なインメモリエンジン (SPICE) を使用して応答性の高いパフォーマンスを実現できます。QuickSight コンソールで AWS IoT Analytics データセットを選択し、ダッシュボードと視覚化の作成を開始できます。QuickSight は、これらのリージョンで使用できます。

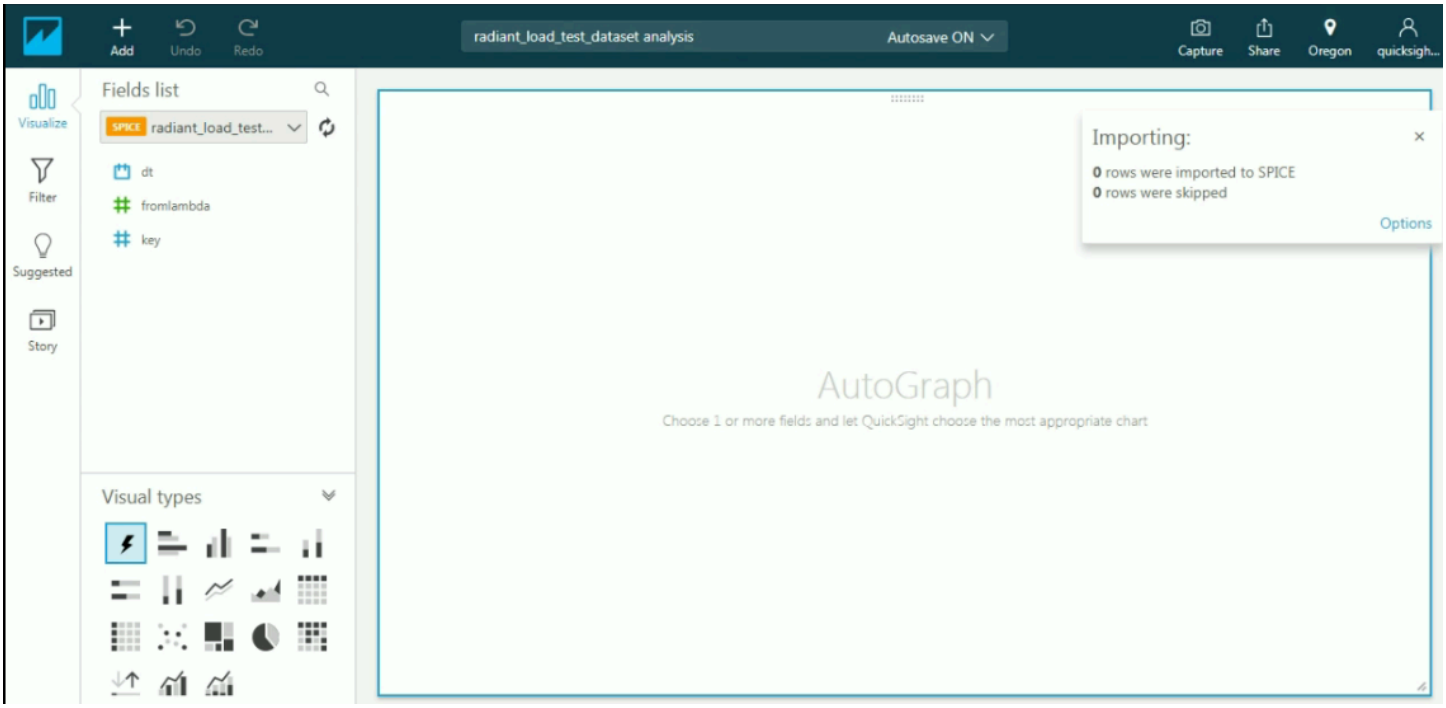
QuickSight の可視化を開始するには、QuickSight アカウントを作成する必要があります。アカウントを設定するときは、QuickSight に AWS IoT Analytics データへのアクセスを許可してください。すでにアカウントをお持ちの場合は、Admin、Manage QuickSight、Security & permissions を選択して、QuickSight に AWS IoT Analytics データへのアクセスを許可します。 QuickSight AWS サービスへの QuickSight アクセスで、追加または削除を選択し、 の横にあるチェックボックスAWS IoT Analyticsをオンにして更新を選択します。



アカウントを設定したら、管理者 QuickSight コンソールページから新しい分析と新しいデータセットを選択し、ソース AWS IoT Analytics として を選択します。データソースの名前を入力し、インポートするデータセットを選択して、[Create data source] (データソースの作成) を選択します。



データソースを作成したら、QuickSight で視覚化を作成できます。



QuickSight ダッシュボードとデータセットの詳細については、QuickSight ドキュメントを参照してください。

AWS IoT Analytics リソースのタグ付け

チャネル、データセットおよびパイプラインの管理を支援するため、これらの各リソースにはタグという形式で独自のメタデータをオプションで割り当てることができます。この章では、タグとその作成方法について説明します。

トピック

- タグの基本
- IAM ポリシーでのタグの使用
- タグの制限

タグの基本

タグを使用すると、 AWS IoT Analytics リソースを目的、所有者、環境などさまざまな方法で分類できます。これは、同じ型のリソースが多数ある場合に役立ちます。割り当てたタグに基づいて特定のリソースをすばやく識別できます。タグはそれぞれ、1 つのキーとオプションの値で設定され、どちらもユーザーが定義します。たとえば、チャネルに一連のタグを定義して、各チャネルのメッセージソースを担当するデバイスのタイプを追跡することができます。各リソースタイプのニーズを満たす一連のタグキーを考案することをお勧めします。一貫性のある一連のタグキーを使用することで、リソースの管理が容易になります。追加したタグに基づいてリソースを検索およびフィルタリングできます。

また、コストの分類と追跡にもタグを使用できます。チャネル、データセット、あるいはパイプラインにタグを適用すると、 AWS はタグ別に利用量とコストを集計したカンマ区切り値 CSVファイルとしてコスト配分レポートを作成します。自社のカテゴリ たとえばコストセンター、アプリケーション名、所有者を表すタグを適用すると、複数のサービスにわたってコストを分類することができます。タグを使ったコスト配分の詳細については、AWS Billing ユーザーガイドの「コスト配分タグの使用」を参照してください。

使いやすくするために、 AWS Billing and Cost Management コンソールでタグエディタを使用します。これにより、タグを一元的に作成および管理できます。詳細については、AWS Management Consoleの開始方法のタグエディタの使用を参照してください。

AWS CLI および AWS IoT Analytics API を使用してタグを操作することもできます。タグの作成時に、チャネル、データセット、データストア、およびパイプラインとタグを関連付けることができます。次のコマンドでタグフィールドを使用します。

- CreateChannel
- CreateDataset
- CreateDatastore
- CreatePipeline

タグ付けに対応している既存のリソースに対して、タグの追加、変更、削除を行うことができます。次のコマンドを使用します。

- TagResource
- ListTagsForResource
- UntagResource

タグのキーと値は編集でき、タグはリソースからいつでも削除できます。タグの値を空の文字列に設定することはできますが、タグの値を null に設定することはできません。特定のリソースについて既存のタグと同じキーを持つタグを追加した場合、古い値は新しい値によって上書きされます。リソースを削除すると、リソースに関連付けられているすべてのタグも削除されます。

IAM ポリシーでのタグの使用

IAM ポリシーの以下の条件コンテキストのキーと値とともに `Condition` 要素 `Condition` ブロックとも呼ばれるを使用して、リソースのタグに基づいてユーザーアクセス アクセス許可を制御できます。

- 特定のタグを持つリソースに対してユーザーアクションを許可または拒否するには、`iotanalytics:ResourceTag/<tag-key>: <tag-value>` を使用します。
- タグが許可されているリソースを作成または変更する API リクエストを作成する場合に、特定のタグが使用されている または、使用されていないことを要求するには、`aws:RequestTag/<tag-key>: <tag-value>` を使用します。
- タグが許可されているリソースを作成または変更する API リクエストを作成する場合に、特定の一連のタグが使用されている または、使用されていないことを要求するには、 `aws:TagKeys: [<tag-key>, ...]` を使用します。

> **Note**
>
> IAM ポリシーの条件コンテキストのキー/値は、タグ付け可能なリソースの ID を必須パラメータとする AWS IoT Analytics アクションにのみ適用されます。たとえば、タグ付け可

能なリソース チャンネル、データセット、データストアまたはパイプラインがこのリクエスト内で参照されないため、DescribeLoggingOptions の使用は条件コンテキストキー/値に基づいて許可/拒否されません。

詳細については、IAM ユーザーガイドの「タグを使用したアクセス制御」を参照してください。そのガイドの「IAM JSON ポリシーリファレンス」セクションには、IAM での JSON ポリシーの要素、変数、および評価ロジックの詳細な構文、説明、および例が記載されています。

次のポリシー例では、タグベースの 2 つの制約が適用されています。このポリシーによって制限されている IAM ユーザー

1. リソースにタグ "env=prod" を付与できません この例の `"aws:RequestTag/env" : "prod"` の行を参照)。
2. 既存のタグ "env=prod" を持つリソースに対しては変更またはアクセスできません この例の `"iotanalytics:ResourceTag/env" : "prod"` の行を参照)。

```
{
  "Version" : "2012-10-17",
  "Statement" :
  [
    {
      "Effect" : "Deny",
      "Action" : "iotanalytics:*",
      "Resource" : "*",
      "Condition" : {
        "StringEquals" : {
          "aws:RequestTag/env" : "prod"
        }
      }
    },
    {
      "Effect" : "Deny",
      "Action" : "iotanalytics:*",
      "Resource" : "*",
      "Condition" : {
        "StringEquals" : {
          "iotanalytics:ResourceTag/env" : "prod"
        }
      }
```

```
    },
    {
      "Effect": "Allow",
      "Action": [
        "iotanalytics:*"
      ],
      "Resource": "*"
    }
  ]
}
```

次の例ように、リストで囲むことにより、特定のタグキーに複数のタグ値を指定することもできます。

```
"StringEquals" : {
  "iotanalytics:ResourceTag/env" : ["dev", "test"]
}
```

> **Note**
>
> タグに基づいてリソースへのユーザーのアクセスを許可/拒否する場合は、ユーザーが同じリソースに対してそれらのタグを追加または削除する機能を明示的に拒否することを検討してください。そうしなければ、ユーザーはそのリソースのタグを変更することで、制限を回避してリソースにアクセスできます。

タグの制限

タグには以下のベーシックな制限があります。

- リソースあたりのタグの最大数: 50
- キーの最大長: 127 文字 UnicodeUTF-8)
- 値の最大長: 255 文字 UnicodeUTF-8)
- タグのキーと値は大文字と小文字が区別されます。
- タグの名前または値`aws: prefix`に を使用しないでください。 は AWS 用に予約されています。このプレフィックスが含まれるタグの名前または値は編集または削除できません。このプレフィックスを持つタグは、ソースあたりのタグ数の制限には計算されません。

- 複数の のサービス間およびリソース間でタグ付けスキーマを使用する場合、他のサービスにも許容される文字数に制限がある可能性があることに注意してください。一般的に、使用が許可される文字は、UTF-8 で表現できる文字、スペース、および数字と特殊文字 + - = . _ : / @です。

の SQL 式 AWS IoT Analytics

データセットは、データストア内のデータの SQL 式を使用して生成されます。 は、Amazon Athena と同じ SQL クエリ、関数、演算子 AWS IoT Analytics を使用します。

AWS IoT Analytics は、ANSI 標準 SQL 構文のサブセットをサポートしています。

```
SELECT [ ALL | DISTINCT ] select_expression [, ...]
[ FROM from_item [, ...] ]
[[ INNER | OUTER ] LEFT | RIGHT | FULL | CROSS JOIN join_item [ ON join_condition ]]
[ WHERE condition ]
[ GROUP BY [ ALL | DISTINCT ] grouping_element [, ...] ]
[ HAVING condition ]
[ UNION [ ALL | DISTINCT ] union_query ]
[ ORDER BY expression [ ASC | DESC ] [ NULLS FIRST | NULLS LAST] [, ...] ]
[ LIMIT [ count | ALL ] ]
```

パラメータの説明については、Amazon Athena のドキュメントの「パラメータ」を参照してください。

AWS IoT Analytics および Amazon Athena では、以下はサポートされていません。

- `WITH` 句
- `CREATE TABLE AS SELECT` ステートメント
- `INSERT INTO` ステートメント
- プリペアドステートメント (`EXECUTE` を `USING` では実行できません。)
- `CREATE TABLE LIKE`
- `DESCRIBE INPUT` および `DESCRIBE OUTPUT`
- `EXPLAIN` ステートメント
- ユーザー定義関数 (UDF または UDAF)。
- ストアドプロシージャ
- フェデレーションコネクタ

トピック

- でサポートされている SQL 機能 AWS IoT Analytics

- AWS IoT Analyticsの SQL クエリに関する一般的な問題のトラブルシューティング

でサポートされている SQL 機能 AWS IoT Analytics

データセットは、データストアのデータで SQL 式を使用して生成されます。で実行するクエリ AWS IoT Analytics は、Presto 0.217 に基づいています。

サポートされているデータ型

AWS IoT Analytics と Amazon Athena は、これらのデータ型をサポートしています。

- primitive_type
  - `TINYINT`
  - `SMALLINT`
  - `INT`
  - `BIGINT`
  - `BOOLEAN`
  - `DOUBLE`
  - `FLOAT`
  - `STRING`
  - `TIMESTAMP`
  - `DECIMAL(precision, scale)`
  - `DATE`
  - `CHAR` (指定した長さの固定長文字データ)
  - `VARCHAR` (指定した長さの可変長文字データ)
- array_type
  - `ARRAY<data_type>`
- map_type
  - `MAP<primitive_type, data_type>`
- struct_type
  - `STRUCT<col_name:data_type[COMMENT col_comment][,...]>`

> **Note**
>
> AWS IoT Analytics と Amazon Athena は、一部のデータ型をサポートしていません。

サポートされている関数

Amazon Athena と AWS IoT Analytics SQL の機能は、Presto 0.217 に基づいています。関連する関数、演算子、および式については、「関数および演算子」および Presto ドキュメントの以下の特定のセクションを参照してください。

- 論理演算子
- 比較関数および演算子
- 条件式
- 変換関数
- 数学関数と演算子
- ビット単位関数
- 進数関数および演算子
- 文字列関数および演算子
- バイナリ関数
- 日付/時刻関数と演算子
- 正規表現関数
- JSON 関数および演算子
- URL 関数
- 集計関数
- Window 関数
- カラー関数
- 配列関数と演算子
- マッピング関数と演算子
- Lambda 式および関数
- Teradata 関数

> **Note**
>
> AWS IoT Analytics および Amazon Athena は、ユーザー定義関数 (UDFs または UDAFs) またはストアドプロシージャをサポートしていません。

AWS IoT Analyticsの SQL クエリに関する一般的な問題のトラブルシューティング

以下の情報は、 AWS IoT Analyticsの SQL クエリに関する問題のトラブルシューティングに役立ちます。

- 一重引用符をエスケープするには、その前に別の一重引用符を付けます。これを二重引用符と混同しないでください。

  Example 例

  ```
  SELECT 'O''Reilly'
  ```

- 下線をエスケープするには、下線で始まるデータストア列名をバックティックで囲みます。

  Example 例

  ```
  SELECT `_myMessageAttribute` FROM myDataStore
  ```

- 数字を含む名前をエスケープするには、数字を含むデータストア名を二重引用符で囲みます。

  Example 例

  ```
  SELECT * FROM "myDataStore123"
  ```

- 予約キーワードをエスケープするには、予約キーワードを二重引用符で囲みます。詳細については、SQL SELECT ステートメントの予約キーワードのリストを参照してください。

のセキュリティ AWS IoT Analytics

のクラウドセキュリティが最優先事項 AWS です。お客様は AWS 、セキュリティを最も重視する組織の要件を満たすように構築されたデータセンターとネットワークアーキテクチャを活用できます。

セキュリティは、 AWS お客様とお客様の間の責任共有です。責任共有モデルでは、これをクラウドのセキュリティおよびクラウド内のセキュリティとして説明しています。

- クラウドのセキュリティ - AWS クラウドで AWS サービスを実行するインフラストラクチャを保護する AWS 責任があります。 AWS また、 では、安全に使用できるサービスも提供しています。セキュリティの有効性は、AWS コンプライアンスプログラムの一環として、サードパーティーの審査機関によって定期的にテストおよび検証されています。が適用されるコンプライアンスプログラムの詳細については AWS IoT Analytics、AWS 「コンプライアンスプログラムによる対象範囲内のサービス」を参照してください。
- クラウド内のセキュリティ - お客様の責任は、使用する AWS サービスによって決まります。また、お客様は、お客様のデータの機密性、組織の要件、および適用可能な法律および規制などの他の要因についても責任を担います。

このドキュメントは、 を使用する際の責任共有モデルの適用方法を理解するのに役立ちます AWS IoT Analytics。以下のトピックでは、セキュリティとコンプライアンスの目的を達成する AWS IoT Analytics ように を設定する方法を示します。また、 AWS IoT Analytics リソースのモニタリングや保護に役立つ他の AWS サービスの使用方法についても説明します。

AWS Identity and Access Management in AWS IoT Analytics

AWS Identity and Access Management (IAM) は、管理者が AWS リソースへのアクセスを安全に制御するのに役立つ AWS サービスです。IAM 管理者は、誰を認証 (サインイン) し、誰に AWS IoT Analytics リソースの使用を許可する (アクセス許可を付与する) かを制御します。IAM は、追加料金なしで使用できる AWS サービスです。

対象者

AWS Identity and Access Management (IAM) の使用方法は、作業内容によって異なります AWS IoT Analytics。

サービスユーザー – AWS IoT Analytics サービスを使用してジョブを実行する場合、管理者から必要な認証情報とアクセス許可が提供されます。さらに多くの AWS IoT Analytics 機能を使用して作業を

行う場合は、追加のアクセス許可が必要になることがあります。アクセスの管理方法を理解すると、管理者から適切な権限をリクエストするのに役に立ちます。 AWS IoT Analytics機能にアクセスできない場合は、「AWS IoT Analytics ID とアクセスのトラブルシューティング」を参照してください。

サービス管理者 – 社内の AWS IoT Analytics リソースを担当している場合は、通常、 へのフルアクセスがあります AWS IoT Analytics。サービスユーザーがどの AWS IoT Analytics 機能やリソースにアクセスするかを決めるのは管理者の仕事です。その後、IAM 管理者にリクエストを送信して、サービスユーザーの権限を変更する必要があります。このページの情報を点検して、IAM の基本概念を理解してください。会社で IAM を使用する方法の詳細については AWS IoT Analytics、「」を参照してくださいが IAM と AWS IoT Analytics 連携する方法。

IAM 管理者 - 管理者は、 AWS IoT Analyticsへのアクセスを管理するポリシーの書き込み方法の詳細について確認する場合があります。IAM で使用できる AWS IoT Analytics アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいAWS IoT Analytics アイデンティティベースのポリシーの例。

アイデンティティを使用した認証

認証とは、ID 認証情報 AWS を使用して にサインインする方法です。として、IAM ユーザーとして AWS アカウントのルートユーザー、または IAM ロールを引き受けることによって、認証 (にサインイン AWS) される必要があります。

ID ソースを介して提供された認証情報を使用して、フェデレーティッド ID AWS として にサインインできます。 AWS IAM Identity Center （ IAM Identity Center) ユーザー、会社のシングルサインオン認証、Google または Facebook 認証情報は、フェデレーション ID の例です。フェデレーティッド ID としてサインインする場合、IAM ロールを使用して、前もって管理者により ID フェデレーションが設定されています。フェデレーション AWS を使用して にアクセスすると、間接的にロールを引き受けることになります。

ユーザーの種類に応じて、 AWS Management Console または AWS アクセスポータルにサインインできます。へのサインインの詳細については AWS、 AWS サインイン ユーザーガイドの「 へのサインイン方法 AWS アカウント」を参照してください。

AWS プログラムで にアクセスする場合、 はソフトウェア開発キット (SDK) とコマンドラインインターフェイス (CLI) AWS を提供し、認証情報を使用してリクエストを暗号化して署名します。AWS ツールを使用しない場合は、自分でリクエストに署名する必要があります。リクエストに自分で署名する推奨方法の使用については、「IAM ユーザーガイド」の「API リクエストに対するAWS Signature Version 4」を参照してください。

使用する認証方法を問わず、追加セキュリティ情報の提供をリクエストされる場合もあります。たとえば、 では、アカウントのセキュリティを高めるために多要素認証 (MFA) を使用する AWS ことをお勧めします。詳細については、「AWS IAM Identity Center ユーザーガイド」の「多要素認証」および「IAM ユーザーガイド」の「IAM のAWS 多要素認証」を参照してください。

AWS アカウント ルートユーザー

を作成するときは AWS アカウント、アカウント内のすべての およびリソースへの AWS のサービス完全なアクセス権を持つ 1 つのサインインアイデンティティから始めます。この ID は AWS アカウント ルートユーザーと呼ばれ、アカウントの作成に使用した E メールアドレスとパスワードでサインインすることでアクセスできます。日常的なタスクには、ルートユーザーを使用しないことを強くお勧めします。ルートユーザーの認証情報は保護し、ルートユーザーでしか実行できないタスクを実行するときに使用します。ルートユーザーとしてサインインする必要があるタスクの完全なリストについては、「IAM ユーザーガイド」の「ルートユーザー認証情報が必要なタスク」を参照してください。

IAM ユーザーとグループ

IAM ユーザーは、単一のユーザーまたはアプリケーションに対して特定のアクセス許可 AWS アカウント を持つ 内の ID です。可能であれば、パスワードやアクセスキーなどの長期的な認証情報を保有する IAM ユーザーを作成する代わりに、一時的な認証情報を使用することをお勧めします。ただし、IAM ユーザーでの長期的な認証情報が必要な特定のユースケースがある場合は、アクセスキーをローテーションすることをお勧めします。詳細については、「IAM ユーザーガイド」の「長期的な認証情報を必要とするユースケースのためにアクセスキーを定期的にローテーションする」を参照してください。

IAM グループは、IAM ユーザーの集団を指定するアイデンティティです。グループとしてサインインすることはできません。グループを使用して、複数のユーザーに対して一度に権限を指定できます。多数のユーザーグループがある場合、グループを使用することで権限の管理が容易になります。例えば、IAMAdmins という名前のグループを設定して、そのグループに IAM リソースを管理する許可を与えることができます。

ユーザーは、ロールとは異なります。ユーザーは 1 人の人または 1 つのアプリケーションに一意に関連付けられますが、ロールはそれを必要とする任意の人が引き受けるようになっています。ユーザーには永続的な長期の認証情報がありますが、ロールでは一時認証情報が提供されます。詳細については、「IAM ユーザーガイド」の「IAM ユーザーに関するユースケース」を参照してください。

IAM ロール

IAM ロールは、特定のアクセス許可 AWS アカウント を持つ 内の ID です。これは IAM ユーザーに似ていますが、特定のユーザーには関連付けられていません。で IAM ロールを一時的に引き受けるには AWS Management Console、ユーザーから IAM ロール (コンソール) に切り替えることができます。ロールを引き受けるには、 または AWS API オペレーションを AWS CLI 呼び出すか、カスタム URL を使用します。ロールを使用する方法の詳細については、「IAM ユーザーガイド」の「ロールを引き受けるための各種方法」を参照してください。

IAM ロールと一時的な認証情報は、次の状況で役立ちます:

- フェデレーションユーザーアクセス – フェデレーティッド ID に許可を割り当てるには、ロールを作成してそのロールの許可を定義します。フェデレーティッド ID が認証されると、その ID はロールに関連付けられ、ロールで定義されている許可が付与されます。フェデレーションのロールについては、「IAM ユーザーガイド」の「サードパーティー ID プロバイダー (フェデレーション) 用のロールを作成する」を参照してください。IAM Identity Center を使用する場合は、許可セットを設定します。アイデンティティが認証後にアクセスできるものを制御するため、IAM Identity Center は、権限セットを IAM のロールに関連付けます。アクセス許可セットの詳細については、「AWS IAM Identity Center User Guide」の「Permission sets」を参照してください。
- 一時的な IAM ユーザー権限 - IAM ユーザーまたはロールは、特定のタスクに対して複数の異なる権限を一時的に IAM ロールで引き受けることができます。
- クロスアカウントアクセス - IAM ロールを使用して、自分のアカウントのリソースにアクセスすることを、別のアカウントの人物 (信頼済みプリンシパル) に許可できます。クロスアカウントアクセス権を付与する主な方法は、ロールを使用することです。ただし、一部の では AWS のサービス、 (プロキシとしてロールを使用する代わりに) リソースに直接ポリシーをアタッチできます。クロスアカウントアクセスにおけるロールとリソースベースのポリシーの違いについては、「IAM ユーザーガイド」の「IAM でのクロスアカウントのリソースへのアクセス」を参照してください。
- クロスサービスアクセス – 一部の は他の の機能 AWS のサービス を使用します AWS のサービス。例えば、あるサービスで呼び出しを行うと、通常そのサービスによって Amazon EC2 でアプリケーションが実行されたり、Amazon S3 にオブジェクトが保存されたりします。サービスでは、呼び出し元プリンシパルの許可、サービスロール、またはサービスリンクロールを使用してこれを行う場合があります。
  - 転送アクセスセッション (FAS) – IAM ユーザーまたはロールを使用してアクションを実行すると AWS、プリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、 を呼び出すプリンシパルのアクセス許可を AWS のサービス、ダウンストリームサービス AWS のサービ

ス へのリクエストをリクエストする と組み合わせて使用します。FAS リクエストは、サービスが他の AWS のサービス またはリソースとのやり取りを完了する必要があるリクエストを受け取った場合にのみ行われます。この場合、両方のアクションを実行するためのアクセス許可が必要です。FAS リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。

- サービスロール - サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除することができます。詳細については、「IAM ユーザーガイド」の「AWS のサービスに許可を委任するロールを作成する」を参照してください。
- サービスにリンクされたロール – サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント 、サービスによって所有されます。IAM 管理者は、サービスリンクロールのアクセス許可を表示できますが、編集することはできません。

- Amazon EC2 で実行されているアプリケーション – IAM ロールを使用して、EC2 インスタンスで実行され、 AWS CLI または AWS API リクエストを行うアプリケーションの一時的な認証情報を管理できます。これは、EC2 インスタンス内でのアクセスキーの保存に推奨されます。EC2 インスタンスに AWS ロールを割り当て、そのすべてのアプリケーションで使用できるようにするには、インスタンスにアタッチされたインスタンスプロファイルを作成します。インスタンスプロファイルにはロールが含まれ、EC2 インスタンスで実行されるプログラムは一時的な認証情報を取得できます。詳細については、「IAM ユーザーガイド」の「Amazon EC2 インスタンスで実行されるアプリケーションに IAM ロールを使用して許可を付与する」を参照してください。

ポリシーを使用したアクセスの管理

でアクセスを制御する AWS には、ポリシーを作成し、ID AWS またはリソースにアタッチします。ポリシーは AWS 、アイデンティティまたはリソースに関連付けられているときにアクセス許可を定義する のオブジェクトです。 は、プリンシパル (ユーザー、ルートユーザー、またはロールセッション) がリクエストを行うときに、これらのポリシー AWS を評価します。ポリシーでの権限により、リクエストが許可されるか拒否されるかが決まります。ほとんどのポリシーは JSON ドキュメント AWS として に保存されます。JSON ポリシードキュメントの構造と内容の詳細については、「IAM ユーザーガイド」の「JSON ポリシー概要」を参照してください。

管理者は JSON AWS ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

デフォルトでは、ユーザーやロールに権限はありません。IAM 管理者は、リソースで必要なアクションを実行するための権限をユーザーに付与する IAM ポリシーを作成できます。その後、管理者はロールに IAM ポリシーを追加し、ユーザーはロールを引き受けることができます。

IAM ポリシーは、オペレーションの実行方法を問わず、アクションの許可を定義します。例えば、`iam:GetRole` アクションを許可するポリシーがあるとします。そのポリシーを持つユーザーは、 AWS Management Console、、 AWS CLIまたは AWS API からロール情報を取得できます。

アイデンティティベースのポリシー

アイデンティティベースポリシーは、IAM ユーザーグループ、ユーザーのグループ、ロールなど、アイデンティティにアタッチできる JSON 許可ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件をコントロールします。アイデンティティベースポリシーの作成方法については、「IAM ユーザーガイド」の「カスタマー管理ポリシーでカスタム IAM アクセス許可を定義する」を参照してください。

アイデンティティベースのポリシーは、さらにインラインポリシーまたはマネージドポリシーに分類できます。インラインポリシーは、単一のユーザー、グループ、またはロールに直接埋め込まれています。管理ポリシーは、 内の複数のユーザー、グループ、ロールにアタッチできるスタンドアロンポリシーです AWS アカウント。管理ポリシーには、 AWS 管理ポリシーとカスタマー管理ポリシーが含まれます。マネージドポリシーまたはインラインポリシーのいずれかを選択する方法については、「IAM ユーザーガイド」の「管理ポリシーとインラインポリシーのいずれかを選択する」を参照してください。

その他のポリシータイプ

AWS は、一般的でない追加のポリシータイプをサポートしています。これらのポリシータイプでは、より一般的なポリシータイプで付与された最大の権限を設定できます。

- アクセス許可の境界 - アクセス許可の境界は、アイデンティティベースポリシーによって IAM エンティティ (IAM ユーザーまたはロール) に付与できる権限の上限を設定する高度な機能です。エンティティにアクセス許可の境界を設定できます。結果として得られる権限は、エンティティのアイデンティティベースポリシーとそのアクセス許可の境界の共通部分になります。`Principal` フィールドでユーザーまたはロールを指定するリソースベースのポリシーでは、アクセス許可の境界は制限されません。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。アクセス許可の境界の詳細については、「IAM ユーザーガイド」の「IAM エンティティのアクセス許可の境界」を参照してください。
- サービスコントロールポリシー (SCPs) – SCPs は、 の組織または組織単位 (OU) の最大アクセス許可を指定する JSON ポリシーです AWS Organizations。 AWS Organizations は、ビジネスが所

有する複数の AWS アカウント をグループ化して一元管理するためのサービスです。組織内のすべての機能を有効にすると、サービスコントロールポリシー (SCP) を一部またはすべてのアカウントに適用できます。SCP は、各 を含むメンバーアカウントのエンティティのアクセス許可を制限します AWS アカウントのルートユーザー。Organizations と SCP の詳細については、「AWS Organizations ユーザーガイド」の「サービスコントロールポリシー (SCP)」を参照してください。

- リソースコントロールポリシー (RCP) – RCP は、所有する各リソースにアタッチされた IAM ポリシーを更新することなく、アカウント内のリソースに利用可能な最大数のアクセス許可を設定するために使用できる JSON ポリシーです。RCP は、メンバーアカウントのリソースのアクセス許可を制限し、組織に属しているかどうかにかかわらず AWS アカウントのルートユーザー、 を含む ID の有効なアクセス許可に影響を与える可能性があります。RCP をサポートする のリストを含む Organizations と RCP の詳細については、AWS Organizations RCPs「リソースコントロールポリシー (RCPs」を参照してください。 AWS のサービス
- セッションポリシー - セッションポリシーは、ロールまたはフェデレーションユーザーの一時的なセッションをプログラムで作成する際にパラメータとして渡す高度なポリシーです。結果としてセッションの権限は、ユーザーまたはロールのアイデンティティベースポリシーとセッションポリシーの共通部分になります。また、リソースベースのポリシーから権限が派生する場合もあります。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。詳細については、「IAM ユーザーガイド」の「セッションポリシー」を参照してください。

複数のポリシータイプ

1 つのリクエストに複数のタイプのポリシーが適用されると、結果として作成される権限を理解するのがさらに難しくなります。が複数のポリシータイプが関与する場合にリクエストを許可するかどうか AWS を決定する方法については、「IAM ユーザーガイド」の「ポリシー評価ロジック」を参照してください。

が IAM と AWS IoT Analytics 連携する方法

IAM を使用して へのアクセスを管理する前に AWS IoT Analytics、使用できる IAM 機能を理解しておく必要があります AWS IoT Analytics。 AWS IoT Analytics およびその他の AWS のサービスが IAM と連携する方法の概要については、「IAM ユーザーガイド」のAWS 「IAM と連携する のサービス」を参照してください。

このページのトピック

- AWS IoT Analytics ID ベースのポリシー
- AWS IoT Analytics リソースベースのポリシー

- AWS IoT Analytics タグに基づく認可
- AWS IoT Analytics IAM ロール

AWS IoT Analytics ID ベースのポリシー

IAM アイデンティティベースのポリシーでは、許可または拒否されたアクションとリソース、およびアクションが許可または拒否される条件を指定できます。 は、特定のアクション、リソース、および条件キー AWS IoT Analytics をサポートします。JSON ポリシーで使用するすべての要素については「IAM ユーザーガイド」の「IAM JSON ポリシーエレメントのリファレンス」を参照してください。

アクション

IAM アイデンティティベースのポリシーの `Action` エレメントは、そのポリシーにより許可または拒否される特定のアクションについて説明します。ポリシーアクションの名前は通常、関連付けられた AWS API オペレーションと同じです。このアクションは、関連付けられたオペレーションを実行するためのアクセス許可を付与するポリシーで使用されます。

のポリシーアクションは、アクションの前に次のプレフィックス AWS IoT Analytics を使用します。たとえば、API `iotanalytics:` オペレーションで AWS IoT Analytics `CreateChannel`チャネルを作成する AWS IoT Analytics アクセス許可を付与するには、ポリシーに `iotanalytics:BatchPuMessage`アクションを含めます。ポリシーステートメントには、`Action`または `NotAction` element を含める必要があります。 は、このサービスで実行できるタスクを記述する独自のアクションのセット AWS IoT Analytics を定義します。

単一のステートメントに複数の アクションを指定するには、次のようにコンマで区切ります。

```
"Action": [
    "iotanalytics:action1",
    "iotanalytics:action2"
    ]
```

ワイルドカード *を使用して複数のアクションを指定することができます。例えば、`Describe` という単語で始まるすべてのアクションを指定するには、次のアクションを含めます。

```
"Action": "iotanalytics:Describe*"
```

AWS IoT Analytics アクションのリストを確認するには、IAM ユーザーガイドの「 で定義されるアクション AWS IoT Analytics」を参照してください。

リソース

Resource エレメントは、アクションが適用されるオブジェクトを指定します。ステートメントには、Resource または NotResource エレメントを含める必要があります。ARN を使用して、またはステートメントがすべてのリソースに適用されることを示すワイルドカード *を使用して、リソースを指定します。

AWS IoT Analytics データセットリソースには次の ARN があります。

```
arn:${Partition}:iotanalytics:${Region}:${Account}:dataset/${DatasetName}
```

ARN の形式の詳細については、「Amazon リソースネーム ARNと AWS のサービスの名前空間」を参照してください。

たとえば、ステートメントで Foobar データセットを指定するには、次の ARN を使用します。

```
"Resource": "arn:aws:iotanalytics:us-east-1:123456789012:dataset/Foobar"
```

特定のアカウントに属するすべてのインスタンスを指定するには、ワイルドカード *を使用します。

```
"Resource": "arn:aws:iotanalytics:us-east-1:123456789012:dataset/*"
```

リソースを作成するためのアクションなど、一部の AWS IoT Analytics アクションは、特定のリソースで実行できません。このような場合はワイルドカード *を使用する必要があります。

```
"Resource": "*"
```

一部の AWS IoT Analytics API アクションには、複数のリソースが含まれます。たとえば、CreatePipeline はチャネルとデータセットを参照するため、IAM ユーザーはチャネルとデータセットを使用する権限を持っている必要があります。複数リソースを単一ステートメントで指定するには、ARN をカンマで区切ります。

```
"Resource": [
    "resource1",
    "resource2"
     ]
```

AWS IoT Analytics リソースタイプとその ARNs「 で定義されるリソース AWS IoT Analytics」を参照してください。 どのアクションで各リソースの ARN を指定できるかについては、「AWS IoT Analyticsで定義されるアクション」を参照してください。

条件キー

Condition エレメント またはCondition ブロックを使用すると、ステートメントが有効な条件を指定できます。Condition エレメントはオプションです。イコールや以下などの条件演算子を使用する条件表現を構築して、リクエスト内に値のあるポリシーの条件に一致させることができます。

1 つのステートメントに複数の Condition エレメントを指定する場合、または 1 つの Condition エレメントに複数のキーを指定する場合、 AWS が論理 AND 演算を使用してそれらを評価します。単一の条件キーに複数の値を指定する場合、 AWS では OR 論理演算子を使用して条件を評価します。ステートメントの権限が付与される前にすべての条件が満たされる必要があります。

条件を指定する際にプレースホルダー変数も使用できます。例えば、ユーザー名でタグ付けされている場合のみ、リソースにアクセスするユーザーアクセス許可を付与できます。詳細については、「IAM ユーザーガイド」の「IAM ポリシーエレメント。可変およびタグ」を参照してください。

AWS IoT Analytics はサービス固有の条件キーを提供しませんが、一部のグローバル条件キーの使用をサポートしています。すべての AWS グローバル条件キーを確認するには、IAM ユーザーガイドのAWS 「 グローバル条件コンテキストキー」を参照してください。

例

AWS IoT Analytics アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいAWS IoT Analytics アイデンティティベースのポリシーの例。

AWS IoT Analytics リソースベースのポリシー

AWS IoT Analytics はリソースベースのポリシーをサポートしていません。詳細なリソースベースポリシーページの例を表示するには、AWS Lambda デベロッパーガイドの「AWS Lambdaのリソースベースのポリシーを使用する」を参照してください。

AWS IoT Analytics タグに基づく認可

AWS IoT Analytics リソースにタグをアタッチしたり、リクエストでタグを渡すことができます AWS IoT Analytics。タグに基づいてアクセスを制御するには、`iotanalytics:ResourceTag/{key-name}`, `aws:RequestTag/{key-name}` または `aws:TagKeys` の条件キーを使用して、ポリシーの条件要素でタグ情報を指定します。 AWS IoT Analytics リソースのタグ付けの詳細については、「リソースの AWS IoT Analytics タグ付け」を参照してください。

リソースのタグに基づいてリソースへのアクセスを制限するためのアイデンティティベースのポリシーの例を表示するには、「タグに基づく AWS IoT Analytics チャネルの表示」を参照してください。

AWS IoT Analytics IAM ロール

IAM ロール は、特定の権限を持つ、 AWS アカウント 内のエンティティです。

での一時的な認証情報の使用 AWS IoT Analytics

テンポラリ認証情報を使用して、フェデレーションでサインイン、IAM ロールを引き受ける、またはクロスアカウントロールを引き受けることができます。テンポラリセキュリティ認証情報を取得するには、AssumeRole または GetFederationToken などの AWS Security Token Service AWS STS API オペレーションを呼び出します。

AWS IoT Analytics は一時的な認証情報の使用をサポートしていません。

サービスにリンクされた役割

サービスラインロールを使用すると、 AWS サービスが他の サービスのリソースにアクセスして、ユーザーに代わってアクションを実行できます。サービスリンクロールは IAM アカウント内に表示され、サービスによって所有されます。IAM 管理者は、サービスリンクロールの許可を表示できますが、編集することはできません。

AWS IoT Analytics は、サービスにリンクされたロールをサポートしていません。

サービス役割

この機能により、ユーザーに代わってサービスがサービス役割を引き受けることが許可されます。この役割により、サービスがお客様に代わって他のサービスのリソースにアクセスし、アクションを完了することが許可されます。サービス役割はIAM アカウントに表示され、アカウントによって所有されます。つまり、IAM 管理者はこの役割の権限を変更できます。ただし、それにより、サービスの機能が損なわれる場合があります。

AWS IoT Analytics はサービスロールをサポートします。

サービス間の混乱した代理の防止

混乱した代理問題は、アクションを実行するためのアクセス許可を持たないエンティティが、より特権のあるエンティティにアクションの実行を強制できてしまう場合に生じる、セキュリティ上の問題です。では AWS、サービス間のなりすましにより、混乱した代理問題が発生する可能性があります。サービス間でのなりすましは、あるサービス (呼び出し元サービス) が、別のサービス (呼び出し対象サービス) を呼び出すときに発生する可能性があります。呼び出し元サービスが操作され、それ自身のアクセス許可を使用して、本来アクセス許可が付与されるべきではない方法で別の顧客のリ

ソースに対して働きかけることがあります。これを防ぐために AWS では、顧客のすべてのサービスのデータを保護するのに役立つツールを提供しています。これには、アカウントのリソースへのアクセス許可が付与されたサービスプリンシパルを使用します。

リソースポリシーには、`aws:SourceArn` および `aws:SourceAccount` のグローバル条件コンテキストキーを使用することをお勧めします。これにより、 が別のサービス AWS IoT Analytics に付与するアクセス許可がリソースに制限されます。両方のグローバル条件コンテキストキーを同じポリシーステートメントで使用する場合は、`aws:SourceAccount` 値と、`aws:SourceArn` 値に含まれるアカウントが、同じアカウント ID を示している必要があります。

「混乱した代理」問題から保護するための最も効果的な方法は、リソースの完全なAmazon リソースネーム (ARN)を指定しながら、グローバル条件コンテキストキー `aws:SourceArn` を使用することです。リソースの ARN 全体が不明または複数のリソースを指定する場合、ARN の未知部分にワイルドカード `*`が付いた `aws:SourceArn` グローバルコンテキスト条件キー を使用します。例えば、`arn:aws:iotanalytics::123456789012:*` と指定します。

トピック

- Amazon S3 バケットの防止
- Amazon CloudWatch Logs での防止
- カスタマーマネージド AWS IoT Analytics リソースの混乱した代理の防止

Amazon S3 バケットの防止

AWS IoT Analytics データストアにカスタマーマネージド Amazon S3 ストレージを使用すると、データを保存する Amazon S3 バケットが混乱した代理問題にさらされる可能性があります。

たとえば、Nikki Wolf は *DOC-EXAMPLE-BUCKET* という顧客所有の Amazon S3 バケットを使用しています。このバケットには、*us-east-1* リージョンで作成された AWS IoT Analytics データストアの情報が保存されます。彼女は、 AWS IoT Analytics サービスプリンシパルが自分に代わって *DOC-EXAMPLE-BUCKET* をクエリできるようにするポリシーを指定します。Nikki の同僚である Li Juan は、自分のアカウントから *DOC-EXAMPLE-BUCKET* にクエリを実行し、その結果を含むデータセットを作成します。その結果、 AWS IoT Analytics サービスプリンシパルは、Li がアカウントからクエリを実行したにもかかわらず、Li に代わって Nikki の Amazon S3 バケットにクエリを実行しました。

これを防ぐため、Nikki は *DOC-EXAMPLE-BUCKET* のポリシーで `aws:SourceAccount` 条件または `aws:SourceArn` 条件を指定できます。

aws:SourceAccount 条件を指定する - 次のバケットポリシーの例では、Nikki のアカウント (*123456789012*) の AWS IoT Analytics リソースのみが *DOC-EXAMPLE-BUCKET* にアクセスできるように指定しています。

```
{
    "Version": "2012-10-17",
    "Id": "MyPolicyID",
    "Statement": [
        {
            "Sid": "ConfusedDeputyPreventionExamplePolicy",
            "Effect": "Allow",
            "Principal": {
                "Service": "iotanalytics.amazonaws.com"
            },
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads",
                "s3:ListMultipartUploadParts",
                "s3:AbortMultipartUpload",
                "s3:PutObject",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "123456789012"
                }
            }
        }
    ]
}
```

aws:SourceArn 条件を指定してください-代わりに、Nikki が **aws:SourceArn** 条件を使用することもできます。

```
{
    "Version": "2012-10-17",
```

```
    "Id": "MyPolicyID",
    "Statement": [
        {
            "Sid": "ConfusedDeputyPreventionExamplePolicy",
            "Effect": "Allow",
            "Principal": {
                "Service": "iotanalytics.amazonaws.com"
            },
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads",
                "s3:ListMultipartUploadParts",
                "s3:AbortMultipartUpload",
                "s3:PutObject",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ],
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": [
                        "arn:aws:iotanalytics:us-east-1:123456789012:dataset/DOC-
EXAMPLE-DATASET",
                        "arn:aws:iotanalytics:us-east-1:123456789012:datastore/DOC-
EXAMPLE-DATASTORE"
                    ]
                }
            }
        }
    ]
}
```

Amazon CloudWatch Logs での防止

Amazon CloudWatch Logs を使用してモニタリングを行うと、混乱した代理問題が発生するのを防ぐことができます。以下のリソースポリシーは、以下のいずれかの問題に関する混乱した代理問題を防止する方法を説明しています。

- グローバル条件コンテキストキー `aws:SourceArn`

- AWS アカウント ID `aws:SourceAccount`を持つ
- の`sts:AssumeRole`リクエストに関連付けられている顧客リソース AWS IoT Analytics

次の例では、*123456789012* を AWS アカウント ID に、*us-east-1* を AWS IoT Analytics アカウントのリージョンに置き換えます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "iotanalytics.amazonaws.com"
            },
            "Action": "logs:PutLogEvents",
            "Resource": "*",
            "Condition":{
                "ArnLike":{
                    "aws:SourceArn":"arn:aws:iotanalytics:us-east-1:123456789012:*/*"
                },
                "StringEquals":{
                    "aws:SourceAccount":"123456789012"
                }
            }
    ]
}
```

Amazon CloudWatch Logsの詳細については、「the section called “ログ記録とモニタリング”」を参照してください。

カスタマーマネージド AWS IoT Analytics リソースの混乱した代理の防止

AWS IoT Analytics リソースに対してアクションを実行する AWS IoT Analytics アクセス許可を付与すると、リソースが混乱した代理問題にさらされる可能性があります。混乱した代理問題を回避するには、次のリソースポリシーの例 AWS IoT Analytics を使用して、 に付与されるアクセス許可を制限できます。

トピック

- AWS IoT Analytics チャネルとデータストアの防止
- AWS IoT Analytics データセットコンテンツ配信ルールのサービス間の混乱した代理の防止

AWS IoT Analytics チャネルとデータストアの防止

IAM ロールを使用して、 がユーザーに代わって AWS IoT Analytics アクセスできる AWS リソースを制御します。混乱した代理問題にロールが公開されないようにするには、 `aws:SourceAccount`要素で AWS アカウントを指定し、ロールにアタッチする信頼ポリシーの `aws:SourceArn`要素で AWS IoT Analytics リソースの ARN を指定できます。

次の例では、*123456789012* を AWS アカウント ID に置き換え、*arn:aws:iotanalytics:aws-region:123456789012:channel/DOC-EXAMPLE-CHANNEL* を AWS IoT Analytics チャネルまたはデータストアの ARN に置き換えます。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ConfusedDeputyPreventionExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "Service": "iotanalytics.amazonaws.com"
       },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:iotanalytics:aws-region:123456789012:channel/DOC-
EXAMPLE-CHANNEL"
        }
      }
    }
  ]
}
```

チャネルとデータストア用のカスタマーマネージド S3 ストレージオプションの詳細については、「AWS IoT Analytics API リファレンス」の「CustomerManagedChannelS3Storage」と「CustomerManagedDatastoreS3Storage」を参照してください。

AWS IoT Analytics データセットコンテンツ配信ルールのサービス間の混乱した代理の防止

Amazon S3 または にデータセットクエリ結果を配信するために が AWS IoT Analytics 引き受ける IAM ロール AWS IoT Events は、混乱した代理問題にさらされる可能性があります。混乱した代理問

題を防ぐには、 `aws:SourceAccount`要素で AWS アカウントを指定し、ロールにアタッチする信頼ポリシーの `aws:SourceArn`要素で AWS IoT Analytics リソースの ARN を指定します。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ConfusedDeputyPreventionExampleTrustPolicyDocument",
      "Effect": "Allow",
      "Principal": {
        "Service": "iotanalytics.amazonaws.com"
       },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:iotanalytics:aws-region:123456789012:dataset/DOC-
EXAMPLE-DATASET"
        }
      }
    }
  ]
}
```

データセットのコンテンツ配信ルールの設定について詳しくは、「AWS IoT Analytics API リファレンス」の「`contentDeliveryRules`」を参照してください。

AWS IoT Analytics アイデンティティベースのポリシーの例

デフォルトでは、 ユーザーおよびロールには、 AWS IoT Analytics リソースを作成または変更する権限はありません。また、 AWS Management Console、 AWS CLI、または AWS API を使用してタスクを実行することはできません。IAM 管理者は、ユーザーとロールに必要な、指定されたリソースで特定の API オペレーションを実行する権限をユーザーとロールに付与する IAM ポリシーを作成する必要があります。続いて、管理者はそれらのアクセス許可が必要なユーザーまたはグループにそのポリシーをアタッチします。

これらの JSON ポリシードキュメント例を使用して IAM のアイデンティティベースのポリシーを作成する方法については、IAM ユーザーガイドの「JSON タブでのポリシーの作成」を参照してください。

このページのトピック

- ポリシーに関するベストプラクティス
- AWS IoT Analytics コンソールの使用
- 自分の権限の表示をユーザーに許可する
- 1 つの AWS IoT Analytics 入力へのアクセス
- タグに基づく AWS IoT Analytics チャネルの表示

ポリシーに関するベストプラクティス

アイデンティティベースポリシーは非常に強力です。アカウント内の AWS IoT Analytics リソースを作成、アクセス、または削除できるかどうかを決定します。これらのアクションを実行すると、AWS アカウントに追加料金が発生する可能性があります。アイデンティティベースポリシーを作成したり編集したりする際には、以下のガイドラインと推奨事項に従ってください:

- AWS 管理ポリシーの使用を開始する - の使用 AWS IoT Analytics をすばやく開始するには、 AWS 管理ポリシーを使用して、従業員に必要なアクセス許可を付与します。これらのポリシーはアカウントですでに有効になっており、 AWSによって管理および更新されています。詳細については、IAM ユーザーガイドの「 AWS 管理ポリシーでアクセス許可の使用を開始する」を参照してください。
- 最小特権 - を付与する - カスタムポリシーを作成するときは、タスクの実行に必要な許可のみを付与します。最小限の許可からスタートし、必要に応じて追加の許可を付与します。この方法は、寛容過ぎる許可から始めて、後から厳しくしようとするよりも安全です。詳細については、IAM ユーザーガイドの「最小特権を認める」を参照してください。
- 機密性の高いオペレーションのために MFA を有効にする追加のセキュリティとして、機密性の高いリソースや API オペレーションにアクセスする際に Multi-Factor Authentication (MFA)を使用することを ユーザーに要求します。詳細については、IAM ユーザーガイドの「AWSでの多要素認証 MFAの使用」を参照してください。
- 追加のセキュリティとしてポリシー条件を使用する - 実行可能な範囲内で、アイデンティティベースのポリシーでリソースへのアクセスを許可する条件を定義します。例えば、要求が発生しなければならない許容 IP アドレスの範囲を指定するための条件を記述できます。指定された日付または時間範囲内でのみリクエストを許可する条件を書くことも、SSL や MFA の使用を要求することもできます。詳細については、IAM ユーザーガイド」の「IAM JSON ポリシー要素: 条件」を参照してください。

AWS IoT Analytics コンソールの使用

AWS IoT Analytics コンソールにアクセスするには、最小限のアクセス許可のセットが必要です。これらのアクセス許可により、 内の AWS IoT Analytics リソースの詳細を一覧表示および表示できます AWS アカウント。最小限必要な許可よりも制限が厳しいアイデンティティベースのポリシーを作成すると、そのポリシーを持つエンティティ ユーザーまたはロールに対してコンソールが意図したとおりに機能しません。

これらのエンティティが引き続き AWS IoT Analytics コンソールを使用できるようにするには、エンティティに次の AWS 管理ポリシーもアタッチします。詳細については、「IAM ユーザーガイド」の「ユーザーへのアクセス許可の追加」を参照してください。

```
{
     "Version": "2012-10-17",
     "Statement": [
         {
             "Effect": "Allow",
             "Action": [
                 "iotanalytics:BatchPutMessage",
                 "iotanalytics:CancelPipelineReprocessing",
                 "iotanalytics:CreateChannel",
                 "iotanalytics:CreateDataset",
                 "iotanalytics:CreateDatasetContent",
                 "iotanalytics:CreateDatastore",
                 "iotanalytics:CreatePipeline",
                 "iotanalytics:DeleteChannel",
                 "iotanalytics:DeleteDataset",
                 "iotanalytics:DeleteDatasetContent",
                 "iotanalytics:DeleteDatastore",
                 "iotanalytics:DeletePipeline",
                 "iotanalytics:DescribeChannel",
                 "iotanalytics:DescribeDataset",
                 "iotanalytics:DescribeDatastore",
                 "iotanalytics:DescribeLoggingOptions",
                 "iotanalytics:DescribePipeline",
                 "iotanalytics:GetDatasetContent",
                 "iotanalytics:ListChannels",
                 "iotanalytics:ListDatasetContents",
                 "iotanalytics:ListDatasets",
                 "iotanalytics:ListDatastores",
                 "iotanalytics:ListPipelines",
                 "iotanalytics:ListTagsForResource",
```

```
                "iotanalytics:PutLoggingOptions",
                "iotanalytics:RunPipelineActivity",
                "iotanalytics:SampleChannelData",
                "iotanalytics:StartPipelineReprocessing",
                "iotanalytics:TagResource",
                "iotanalytics:UntagResource",
                "iotanalytics:UpdateChannel",
                "iotanalytics:UpdateDataset",
                "iotanalytics:UpdateDatastore",
                "iotanalytics:UpdatePipeline"
            ],
            "Resource": "arn:${Partition}:iotanalytics:${Region}:${Account}:channel/
${channelName}",
            "Resource": "arn:${Partition}:iotanalytics:${Region}:${Account}:dataset/
${datasetName}",
            "Resource": "arn:${Partition}:iotanalytics:${Region}:${Account}:datastore/
${datastoreName}",
            "Resource": "arn:${Partition}:iotanalytics:${Region}:${Account}:pipeline/
${pipelineName}"
        }
    ]
 }
```

AWS CLI または AWS API のみを呼び出すユーザーには、最小限のコンソールアクセス許可を付与する必要はありません。代わりに、実行しようとしている API オペレーションに一致するアクションのみへのアクセスが許可されます。

自分の権限の表示をユーザーに許可する

この例では、ユーザー ID にアタッチされたインラインおよび管理ポリシーの表示を ユーザーに許可するポリシーを作成する方法を示します。このポリシーには、コンソールで、または AWS CLI または AWS API を使用してプログラムでこのアクションを実行するアクセス許可が含まれています。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
```

```
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": [
                "arn:aws:iam::*:user/${aws:username}"
            ]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

1 つの AWS IoT Analytics 入力へのアクセス

この例では、チャネルの 1 つである AWS IoT Analytics AWS アカウント へのアクセスを のユーザーに許可しますexampleChannel。また、ユーザーに対してチャネルの追加、更新、削除の実行も許可します。

このポリシーでは、ユーザーに iotanalytics:ListChannels, iotanalytics:DescribeChannel, iotanalytics:CreateChannel, iotanalytics:DeleteChannel, and iotanalytics:UpdateChannel アクセス許可を付与します。コンソールを使用して、ユーザーにアクセス許可を付与しテストする Amazon S3 サービス例の解説については、「チュートリアル例: ユーザーポリシーを使用したバケットへのアクセスの制御」を参照してください。

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
```

```
         "Sid":"ListChannelsInConsole",
         "Effect":"Allow",
         "Action":[
            "iotanalytics:ListChannels"
         ],
         "Resource":"arn:aws:iotanalytics:::*"
      },
      {
         "Sid":"ViewSpecificChannelInfo",
         "Effect":"Allow",
         "Action":[
            "iotanalytics:DescribeChannel"
         ],
         "Resource":"arn:aws:iotanalytics:::exampleChannel"
      },
      {
         "Sid":"ManageChannels",
         "Effect":"Allow",
         "Action":[
            "iotanalytics:CreateChannel",
            "iotanalytics:DeleteChannel",
            "iotanalytics:DescribeChannel",
            "iotanalytics:ListChannels",
            "iotanalytics:UpdateChannel"
         ],
         "Resource":"arn:aws:iotanalytics:::exampleChannel/*"
      }
   ]
}
```

タグに基づく AWS IoT Analytics チャネルの表示

アイデンティティベースのポリシーの条件を使用して、タグに基づいて AWS IoT Analytics リソースへのアクセスを制御できます。この例では、`channel` を表示できるポリシーを作成する方法を示します。ただし、アクセス許可は、`channel` タグ `Owner` にそのユーザーのユーザー名の値がある場合のみ付与されます。このポリシーでは、このアクションをコンソールで実行するために必要なアクセス権限も付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListChannelsInConsole",
```

```
            "Effect": "Allow",
            "Action": "iotanalytics:ListChannels",
            "Resource": "*"
        },
        {
            "Sid": "ViewChannelsIfOwner",
            "Effect": "Allow",
            "Action": "iotanalytics:ListChannels",
            "Resource": "arn:aws:iotanalytics:*:*:channel/*",
            "Condition": {
                "StringEquals": {"iotanalytics:ResourceTag/Owner": "${aws:username}"}
            }
        }
    ]
}
```

このポリシーをアカウントの ユーザーにアタッチできます。という名前のユーザーが を表示`richard-roe`しようとする場合 AWS IoT Analytics `channel`、 には というタグを付ける`channel`必要があります`Owner=richard-roe or owner=richard-roe`。それ以外の場合、そのユーザーのアクセスは拒否されます。条件キー名では大文字と小文字は区別されないため、条件タグキー `Owner` は `Owner` と `owner` に一致します。詳細については、IAM ユーザーガイド」の「IAM JSON ポリシー要素: 条件」を参照してください。

AWS IoT Analytics ID とアクセスのトラブルシューティング

以下の情報は、作業中に発生する可能性がある一般的な問題の診断と修正に役立ちます AWS IoT Analytics。

トピック

- でアクションを実行する権限がありません AWS IoT Analytics
- iam:PassRole を実行する権限がない
- 自分の AWS アカウント 以外のユーザーに自分の AWS IoT Analytics リソースへのアクセスを許可したい

でアクションを実行する権限がありません AWS IoT Analytics

でアクションを実行する権限がないと AWS Management Console 通知された場合は、管理者に連絡してサポートを依頼する必要があります。管理者とは、現在使用しているユーザー名とパスワードを発行した人物です。

以下の例のエラーは、mateojackson IAM ユーザーがコンソールを使用して、channel の詳細を表示しようとしているが、iotanalytics:ListChannels アクセス許可がない場合に発生します。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 iotanalytics:``ListChannels`` on resource: ``my-example-channel``
```

この場合、Mateo は、iotanalytics:ListChannel アクションを使用して my-example-channel リソースにアクセスできるように、ポリシーの更新を管理者に依頼します。

iam:PassRole を実行する権限がない

iam:PassRole アクションを実行する権限がないというエラーが表示された場合は、ポリシーを更新して AWS IoT Analyticsにロールを渡すことができるようにする必要があります。

一部の AWS のサービス では、新しいサービスロールまたはサービスにリンクされたロールを作成する代わりに、そのサービスに既存のロールを渡すことができます。そのためには、サービスにロールを渡す権限が必要です。

以下の例のエラーは、marymajor という IAM ユーザーがコンソールを使用して AWS IoT Analytics でアクションを実行しようする場合に発生します。ただし、このアクションをサービスが実行するには、サービスロールから付与された権限が必要です。メアリーには、ロールをサービスに渡す許可がありません。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

この場合、Mary のポリシーを更新してメアリーに iam:PassRole アクションの実行を許可する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン資格情報を提供した担当者が管理者です。

自分の AWS アカウント 以外のユーザーに自分の AWS IoT Analytics リソースへのアクセスを許可したい

他のアカウントのユーザーや組織外の人が、リソースにアクセスするために使用できるロールを作成できます。ロールの引き受けを委託するユーザーを指定できます。リソースベースのポリシーまたはアクセスコントロールリスト ACLをサポートするサービスの場合、それらのポリシーを使用して、リソースへのアクセスを付与できます。

詳細については、以下を参照してください。

- がこれらの機能 AWS IoT Analytics をサポートしているかどうかを確認するには、「 AWS IoT Analytics が IAM と連携する方法」を参照してください。
- 所有 AWS アカウント している のリソースへのアクセスを提供する方法については、「IAM ユーザーガイド」の「所有 AWS アカウント している別の の IAM ユーザーへのアクセスを提供する」を参照してください。
- リソースへのアクセスをサードパーティーに提供する方法については AWS アカウント、IAM ユーザーガイドの「サードパーティー AWS アカウント が所有する へのアクセスを提供する」を参照してください。
- ID フェデレーションを介してアクセスを提供する方法については、IAM ユーザーガイドの外部で認証されたユーザー (ID フェデレーション) へのアクセスの許可を参照してください。
- クロスアカウントアクセスでのロールとリソースベースのポリシーの使用の違いの詳細については、IAM ユーザーガイドのIAM ロールとリソースベースのポリシーとの相違点を参照してください。

でのログ記録とモニタリング AWS IoT Analytics

AWS には、モニタリングに使用できるツールが用意されています AWS IoT Analytics。自動的にモニタリングが行われるように、これらのツールを設定できます。手動操作を必要とするツールもあります。モニタリングタスクをできるだけ自動化することをお勧めします。

自動モニタリングツール

次の自動モニタリングツールを使用して、問題を監視 AWS IoT して報告できます。

- Amazon CloudWatch Logs - AWS CloudTrail またはその他のソースからのログファイルをモニタリング、保存、アクセスします。詳細については、Amazon CloudWatch ユーザーガイドの「AWS CloudTrail モニタリングログファイルとは」を参照してください。
- AWS CloudTrail ログモニタリング - アカウント間でログファイルを共有し、CloudWatch Logs に送信CloudWatch CloudTrail ログファイルをリアルタイムでモニタリングし、Java でログ処理アプリケーションを書き込み、CloudTrail による配信後にログファイルが変更されていないことを確認します。詳細については、AWS CloudTrail ユーザーガイドのCloudTrail ログファイルの操作を参照してください。

手動モニタリングツール

モニタリングのもう 1 つの重要な点は AWS IoT 、CloudWatch アラームでカバーされていない項目を手動でモニタリングすることです。 AWS IoT、CloudWatch、およびその他の AWS サービスコンソールダッシュボードには、 AWS 環境の状態がat-a-glanceビューが表示されます。ログファイルも確認することをお勧めします AWS IoT Analytics。

- AWS IoT Analytics コンソールには以下が表示されます。
  - チャンネル
  - Pipelines
  - データストア
  - データセット
  - Notebooks
  - 設定
  - 学習
- CloudWatch のホームページには、以下の情報が表示されます。
  - 現在のアラームとステータス
  - アラームとリソースのグラフ
  - サービスのヘルスステータス

  また、CloudWatch を使用して以下のことを行えます。
  - 重視するサービスをモニタリングするためのカスタマイズしたダッシュボードを作成します
  - メトリクスデータをグラフ化して、問題のトラブルシューティングを行い、傾向を確認する
  - すべての AWS リソースメトリクスを検索して参照する
  - 問題があることを通知するアラームを作成/編集する

Amazon CloudWatch Logs によるモニタリング

AWS IoT Analytics は Amazon CloudWatch でのログ記録をサポートしています。`PutLoggingOptions`API operation を使用することで、 AWS IoT Analytics のAmazon CloudWatch loggingを有効にし、設定することができる。このセクションでは、 AWS Identity and Access Management (IAM) `PutLoggingOptions` で を使用して Amazon CloudWatch ログ記録を設定および有効にする方法について説明します AWS IoT Analytics。

CloudWatch Logs の詳細については、Amazon CloudWatch Logs ユーザーガイドを参照してください。IAM の詳細については、 AWS 「 AWS Identity and Access Management ユーザーガイド」を参照してください。

> **Note**
>
> AWS IoT Analytics ログ記録を有効にする前に、CloudWatch Logs のアクセス許可を理解していることを確認してください。CloudWatch Logs に対するアクセス権限のあるユーザーは、デバッグ情報を表示できます。詳細については、「Amazon CloudWatch Logs に対する認証とアクセスコントロール」を参照してください。

IAM ロールを作成してログ記録を有効にする

Amazon CloudWatch のログ記録を有効にする IAM ロールを作成するには

1. AWS IAM コンソールまたは次の AWS IAM CLI コマンド CreateRole を使用して、信頼関係ポリシー (信頼ポリシー) を持つ新しい IAM ロールを作成します。信頼ポリシーは、Amazon CloudWatch などのエンティティにロールを継承するために許可を付与します。

   ```
   aws iam create-role --role-name exampleRoleName --assume-role-policy-document
          exampleTrustPolicy.json
   ```

   `exampleTrustPolicy.json` ファイルには次のコンテンツが含まれます。

   > **Note**
   >
   > この例には、「混乱した代理」問題から保護するためのグローバル条件コンテキストキーが含まれています。*123456789012* を AWS アカウント ID に置き換え、*aws-region* を AWS リソースの AWS リージョンに置き換えます。詳細については、「the section called “サービス間の混乱した代理の防止”」を参照してください。

   ```
   {
      "Version": "2012-10-17",
      "Statement": [
         {
            "Effect": "Allow",
            "Principal": {
   ```

```
            "Service": "iotanalytics.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
        "Condition": {
          "StringEquals": {
            "aws:SourceAccount": "123456789012"
          },
          "ArnLike": {
            "aws:SourceArn": "arn:aws:iotanalytics:aws-region:123456789012:*"
          }
        }

      }
   ]
}
```

このロールの ARN は、後で コマンドを AWS IoT Analytics `PutLoggingOptions`呼び出すときに使用します。

2. AWS IAM PutRolePolicy を使用して、ステップ 1 で作成したロールにアクセス許可ポリシー (`role policy`) をアタッチします。

   ```
   aws iam put-role-policy --role-name exampleRoleName --policy-name
   examplePolicyName --policy-document exampleRolePolicy.json
   ```

   exampleRolePolicy.json ファイルには次のコンテンツが含まれます。

   ```
   {
      "Version": "2012-10-17",
      "Statement": [
      {
      "Effect": "Allow",
      "Action": [
       "logs:CreateLogGroup",
       "logs:CreateLogStream"
      ],
      "Resource": [
       "arn:aws:logs:*:*:*"
      ]
      }
      ]
   }
   ```

3. Amazon CloudWatch にログイベントを配置する AWS IoT Analytics アクセス許可を付与するには、Amazon CloudWatch コマンド PutResourcePolicy を使用します。

> **Note**
>
> 混乱した代理のセキュリティ上の問題を防ぐため、リソースポリシーで `aws:SourceArn` を指定することをお勧めします。これにより、指定したアカウントからのリクエストのみを許可するようにアクセスを制限できます。混乱した代理に関する問題の詳細については、「the section called “サービス間の混乱した代理の防止”」を参照してください。

```
aws logs put-resource-policy --policy-in-json
exampleResourcePolicy.json
```

`exampleResourcePolicy.json` リソースには、次に示すようなフィールドがあります。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "iotanalytics.amazonaws.com"
            },
            "Action": "logs:PutLogEvents",
            "Resource": "*",
            "Condition":{
                "ArnLike":{
                    "aws:SourceArn":"arn:aws:iotanalytics:us-east-1:123456789012:*/
*"
                },
                "StringEquals":{
                    "aws:SourceAccount":"123456789012"
                }
            }
    ]
}
```

ログ記録の設定および有効

`PutLoggingOptions` コマンドを使用して、 AWS IoT Analytics用に Amazon CloudWatch ログ記録を設定し、有効化します。`loggingOptions` フィールドの `roleArn` は、前のセクションで作成したロールの ARN にする必要があります。また、`DecribeLoggingOptions` コマンドを使用してログ記録オプションの設定を確認することもできます。

PutLoggingOptions

AWS IoT Analytics ログ記録オプションを設定または更新します。いずれかの `loggingOptions` フィールドの値を更新する場合、変更が有効になるまでに最大で 1 分かかることに注意してください また、`roleArn` フィールドで指定したロールにアタッチされるポリシーを変更する場合 たとえば、無効なポリシーを修正するなど)、この変更が有効になるまでには最大で 5 分かかります。詳細については、「`PutLoggingOptions`」を参照してください。

DescribeLoggingOptions

AWS IoT Analytics ログ記録オプションの現在の設定を取得します。詳細については、「`DescribeLoggingOptions`」を参照してください

名前空間、メトリクス、ディメンション

AWS IoT Analytics は、次のメトリクスを Amazon CloudWatch リポジトリに配置します。

名前空間
AWS/IoTAnalytics

メトリクス	説明
ActionExecution	実行されるアクションの数。
ActionExecutionThrottled	スロットリングされたアクションの数。
ActivityExecutionError	パイプラインアクティビティの実行中に生成されたエラーの数。
IncomingMessages	チャンネルに送信されるメッセージの数。

メトリクス	説明
PipelineConcurrentExecutionCount	同時に実行されたパイプラインアクティビティの数。

ディメンション	説明
ActionType	モニタリングされているアクションのタイプ。
ChannelName	モニタリングされているチャネルの名前。
DatasetName	モニタリングされているデータセットの名前。
DatastoreName	モニタリングされているデータストアの名前。
PipelineActivityName	モニタリングされているパイプラインアクティビティの名前。
PipelineActivityType	モニタリングされているパイプラインアクティビティのタイプ。
PipelineName	モニタリングされているパイプラインの名前。

Amazon CloudWatch Events によるモニタリング

AWS IoT Analytics アクティビティ中にランタイムエラーが発生すると、 は自動的に Amazon CloudWatch Events にイベントを発行します AWS Lambda 。このイベントには、詳細なエラーメッセージと、未処理チャネルメッセージを格納する Amazon Simple Storage Service Amazon S3オブジェクトのキーが含まれます。Amazon S3 キーを使用して、未処理のチャネルメッセージを再処理できます。詳細については、チャネルメッセージの再処理、AWS IoT Analytics API リファレンスのStartPipelineReprocessing API、Amazon CloudWatch Events ユーザーガイドの「Amazon CloudWatch Events とは」を参照してください。

また、Amazon CloudWatch Events による通知の送信や高度なアクションの実行が可能になるようにターゲットを設定することもできます。たとえば、Amazon Simple Queue Service Amazon SQS キューに通知を送信し、`StartReprocessingMessage` API を呼び出して Amazon S3 オブジェク

トに保存されているチャネルメッセージを処理することができます。Amazon CloudWatch Events では、次のような数多くのタイプのターゲットがサポートされています。

- Amazon Kinesis Streams
- AWS Lambda 関数
- Amazon Simple Notification Service (Amazon SNS)のトピック
- Amazon Simple Queue Service Amazon SQSキュー

詳細については、Amazon EventBridge ユーザーガイドの「Amazon EventBridge ターゲット」を参照してください。

CloudWatch Events リソースおよび関連するターゲットは、 AWS IoT Analytics リソースを作成した AWS リージョンにある必要があります。詳細については、「AWS 全般のリファレンス」の「サービスエンドポイントとクォータ」を参照してください。

AWS Lambda アクティビティのランタイムエラーについて Amazon CloudWatch Events に送信される通知は、次の形式を使用します。

```
{
    "version": "version-id",
    "id": "event-id",
    "detail-type": "IoT Analytics Pipeline Failure Notification",
    "source": "aws.iotanalytics",
    "account": "aws-account",
    "time": "timestamp",
    "region": "aws-region",
    "resources": [
        "pipeline-arn"
    ],
    "detail": {
        "event-detail-version": "1.0",
        "pipeline-name": "pipeline-name",
        "error-code": "LAMBDA_FAILURE",
        "message": "error-message",
        "channel-messages": {
            "s3paths": [
                "s3-keys"
            ]
        },
        "activity-name": "lambda-activity-name",
        "lambda-function-arn": "lambda-function-arn"
```

```
    }
}
```

通知の例

```
{
    "version": "0",
    "id": "204e672e-ef12-09af-4cfd-de3b53673ec6",
    "detail-type": "IoT Analytics Pipeline Failure Notification",
    "source": "aws.iotanalytics",
    "account": "123456789012",
    "time": "2020-10-15T23:47:02Z",
    "region": "ap-southeast-2",
    "resources": [
        "arn:aws:iotanalytics:ap-southeast-2:123456789012:pipeline/
test_pipeline_failure"
    ],
    "detail": {
        "event-detail-version": "1.0",
        "pipeline-name": "test_pipeline_failure",
        "error-code": "LAMBDA_FAILURE",
        "message": "Temp unavaliable",
        "channel-messages": {
        "s3paths": [
            "test_pipeline_failure/channel/cmr_channel/__dt=2020-10-15
 00:00:00/1602805530000_1602805560000_123456789012_cmr_channel_0_257.0.json.gz"
        ]
    },
    "activity-name": "LambdaActivity_33",
    "lambda-function-arn": "arn:aws:lambda:ap-
southeast-2:123456789012:function:lambda_activity"
    }
}
```

Amazon CloudWatch Events を通じた遅延データ通知の取得

指定した時間枠のデータを使用してデータセットコンテンツを作成する場合、一部のデータが処理に間に合わない可能性があります。遅延を許可するには、 `queryAction` (SQL クエリ) を適用してデータセットを作成する`QueryFilter`ときに の`deltaTime`オフセットを指定できます。 AWS IoT Analytics は差分時間内に到着したデータを処理し、データセットの内容にはタイムラグがあります。遅延データ通知機能を使えば、データがデルタ時間後に到着した場合に AWS IoT Analytics では Amazon CloudWatch Events を通じて通知を送信できます。

AWS IoT Analytics コンソール、API、 AWS Command Line Interface（AWS CLI）、または AWS SDK を使用して、データセットの遅延データルールを指定できます。

AWS IoT Analytics API では、 `LateDataRuleConfiguration` オブジェクトはデータセットの遅延データルール設定を表します。このオブジェクトは、`CreateDataset` と `UpdateDataset` API オペレーションに関連する `Dataset` オブジェクトの一部です。

パラメータ

を使用してデータセットの遅延データルールを作成する場合は AWS IoT Analytics、次の情報を指定する必要があります。

`ruleConfiguration` (**`LateDataRuleConfiguration`**)

遅延データルールの設定情報を含む構造体。

`deltaTimeSessionWindowConfiguration`

デルタ時間セッションウィンドウの設定情報を含む構造体。

DeltaTime は時間間隔を指定します。`DeltaTime` を使用して、前回の実行以降にデータストアに到着したデータでデータセットコンテンツを作成できます。`DeltaTime` の例については、「デルタウィンドウを使用して SQL データセットを作成する (CLI)」を参照してください。

`timeoutInMinutes`

時間間隔。を使用すると`timeoutInMinutes`、 AWS IoT Analytics は前回の実行以降に生成された遅延データ通知をバッチアップできます。 AWS IoT Analytics は、一度に 1 つの通知のバッチを CloudWatch Events に送信します。

タイプ: 整数

有効な範囲: 1 ~ 60

`ruleName`

遅延データルールの名前。

タイプ: 文字列

> **Important**
>
> `lateDataRules` を指定するには、データセットで `DeltaTime` フィルターを使用する必要があります。

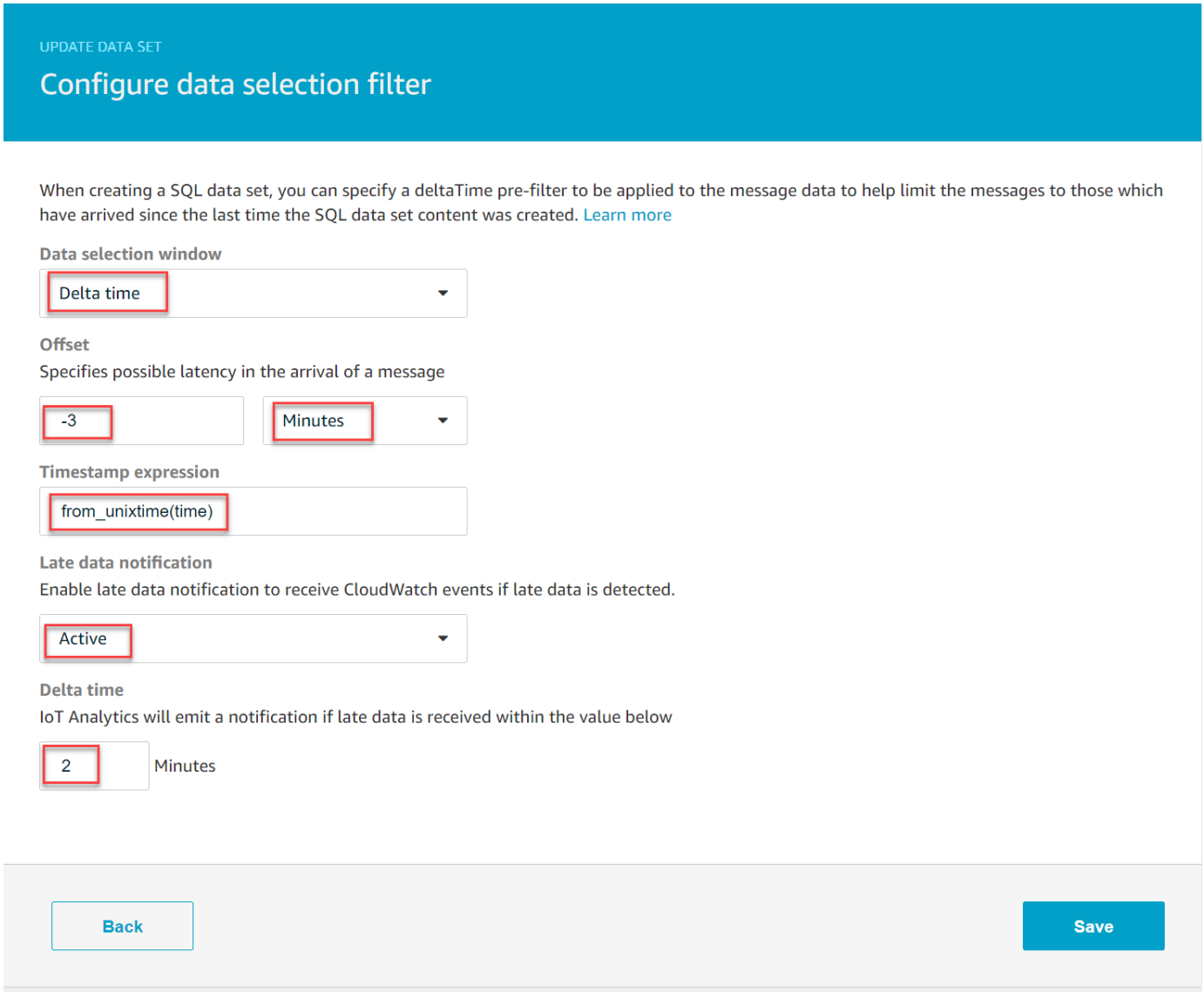
遅延データルールの設定 (コンソール)

以下の手順は、 AWS IoT Analytics コンソールでデータセットの遅延データルールを設定する方法を示しています。

遅延データルールの設定

1. AWS IoT Analytics コンソール にサインインします。
2. ナビゲーションペインで、[Data sets] (データセット) を選択します。
3. [Data sets] (データセット) でターゲットのデータセットを選択します。
4. ナビゲーションペインで、[Details] (詳細) を選択します。
5. [Delta window] (デルタウィンドウ) セクションで [Edit] (編集) を選択します。
6. [Configure data selection filter] (データ選択フィルターの設定) で以下を実行します。
   a. [Data selection window] (データ選択ウィンドウ) で [Delta time] (デルタ時間) を選択します。
   b. [Offset] (オフセット) で、期間を入力して単位を選択します。
   c. [Timestamp expression] (タイムスタンプ式) で式を入力します。これは、タイムスタンプフィールド名、または時間を取得できる SQL 式になります (*`from_unixtime(time)`* など)。

      タイムスタンプ式の記述方法の詳細については、Presto 0.172 ドキュメントの「日付と時刻の関数と演算子」を参照してください。
   d. [Late data notification] (遅延データ通知) で [Active] (アクティブ) を選択します。
   e. [Delta time] (デルタ時間) で、整数を入力します。有効な範囲は 1 ~ 60 です。
   f. [Save] を選択します。



遅延データルールの設定 (CLI)

AWS IoT Analytics API では、 `LateDataRuleConfiguration` オブジェクトはデータセットの遅延データルール設定を表します。このオブジェクトは、`CreateDataset` と `UpdateDataset` に関連する `Dataset` オブジェクトの一部です。API、AWS CLI、または AWS SDK を使用して、データセット用に遅延データルールを指定することができます。次の例では AWS CLIを使用しています。

指定した遅延データルールを使用してデータセットを作成するには、以下のコマンドを実行します。このコマンドでは、`dataset.json` ファイルが現在のディレクトリ内にあると想定します。

> **Note**
>
> UpdateDataset API を使用して既存のデータセットを更新できます。

```
aws iotanalytics create-dataset --cli-input-json file://dataset.json
```

`dataset.json` ファイルには次の内容が含まれます。

- *demo_dataset* をターゲットのデータセット名に置換します。
- *demo_datastore* をターゲットのデータセット名に置換します。
- *from_unixtime(time)* をタイムスタンプフィールド名、または時間を取得できる SQL 式に置換します。

  タイムスタンプ式の記述方法の詳細については、Presto 0.172 ドキュメントの「日付と時刻の関数と演算子」を参照してください。
- *timeout* を 1 ~ 60 の整数に置換します。
- *demo_rule* を任意の名前に置換します。

```
{
    "datasetName": "demo_dataset",
    "actions": [
        {
            "actionName": "myDatasetAction",
            "queryAction": {
                "filters": [
                    {
                        "deltaTime": {
                            "offsetSeconds": -180,
                            "timeExpression": "from_unixtime(time)"
                        }
                    }
                ],
                "sqlQuery": "SELECT * FROM demo_datastore"
            }
        }
    ],
    "retentionPeriod": {
```

```
        "unlimited": false,
        "numberOfDays": 90
    },
    "lateDataRules": [
        {
            "ruleConfiguration": {
                "deltaTimeSessionWindowConfiguration": {
                    "timeoutInMinutes": timeout
                }
            },
            "ruleName": "demo_rule"
        }
    ]
}
```

遅延データ通知を受信するためのサブスクライブ

CloudWatch Events で、 AWS IoT Analyticsから送信された遅延データ通知の処理方法を定義するルールを作成できます。CloudWatch Events により通知が受信されると、ルールで定義されている指定のターゲットアクションが呼び出されます。

CloudWatch Events ルール作成の前提条件

の CloudWatch Events ルールを作成する前に AWS IoT Analytics、以下を実行する必要があります。

- CloudWatch Events のイベント、ルール、およびターゲットをしっかりと理解しておきます。
- CloudWatch Events ルールによって呼び出されるターゲットを作成して設定します。ルールにより、以下のような数多くのタイプのターゲットを呼び出すことができます。
  - Amazon Kinesis Streams
  - AWS Lambda 関数
  - Amazon Simple Notification Service (Amazon SNS)のトピック
  - Amazon Simple Queue Service Amazon SQS キュー

  CloudWatch Events ルールと関連するターゲットは、 AWS IoT Analytics リソースを作成した AWS リージョンにある必要があります。詳細については、「AWS 全般のリファレンス」の「サービスエンドポイントとクォータ」を参照してください。

詳細については、Amazon CloudWatch Events ユーザーガイドの「CloudWatch Events とは」と「Amazon CloudWatch Events の開始方法」を参照してください。

遅延データ通知イベント

遅延データ通知のイベントでは次の形式を使用します。

```
{
 "version": "0",
 "id": "7f51dfa7-ffef-97a5-c625-abddbac5eadd",
 "detail-type": "IoT Analytics Dataset Lifecycle Notification",
 "source": "aws.iotanalytics",
 "account": "123456789012",
 "time": "2020-05-14T02:38:46Z",
 "region": "us-east-2",
 "resources": ["arn:aws:iotanalytics:us-east-2:123456789012:dataset/demo_dataset"],
 "detail": {
  "event-detail-version": "1.0",
  "dataset-name": "demo_dataset",
  "late-data-rule-name": "demo_rule",
  "version-ids": ["78244852-8737-4650-aa4d-3071a01338fa"],
  "message": null
 }
}
```

CloudWatch Events ルールを作成して遅延データ通知を受信する

次の手順では、 AWS IoT Analytics 遅延データ通知を Amazon SQS キューに送信するルールを作成する方法を示します。

CloudWatch Events ルールの作成方法

1. Amazon CloudWatch コンソールにサインインします。
2. ナビゲーションペインの [Events (イベント)] で、[Rules (ルール)] を選択します。
3. [Rules] (ルール) ページで、[Create rule] (ルールの作成) を選択します。
4. [Event Source] (イベントソース) で、[Event Pattern] (イベントパターン) を選択します。
5. [Build event pattern to match events by service] (サービス別のイベントに一致するイベントパターンの構築) セクションで以下の作業を実行します。
   a. [Service Name] (サービス名) で [IoT Analytics] を選択します。
   b. [Event Type] (イベントタイプ) で、[IoT Analytics Dataset Lifecycle Notification] (IoT Analytics データセットライフサイクル通知) を選択します。

c. [Specific dataset name(s)] (特定のデータセット名) を選択し、ターゲットデータセットの名前を入力します。

6. [Targets] (ターゲット) で、[Add target*] (ターゲットの追加) を選択します。
7. [SQS queue] (SQS キュー) を選択して、次の作業を行います。

   - [Queue*] (キュー) でターゲットキューを選択します。

8. [Configure details] (詳細の設定) を選択します。
9. [Step 2: Configure rule details] (ステップ 2: ルールの詳細を設定する) ページで、名前と説明を入力します。
10. [Create rule] (ルールの作成) を選択します。

を使用した AWS IoT Analytics API コールのログ記録 AWS CloudTrail

AWS IoT Analytics は、ユーザー AWS CloudTrail、ロール、または のサービスによって実行されたアクションを記録する AWS サービスである と統合されています AWS IoT Analytics。CloudTrail は、 AWS IoT Analytics コンソールからの呼び出しや API へのコード呼び出しを含む、 の API コールのサブセットをイベント AWS IoT Analytics としてキャプチャします。 AWS IoT Analytics APIs 証跡を作成する場合は、 イベントなど、Amazon S3 バケットへの CloudTrail イベントの継続的な配信を有効にすることができます AWS IoT Analytics。証跡を設定しない場合でも、CloudTrail コンソールの [イベント履歴] で最新のイベントを表示できます。CloudTrail によって収集された情報を使用して、リクエストの実行元の IP アドレス AWS IoT Analytics、リクエストの実行者、リクエストの実行日時などの詳細を確認できます。

CloudTrail の詳細については、「AWS CloudTrail ユーザーガイド」を参照してください。

AWS IoT Analytics の情報 AWS CloudTrail

CloudTrail は、 AWS アカウントの作成時にアカウントで有効になります。でアクティビティが発生すると AWS IoT Analytics、そのアクティビティはイベント履歴の他の AWS サービスイベントとともに CloudTrail イベントに記録されます。 AWS アカウントで最近のイベントを表示、検索、ダウンロードできます。詳細については、「Viewing events with CloudTrail event history」(CloudTrail イベント履歴でのイベントの表示) を参照してください。

のイベントなど、 AWS アカウントのイベントの継続的な記録については AWS IoT Analytics、証跡を作成します。証跡により、ログファイルを CloudTrail で Amazon S3 バケットに配信できます。デフォルトでは、コンソールで証跡を作成すると、すべての リージョンに証跡が適用されます。証跡は、 AWS パーティション内のすべてのリージョンからのイベントをログに記録し、指定した

Amazon S3 バケットにログファイルを配信します。さらに、CloudTrail ログで収集されたイベントデータをより詳細に分析し、それに基づいて行動するように、他の AWS サービスを設定できます。詳細については、以下を参照してください。

- 追跡を作成するための概要
- CloudTrail のサポートされているサービスと統合
- 「CloudTrail の Amazon SNS 通知の設定」
- 「複数のリージョンから CloudTrail ログファイルを受け取る」および「複数のアカウントから CloudTrail ログファイルを受け取る」

AWS IoT Analytics は、CloudTrail ログファイルのイベントとして次のアクションのログ記録をサポートします。

- CancelPipelineReprocessing
- CreateChannel
- CreateDataset
- CreateDatasetContent
- CreateDatastore
- CreatePipeline
- DeleteChannel
- DeleteDataset
- DeleteDatasetContent
- DeleteDatastore
- DeletePipeline
- DescribeChannel
- DescribeDataset
- DescribeDatastore
- DescribeLoggingOptions
- DescribePipeline
- GetDatasetContent
- ListChannels
- ListDatasets

- ListDatastores
- ListPipelines
- PutLoggingOptions
- RunPipelineActivity
- SampleChannelData
- StartPipelineReprocessing
- UpdateChannel
- UpdateDataset
- UpdateDatastore
- UpdatePipeline

各イベントまたはログエントリには、誰がリクエストを生成したかという情報が含まれます。アイデンティティ情報は、以下を判別するのに役立ちます。

- リクエストがルート認証情報と AWS Identity and Access Management ユーザー認証情報のどちらを使用して行われたか。
- リクエストがロールまたはフェデレーションユーザーのテンポラリなセキュリティ認証情報を使用して行われたかどうか。
- リクエストが別の AWS サービスによって行われたかどうか。

詳細については、「CloudTrail userIdentity エレメント」を参照してください。

AWS IoT Analytics ログファイルエントリについて

証跡は、指定した S3 バケットにイベントをログファイルとして配信するように設定できます。CloudTrail のログファイルには、単一か複数のログエントリがあります。イベントは任意の発生元からの 1 つの要求を表し、要求されたアクション、アクションの日時、要求のパラメータなどに関する情報が含まれます。CloudTrail ログファイルは、公開 API コールの順序付けられたスタックトレースではないため、特定の順序では表示されません。

次は、`CreateChannel` アクションを示す CloudTrail ログエントリの例です。

```
{
"eventVersion": "1.05",
"userIdentity": {
```

```
"type": "AssumedRole",
"principalId": "ABCDE12345FGHIJ67890B:AnalyticsChannelTestFunction",
"arn": "arn:aws:sts::123456789012:assumed-role/AnalyticsRole/
AnalyticsChannelTestFunction",
"accountId": "123456789012",
"accessKeyId": "ABCDE12345FGHIJ67890B",
"sessionContext": {
"attributes": {
 "mfaAuthenticated": "false",
 "creationDate": "2018-02-14T23:43:12Z"
},
"sessionIssuer": {
 "type": "Role",
 "principalId": "ABCDE12345FGHIJ67890B",
 "arn": "arn:aws:iam::123456789012:role/AnalyticsRole",
 "accountId": "123456789012",
 "userName": "AnalyticsRole"
}
}
},
"eventTime": "2018-02-14T23:55:14Z",
"eventSource": "iotanalytics.amazonaws.com",
"eventName": "CreateChannel",
"awsRegion": "us-east-1",
"sourceIPAddress": "198.162.1.0",
"userAgent": "aws-internal/3 exec-env/AWS_Lambda_java8",
"requestParameters": {
"channelName": "channel_channeltest"
},
"responseElements": {
"retentionPeriod": {
"unlimited": true
},
"channelName": "channel_channeltest",
"channelArn": "arn:aws:iotanalytics:us-east-1:123456789012:channel/channel_channeltest"
},
"requestID": "7f871429-11e2-11e8-9eee-0781b5c0ac59",
"eventID": "17885899-6977-41be-a6a0-74bb95a78294",
"eventType": "AwsApiCall",
"recipientAccountId": "123456789012"
}
```


以下の例は、`CreateDataset` アクションを示す CloudTrail ログエントリです。

```
{
"eventVersion": "1.05",
"userIdentity": {
"type": "AssumedRole",
"principalId": "ABCDE12345FGHIJ67890B:AnalyticsDatasetTestFunction",
"arn": "arn:aws:sts::123456789012:assumed-role/AnalyticsRole/
AnalyticsDatasetTestFunction",
"accountId": "123456789012",
"accessKeyId": "ABCDE12345FGHIJ67890B",
"sessionContext": {
"attributes": {
 "mfaAuthenticated": "false",
 "creationDate": "2018-02-14T23:41:36Z"
},
"sessionIssuer": {
 "type": "Role",
 "principalId": "ABCDE12345FGHIJ67890B",
 "arn": "arn:aws:iam::123456789012:role/AnalyticsRole",
 "accountId": "123456789012",
 "userName": "AnalyticsRole"
}
}
},
"eventTime": "2018-02-14T23:53:39Z",
"eventSource": "iotanalytics.amazonaws.com",
"eventName": "CreateDataset",
"awsRegion": "us-east-1",
"sourceIPAddress": "198.162.1.0",
"userAgent": "aws-internal/3 exec-env/AWS_Lambda_java8",
"requestParameters": {
"datasetName": "dataset_datasettest"
},
"responseElements": {
"datasetArn": "arn:aws:iotanalytics:us-east-1:123456789012:dataset/
dataset_datasettest",
"datasetName": "dataset_datasettest"
},
"requestID": "46ee8dd9-11e2-11e8-979a-6198b668c3f0",
"eventID": "5abe21f6-ee1a-48ef-afc5-c77211235303",
"eventType": "AwsApiCall",
"recipientAccountId": "123456789012"
}
```

のコンプライアンス検証 AWS IoT Analytics

AWS のサービス が特定のコンプライアンスプログラムの範囲内にあるかどうかを確認するには、AWS のサービス 「コンプライアンスプログラムによる範囲内」を参照して、関心のあるコンプライアンスプログラムを選択します。一般的な情報については、AWS 「 Compliance ProgramsAssurance」を参照してください。

を使用して、サードパーティーの監査レポートをダウンロードできます AWS Artifact。詳細については、「Downloading Reports in AWS Artifact」を参照してください。

を使用する際のお客様のコンプライアンス責任 AWS のサービス は、お客様のデータの機密性、貴社のコンプライアンス目的、適用される法律および規制によって決まります。 は、コンプライアンスに役立つ以下のリソース AWS を提供します。

- セキュリティのコンプライアンスとガバナンス – これらのソリューション実装ガイドでは、アーキテクチャ上の考慮事項について説明し、セキュリティとコンプライアンスの機能をデプロイする手順を示します。
- HIPAA 対応サービスのリファレンス – HIPAA 対応サービスの一覧が提供されています。すべての AWS のサービス が HIPAA の対象となるわけではありません。
- AWS コンプライアンスリソース – このワークブックとガイドのコレクションは、お客様の業界や地域に適用される場合があります。
- AWS カスタマーコンプライアンスガイド – コンプライアンスの観点から責任共有モデルを理解します。このガイドは、複数のフレームワーク (米国国立標準技術研究所 (NIST)、Payment Card Industry Security Standards Council (PCI)、国際標準化機構 (ISO) を含む) のセキュリティコントロールを保護し、そのガイダンスに AWS のサービス マッピングするためのベストプラクティスをまとめたものです。
- 「 デベロッパーガイド」の「ルールによるリソースの評価」 – この AWS Config サービスは、リソース設定が内部プラクティス、業界ガイドライン、および規制にどの程度準拠しているかを評価します。 AWS Config
- AWS Security Hub – これにより AWS のサービス 、 内のセキュリティ状態を包括的に把握できます AWS。Security Hub では、セキュリティコントロールを使用して AWS リソースを評価し、セキュリティ業界標準とベストプラクティスに対するコンプライアンスをチェックします。サポートされているサービスとコントロールの一覧については、Security Hub のコントロールリファレンスを参照してください。
- Amazon GuardDuty – 不審なアクティビティや悪意のあるアクティビティがないか環境をモニタリングすることで AWS アカウント、、ワークロード、コンテナ、データに対する潜在的な脅威 AWS のサービス を検出します。GuardDuty を使用すると、特定のコンプライアンスフレームワー

クで義務付けられている侵入検知要件を満たすことで、PCI DSS などのさまざまなコンプライアンス要件に対応できます。

- AWS Audit Manager – これにより AWS のサービス 、 AWS 使用状況を継続的に監査し、リスクの管理方法と規制や業界標準への準拠を簡素化できます。

の耐障害性 AWS IoT Analytics

AWS グローバルインフラストラクチャは、 AWS リージョンとアベイラビリティーゾーンを中心に構築されています。 AWS リージョンには、低レイテンシー、高いスループット、そして高度の冗長ネットワークで接続されている複数の物理的に独立し隔離されたアベイラビリティーゾーンがあります。アベイラビリティーゾーンでは、アベイラビリティーゾーン間で中断することなく自動的にフェイルオーバーするアプリケーションとデータベースを設計および運用することができます。アベイラビリティーゾーンは、従来の単一または複数のデータセンターインフラストラクチャよりも可用性、耐障害性、およびスケーラビリティが優れています。

AWS リージョンとアベイラビリティーゾーンの詳細については、AWS 「 グローバルインフラストラクチャ」を参照してください。

のインフラストラクチャセキュリティ AWS IoT Analytics

マネージドサービスである AWS IoT Analytics は、 AWS グローバルネットワークセキュリティで保護されています。 AWS セキュリティサービスと がインフラストラクチャ AWS を保護する方法については、AWS 「 クラウドセキュリティ」を参照してください。インフラストラクチャセキュリティのベストプラクティスを使用して AWS 環境を設計するには、「Security Pillar AWS Well-Architected Framework」の「Infrastructure Protection」を参照してください。

AWS が発行した API コールを使用して、ネットワーク AWS IoT Analytics 経由で にアクセスします。クライアントは以下をサポートする必要があります。

- Transport Layer Security (TLS)。TLS 1.2 が必須で、TLS 1.3 をお勧めします。
- DHE (楕円ディフィー・ ヘルマン鍵共有) や ECDHE (楕円曲線ディフィー・ ヘルマン鍵共有) などの完全前方秘匿性 (PFS) による暗号スイート。これらのモードはJava 7 以降など、ほとんどの最新システムでサポートされています。

また、リクエストにはアクセスキー ID と、IAM プリンシパルに関連付けられているシークレットアクセスキーを使用して署名する必要があります。または、AWS Security Token ServiceAWS STSを使用して、一時的なセキュリティ認証情報を生成し、リクエストに署名することもできます。

AWS IoT Analytics クォータ

AWS 全般のリファレンス ガイドには、 AWS アカウントの AWS IoT Analytics のデフォルトのクォータが記載されています。指定されていない限り、各クォータは AWS リージョンごとです。詳細については、AWS 全般のリファレンス 全般リファレンスガイドの「AWS IoT Analytics エンドポイントとクォータ」および「AWS Service Quotas」を参照してください。

サービスクォータ の増加を要求するには、サポートセンターコンソールでサポートケースを送信します。詳細については、「Service Quotasユーザーガイド」の「クォータ引き上げのリクエスト」を参照してください。

AWS IoT Analytics コマンド

このトピックでは、サポートされているウェブサービスプロトコルのサンプルリクエスト AWS IoT Analytics、レスポンス、エラーなど、 の API オペレーションについて説明します。

AWS IoT Analytics アクション

AWS IoT Analytics API コマンドを使用して、IoT データを収集、処理、保存、分析できます。詳細については、 AWS IoT Analytics API リファレンスの AWS IoT Analytics でサポートされているアクションを参照してください。

AWS CLI コマンドリファレンスAWS IoT Analytics のセクションには、管理および操作に使用できる AWS CLI コマンドが含まれています AWS IoT Analytics。

AWS IoT Analytics データ

AWS IoT Analytics Data API コマンドを使用して、 AWS IoT Analytics `channel`、、`datastore`、および `pipeline`で高度なアクティビティを実行できます`dataset`。詳細については、AWS IoT Analytics 「 API リファレンス」の「Data」でサポートされている AWS IoT Analytics データ型を参照してください。

トラブルシューティング AWS IoT Analytics

次のセクションを参照して、エラーのトラブルシューティングを行い、問題を解決するための および考えられる解決策を見つけます AWS IoT Analytics。

トピック

- どうすれば AWS IoT Analyticsにメッセージが取り込まれているかどうかを確認できますか?
- パイプラインからメッセージが欠落するのはなぜですか? 解決策は?
- データストア内にデータがないのはなぜですか?
- データセットに __dt が表示されるのはなぜですか?
- データセットの完了に伴ってイベントを発生させるコードはどのように記述しますか?
- AWS IoT Analyticsを使用するようにノートブックインスタンスを正しく設定するにはどうしますか?
- インスタンスでノートブックを作成できないのはなぜですか?
- QuickSight でデータセットが表示されないのはなぜですか ?
- 既存の Jupyter Notebook にコンテナ化ボタンが表示されないのはなぜですか?
- コンテナ化プラグインのインストールが失敗する原因は何ですか?
- なぜコンテナ化プラグインによってエラーが返されるのですか?
- コンテナ化中に使用する変数が表示されません。
- どのような変数を入力としてコンテナに追加できますか?
- コンテナ出力をこの先の入力として設定するにはどうすればよいですか?
- コンテナデータセットが失敗する原因は何ですか?

どうすれば AWS IoT Analyticsにメッセージが取り込まれているかどうかを確認できますか?

ルールエンジンを通じてチャネルにデータを取り込むためのルールが正しく設定されているかどうかを確認してください。

```
aws iot get-topic-rule --rule-name your-rule-name
```

レスポンスは次のようになります。

```
{
    "ruleArn": "arn:aws:iot:us-west-2:your-account-id:rule/your-rule-name",
    "rule": {
        "awsIotSqlVersion": "2016-03-23",
        "sql": "SELECT * FROM 'iot/your-rule-name'",
        "ruleDisabled": false,
        "actions": [
            {
                "iotAnalytics": {
                    "channelArn":
 "arn:aws:iotanalytics:region:your_account_id:channel/your-channel-name"
                }
            }
        ],
        "ruleName": "your-rule-name"
    }
}
```

ルールで使用されているリージョンとチャネル名が正しいことを確認してください。データがルールエンジンに到達してルールが正しく実行されていることを確認するには、新しいターゲットを追加して一時的に着信メッセージを Amazon S3 バケットに保存することもできます。

パイプラインからメッセージが欠落するのはなぜですか? 解決策は?

- アクティビティで受信した JSON 入力が無効です。

  すべてのアクティビティ Lambda アクティビティを除くは、有効な JSON 文字列が入力として特に必要です。アクティビティで受信した JSON が無効である場合、メッセージは欠落してデータストアに到達しません。有効な JSON メッセージをサービスに取り込んでいることを確認してください。バイナリ入力の場合、パイプラインの最初のアクティビティが Lambda アクティビティであり、これによってバイナリデータが有効な JSON に変換されてから、次のアクティビティに渡されるか、データストアに保存されていることを確認します。詳細については、「Lambda 関数の例 2」を参照してください。
- Lambda アクティビティで呼び出された Lambda 関数のアクセス許可が不十分です。

  Lambda アクティビティの各 Lambda 関数に、 AWS IoT Analytics サービスから呼び出すアクセス許可があることを確認します。次の AWS CLI コマンドを使用して、 アクセス許可を付与できます。

```
aws lambda add-permission --function-name <name> --region <region> --statement-id
 <id> --principal iotanalytics.amazonaws.com --action lambda:InvokeFunction
```

- フィルタまたは removeAttribute アクティビティの定義が正しくありません。

  `filter` または `removeAttribute` アクティビティの定義が正しいかどうか確認してください。メッセージをフィルタで除外した場合やメッセージからすべての属性を削除した場合は、メッセージがデータストアに追加されません。

データストア内にデータがないのはなぜですか?

- データを取り込んでからデータが使用可能になるまでに遅延があります。

  データをチャネルに取り込んでからデータがデータストアで使用可能になるまでに数分かかることがあります。パイプラインアクティビティ数とパイプラインのカスタム Lambda アクティビティの定義に応じて、所要時間は異なります。

- メッセージがフィルタによってパイプラインから除外されています。

  パイプラインでメッセージを削除していないことを確認してください 前の質問と回答を参照してください)。

- データセットのクエリが正しくありません。

  データストアからデータセットを生成するクエリが正しいことを確認してください。データがデータストアに到達するように、クエリから不要なフィルタを削除します。

データセットに `__dt` が表示されるのはなぜですか?

- この列は、サービスによって自動的に追加され、データ取り込みの推定時間を示します。クエリを最適化するために使用できます。データセットの内容がこれだけの場合は、前の質問と回答を参照してください。

データセットの完了に伴ってイベントを発生させるコードはどのように記述しますか?

- `describe-dataset` コマンドに基づくポーリングをセットアップして、特定のタイプスタンプを持つデータセットのステータスが 「成功しました」 であるかどうかを確認する必要があります。

AWS IoT Analyticsを使用するようにノートブックインスタンスを正しく設定するにはどうしますか?

以下の手順に従って、ノートブックインスタンスの作成に使用している IAM ロールに必要なアクセス許可があることを確認してください。

1. SageMaker AI コンソールに移動し、ノートブックインスタンスを作成します。
2. 詳細を入力し、新しいロールの作成を選択します。ロールの ARN をメモします。
3. ノートブックインスタンスを作成します。これにより、SageMaker AI が使用できるロールも作成されます。
4. IAM コンソールに移動し、新しく作成した SageMaker AI ロールを変更します。このロールを開くと、管理ポリシーがあります。
5. インラインポリシーの追加をクリックして、サービスとして IoTAnalytics を選択します。読み取りアクセス許可で、GetDatasetContent を選択します。
6. ポリシーを確認し、ポリシー名を追加して、ポリシーを 作成 します。新しく作成されたロールには、データセットを読み取るポリシーアクセス許可が付与されるようになりました AWS IoT Analytics。
7. AWS IoT Analytics コンソールに移動し、ノートブックインスタンスにノートブックを作成します。
8. ノートブックインスタンスの状態が「進行中」になるのを待ちます。
9. ノートブックの作成 を選択し、作成したノートブックインスタンスを選択します。これにより、選択したテンプレートを使用してデータセットにアクセスできる Jupyter ノートブックが作成されます。

インスタンスでノートブックを作成できないのはなぜですか?

- 正しい IAM ポリシーを使用してノートブックインスタンスを作成していることを確認してください 前の質問の手順に従います)。
- ノートブックインスタンスの状態が 「進行中」であることを確認します インスタンスを作成するときの状態は 「保留中」です。通常、状態が「進行中」になるまでに約 5 分かかります。約 5 分後にノートブックインスタンスの状態が 「失敗」になった場合は、アクセス許可を再度確認します。

QuickSight でデータセットが表示されないのはなぜですか ?

QuickSight では、 AWS IoT Analytics データセットのコンテンツを読み取るためのアクセス許可が必要になる場合があります。アクセス許可を付与するには、次の手順に従います。

1. QuickSight の右上隅にあるアカウント名を選択し、QuickSight の管理を選択します。
2. 左側のナビゲーションペインで、キュリティとアクセス権限許可を選択します。QuickSight access to AWS services AWS サービスへの QuickSight アクセスで、アクセスが AWS IoT Analyticsに付与されていることを確認します。
   a. にアクセス AWS IoT Analytics できない場合は、追加または削除を選択します。
   b. AWS IoT Analytics の横のボックスを選択して、Update 更新を選択します。これにより、QuickSight にデータセットのコンテンツを読み取るアクセス許可が付与されます。
3. データの可視化をもう一度試してください。

AWS IoT Analytics と QuickSight の両方に同じ AWS リージョンを選択していることを確認してください。そうしないと、 AWS リソースへのアクセスに問題がある可能性があります。サポートされているリージョンのリストについては、『』の「 AWS IoT Analytics エンドポイントとクォータ」およびQuickSight エンドポイントとクォータ」を参照してくださいAmazon Web Services 全般のリファレンス。

既存の Jupyter Notebook にコンテナ化ボタンが表示されないのはなぜですか?

- これは、 AWS IoT Analytics コンテナ化プラグインが見つからないことが原因です。SageMaker ノートブックインスタンスを 2018 年 8 月 23 日以前に作成した場合には、「ノートブックのコンテナ化」の説明に従って、手動でプラグインをインストールする必要があります。
- AWS IoT Analytics コンソールから SageMaker ノートブックインスタンスを作成した後、または手動でインストールした後にコンテナ化ボタンが表示されない場合は、 AWS IoT Analytics テクニカルサポートにお問い合わせください。

コンテナ化プラグインのインストールが失敗する原因は何ですか?

- 通常の場合、プラグインのインストールの失敗は、SageMaker ノートブックインスタンスへのアクセス権限がないことが原因です。ノートブックインスタンスに必要なアクセス許可については、「アクセス許可」を参照し、ノートブックインスタンスロールに必要なアクセス許可を追加します。問題が解決しない場合は、 AWS IoT Analytics コンソールから新しいノートブックインスタンスを作成します。
- プラグインのインストール中にログに表示される「To initialize this extension in the browser every time the notebook or other apploads.」ノートブック または他のアプリが読み込まれるたびに、この拡張機能をブラウザで初期化する方法というメッセージについては、無視しても問題ありません。

なぜコンテナ化プラグインによってエラーが返されるのですか?

- コンテナ化は複数の原因により失敗し、エラーを生成することがあります。ノートブックをコンテナ化する前に、正しいカーネルを使用していることを確認してください。コンテナ化されたカーネルは、「Containerized」というプレフィックスで始まります。
- プラグインは Docker イメージを ECR リポジトリに作成して保存するため、ノートブックインスタンスロールに ECR リポジトリを読み取り、リストして作成するために十分なアクセス権限があることを確認してください。ノートブックインスタンスに必要なアクセス許可については、「アクセス許可」を参照し、ノートブックインスタンスロールに必要なアクセス許可を追加します。
- また、リポジトリの名前が ECR 要件を準拠していることを確認します。ECR リポジトリ名は英字で始まる必要があり、小文字、数字、ハイフン、下線、スラッシュのみを含めることができます。"

- コンテナ化プロセスがエラーで失敗する場合：「このインスタンスでは、コンテナ化を実行するための空き領域が不足しています。
- 接続エラーあるいはイメージ作成エラーが表示されたら、再試行してください。問題が解決しない場合は、インスタンスを再起動し、最新のプラグインのバージョンをインストールします。

コンテナ化中に使用する変数が表示されません。

- AWS IoT Analytics コンテナ化プラグインは、「コンテナ化」カーネルでノートブックを実行した後、ノートブック内のすべての変数を自動的に認識します。コンテナ化されたカーネルの 1 つを使用してノートブックを実行し、コンテナ化を実行します。

どのような変数を入力としてコンテナに追加できますか?

- ランタイム中に変更する値を含む任意の変数をコンテナへの入力として追加できます。これにより、データセットの作成時に提供される必要がある別のパラメータを使用して、同じコンテナを実行できます。 AWS IoT Analytics コンテナ化 Jupyter プラグインは、ノートブック内の変数を自動的に認識し、コンテナ化プロセスの一部として使用できるようにすることで、このプロセスを簡素化します。

コンテナ出力をこの先の入力として設定するにはどうすればよいですか?

- 実行されたアーティファクトを保存できる特定の S3 の場所は、コンテナデータセットの実行ごとに作成されます。この出力場所にアクセスするには、コンテナデータセットで `outputFileUriValue` と入力してデータセット変数を作成します。この変数の値は、追加の出力ファイルを保存するために使用される S3 パスとする必要があります。これ以降の実行で上記で保存したアーティファクトにアクセスするには、`getDatasetContent` API を使用して、後続の実行で必要となる適切な出力ファイルを選択できます。

コンテナデータセットが失敗する原因は何ですか?

- 正しい `executionRole` をコンテナデータセットに渡そうとしていることを確認してください。`executionRole` の信頼ポリシーには `iotanalytics.amazonaws.com` と `sagemaker.amazonaws.com` が含まれていなければなりません。

- 失敗の原因として「`AlgorithmError`」と表示される場合には、手動でコンテナのデバッグを試行してください。これは、コンテナコードにバグがある場合、あるいは実行ロールにコンテナを実行できるアクセス権限がない場合に発生します。 AWS IoT Analytics Jupyter プラグインを使用してコンテナ化した場合には、同じロールを containerDataset の executionRole として使用して新しい SageMaker ノートブックインスタンスを作成し、手動でノートブックの実行を試行してください。Jupyter プラグイン外でコンテナが作成された場合には、手動でコードを実行して、executionRole のアクセス権限を制限してみてください。

ドキュメント履歴

次の表は、2020 年 11 月 3 日以降に加えられた AWS IoT Analytics ユーザーガイドに対する重要な変更点の一覧です。このドキュメントの更新情報は、RSS フィードに加入して取得できます。

変更	説明	日付
サポート終了通知	サポート終了通知: 2025 年 12 月 15 日、 AWS はサポートを終了します AWS IoT Analytics 。2025 年 12 月 15 日以降、AWS IoT Analytics コンソールまたは AWS IoT Analytics リソースにアクセスできなくなります。詳細については、AWS IoT Analytics 「サポート終了」を参照してください。	2025 年 5 月 20 日
AWS IoT Analytics は新規顧客には利用できなくなりました	AWS IoT Analytics は新規顧客には利用できなくなりました。の既存のお客様は、通常どおりサービスを AWS IoT Analytics 引き続き使用できます。詳細はこちら	2024 年 8 月 8 日
リージョンへの参入	AWS IoT Analytics がアジアパシフィック (ムンバイ) リージョンで利用可能になりました。	2021 年 8 月 18 日
`JOIN` によるクエリ	この更新により、 `JOIN`を使用して AWS IoT Analytics データセットをクエリできます。	2021 年 7 月 27 日
との統合 AWS IoT SiteWise	AWS IoT Analytics を使用して AWS IoT SiteWise データをク	2021 年 7 月 27 日

	エリできるようになりました。	
カスタムパーティション	AWS IoT Analytics では、通常、パイプラインアクティビティを通じて追加されたメッセージ属性に従ってデータをパーティション化できるようになりました。	2021 年 6 月 14 日
チャネルメッセージの再処理	この更新により、指定された Amazon S3 オブジェクトのチャネルデータを再処理できるようになりました。	2020 年 12 月 15 日
Parquet スキーマ	AWS IoT Analytics データストアが Parquet ファイル形式をサポートするようになりました。	2020 年 12 月 15 日
CloudWatch Events によるモニタリング	AWS IoT Analytics アクティビティ中にランタイムエラーが発生すると、 は Amazon CloudWatch Events にイベントを自動的に発行します AWS Lambda 。	2020 年 12 月 15 日
遅延データ通知	この機能を使用すれば、遅延データが到着したときに Amazon CloudWatch Events を通じて通知を受信できます。	2020 年 11 月 9 日
発売地域	中国 (北京) AWS IoT Analytics で開始されました。	2020 年 11 月 4 日

以前の更新

次の表は、2020 年 11 月 4 日以前に加えられた AWS IoT Analytics ユーザーガイドに対する重要な変更点の一覧です。

変更	説明	日付
リージョンへの参入	アジアパシフィック (シドニー) リージョン AWS IoT Analytics で開始されました。	2020 年 7 月 16 日
更新	ドキュメントが再編されました。	2020 年 5 月 7 日

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。