



ユーザーガイド

# AWS IoT Analytics



# AWS IoT Analytics: ユーザーガイド

Copyright © 2024 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

AWS IoT Analytics とは? .....	1
AWS IoT Analytics を使用する方法 .....	1
主な特徴 .....	2
AWS IoT Analytics のコンポーネントおよび概念 .....	4
AWS IoT Analytics へのアクセス .....	6
ユースケース .....	7
使用開始 コンソール .....	9
AWS IoT Analytics コンソールにサインインする .....	10
チャンネルの作成 .....	10
データストアの作成 .....	12
パイプラインを作成する .....	13
データセットの作成 .....	15
でメッセージデータを送信する AWS IoT .....	17
AWS IoT メッセージの進行状況を確認する .....	18
クエリ結果へのアクセス .....	19
データの探索 .....	19
ノートブックテンプレート .....	22
開始方法 .....	23
チャンネルの作成 .....	23
データストアの作成 .....	25
Amazon S3 ポリシー .....	25
ファイル形式 .....	27
カスタムパーティション .....	31
パイプラインの作成 .....	34
AWS IoT Analytics へのデータの取り込み .....	35
AWS IoT メッセージブローカーの使用 .....	35
BatchPutMessage API の使用 .....	39
取り込まれたデータのモニタリング .....	40
データセットの作成 .....	42
データのクエリ .....	43
クエリされたデータへのアクセス .....	43
AWS IoT Analytics データの調査 .....	19
Amazon S3 .....	44
AWS IoT Events .....	45

Amazon QuickSight .....	45
Jupyter Notebook .....	46
複数のバージョンのデータセットを保持する .....	46
メッセージペイロード構文 .....	47
AWS IoT SiteWise データの使用 .....	47
データセットの作成 .....	48
データセットコンテンツへのアクセス .....	52
チュートリアル: AWS IoT SiteWise データのクエリ .....	53
パイプラインアクティビティ .....	61
チャンネルアクティビティ .....	61
データストアアクティビティ .....	61
AWS Lambda アクティビティ .....	62
Lambda 関数の例 1 .....	62
Lambda 関数の例 2 .....	65
AddAttributes アクティビティ .....	66
RemoveAttributes アクティビティ .....	67
SelectAttributes アクティビティ .....	68
フィルターアクティビティ .....	69
DeviceRegistryEnrich アクティビティ .....	69
DeviceShadowEnrich アクティビティ .....	71
Math アクティビティ .....	73
Math アクティビティ演算子と関数 .....	74
RunPipelineActivity .....	90
チャンネルメッセージの再処理 .....	92
パラメータ .....	92
チャンネルメッセージの再処理 (コンソール) .....	93
チャンネルメッセージの再処理 (API) .....	94
チャンネルの再処理アクティビティのキャンセル .....	95
ワークフローの自動化 .....	96
ユースケース .....	97
Docker コンテナの使用 .....	98
カスタム Docker コンテナの入力/出力変数 .....	101
許可 .....	103
データセットを作成 (Java と AWS CLI) .....	105
例 1 -- SQL データセットを作成する java) .....	106
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) .....	106

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) .....	108
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) .....	109
例 5 -- SQL データセットを作成する (CLI) .....	110
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) .....	110
ノートブックをコンテナ化する .....	111
AWS IoT Analytics コンソールを介して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化する .....	112
ノートブックのコンテナ化拡張機能を更新する .....	115
コンテナ化イメージを作成する .....	115
カスタムコンテナの使用 .....	121
データの可視化 .....	130
可視化 (コンソール) .....	130
可視化 (QuickSight) .....	131
タグ付け .....	135
ベーシックタグ .....	135
IAM ポリシーでのタグの使用 .....	136
タグの制限 .....	138
SQL 式 .....	140
ポートされている SQL 機能 .....	141
サポートされているデータ型 .....	141
サポートされている関数 .....	142
一般的な問題のトラブルシューティング .....	143
セキュリティ .....	144
AWS Identity and Access Management .....	144
対象者 .....	144
アイデンティティを使用した認証 .....	145
アクセスの管理 .....	148
IAM の操作 .....	150
サービス間の混乱した代理の防止 .....	154
IAM ポリシーの例 .....	160
ID とアクセスのトラブルシューティング .....	166
ログ記録とモニタリング .....	168
自動モニタリングツール .....	168
手動モニタリングツール .....	169
CloudWatch Logs によるモニタリング .....	169

CloudWatch Events によるモニタリング .....	174
CloudTrail による API コールのログ記録 .....	183
コンプライアンス検証 .....	188
耐障害性 .....	189
インフラストラクチャセキュリティ .....	189
クォータ .....	191
コマンド .....	192
AWS IoT Analytics アクション .....	192
AWS IoT Analytics データ .....	192
トラブルシューティング .....	193
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? .....	193
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? .....	194
データストア内にデータがないのはなぜですか? .....	195
データセットに __dt が表示されるのはなぜですか? .....	195
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? .....	196
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? .....	196
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? .....	196
Amazon QuickSight で自分のデータセットが表示されないのはなぜですか? .....	197
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? .....	198
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? .....	198
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? .....	198
コンテナ化中に使用する変数が表示されません。 .....	199
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? .....	199
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? .....	199
コンテナデータセットが失敗する原因は何ですか? .....	199
ドキュメント履歴 .....	201
以前の更新 .....	202
.....	cciii

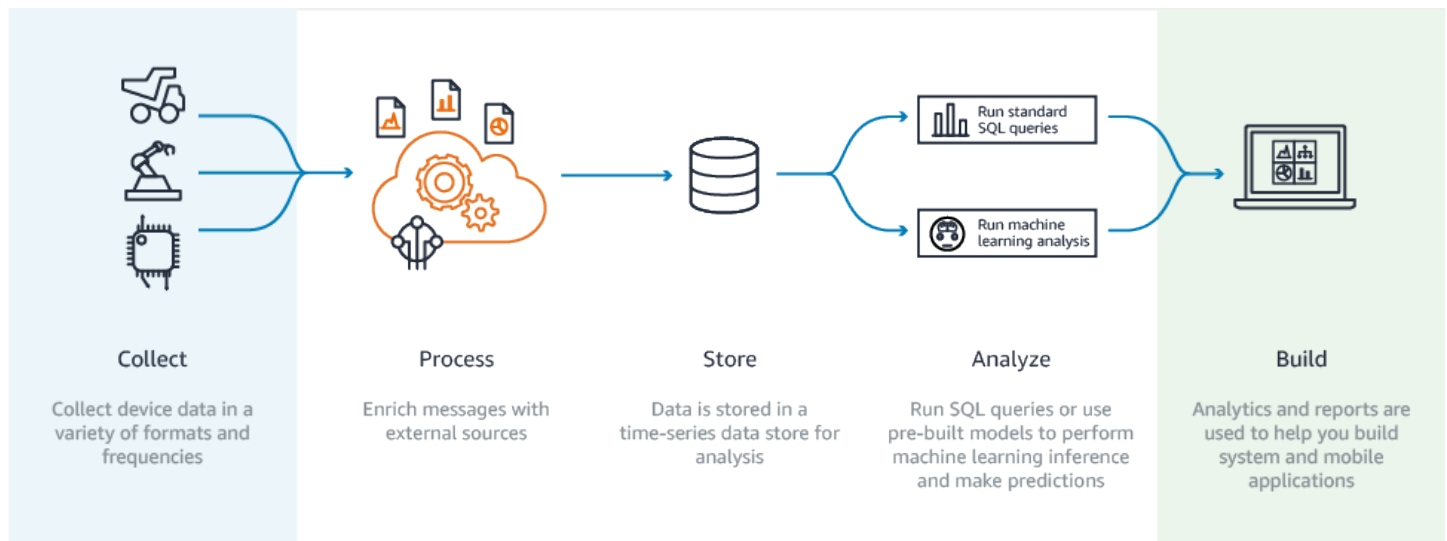
# AWS IoT Analytics とは？

AWS IoT Analytics では、IoT デバイスのデータの分析に必要なステップが自動化されます。AWS IoT Analytics では、分析用の時系列データストアに保存する前の IoT データのフィルタリング、変換、強化が行われます。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。また、内蔵の 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、複雑な分析や機械学習の推論を実行したりすることができます。AWS IoT Analytics では [Jupyter Notebook](#) との統合により高度なデータ探索が可能です。AWS IoT Analytics ではさらに [Amazon QuickSight](#) との統合によりデータの可視化も可能になります。Amazon QuickSight は次の [リージョン](#) で利用可能です。

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 のデータは、外部ソースの他のデータとの関連性の中でしか意味を持たないことが多いものです。AWS IoT Analytics を利用して、これらの問題への対処、大量のデバイスデータの収集、メッセージの処理と保存を実行できます。さらにデータのクエリと分析が可能になります。AWS IoT Analytics には一般的な IoT ユースケース用のモデルが組み込まれているため、どのデバイスが故障する可能性があるか、どの顧客がウェアラブルデバイスを廃棄する可能性があるかなどの質問に回答できます。

## AWS IoT Analytics を使用する方法

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

- AWS IoT Core への統合 — AWS IoT Analytics は AWS IoT Core と完全に統合されているため、コネクテッドデバイスからストリーミングされたメッセージを受信できます。
- バッチ API を使用した任意のソースからのデータの追加 — AWS IoT Analytics は HTTP を介して任意のソースからのデータを受信できます。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・分析するデータに限定した収集 — AWS IoT Analytics コンソールを使用すれば、MQTT トピックフィルターを通じてデバイスからのメッセージがさまざまな形式と頻度で受信されるように AWS IoT Analytics を設定することができます。AWS IoT Analytics では、定義した特定のパラメータ内にそのデータがあることが検証され、チャンネルが作成されます。次に、これらのチャン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

- 最適化とフィルター — AWS IoT Analytics では、AWS IoT Analytics が欠落データを検出したときにトリガーされる AWS Lambda 関数を定義すれば、コードを実行して欠落を推定し、それを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 — AWS IoT Analytics では、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変換できるため、摂氏のような一般的な計算を華氏に変換することができます。



- 強化 — AWS IoT Analytics では、天気予報などの外部データソースを使用してデータを強化し、そのデータを AWS IoT Analytics データストアにルーティングできます。

## 保存

- 時系列データストア - AWS IoT Analytics では、最適化した高速な取り出しと分析用の時系列データストアにデバイスデータを保存します。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。
- 処理済みデータと生データの保存 — AWS IoT Analytics では、処理されたデータを保存するだけでなく、取り込まれた未加工データも自動的に保存されるため、後でそのデータを処理することができます。

## 分析

- アドホック SQL クエリの実行 — AWS IoT Analytics には、SQL クエリエンジンが用意されているため、アドホッククエリを実行して結果を迅速に取得できます。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 - AWS IoT Analytics は時系列分析に対応しているため、時間の経過とともにデバイスのパフォーマンスを分析したり、使用方法や使用場所を把握するだけでなく、デバイスデータを継続的に監視してメンテナンスの問題を予測したり、センサーを監視して環境条件を予測して反応させることができます。
- 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 により作成された機械学習モデルと可視化が含まれている一連のノートブックテンプレートがあります。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

- Amazon QuickSight の統合 - AWS IoT Analytics には Amazon QuickSight へのコネクタがあるので、QuickSight ダッシュボードでデータセットを可視化することができます。
- コンソールの統合 - また、AWS IoT Analytics コンソール内に組み込まれている Jupyter Notebook で、結果やアドホック分析を可視化することもできます。

## AWS IoT Analytics のコンポーネントおよび概念

### チャンネル

チャンネルは、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、お客様または AWS IoT Analytics が管理する 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ン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、メッセージのスケラブルでクエリ可能なりポジトリです。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。未処理のチャンネルメッセージと同様に、データストアの処理済みメッセージは、お客様または AWS IoT Analytics によって管理されている [Amazon S3](#) バケットに保存されます。

### データセット

データセットを作成することでデータストアからデータを取得します。AWS IoT Analytics では 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、[Amazon QuickSight](#) と連携してデータを探索し、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することも、SageMaker で提供されるオプションを使用することもできます。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することも、[SageMaker](#) で

提供されるオプションを使用することもできます。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。

## AWS IoT Analytics へのアクセス

AWS IoT の一部として、AWS IoT Analytics には以下のインターフェイスがあり、これを使って、デバイスでデータを生成し、そのデータをアプリケーションで操作することができます。

### AWS Command Line Interface (AWS CLI)

Windows、OS X、Linux で AWS IoT Analytics 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。使用を開始するには、「[AWS Command Line Interface ユーザーガイド](#)」を参照してください。AWS IoT のコマンドの詳細については、AWS Command Line Interface リファレンスの [iot](#) を参照してください。

#### Important

AWS IoT Analytics を操作するには `aws iotanalytics` コマンドを使用します。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 SDK

言語固有の API 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 IoTDevice SDKs

AWS IoT Analytics に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。詳細については、[AWS IoTSDK](#) をご参照ください。

## 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て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。たとえば、AWS IoT Analytics では、コネクテッド貨物車で冷暖房装置が故障する可能性が高い時期を予測し、車両の経路を変更して、輸送中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、在庫をリアルタイムで監視できる 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

AWS IoT Analytics では、さまざまなプロセスの効率を常に監視し、プロセスを改善する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。AWS IoT Analytics で

は、時間の経過とともに場所やトラックに最も効率的な積載量を見極め、目標量との偏差をリアルタイムで比較して積載ガイドラインを改善することで、効率を高めることができます。

## スマートアグリカルチャー

AWS IoT Analytics では、時間や場所、温度、高度などの環境条件が分析に反映されるように、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、環境に応じたメタデータで IoT デバイスデータを強化できます。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。

# 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、チャンネル、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開始機能があります。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。

チャンネル、パイプライン AWS IoT Analytics、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、コンソールを使用して AWS IoT Core、に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](#)
- [チャンネルの作成](#)
- [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](#)
- [で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。AWS アカウントを既にお持ちの場合は、<https://console.aws.amazon.com/iotanalytics/> に移動します。

AWS アカウントをお持ちでない場合は、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、通話呼び出しを受け取り、電話キーパッドで検証コードを入力するように求められます。

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 サービス とリソースへのアクセス権があります。セキュリティのベストプラクティスとして、ユーザーに管理アクセスを割り当て、ルートユーザーのみを使用して [ルートユーザーアクセスが必要なタスク](#) を実行してください。

3. にサインイン AWS Management Console し、<https://console.aws.amazon.com/iotanalytics/> に移動します。

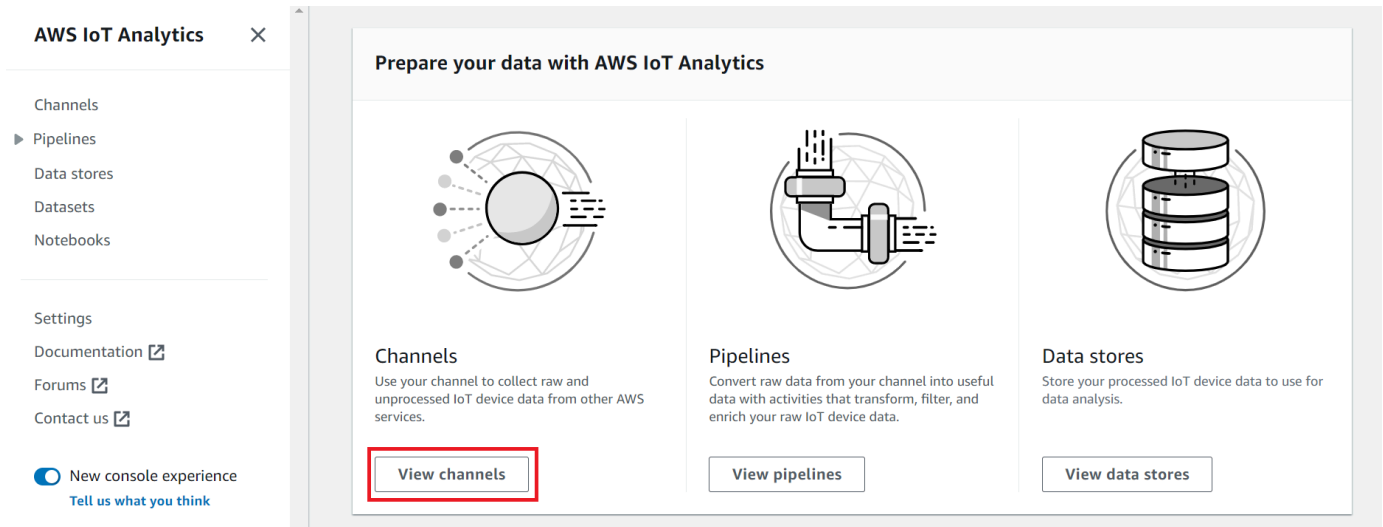
## チャネルの作成

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

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。



**i** Tip

ナビゲーションペインから **チャンネル** を選択することもできます。

2. チャンネル ページで、チャンネルの作成 を選択します。
3. チャンネルの詳細の指定 ページで、チャンネルの詳細情報を入力します。
  - a. チャン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 をチャンネルに追加します。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] をクリックします。
4. AWS IoT Analytics は、未処理の生の 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することも、Amazon S3 バケット AWS IoT Analytics を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 をクリックします。
5. 「ソースの設定」ページで、 が から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. などの AWS IoT Core トピックフィルターを入力します 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ンネルに送信します。
  - 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、チャンネルの作成 を選択します。
  7. 新しいチャンネルがチャン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ptional 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. 次へをクリックします。
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. 次へをクリックします。
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 をクリックします。

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ン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ン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ン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 チャンネルです。
  - 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。

- チャン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、IoT デバイスに関する有用なインサイトを発見するための AWS IoT Analytics SQL データセットを作成します。

# 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、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。

これで、チャン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。[Amazon スケジュール式と同様の CloudWatch 式を使用して](#)、オプションのスケジュールを作成できます。


## 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 [AWS IoT Analytics の SQL 式](#)。

- 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行う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、**バージョンング** 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以外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。

## で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ンネルがある場合、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**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 AWS アカウントまたは AWS リージョン間でメッセージを送信できません。

で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、チャン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、チャン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、チャンネル を選択し、前の段階で作成したチャン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 Datasets 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 データの可視化](#)」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 フィールド `contentDeliveryRules::destination::s3DestinationConfiguration` を参照してください。

## AWS IoT Events

データセットのコンテンツを への入力として送信できます。これは AWS IoT Events、デバイスまたはプロセスをモニタリングしてオペレーションの障害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るサービスです。

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 を指定する必要があります `iotevents:BatchPutMessage`。データセットの内容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、) を認識する AWS IoT Events 入力を作成し、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したりwho、内部変数を設定したりするディテクターモデルdt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 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 の間に関係を作成する必要があります SageMaker。

1. [SageMaker 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、ロールを変更します SageMaker。
  - 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、 から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 IoT Analytics 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、作成したノートブックインスタンスを選択します SageMaker。
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](#) 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 IoT Analytics ユースケースの開始に役立つ、AWS オーサリング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。

# AWS IoT Analytics の開始方法

このセクションでは、AWS IoT Analytics 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。ここに示す例では AWS Command Line Interface (AWS CLI) を使用しています。AWS CLI の詳細については、[AWS Command Line Interface ユーザーガイド](#) を参照してください。AWS IoT に使用できる CLI コマンドの詳細については、AWS Command Line Interface リファレンスの [iot](#) を参照してください。

## Important

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

## Note

以下の例で AWS IoT Analytics エンティティの名前 (チャンネル、データセット、データストア およびパイプライン) 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

チャンネルは、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ンネルに送信されるため、最初のステップとして、データ用のチャンネルを作成します。

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

AWS IoT メッセージを AWS IoT Analytics に取り込む場合は、AWS IoT ルールエンジンのルールを作成して、このチャンネルにメッセージを送信できます。これについては後ほど [AWS IoT Analytics へのデータの取り込み](#) で説明します。データをチャンネルに取得する別の方法は、AWS IoT Analytics コマンド「BatchPutMessage」を使用することです。

作成済みのチャンネルを一覧表示する方法:

```
aws iotanalytics list-channels
```

チャンネルに関する追加情報を取得する方法。

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

未処理のチャンネルメッセージは、AWS IoT Analytics が管理する Amazon S3 バケット、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 バケットにチャンネルメッセージを保存することを選択した場合、ユーザーに代わって Amazon S3 バケットで、s3:GetBucketLocation (バケットの場所の確認)、s3:PutObject (保存)、s3:GetObject (読み取り)、s3:ListBucket (再処理) のアクションを実行するには、AWS IoT Analytics のアクセス許可を付与する必要があります。

## 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ン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ン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。「[チャン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、メッセージのスケラブルでクエリ可能なりポジトリです。複数のデータストアを作成して異なるデバイスまたは場所からのメッセージを保存したり、1つ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することができます。

```
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、AWS IoT Analytics により管理されている Amazon S3 バケットまたはユーザーが管理するバケットに保存されます。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。

データストアのメッセージを、あなたが管理する Amazon S3 バケットに保存することを選択した場合、お客様のために Amazon S3 バケット上でこれらのアクションを実行する権限を AWS IoT Analytics に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、バケットのコンテンツで Amazon Athena クエリを実行するための 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する必要があります。

### 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ン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ン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 データストアでは現在、JSON と Parquet のファイル形式がサポートされています。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ために、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ます。AWS IoT Analytics データストアでは、JSON と Parquet がサポートされています。

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 では、[Amazon Athena のデータ型](#)ページに記載されているすべてのデータ型 (DECIMAL(*precision*, *scale*) - *precision* を除く) がサポートされています。

## 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. [Specify data store details] (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. [Data store ID] (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. (オプション) [Tags] (タグ) で [Add new tag] (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) をクリックします。
5. [Configure storage type] (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. [Storage type] (ストレージタイプ) で、[Service managed storage] (サービス管理ストレージ) を選択します。
  - b. [Configure how long you want to keep your processed data] (処理データの保存期間の設定) で、[Indefinitely] (無期限) を選択します。
  - c. [Next] (次へ) をクリックします。
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) をクリックします。

7. (オプション) AWS IoT Analytics ではデータストア内のカスタムパーティションがサポートされているため、プルーニングされたデータをクエリすればレイテンシーを改善することができます。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。

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 ではデータのパーティション化がサポートされているため、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. [Specify data store details] (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. [Data store ID] (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. (オプション) [Tags] (タグ) で [Add new tag] (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 に対して作成するリソースを識別できます。
  - c. [Next] (次へ) をクリックします。
5. [Configure storage type] (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. [Storage type] (ストレージタイプ) で、[Service managed storage] (サービス管理ストレージ) を選択します。

- b. [Configure how long you want to keep your processed data] (処理データの保存期間の設定) で、[Indefinitely] (無期限) を選択します。
  - c. [Next] (次へ) をクリックします。
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 でサポートされファイルタイプについては、「[ファイル形式](#)」を参照してください。

 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) をクリックします。
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。  
  
[Sample source] (サンプルソース) を選択して、このデータストアのメッセージを収集する AWS IoT Analytics チャンネルを選択します。
  - 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。
- [Attribute name] (属性名) および [Dimension name] (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、AWS IoT Analytics により、`__year`、`__month`、`__day`、`__hour` という名前の 4 つのディメンションが作成されます。
- [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。表示されている AWS IoT Analytics 形式オプションのうちの 1 つを選択するか、またはデータの形式に合うものを指定することができます。[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) をクリックします。
8. [Review and create] (確認して作成) ページで、選択内容を認め、[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、チャン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ン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ン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ン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
```

## AWS IoT Analytics へのデータの取り込み

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ンネルがある場合、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。AWS IoT メッセージブローカーまたは AWS IoT Analytics BatchPutMessage API を使用してメッセージを送信することができます。。

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、AWS IoT ルールエンジンを使用してルールを作成します。このルールにより、特定のトピックを含むメッセージが AWS IoT Analytics にルーティングされます。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 チャン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、その前にメッセージデータを AWS IoT Analytics チャンネルに送信するためのアクセス許可をルールに付与する 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

チャン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。チャン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。

## MQTT メッセージの AWS IoT Analytics への送信

ルールをチャンネルに、チャンネルをパイプラインに、パイプラインをデータストアに結合すると、ルールに一致するデータは、AWS IoT Analytics を経由して、クエリを実行する準備ができていてデータストアに流れるようになります。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できます。

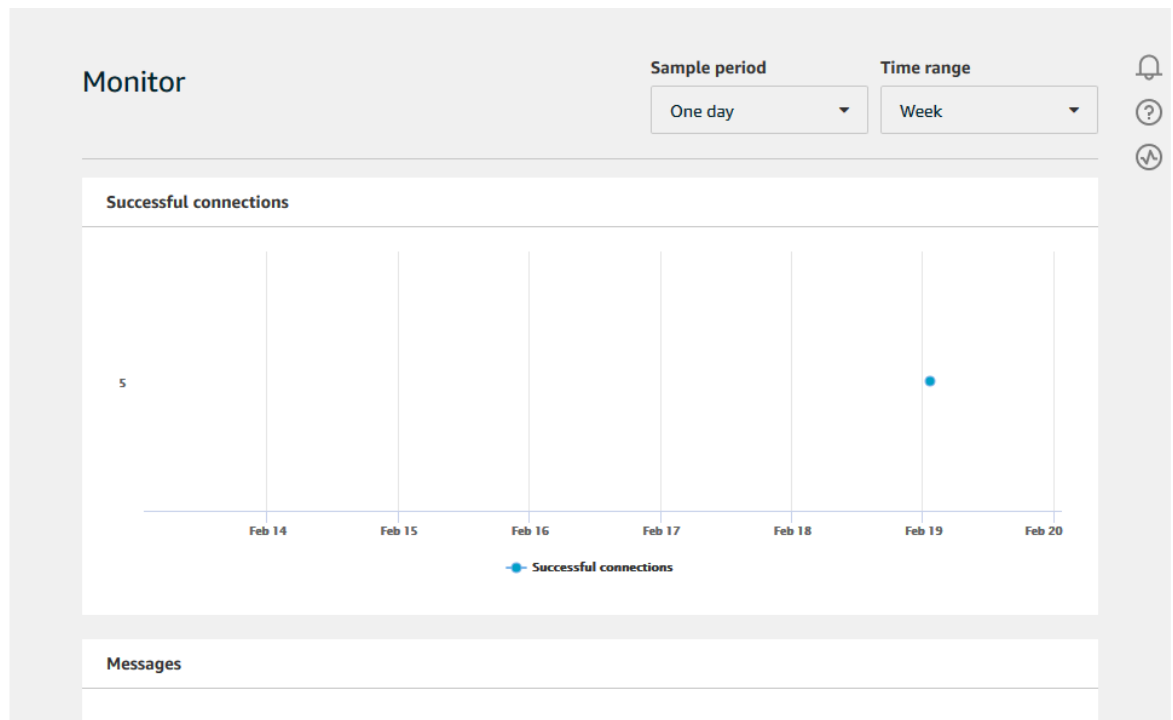
### Note

AWS IoT Analytics に送信するメッセージペイロードのフィールド名 (データ)。

- 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 F00 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。

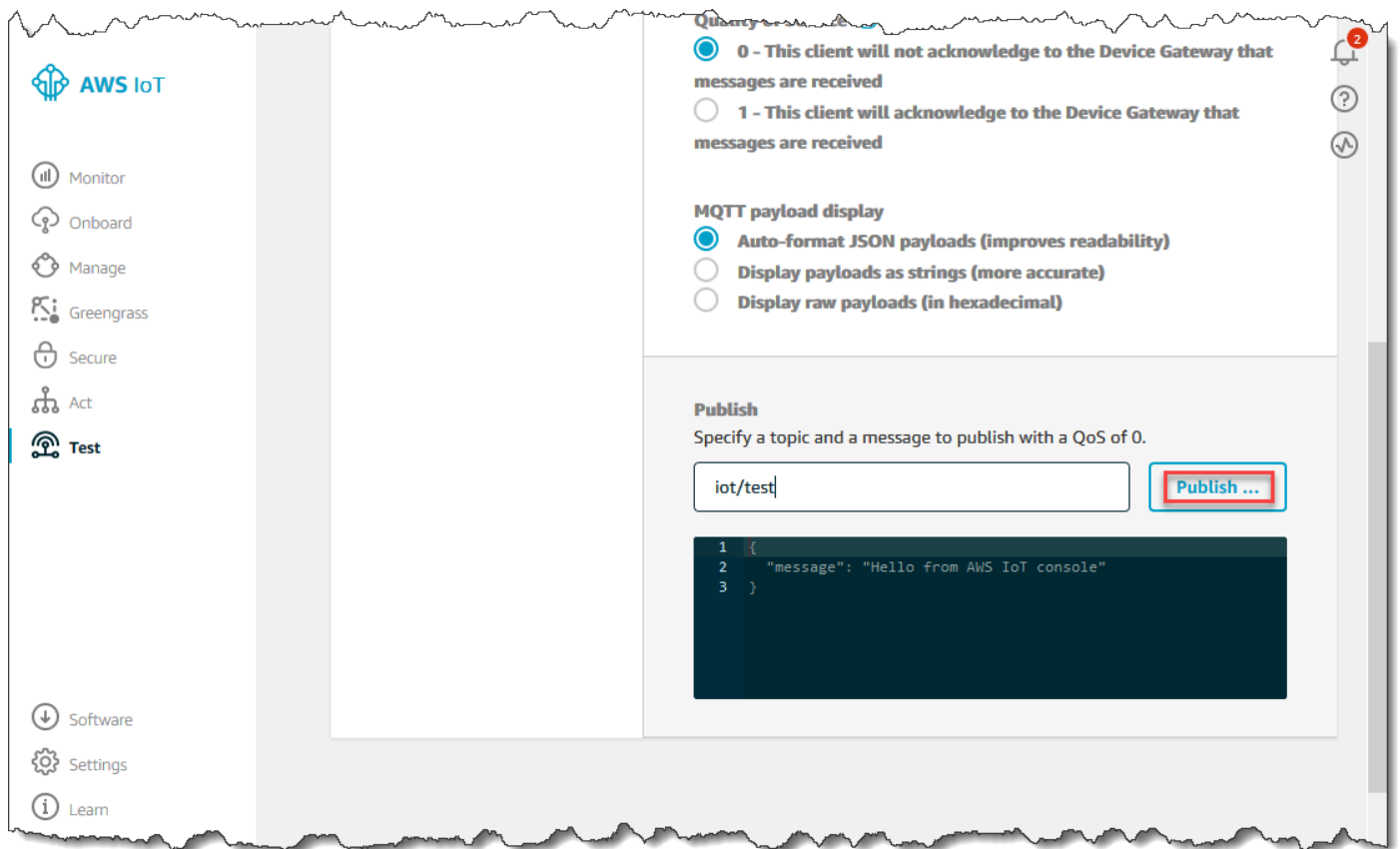
1. [AWS IoT コンソール](#)の、左のナビゲーションペインで、[Test (テスト)] を選択します。



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 AWS IoT Analytics に取り込むもう一つの方法として、BatchPutMessage API コマンドを使用する方法があります。このメソッドでは、メッセージをルートするためにチャンネルに特定のトピックを使用して 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ンネルに送信するデバイスが AWS SDK で作成したソフトウェアを実行できること、あるいは AWS CLI を使用して BatchPutMessage を呼び出すことができることが必須条件となります。

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\n\" }" }
```

]

## 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
```

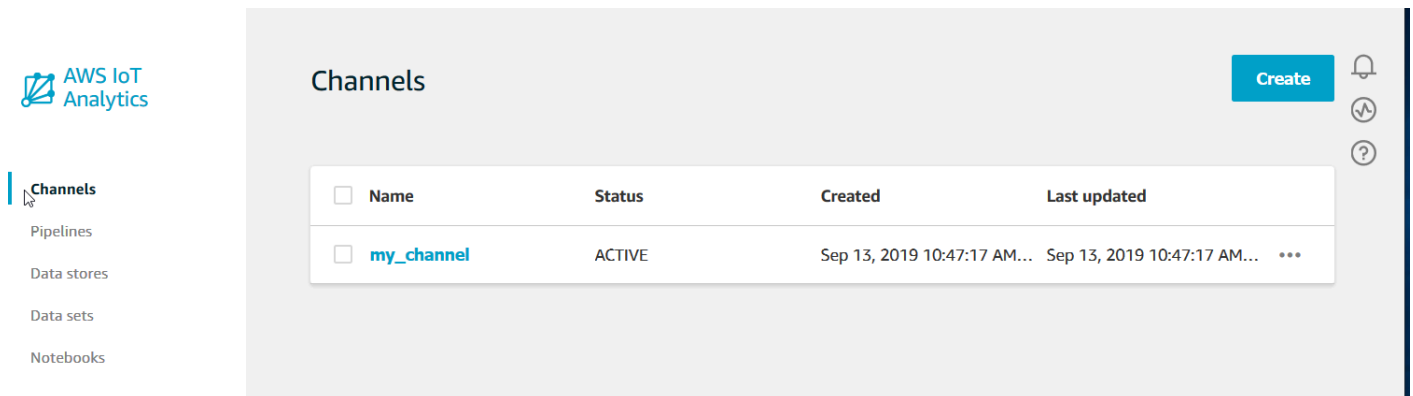
エラーがなければ次の出力が表示されます。

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

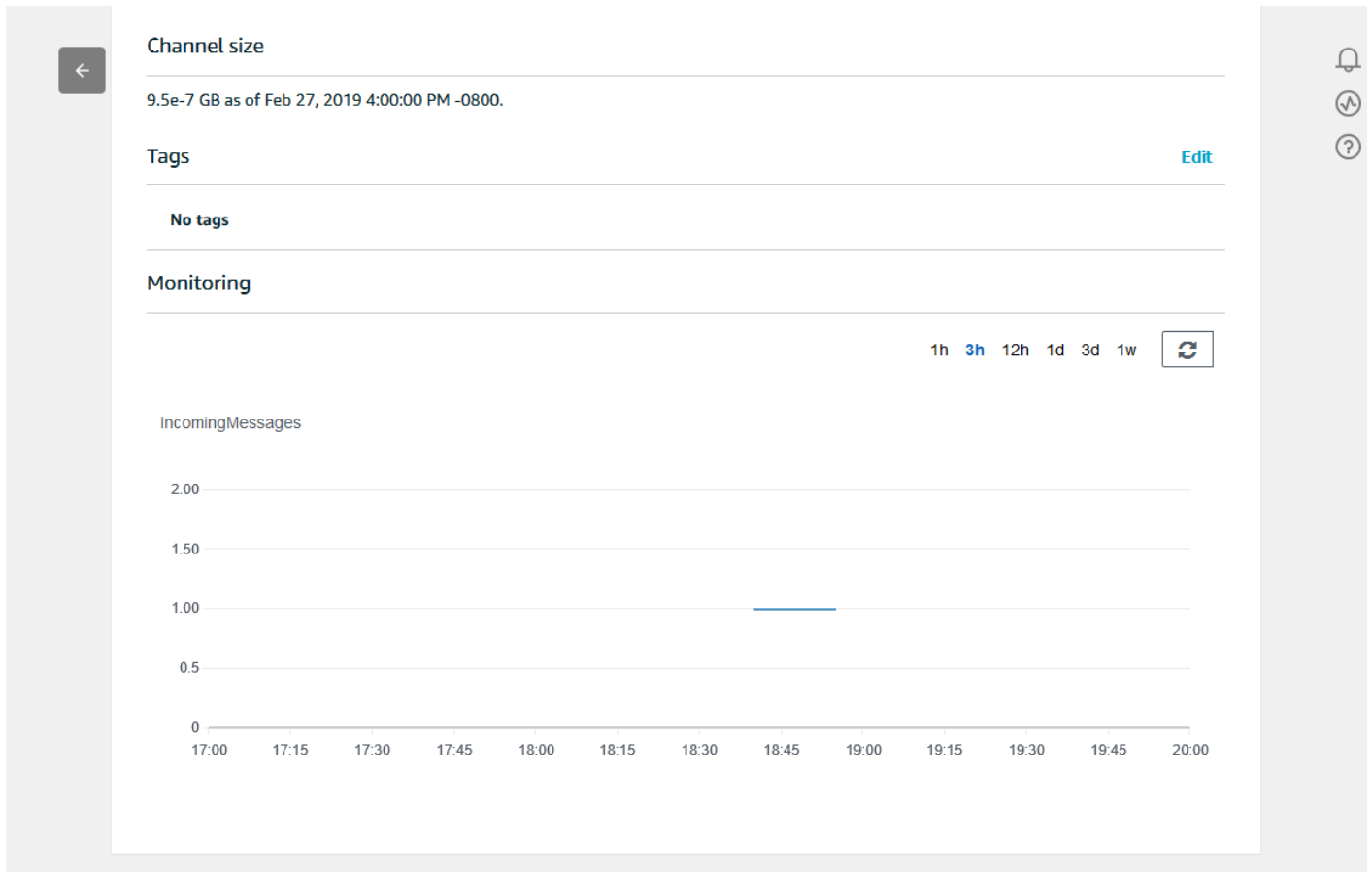
## 取り込まれたデータのモニタリング

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ン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] (チャンネル) を選択し、前に作成したチャンネルの名前を選択します。

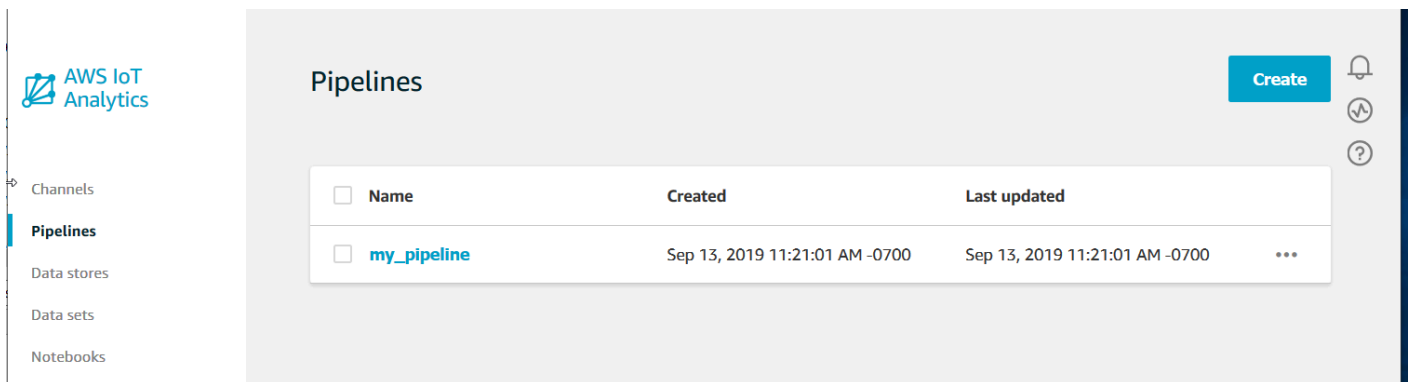


2. チャンネルの詳細ページで、[Monitoring (モニタリング)] セクションまで下にスクロールします。必要に応じて、時間ウィンドウインジケータ [1h 3h 12h 1d 3d 1w] (1 時間 3 時間 12 時間 1 日 3 日 1 週間) のいずれかを選択することで表示される時間枠ウィンドウを調整します。指定された時間ウィンドウ内にチャンネルに取り込まれたメッセージの数を示す線グラフが表示されます。



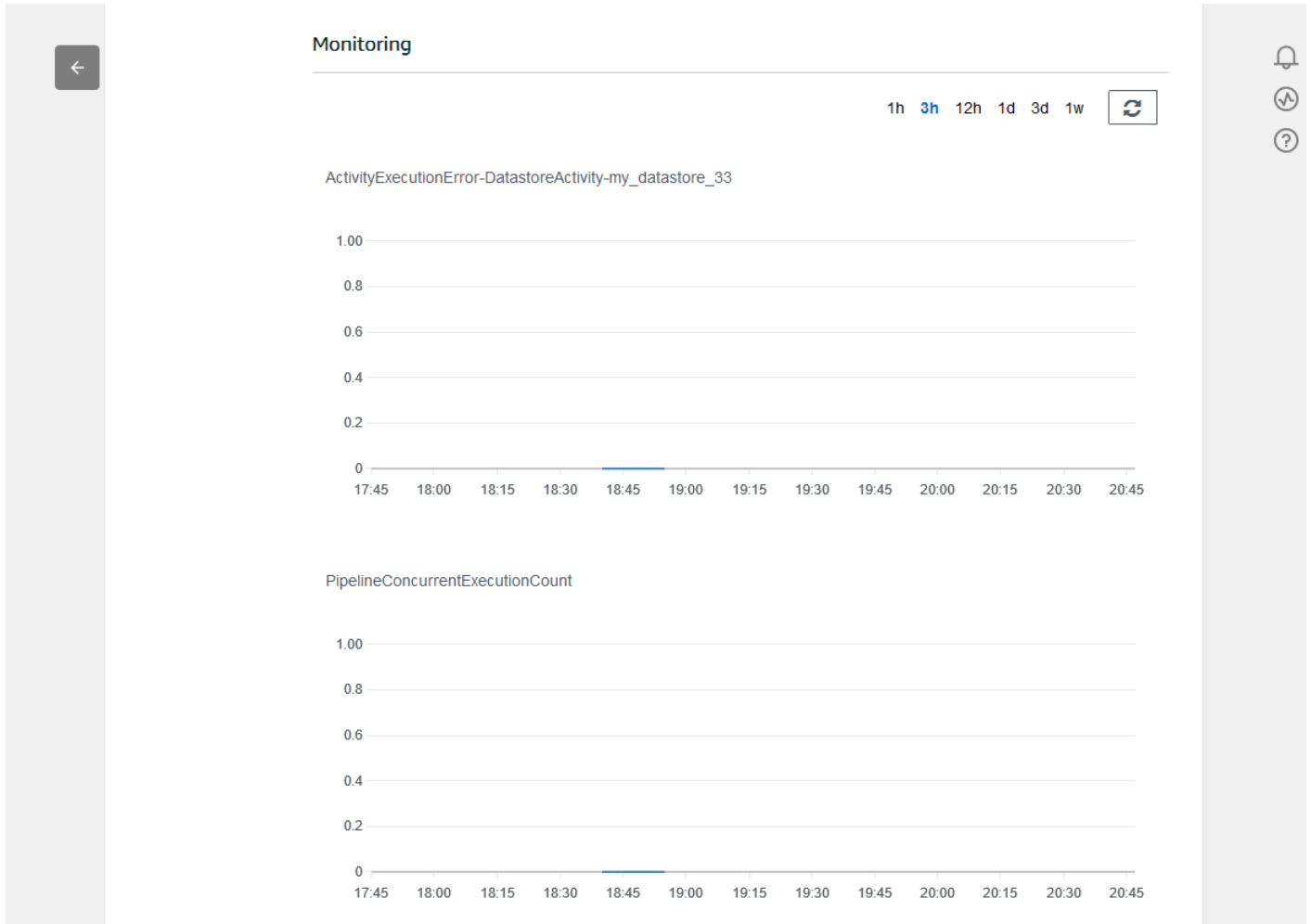
パイプラインのアクティビティの実行を確認する同様のモニタリング機能があります。パイプラインの詳細ページでアクティビティ実行エラーをモニタリングできます。アクティビティをパイプラインの一部として指定していない場合、実行エラー 0 が表示されます。

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 では、データをクエリして分析関連の質問に答えることができます。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 データの調査

AWS IoT Analytics 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。

このページのトピック

- [Amazon S3](#)
- [AWS IoT Events](#)
- [Amazon 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。

これを実行するには、[CreateDataset](#) を使用してデータセットを作成し、フィールド `contentDeliveryRules :: destination :: iotEventsDestinationConfiguration :: inputName` で AWS IoT Events 入力を指定します。"iotevents:BatchPutMessage" を実行するために AWS IoT Analytics アクセス許可を付与するロールの `roleArn` も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、AWS IoT Analytics により、データセットコンテンツの各入力がメッセージとして、指定した AWS IoT Events 入力に送信されます。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"
...
```

AWS IoT Analytics により次のようなフィールドを含むメッセージが送信されます。

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

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

そして、関心のあるフィールドを認識した AWS IoT Events 入力を作成したり (what、who、dt のうちの1つまたは複数)、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のトリガーや内部変数の設定を行う AWS IoT Events 検出器モデルを作成したりします。

## Amazon QuickSight

AWS IoT Analytics は [Amazon QuickSight](#) との直接統合を実現します。Amazon QuickSight は、可視化の構築、アドホック分析の実行、データからの迅速なビジネス上の洞察の取得に使用できる高速なビジネス分析サービスです。Amazon QuickSight は、組織が数十万人のユーザーにスケールするのを可能にし、堅牢なインメモリエンジン (SPICE) を使用することで応答性の高いパフォーマンスを実現します。Amazon QuickSight は、[これらのリージョン](#) で利用可能です。

## Jupyter Notebook

AWS IoT Analytics データセットは、高度な分析とデータ検索を実行するために、Jupyter Notebook によって直接消費されることもあります。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。さらに Amazon がホストするノートブックソリューションである SageMaker と統合することもできます。

## 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	retentionPeriod:	retentionPeriod:
	[指定されていません]	無制限 = TRUE、numberOfDays = 未設定	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

AWS IoT Analytics に送信するメッセージペイロードのフィールド名 (データ) には以下の条件があります。

- 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、産業機器のデータの大規模な収集、整理、分析を可能にす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 アセットモデルでは、消費すべき産業機器データ、および、データを複雑なメトリクスに処理する方法を定義します。AWS クラウドでデータの収集と処理が行われるようにアセットモデルを設定することができます。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 AWS IoT SiteWise と統合されるので、AWS IoT SiteWise データ上で SQL クエリの実行とスケジューリングを行うことができます。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[ストレージの設定](#)の手順に従ってデータストアを作成します。次に、[AWS IoT SiteWise データでのデータセットの作成 \(コンソール\)](#) または [AWS IoT SiteWise データでのデータセットの作成 \(AWS CLI\)](#) の手順に従って、AWS IoT Analytics データセットを作成し、産業データ上で SQL クエリを実行します。

## トピック

- [AWS IoT SiteWise データでの AWS IoT Analytics データセットの作成](#)
- [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 データでの AWS IoT Analytics データセットの作成

AWS IoT Analytics データセットには、データストアでのデータのクエリに使用する SQL のステートメント式、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。

### 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、AWS IoT Analytics では、データストアのデータに SQL クエリを適用することでデータセットを作成します。

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 データでのデータセットの作成 \(コンソール\)](#)
- [AWS IoT SiteWise データでのデータセットの作成 \(AWS CLI\)](#)

## AWS IoT SiteWise データ でのデータセットの作成 (コンソール)

次の手順を使用して、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方法

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. [Specify dataset details](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. [Data store source] (データストアのソース) 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) [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
```

AWS IoT Analytics でサポートされている SQL 機能の詳細については、「[AWS IoT Analytics の SQL 式](#)」を参照してください。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 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、SQL クエリを合理的なサイズに制限すべきです。

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行うように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

6 時間ごとに AWS IoT SiteWise のデータが AWS IoT Analytics に取り込まれます。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 により、このデータセットコンテンツのバージョンが作成され、分析結果が指定期間保存されます。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. [Datasets] (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 データでのデータセットの作成 (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 (AWS CLI) を使用しています。AWS CLI の詳細については、[AWS Command Line Interface ユーザーガイド](#) を参照してください。AWS IoT Analytics に使用できる CLI コマンドの詳細については、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方法

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"
      }
    }
  ]
}
```

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

- [AWS IoT Analytics でのデータセットコンテンツへのアクセス \(コンソール\)](#)
- [AWS IoT Analytics でのデータセットコンテンツへのアクセス \(AWS CLI\)](#)

### AWS IoT Analytics でのデータセットコンテンツへのアクセス (コンソール)

データセットにデータが含まれる場合、AWS IoT Analytics コンソールでの 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。

#### AWS IoT Analytics データセットの結果にアクセスする方法

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 でのデータセットコンテンツへのアクセス (AWS CLI)

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 (AWS CLI) を使用しています。AWS CLI の詳細については、[AWS Command Line Interface ユーザーガイド](#) を参照してください。AWS IoT Analytics に使用できる CLI コマンドの詳細については、AWS Command Line Interface リファレンスの [iotanalytics](#) を参照してください。

#### AWS IoT Analytics データセット結果にアクセスする方法 (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、風力発電所用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 と 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。
- AWS Management Consoleにアクセスするための、Windows、macOS、Linux、または Unix を実行してい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の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、「[ユーザーガイド](#)」の [AWS IoT SiteWise 「デモの作成](#)」の手順に従います。
- AWS IoT SiteWise デモ風力発電所の機器データは、管理する既存のデータストアに格納されます。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、「[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。

**i 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、未加工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の SQL クエリを実行するには

1. [AWS IoT SiteWise データでのデータセットの作成 \(コンソール\)](#) または [この手順に従って、AWS IoT SiteWise データの AWS IoT Analytics データセット AWS IoT SiteWise データでのデータセットの作成 \(AWS CLI\)](#) 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. データセットの概要ページで、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。

または、次の update-dataset コマンドを実行して、で SQL クエリを変更できます AWS CLI。

```
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クエリを実行してアセットに関するインサイト AWS IoT Analytics 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 クエリを使用してデータを調べるには

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. を使用してasset\_metadataテーブルSELECT DISTINCTをクエリし、AWS IoT SiteWise アセットの(一意の)名前を一覧表示します。

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

- 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'
```

- では 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 デモ風力発電施設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) の数値を持つ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 (デモタービンアセット 2) の特定のプロパティ (風速) の平均値を検索するには、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削除するにはdelete-datastore、次の例のように CLI コマンドを実行します。

```
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

- データセットを削除するにはdelete-dataset、次の例のように CLI コマンドを実行します。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、チャン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。

## チャン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け取っ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。

AWS IoT Analytics 許可を付与するためのポリシーを追加して Lambda 関数を呼び出す必要があります。たとえば、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、特定の 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": "iotanalytica", "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 アクティビティでは、AWS IoT デバイスレジストリのデータをメッセージのペイロードに追加できます。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"
  }
}
```

出カ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-dddeef-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 デバイスシャドウサービス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、次の例の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。

+	加算
-	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 .

引数の型	結果
Null	Undefined .
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:  $\text{acos}(0) = 1.5707963267948966$

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:  $\text{asin}(0) = 0.0$

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:  $\text{atan}(0) = 0.0$

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。
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:  $\text{atan}(1, 0) = 1.5707963267948966$

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。

引数の型	引数の型	結果
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。
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:

$\text{ceil}(1.2) = 2$

$\text{ceil}(11.2) = -1$

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:  $\text{cos}(0) = 1$

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。
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。

引数の型	結果
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 .
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」、 $e^{\text{引数}}$ 。
Decimal	Decimal 「倍精度で」、 $e^{\text{引数}}$ 。
String	Decimal 「倍精度で」、 $e^{\text{引数}}$ 。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。



引数の型	結果
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。
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:  $\text{mod}(8, 3) = 3$

左のオペランド	右のオペランド	出力
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。

左のオペランド	右のオペランド	出力
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`

$\text{Round}(1.7) = 2$

$\text{Round}(-1.1) = -1$

$\text{Round}(-1.5) = -2$

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:

$\text{sign}(-7) = -1$

$\text{sign}(0) = 0$ .

$\text{sign}(13) = 1$

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

引数の型	結果
	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」、引数の双曲線サイン。
Decimal	Decimal 「倍精度で」、引数の双曲線サイン。
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:  $\text{sqrt}(9) = 3.0$

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。

引数の型	結果
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。
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:

$\text{trunc}(2.3, 0) = 2$

$\text{trunc}(2.3123, 2) = 2.31$

`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}
```

# チャンネルメッセージの再処理

AWS IoT Analytics を使えばチャン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、チャンネルに対する許可、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

AWS IoT Analytics でパイプラインを通じてチャンネルメッセージを再処理する場合、以下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ンネルメッセージの再処理を開始します。

### ChannelMessages

再処理する 1 つ以上のチャン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

チャン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
```

型: 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: Timestamp

### 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: Timestamp

### pipelineName

再処理を開始するパイプラインの名前。

型: 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、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 AWS IoT Analytics コンソールで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 コンソールには以下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) - チャン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。

## チャン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ン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。

## チャン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、または AWS IoT Analytics コンソールの [Activities] (アクティビティ) で [Cancel reprocessing] (再処理のキャンセル) 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 では AWS IoT に関する高度なデータ分析が可能です。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。AWS IoT Analytics を使用すると、これらの周期ウィンドウのデータを重複することなく集計した分析を自動化し、最終精神の品質で KPI レポートを作成できます。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 を使用すれば、分析をオーケストレーションして、各実行ごとに KPI およびレポートを作成してから、今後の分析用にこのデータを保存できます。

### 異常検出の自動化

AWS IoT Analytics では、データストアに到着した新しいデータで 15 分ごとに手動で実行する必要があった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。AWS IoT Analytics に送信されたデータを使用して (使用可能なプロセスの測定を含む)、分析ワークフローを運用可能にすることでプロセスの成果を予測できます。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。

IoT Analytics 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ンネルを作成します (iotanalytics: [CreateChannel](#))。)
5. チャン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. メッセージデータを AWS IoT Analytics チャンネルに送信するためのアクセス許可を付与する IAM ロールを作成します (iam: [CreateRole](#)。)
7. チャン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、使用するチャンネルにルーティングされます。また、iotanalytics: [BatchPutMessage](#) を使用して、AWS SDK あるいは AWS CLI を使用できるデバイスから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。
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か。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 コンテナを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よう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 `GBvolumeSizeInGB`。
- 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。lotAnalytics 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せん (datasetContentVersionValue:datasetName)。

データセットを生成する SQL クエリとデルタウィンドウ、およびアプリケーションのコンテナを使用して、AWS IoT Analytics は、デルタウィンドウのデータで指定した間隔で実行される本番稼働用データセットをスケジュールし、必要な出力を行っ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。本番稼働用データセットアプリケーションを再開するとき、AWS IoT Analytics のデフォルト設定では、最後の実行以降に到着し、まだ分析済みではないすべてのデータを補足します。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": {
    "OutputUri": {
      "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。OutputUri 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。このアクセスリストは、アップロードされた出力に対して AWS IoT Analytics サービスによるコントロールを付与します。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)
```

## 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、コンテナ化されたノートブックではなく、SageMaker 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 コンソールを使用して新しい SageMaker インスタンスを作成する場合、SageMaker インスタンスを実行してノートブックをコンテナ化するため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するオプションが提供されます。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: <datasetName>, DatasetArn: <datasetARN>, 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: <datasetName>, DatasetArn: <datasetARN>, 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: <datasetName>, DatasetArn: <datasetARN>, 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: <datasetName>, DatasetArn: <datasetARN>}
```

## 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": "<datasetARN>",
  "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 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": "<datasetARN>",
}
```

## 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 は AWS IoT Analytics コンソールに表示される可能性があり、HTML を表示しているコンピュータ上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できます。このコンテナは、定義したデルタ時間ウィンドウ中に AWS IoT Analytics が受信した新しいデータのバッチを定期的に行います。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 インスタンスを作成した場合、コンテナ化の拡張機能のインストールは自動的に行われ、[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 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、SageMaker 拡張機能を変更して Amazon EC2 にコンテナイメージをアップロードできるようにします。こうして、コンテナ化の拡張機能をインストールできます。

## AWS IoT Analytics コンソールを介して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化する

新しい SageMaker インスタンスの作成は、次の手順ではなく、AWS IoT Analytics コンソールを使用して行うことが推奨されます。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で SageMaker インスタンスを再起動する場合、IAM ロールおよびポリシーを再度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. Amazon ECS へのアクセス権をノートブックに付与するには、Amazon SageMaker ページで SageMaker インスタンスを選択します。

Amazon SageMaker

Dashboard

▼ Notebook

    Notebook instances

    Lifecycle configurations

▼ Training

    Training jobs

Amazon SageMaker > Notebook instances

Notebook instances

Open Start Update settings Actions

Search notebook instances

Name	Instance	Creation time
exampleNotebookInstance	ml.t2.medium	Jul 03, 2018 21:25 UTC

2. IAM role ARN IAM ロール ARNで SageMaker Execution Role SageMaker 拡張機能ロールを選択します。

Amazon SageMaker

Dashboard

▼ Notebook

    Notebook instances

    Lifecycle configurations

▼ Training

    Training jobs

    Hyperparameter tuning jobs

▼ Inference

    Models

    Endpoint configurations

    Endpoints

Amazon SageMaker > Notebook instances > exampleNotebookInstance

exampleNotebookInstance

Delete Stop Start Open

Notebook instance settings

Edit

Name	exampleNotebookInstance	Notebook instance type	ml.t2.medium
ARN	arn:aws:sagemaker:us-east-1:[redacted]:notebook-instance/examplenotebookinstance	Storage	5GB EBS
Lifecycle configuration	—	Encryption key	
Status	Pending	IAM role ARN	arn:aws:iam::[redacted]:role/service-role/AmazonSageMaker-ExecutionRole-20180620T141485

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。

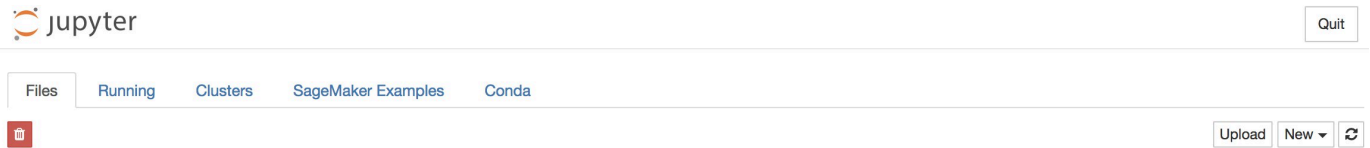
Permissions Trust relationships Access Advisor Revoke sessions

Attach policy

Attached policies: 7

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 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 インスタンス上でノートブックが実行するすべてのプロセスを終了することに注意してください。



```
sh-4.2$ █
```

```
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。

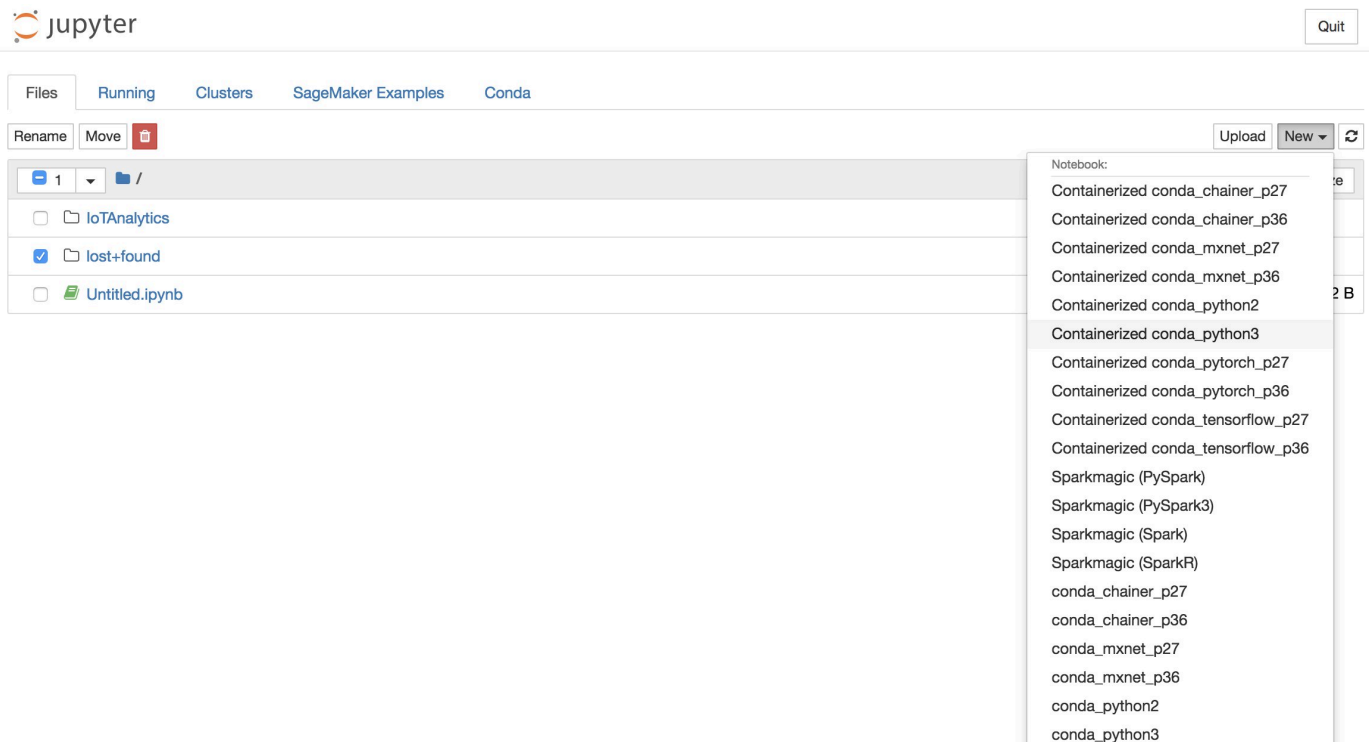
## ノートブックのコンテナ化拡張機能を更新する

2018 年 8 月 23 日以降に AWS IoT Analytics コンソールで SageMaker インスタンスを作成した場合、コンテナ化の拡張機能は自動的にインストールされます。SageMaker コンソールからインスタンスを再起動して拡張機能を更新できます。手動で拡張機能をインストールした場合には、「Enable Containerization Of Notebook Instances Not Created Via IoT Analytics Console AWS IoT Analytics コンソールを使用しないで作成されたノートブックインスタンスのコンテナ化を有効にする)」にリストされるターミナルコマンドを再度実行して更新することができます。

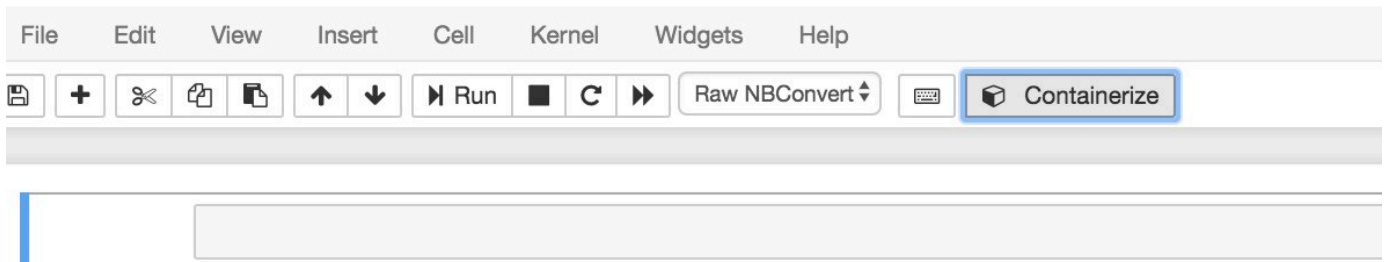
## コンテナ化イメージを作成する

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。

**1. Name**

2. Input Variables

3. Select AWS ECR Repository

4. Review

5. Monitor Progress

**Container Name \***

**Container Description**

**Next**

**Exit**

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。

1. Name

**2. Input Variables**

3. Select AWS ECR Repository

4. Review

5. Monitor Progress

Name	Type	Description	
<input type="text" value="ounces"/>	<input type="text" value="Double"/>	<input type="text"/>	<input type="button" value="X"/>
<input type="text" value="brand"/>	<input type="text" value="String"/>	<input type="text"/>	<input type="button" value="X"/>

Showing 1 to 2 of 2 variables

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。

1. Name    2. Input Variables    **3. Select AWS ECR Repository**    4. Review    5. Monitor Progress

Please upload different notebooks to different repositories.

Repository Name  Create    Search:

Name
my-repo
my-repo2
my-repo3

Showing 1 to 3 of 3 repositories    Previous  Next

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。

1. Name    2. Input Variables    3. Select AWS ECR Repository    **4. Review**    5. Monitor Progress

**Container Name:** Beer-Tastiness-Calculator  
**Container Description:**  
**Upload To:** my-repo

Variable Name	Type	Description
ounces	Double	
brand	String	

Showing 1 to 2 of 2 variables    Previous    **1**    Next

Previous    **Containerize**

Exit

7. 次のページに進行状況が表示されます。

1. Name    2. Input Variables    3. Select AWS ECR Repository    4. Review    **5. Monitor Progress**

The containerization process typically completes within 30 minutes.

Creating Image...

Exit

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。

### Containerize Notebook ×



1. Name

2. Input Variables

3. Select AWS ECR Repository

4. Review

5. Monitor Progress

Creating Image... Uploading Image... 

You can now use this notebook for scheduled analysis of your Data Sets.

[Go To Data Sets](#)[Exit](#)

## 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 は AWS IoT Analytics コンソールに表示される可能性があり、HTML を表示しているコンピュータ上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て、AWS IoT Analytics サービスを使用してこれを実行できます。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 へのアクセス権がある 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 – AWS IoT Analytics から JSON を解析し、オクターブ スクリプトを実行して、Amazon S3 にアーティファクトを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ます。

```
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
```

- 「octave-sample-data-*your-aws-account-id*」という Amazon S3 バケットを作成します。
- 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

- 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* はお客様の AWS アカウント ID に、*aws-region* はお客様の AWS リソースの AWS リージョンに置き換えてください。

#### 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「role1.json」ファイルを使用して、SageMaker および AWS IoT Analytics へのアクセス許可を付与する新しい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ます。

```
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ます。

```
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 データの可視化

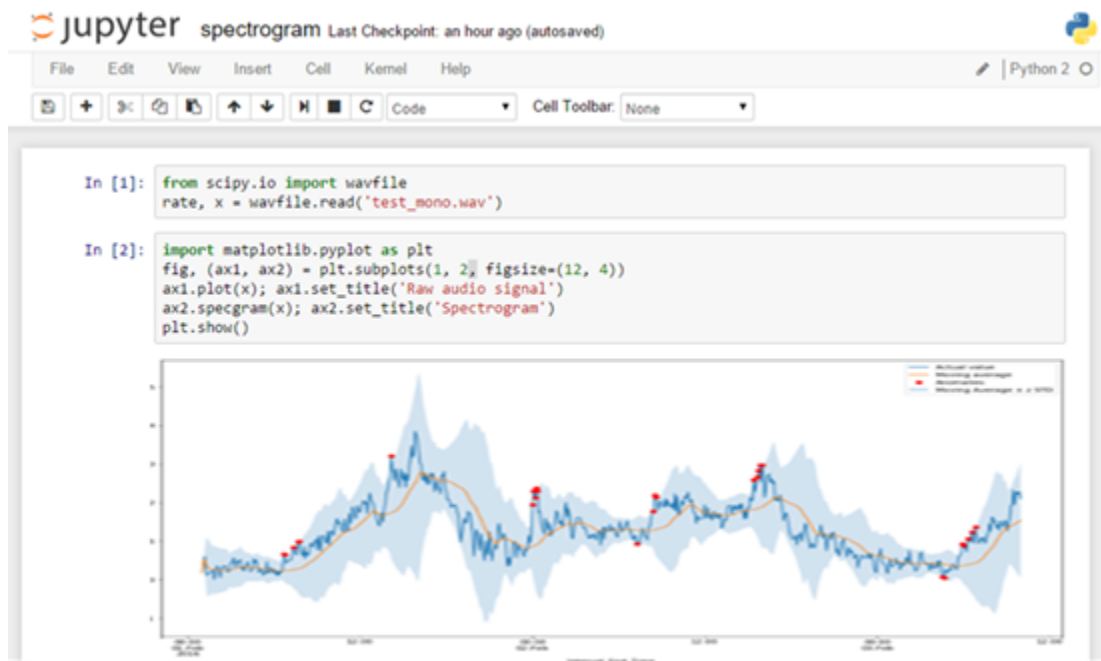
AWS IoT Analytics データの可視化には、AWS IoT Analytics コンソールまたは Amazon QuickSight を使用できます。

## トピック

- [コンソールを使用した AWS IoT Analytics データの可視化](#)
- [Amazon QuickSight を使用した AWS IoT Analytics データの可視化](#)

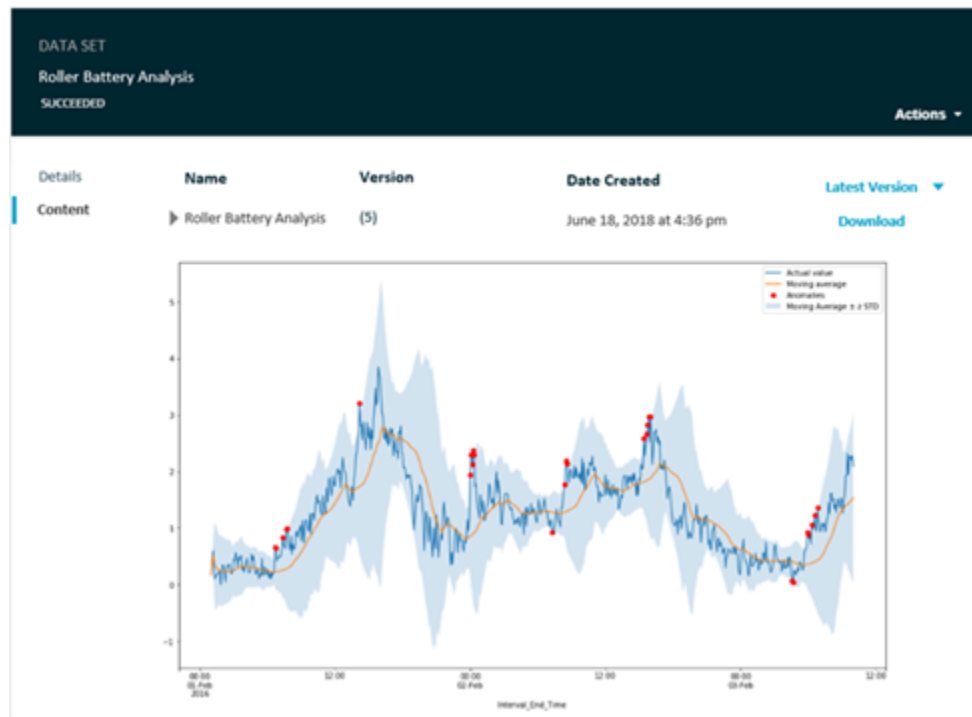
## コンソールを使用した AWS IoT Analytics データの可視化

AWS IoT Analytics では、コンテナデータセット (ファイル output.html にある) の HTML 出力を、[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。

## Amazon QuickSight を使用した AWS IoT Analytics データの可視化

AWS IoT Analytics は [Amazon QuickSight](#) との直接統合を実現します。Amazon QuickSight は、可視化の構築、アドホック分析の実行、データからの迅速なビジネス上の洞察の取得に使用できる高速なビジネス分析サービスです。Amazon QuickSight は、組織が数十万人のユーザーにスケールするのを可能にし、堅牢なインメモリエンジン (SPICE) を使用することで応答性の高いパフォーマンスを実現します。Amazon QuickSight コンソールで AWS IoT Analytics データセットを選択すれば、ダッシュボードと可視化の作成を開始できます。Amazon QuickSight は、[これらのリージョン](#)で利用可能です。

Amazon QuickSight の可視化を開始するには Amazon QuickSight アカウントを作成する必要があります。アカウントのセットアップ時に、Amazon QuickSight から AWS IoT Analytics データにアクセスできるように設定してください。すでにアカウントをお持ちの方は、[Admin] (管理者)、[Manage QuickSight] (QuickSight を管理する)、[Security & permission] (セキュリティと許可) を選択することで、Amazon QuickSight へのアクセス権を AWS IoT Analytics データに付与してください。[QuickSight access to AWS services] (AWS サービスへの QuickSight アクセス) で、[Add or remove] (追加または削除) を選択し、次に AWS IoT Analytics の横のチェックボックスを選択して [Update] (更新) を選択します。

QuickSight

Account name: [Redacted]  
Edition: Enterprise

Manage users  
Your subscriptions  
SPICE capacity  
Account settings  
**Security & permissions**  
Manage VPC connections  
Domains and Embedding

### Security & permissions

QuickSight can control access to AWS resources for the entire account in addition to individual users and groups

#### QuickSight access to AWS services

Amazon Redshift Amazon RDS IAM Amazon S3 AWS IoT Analytics

By configuring access to AWS services, QuickSight can access the data in those services. Access by users and groups can be controlled through the options below.

[Add or remove](#)

#### Default resource access

① Users and groups have access to all connected resources.

QuickSight can allow or deny access to all users and groups by default, when an individual access control is not in effect for a particular user or group

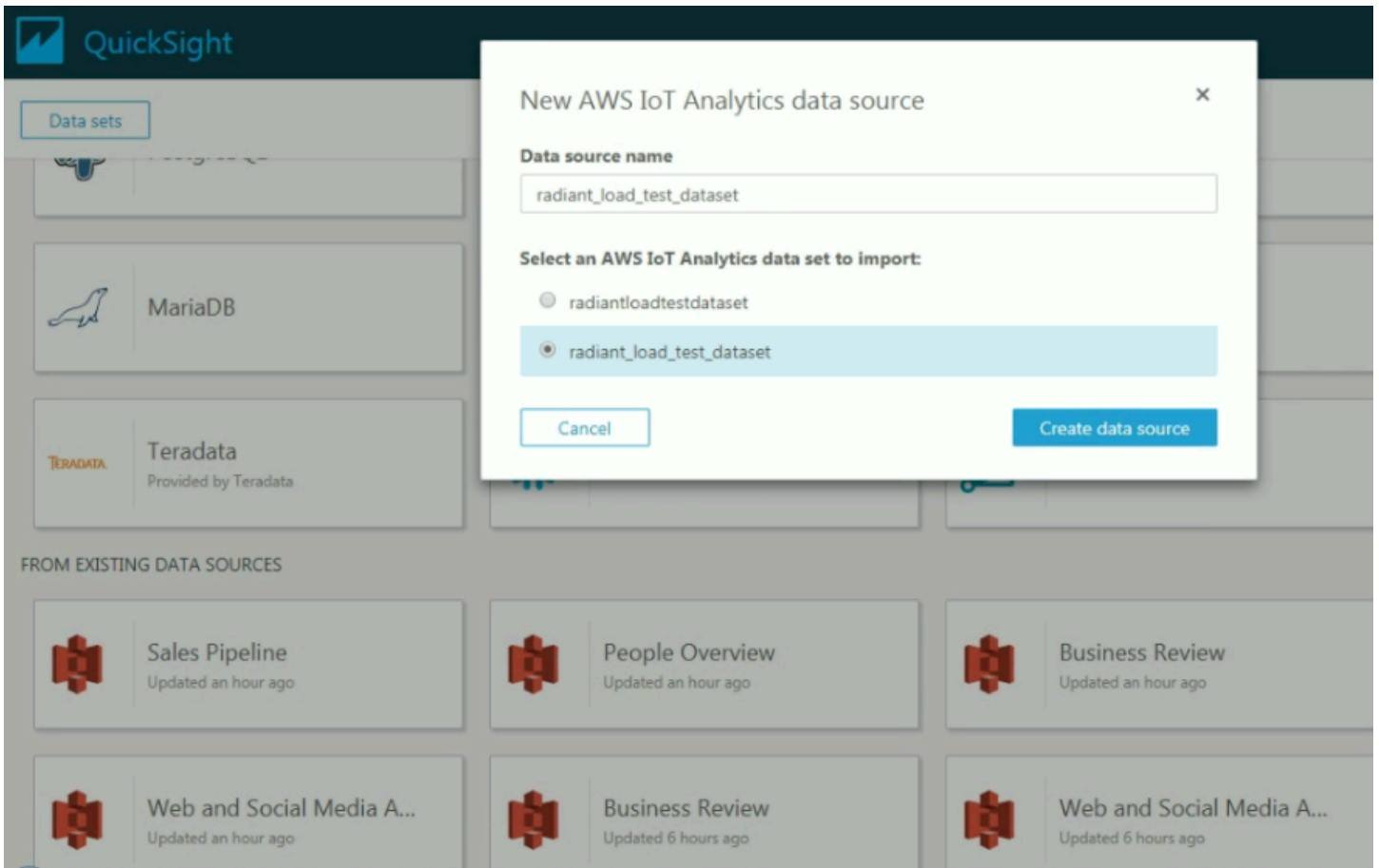
[Change](#)

#### Resource access for individual users and groups

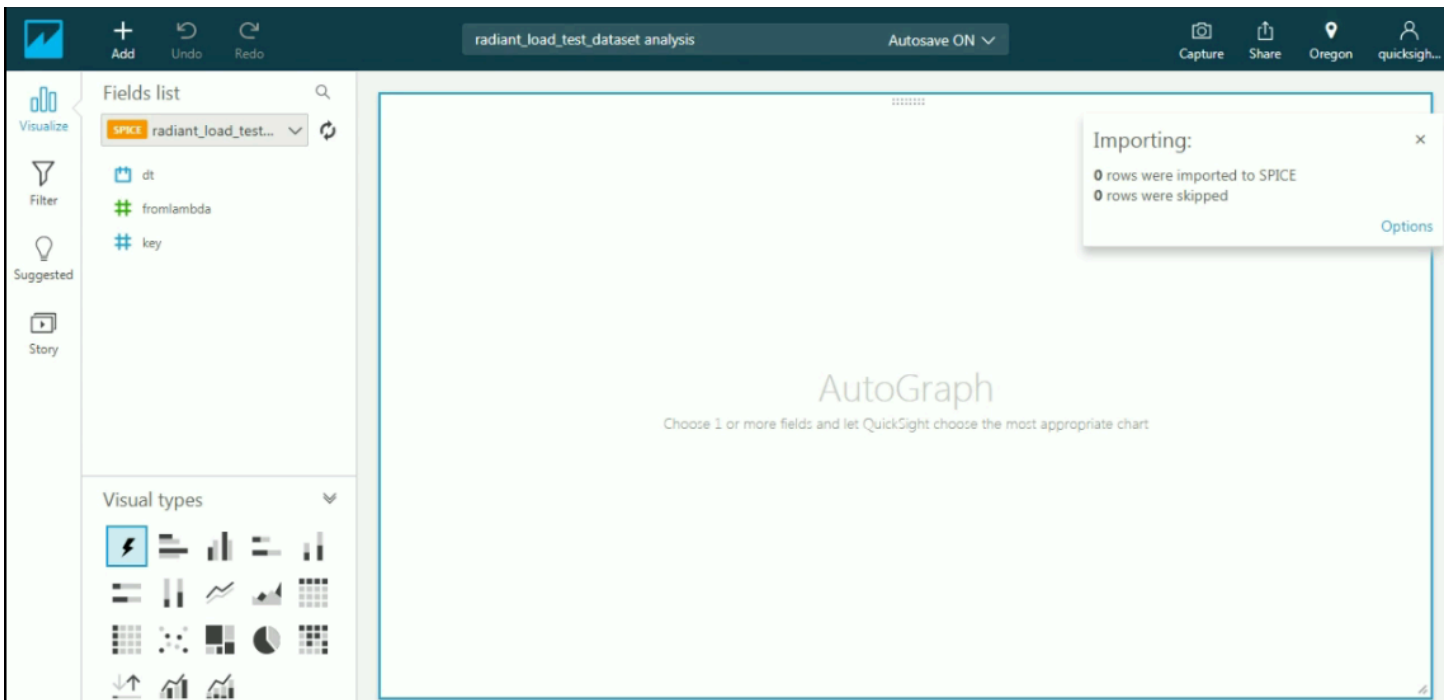
Resource access is controlled by assigning IAM policies.

[IAM policy assignments](#)

アカウントのセットアップが完了したら、管理者用の Amazon QuickSight コンソールページで [New Analysis] (新しい分析) と [New data set] (新しいデータセット) を選択し、ソースとして AWS IoT Analytics 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。



データソースが作成されたら、Amazon QuickSight で可視化を作成できます。



Amazon QuickSight ダッシュボードとデータセットの詳細については、「[Amazon QuickSight のドキュメント](#)」を参照してください。

# AWS IoT Analytics リソースのタグ付け

チャンネル、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

- [ベーシックタグ](#)
- [IAM ポリシーでのタグの使用](#)
- [タグの制限](#)

## ベーシックタグ

タグを使用すると、AWS IoT Analytics リソースをさまざまな方法で分類できます。例えば、目的、所有者、環境などです。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、チャンネルに一連のタグを定義して、各チャン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。チャンネル、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。

利便度を向上させる目的から、AWS Billing and Cost Management コンソールのタグエディタを使用すると、集中管理の統合された方法によるタグの作成と管理ができます。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。

また、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、チャンネル、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": "*"
    }
  ]
}
```

次の例のように、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に `aws: prefix` は使用しないでください。これは 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、他のサービスでも許可される文字に制限が適用されることがあるのでご注意ください。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。

# AWS IoT Analytics の SQL 式

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。AWS IoT Analytics では、Amazon Athena と同じ SQL クエリ、関数、および演算子を使用しています。

AWS IoT Analytics では、ANSI 標準 SQL 構文のサブセットがサポートされています。

```
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 と Amazon Athena では、以下の機能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

- [AWS IoT Analytics でサポートされている SQL 機能](#)

- [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](#)

## AWS IoT Analytics でサポートされている SQL 機能

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。AWS IoT Analytics で実行するクエリは [Presto 0.217](#) に基づきます。

### サポートされているデータ型

AWS IoT Analytics と Amazon Athena ではこれらのデータ型がサポートされています。

- 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 では一部のデータ型がサポートされていません。

## サポートされている関数

Amazon Athena と AWS IoT Analytics SQL 機能は [Presto 0.217](#) に基づきます。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 と Amazon Athena では、ユーザー定義関数 (UDF または UDAF) とストアプロシージャはサポートされていません。

## 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 '0''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からメリットを得られます。

セキュリティは、AWS とユーザーの間で共有される責任です。[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責任 AWS を担います。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。に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、「[コンプライアンスAWS プログラムによる対象範囲内の のサービス](#)」を参照してください。
- クラウドのセキュリティ - 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、 の使用時に責任共有モデルを適用する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、セキュリティおよびコンプライアンスの目的を達成するために AWS IoT Analytics 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 のサービスの使用方法についても説明します。

## AWS Identity and Access Management の 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 リソースの使用を承認する (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 のサービスです。

### 対象者

AWS Identity and Access Management ( IAM) の使用方法は、 で行う作業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与えられます。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。次に、サービスユーザーのアクセス許可を変更するリクエストをIAM管理者に送信する必要があります。このページの情報を確認して、の基本概念を理解してくださいIAM。会社で を と使用する方法の詳細については、IAM AWS IoT Analytics 「」を参照してください [と の AWS IoT Analytics 連携方法 IAM](#)。

IAM 管理者 - IAM管理者は、へのアクセスを管理するポリシーの作成方法の詳細について確認する場合があります AWS IoT Analytics。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IAM、「」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 の認証情報は、フェデレーテッド ID の例です。フェデレーテッド ID としてサインインすると、管理者は以前に IAMロールを使用して ID フェデレーションをセットアップしていました。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となります。

ユーザーのタイプに応じて、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、「ユーザーガイド」の「[にサインインする方法 AWS アカウント](#)AWS サインイン」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に暗号で署名します。AWS ツールを使用しない場合は、リクエストに自分で署名する必要があります。推奨される方法を使用し

てリクエストを自分で署名する方法の詳細については、「IAMユーザーガイド」の[AWS API「リクエストの署名」](#)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。例えば、AWS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)を使用することをお勧めします。詳細については、「ユーザーガイド」の[「多要素認証」](#)および[「ユーザーガイド」の「での多要素認証\(MFA\) AWS IAMの使用」](#)を参照してください。AWS IAM Identity Center

## 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[「ルートユーザーの認証情報を必要とするタスク」](#)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アイデンティティです。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持つIAMユーザーを作成するのではなく、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ユーザーとの長期的な認証情報を必要とする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、「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でアクセスキーを定期的にローテーションするIAM」](#)を参照してください。

[IAM グループ](#)は、IAMユーザーのコレクション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、という名前のグループを作成しIAMAdmins、そのグループにIAMリソースを管理するアクセス許可を付与できます。

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、「[ユーザーガイド](#)」のIAM「[\(ロールの代わりに\) ユーザーを作成する場合IAM](#)」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 内のアイデンティティです。これは IAM ユーザーと似ていますが、特定のユーザーに関連付けられていません。IAM ロール を切り替える AWS Management Console ことで、[で ロール](#)を一時的に引き受けることができます。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 を使用します URL。ロールの使用の詳細については、「[ユーザーガイド](#)」のIAM「[ロールの使用IAM](#)」を参照してください。

IAM 一時的な認証情報を持つ ロールは、以下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の詳細については、「[ユーザーガイド](#)」の「[サードパーティー ID プロバイダーのロールの作成IAM](#)」を参照してください。IAM Identity Center を使用する場合は、アクセス許可セットを設定します。ID が認証後にアクセスできる内容を制御するために、IAM Identity Center はアクセス許可セットを のロールに関連付けますIAM。アクセス許可セットの詳細については、「AWS IAM Identity Center ユーザーガイド」の「[アクセス許可セット](#)」を参照してください。
- 一時的なIAMユーザーアクセス許可 – IAM ユーザーまたはロールは、IAMロールを引き受けて、特定のタスクに対して異なるアクセス許可を一時的に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、別のアカウントのユーザー (信頼されたプリンシパル) がアカウントのリソースにアクセスすることを許可できます。クロスアカウントアクセスを許可する主な方法は、ロールを使用することです。ただし、一部の では AWS サービス、(ロールをプロキシとして使用する代わりに) ポリシーをリソースに直接アタッチできます。クロスアカウントアクセスのロールとリソースベースのポリシーの違いについては、「[ユーザーガイド](#)」の「[でのクロスアカウントリソースアクセスIAMIAM](#)」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 サービス を使用します AWS サービス。例えば、サービスで呼び出しを行うと、そのサービスが Amazon でアプリケーションを実行 EC2したり、Amazon S3 にオブジェクトを保存したりするのが一般的です。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 で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 サービス、ダウンストリームサービス AWS サービスへのリクエストのリクエスト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 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権限が必要です。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 – サービスロールは、ユーザーに代わってアクションを実行するためにサービスが引き受ける [IAMロール](#)です。IAM 管理者は、内からサービスロールを作成、変更、削除できますIAM。詳細については、「[ユーザーガイド](#)」の「[にアクセス許可を委任するロールの作成 AWS サービスIAM](#)」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 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、サービスにリンクされた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 で実行されているアプリケーション EC2 – 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インスタンス内にアクセスキーを保存するよりも望ましいです。AWS ロールをEC2インスタンスに割り当て、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 ロールが含まれており、EC2インスタンスで実行されているプログラムが一時的な認証情報を取得できるようにします。詳細については、「[ユーザーガイド](#)」の「[IAMロールを使用して Amazon EC2インスタンスで実行されているアプリケーションにアクセス許可を付与するIAM](#)」を参照してください。

IAM ロールとIAMユーザーのどちらを使用するかについては、「[ユーザーガイド](#)」の「[\(ユーザーではなく\) IAMロールを作成する場合IAM](#)」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、AWS ID 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、「[ユーザーガイド](#)」の[JSON「ポリシーの概要IAM」](#)を参照してください。

管理者はポリシーを使用して AWS JSON、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。必要なリソースに対してアクションを実行するアクセス許可をユーザーに付与するために、IAM管理者はIAMポリシーを作成できます。その後、管理者はIAMポリシーをロールに追加し、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に使用する方法に関係なく、アクションのアクセス許可を定義します。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 からロール情報を取得できますAPI。

## 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のポリシーは、IAMユーザー、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アクセス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のポリシーを作成する方法については、「[ユーザーガイド](#)」の[IAM「ポリシーの作成IAM」](#)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。管理ポリシーとインラインポリシーのどちらかを選択する方法については、「[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選択](#)」を参照してください。IAM

## その他のポリシータイプ

AWS は、一般的ではない追加のポリシータイプをサポートします。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のポリシーがIAMエンティティ (IAMユーザーまたはロール) に付与できるアクセス許可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) に対する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、AWS アカウント ビジネスが所有する複数の 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 (SCPs) をアカウントの一部またはすべてに適用できます。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SCPを制限します AWS アカウントのルートユーザー。Organizations と の詳細についてはSCPs、「AWS Organizations ユーザーガイド」の[「サービスコントロールポリシー」](#)を参照してください。
- 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があります。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、「ユーザーガイド」の[「セッションポリシーIAM」](#)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。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、「ユーザーガイド」の[「ポリシー評価ロジックIAM」](#)を参照してください。

## との AWS IoT Analytics 連携方法 IAM

IAM を使用してへのアクセスを管理する前に AWS IoT Analytics、で利用できる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のサービスがと連携する方法の概要を把握するにはIAM、「IAMユーザーガイド」の[AWS 「と連携するのサービスIAM」](#)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 タグに基づいた認可](#)
- [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、およびアクションを許可または拒否す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、「ユーザーガイド」の「[IAMJSONポリシー要素リファレンスIAM](#)」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 要素は、ポリシーによって許可または拒否される特定のアクションを記述します。ポリシーアクションの名前は通常、関連する 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。例えば、`iotanalytics:APIオペレーション`で AWS IoT Analytics `CreateChannel` チャンネルを作成する AWS IoT Analytics アクセス許可を付与するには、ポリシーに `iotanalytics:BatchPutMessage` 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. AWS IoT Analytics defines のいずれかを含める必要があります。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、「ユーザーガイド」の「[で定義されるアクション AWS IoT AnalyticsIAM](#)」を参照してください。

## リソース

Resource エlement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lementを含める必要があります。ARN またはワイルドカード (\*) を使用してリソースを指定し、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します。

AWS IoT Analytics データセットリソースには、次の がありますARN。

```
arn:${Partition}:iotanalytics:${Region}:${Account}:dataset/${DatasetName}
```

の形式の詳細についてはARNs、[「Amazon リソースネーム \(ARNs \)」](#) および AWS [「サービス名前空間」](#) を参照してください。

例えば、ステートメントでFoobarデータセットを指定するには、次の を使用しますARN。

```
"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 はチャンネルとデータセットを参照するため、IAM ユーザーはチャン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。1 つのステートメントで複数のリソースを指定するには、 をカンマARNsで区切ります。

```
"Resource": [
  "resource1",
  "resource2"
]
```

AWS IoT Analytics リソースタイプとその のリストを確認するにはARNs、[「IAMユーザーガイド」](#) の [「定義されるリソース AWS IoT Analytics」](#) を参照してください。各リソースARNの を指定できるアクションについては、[「定義されるアクション AWS IoT Analytics」](#) を参照してください。



## 条件キー

Condition エlement 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lement 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lement を指定する場合、または 1つの Condition エlement 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进行评估します。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进行评估します。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「ポリシー要素: 変数とタグ」](#)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、「ユーザーガイド」の[AWS「グローバル条件コンテキストキーIAM」](#)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 タグに基づいた認可

AWS IoT Analytics リソースにタグをアタッチしたり、への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、特定のアクセス許可 AWS アカウント を持つ 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

一時的な認証情報を使用して、フェデレーションでサインインしたり、IAMロールを引き受けたり、クロスアカウントロールを引き受けたりすることができます。一時的なセキュリティ認証情報を取得するには、[AssumeRole](#)や などの (AWS STS) APIオペレーションを呼び出し AWS Security Token Service [まずGetFederationToken](#)。

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。

### サービスリンクロール

[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。サービスにリンクされたロールはIAMアカウントに表示され、サービスによって所有されます。IAM 管理者は、サービスにリンクされた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。

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。

### サービスロール

この機能により、ユーザーに代わってサービスが[サービスロール](#)を引き受けることが許可されます。このロールにより、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。サービスロールは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管理者はこのロールのアクセス許可を変更できます。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

混乱した代理問題とは、アクションを実行する許可を持たないエンティティが、より高い特権を持つエンティティにそのアクションの実行を強制できるというセキュリティ問題です。AWS では、サービス間でのなりすましが、混乱した代理問題を生じさせることがあります。サービス間でのなりすまは、1つのサービス 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。これらのキーは、AWS IoT Analytics がほかのサービスに付与するそのリソースへのアクセス許可を制限します。両方のグローバル条件コンテキストキーを使用しており、それらが同じポリシーステートメントで使用されるときは、aws:SourceAccount 値と、aws:SourceArn 値のアカウントが同じアカウント ID を使用する必要があります。

「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*:\* です。

## トピック

- [Amazon S3 バケットの防止](#)
- [Amazon CloudWatch Logs での防止](#)
- [顧客が管理する 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 データストアに顧客管理の Amazon S3 ストレージを使用している場合、データを保存する Amazon S3 バケットは混乱した代理問題にさらされてい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* にクエリを実行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、Li は自分のアカウントからクエリを実行したにもかかわらず、AWS IoT Analytics サービスプリンシパルは 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-以下のバケットポリシーの例では、ニッキのアカウント **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 の `aws:SourceAccount`
- AWS IoT Analytics 内の `sts:AssumeRole` リクエストに関連付けられているカスタマーリソース

次の例では、`123456789012` を自分の 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。

## 顧客が管理する 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 権限を付与すると、そのリソース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ため、以下のリソースポリシーの例で AWS IoT Analytics に付与する権限を制限できます。

### 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クロスサービスでの混乱した代理の防止](#)

## AWS IoT Analytics チャンネルとデータストアの防止

IAM ロールを使用して、AWS IoT Analytics がユーザーに代わってアクセスできる AWS リソースを制御します。ロールが混乱した代理問題にさらされるのを防ぐため、ロールにアタッチする信頼ポリシーの `aws:SourceAccount` 要素で AWS アカウントを指定し、`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` を自分の 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 チャン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"
        }
      }
    }
  ]
}
```

チャン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クロスサービスでの混乱した代理の防止

AWS IoT Analytics がデータセットのクエリ結果を Amazon S3 または AWS IoT Events に配信することを前提とする IAM ロールは、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、ロールにアタッチする信頼ポリシーの `aws:SourceAccount` 要素で AWS アカウントを指定し、`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、またはを使用してタスクを実行することはできません。AWS API。IAM 管理者は、必要な特定のリソースに対して特定のAPIオペレーションを実行するアクセス許可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。

これらのポリシードキュメント例を使用してIAMアイデンティティベースのJSON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

- [ポリシーの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 入カへのアクセス](#)
- [タグに基づく AWS IoT Analytics チャンネルの表示](#)

## ポリシーの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。これらは、ユーザーのアカウントで誰かが 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、「[ユーザーガイド](#)」の「[AWS マネージドポリシーでアクセス許可の使用を開始するIAM](#)」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、「[ユーザーガイド](#)」の「[最小特権](#)を付与するIAM」を参照してください。
- 機密性の高いオペレーションMFAを有効にする - セキュリティを強化するには、機密性の高いリソースまたはAPIオペレーションにアクセスするために多要素認証 (MFA) の使用をユーザーに要求します。詳細については、「[ユーザーガイド](#)」の「[での多要素認証 \(MFA\) AWS の使用IAM](#)」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ぬ許容 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。また、指定した日付または時間範囲内のリクエストのみを許可する条件を記述したり、SSL または の使用を要求したりすることもできますMFA。詳細については、「[ユーザーガイド](#)」の[IAMJSON](#)「[ポリシー要素: 条件IAM](#)」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、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 AWS IoT Analytics 引き続き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、「[ユーザーガイド](#)」の「[ユーザーへのアクセス許可の追加IAM](#)」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 のみ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 AWS API。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を許可します。

## 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 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API。

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
        "iam:ListGroupsWithUser",
        "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、 のユーザーに、いずれかの AWS IoT Analytics チャンネル AWS アカウント へのアクセスを付与しますexampleChannel。また、ユーザーに対してチャンネルの追加、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 チャンネルの表示

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 `richard-roe` しようとする場合は AWS IoT Analytics channel、に のタグ `channel` を付ける必要があります `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、「ユーザーガイド」の [IAMJSON「ポリシー要素: 条件IAM」](#) 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

次の情報は、の使用時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ない AWS IoT Analytics](#)
- [iam:PassRole を実行する権限がない](#)
- [自分の 以外のユーザーに自分の AWS アカウント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ない AWS IoT Analytics

から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 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。

次の例のエラーは、という IAM ユーザーがコンソールを使用して marymajor でアクションを実行しようする場合に発生します AWS IoT Analytics。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。サインイン認証情報を提供した担当者が管理者です。

## 自分の以外のユーザーに自分の AWS アカウント 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では、これらのポリシーを使用して、ユーザーにリソースへのアクセスを許可できます。

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、[「がと連携する方法 AWS IoT Analytics IAM」](#)を参照してください。
- 所有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、AWS アカウント「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IAM」](#)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の に提供する方法については AWS アカウント、「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IAM」を参照してください。
- ID フェデレーションを通じ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[「外部認証されたユーザーへのアクセスの提供 \(ID フェデレーション\)」](#)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の違い」](#)を参照してください。

## AWS IoT Analyticsでのログ記録とモニタリング

AWS では、のモニタリングに使用できるツールを提供しています。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

以下の自動化されたモニタリングツールを使用して、AWS IoT を監視し、問題が発生したときにレポートできます。

- Amazon CloudWatch Logs - AWS CloudTrail またはその他のソースのログファイルのモニタリング、保存、アクセスを実行します。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 - アカウント間でのログファイルの共有、CloudWatch Logs への送信による CloudTrail ログファイルのリアルタイムモニタリング、ログ処理アプリケーションの Java への書き込み、CloudTrail からの提供後にログファイルが変更されていないことの確認を行います。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

AWS IoT のモニタリングでもう 1 つ重要な点は、CloudWatch のアラームの対象外の項目を手動でモニタリングすることです。AWS IoT、CloudWatch、その他の AWS サービスのコンソールのダッシュボードには、AWS 環境の状態が一目でわかるように表示されます。AWS IoT Analytics のログファイルを確認することもお勧めします。

- AWS IoT Analytics コンソールには以下が表示されます。
  - チャンネル
  - パイプライン
  - データストア
  - データセット
  - ノートブック
  - 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
- AWS リソースのすべての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 では、Amazon CloudWatch を使用したログ記録がサポートされています。 [PutLoggingOptionsAPI 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、PutLoggingOptions と AWS Identity and Access Management IAM を使用して、AWS IoT Analytics の Amazon CloudWatch のログギングを設定および有効にする方法について説明します。

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。AWS IAM の詳細については、[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 へのアクセス権限を理解しておく必要があります。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 [ロールの作成](#) を使用して、信頼関係ポリシー (信頼ポリシー) を含む新しい 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トキーが含まれています。[123456789012](#) を自分の AWS アカウント ID に、[aws-region](#) を AWS リソースの AWS リージョンに置き換えてください。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:*"
        }
    }
}
]
```

後で AWS IoT Analytics PutLoggingOptions コマンドを呼び出すときに、このロールの ARN を使用します。


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 にログインイベントを送信する AWS IoT Analytics 権限を与えるには、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、DescribeLoggingOptions 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 の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 の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 では 以下のメトリクスが Amazon CloudWatch リポジトリに投入されます。

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ンネルの名前。
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、AWS Lambda によりイベントが Amazon CloudWatch Events に自動的に発行されます。このイベントには、詳細なエラーメッセージと、未処理チャン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、未処理のチャンネルメッセージを再処理できます。詳細については、[チャン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ン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 リソースと関連ターゲットは、AWS IoT Analytics リソースを作成した AWS リージョン内になければなりません。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 アクティビティでのランタイムエラーに関して Amazon CloudWatch Events に送信された通知には以下の形式が使用されます。

```
{
  "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。queryAction (SQL クエリ) を適用して [データセットを作成](#) するときに QueryFilter に対して deltaTime オフセットを指定すれば遅延を許可できます。その後も、AWS IoT Analytics によりデルタ時間内に到着したデータは処理され、データセットコンテンツ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、データセット用に遅延データルールを指定することができます。

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

AWS IoT Analytics でデータセット用の遅延データルールを作成する場合は、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。timeoutInMinutes を使用すれば、AWS IoT Analytics において、全開の実行以降に生成された遅延データ通知のバッチアップが可能です。AWS IoT Analytics では CloudWatch Events に対して一度に通知バッチが 1 つ送信されます。

型: 整数

有効な範囲: 1 ~ 60

### **ruleName**

遅延データルールの名前。

型: 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 (保存)] を選択します。



UPDATE DATA SET

## Configure data selection filter

When creating a SQL data set, you can specify a `deltaTime` pre-filter to be applied to the message data to help limit the messages to those which have arrived since the last time the SQL data set content was created. [Learn more](#)

### Data selection window

### Offset

Specifies possible latency in the arrival of a message

### Timestamp expression

### Late data notification

Enable late data notification to receive CloudWatch events if late data is detected.

### Delta time

IoT Analytics will emit a notification if late data is received within the value below

 Minutes[Back](#)[Save](#)

## 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

AWS IoT Analytics 用の CloudWatch Events ルールを作成する前に、以下を行う必要があります。

- 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 ルールと関連ターゲットは、AWS リソースを作成した AWS IoT Analytics リージョン内になければなりません。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

以下の手順は、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ています。

#### 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。

## AWS IoT Analytics による AWS CloudTrail API コールのログ記録

AWS IoT Analytics は AWS CloudTrail と統合されています。このサービスは、ユーザーやロール、または AWS の AWS IoT Analytics のサービスによって実行されたアクションをレコードするサービスです。CloudTrail は、AWS IoT Analytics コンソールからの呼び出しや AWS IoT Analytics API へのコード呼び出しを含む、AWS IoT Analytics の API コールのサブセットをイベントとしてキャプチャします。トレイルを作成すると、AWS IoT Analytics のイベントを含む CloudTrail イベントの Amazon S3 バケットへの継続的な配信が可能になります。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、AWS IoT Analytics に対して行われた要求、要求が行われた IP アドレス、要求を行った人、要求が行われた日時、および追加の詳細を判別できます。

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 内の AWS CloudTrail 情報

CloudTrail は、アカウントを作成すると AWS アカウントで有効になります。AWS IoT Analytics でアクティビティが発生すると、そのアクティビティは Event history イベント履歴で AWS のその他のサービスのイベントと共に CloudTrail イベントに記録されます。最近のイベントは、AWS アカウントで表示、検索、ダウンロードできます。詳細については、[CloudTrail イベント履歴でのイベントの表示](#)をご参照ください。

AWS のイベントなど、AWS IoT Analytics アカウントのイベントの継続的なレコードについては、追跡を作成します。追跡により、CloudTrail はログファイルを Amazon S3 バケットに配信できます。デフォルトでは、コンソールで追跡を作成するときに、追跡がすべてのリージョンに適用されます。追跡は、AWSパーティション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 ログで収集したイベントデータをより詳細に分析し、それに基づく対応するためにその他の AWS のサービスを設定できます。詳細については、次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以下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、リクエストの生成者に関する情報が含まれます。同一性情報は次の判断に役立ちます。

- リクエストが、ルートと AWS Identity and Access Management ユーザー認証情報のどちらを使用して送信されたか。
- 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一時的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 のサービスによって送信されたかどうか。

詳細については、「[CloudTrail userIdentity 要素](#)」をご参照ください。

## AWS IoT Analytics ログファイルエントリの理解

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"
}
```

次の例は、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 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、コンプライアンスプログラム [AWS サービスによる対象範囲内のコンプライアンスプログラム](#) を参照し、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「コンプライアンスプログラム」](#) 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、[「でのレポートのダウンロード AWS Artifact」](#)の「」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 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。では、コンプライアンスに役立つ以下のリソース AWS を提供しています。

- [セキュリティとコンプライアンスのクイックスタートガイド](#) – これらのデプロイガイドでは、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に重点を置いたベースライン環境 AWS をにデプロイする手順について説明します。
- [アマゾン ウェブ サービスHIPAAのセキュリティとコンプライアンスのためのアーキテクチャ](#) – このホワイトペーパーでは、企業が AWS を使用して HIPAA 対象アプリケーションを作成する方法について説明します。

### Note

すべての AWS サービスがHIPAA対象となるわけではありません。詳細については、[HIPAA「対象サービスリファレンス」](#)を参照してください。

- [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。このガイドでは、ガイダンスを保護し AWS サービス、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) など) のセキュリティコントロールにマッピングするためのベストプラクティスをまとめています。
- [「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、リソース設定が社内プラクティス、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 サービス、内のセキュリティ状態を包括的に確認できます 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リストについては、[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](#) – これにより AWS アカウント、疑わしい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監視することで、ワークロード、コンテナ、データに対する潜在的な脅威 AWS サービス を検出します。GuardDuty は、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DSS、PCI などのさまざまなコンプライアンス要件に対応するのに役立ちます。
- [AWS Audit Manager](#) – これにより AWS 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ゾーンがあります。アベイラビリティゾーンでは、アベイラビリティ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 [AWS 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。

が AWS 公開したAPI呼び出しを使用して、ネットワーク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 )。1TLS.2 が必要で、1.3 TLS をお勧めします。
- (Ephemeral Diffie-HellmanPFS) や DHE (Elliptic Curve Ephemeral Diffie-Hellman) などの完全前方秘匿性 ECDHE () を備えた暗号スイート。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。

さらに、リクエストは、IAMプリンシパルに関連付けられたアクセスキー ID と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 Service](#) (AWS STS) を使用して、一時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 のクォータ

AWS 全般のリファレンス全般のリファレンスガイドには、AWS アカウントの AWS IoT Analytics に対するデフォルトのクォータ値が記述されています。特に指定がない限り、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のリクエスト、レスポンス、エラーなどを含め、AWS IoT Analytics API オペレーションについて詳しく説明します。

## AWS IoT Analytics アクション

AWS IoT Analytics API コマンドを使用して、IoT データの収集、処理、保存、分析を行うことができます。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の [AWS IoT Analytics セクション](#) には、AWS IoT Analytics の管理と操作に使用できる AWS CLI コマンドが含まれています。

## AWS IoT Analytics データ

AWS IoT Analytics データ API コマンドを使用すれば、AWS IoT Analytics、channel、pipeline、datastore、dataset で高度なアクティビティを実行できます。詳細については、AWS IoT Analytics API リファレンスの AWS IoT Analytics データでサポートされている [データ型](#) を参照してください。

# AWS IoT Analytics のトラブルシューティング

AWS IoT Analytics に関する問題を解決するためにエラーを解決して実行可能な解決方法を見出す方法については、次のセクションを参照してください。

## 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？](#)
- [Amazon QuickSight で自分の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ン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ン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ためのアクセス許可があることを確認してください。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。

データをチャン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 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 で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 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。新しく作成したロールに、AWS IoT Analytics からデータセットを読み取るポリシーのアクセス許可が付与されます。
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で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。

## Amazon QuickSight で自分のデータセットが表示されないのはなぜですか？

Amazon QuickSight では AWS IoT Analytics データセットコンテンツの読み取りにアクセス許可が必要になる可能性があります。アクセス許可を付与するには、次の手順に従います。

1. Amazon QuickSight の右上隅にあるアカウント名を選択し、Manage QuickSight を選択します。
2. 左側のナビゲーションペインで、セ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. AWS IoT Analytics にアクセス権がない場合は Add or remove 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、データセットコンテンツを読み取るための Amazon QuickSight 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。

必ず、AWS IoT Analytics と Amazon QuickSight に対して同一の AWS リージョンを選択してください。そうしない場合、AWS リソースへのアクセスに関する問題が発生す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、「Amazon Web Services 全般のリファレンス」の「[AWS IoT Analytics エンドポイントとクォータ](#)」と「[Amazon QuickSight エンドポイントとクォータ](#)」を参照してください。

## 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欠落していることが原因で発生します。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 技術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 の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後、ノートブック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。

## 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
<a href="#">リージョン起動</a>	AWS IoT Analytics が、アジアパシフィック (ムンバイ) リージョンで使用できるようになりました。	2021年8月18日
<a href="#">JOIN によるクエリ</a>	今回の更新により JOIN データセットのクエリに AWS IoT Analytics を使用できるようになりました。	2021年7月27日
<a href="#">AWS IoT SiteWise との統合</a>	AWS IoT SiteWise データのクエリに AWS IoT Analytics を使用できるようになりました。	2021年7月27日
<a href="#">カスタムパーティション</a>	AWS IoT Analytics では通常、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沿ったデータのパーティション化がサポートされるようになりました。	2021年6月14日
<a href="#">チャンネルメッセージの再処理</a>	この更新により、指定された Amazon S3 オブジェクトのチャン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
<a href="#">Parquet スキーマ</a>	AWS IoT Analytics データストアでは Parquet ファイル形式	2020年12月15日

がサポートされるようになりました。

### [CloudWatch Events によるモニタリング](#)

AWS Lambda アクティビティ中にランタイムエラーが発生すると、AWS IoT Analytics によりイベントが Amazon CloudWatch Events に自動的に発行されます。

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](#)

AWS IoT Analytics が中国 (北京) で利用可能になりました。

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
発売地域	AWS IoT Analytics がアアジアパシフィック (シドニー) リージョンで利用可能になりました。	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。