aws

開発者ガイド

AWS HealthLake

AWS HealthLake: 開発者ガイド

Table of Contents

とは AWS HealthLake

AWS HealthLake は、ヘルスケア相互運用性 FHIR (R4) 仕様を利用した臨床データの取り込み、保存、分析HIPAAの対象となるサービスです。

> **Note**
>
> 2023 年 2 月 20 日以降、 HealthLake データストアはデフォルトでは統合自然言語処理 (NLP) を使用しません。データストアでこの機能を有効にする場合は、トラブルシューティングの章 HealthLake統合された自然言語処理機能を有効にするにはどうすればよいですか？の「」を参照してください。

ヘルスデータは不完全な場合や一貫性のない場合があります。また、臨床メモ、ラボレポート、保険金請求、医療画像、録音された会話、時系列データ (ハートECGトレースやブレインEEGトレースなど) に含まれる情報を含む、構造化されていないこともよくあります。

ヘルスケアプロバイダーは、 HealthLake を使用して AWS クラウド内のデータを保存、変換、クエリ、分析できます。 HealthLake 統合された医療自然言語処理 (NLP) 機能を使用すると、さまざまなソースからの非構造化臨床テキストを分析できます。 は、自然言語処理モデルを使用して非構造化データを HealthLake 変換し、強力なクエリおよび検索機能を提供します。 HealthLake を使用して、安全で準拠しており、監査可能な方法で、患者情報を整理、インデックス作成、構造化できます。

HealthLake は、Amazon Athena および AWS Lake Formation とも統合されています。この統合を使用して、 を使用してデータストアをクエリできますSQL。

の利点 AWS HealthLake

を使用すると AWS HealthLake、次のことができます。

- ヘルスデータを迅速かつ簡単に取り込む – 臨床メモ、ラボレポート、保険金請求など、オンプレミスの Fast™ Interoperability Resources (FHIR) ファイルを Amazon Simple Storage Service (Amazon S3) バケットに一括インポートできます。その後、ダウンストリームアプリケーションまたはワークフローでデータを使用できます。
- FHIR REST API オペレーションを使用する – HealthLake は、 FHIR REST API オペレーションを使用してデータストアで CRUD (Create/Read/Update/Delete) オペレーションを実行します。FHIR検索もサポートされています。

- 監査可能な安全な - HIPAA権限のある方法で AWS クラウドにデータを保存する – データを FHIR 形式で保存できるため、簡単にクエリできます。 は、各患者の医療履歴の完全な時系列ビュー HealthLake を作成し、R4 FHIR標準形式で構造化します。
- Athena 統合 – HealthLakeと Athena の統合により、複雑なフィルター条件を作成および保存するために使用できる強力な SQLベースのクエリを作成できます。次に、このデータを SageMaker AI などのダウンストリームアプリケーションで使用して、機械学習モデルや Amazon QuickSight をトレーニングし、ダッシュボードやデータの視覚化を作成できます。
- 特殊な機械学習 (ML) モデルを使用して非構造化データを変換 – Amazon Comprehend Medical を使用して統合された医療自然言語処理 (NLP) HealthLake を提供します。生の医療テキストデータは、特殊な ML モデルを使用して変換されます。これらのモデルは、非構造化医療データから意味のある情報を理解し、抽出するようにトレーニングされています。統合された医療 ではNLP、エンティティ (医療処置や薬剤など) 、エンティティの関係 (薬剤やその投与量など) 、エンティティの特性 (陽性または陰性のテスト結果や処置時間など) データを医療テキストから自動的に抽出できます。 HealthLake その後、 は特性の兆候、症状、状態に基づいて新しいリソースを作成します。これらは、新しい条件、観測、および MedicationStatement リソースタイプとして追加されます。

HealthLake ユースケース

は、以下の医療アプリケーション HealthLake に使用できます。

- 母集団健康管理 — 医療組織は、母集団健康の傾向、成果、コストを分析する HealthLake のに役立ちます。これにより、組織は患者層に最適な介入を特定し、より良いケア管理オプションを選択できるようになります。
- 医療の質の向上 – HealthLake 患者の医療履歴の全体像をまとめることで、医療のギャップを埋め、医療の質を向上させ、コストを削減します。
- 医療効率の最適化 – 主要な分析ツールと機械学習ツール HealthLake を提供し、効率を向上させ、医療の無駄を減らします。

アクセス HealthLake

には、、 AWS Command Line Interface （ AWS CLI ） AWS Management Console、または HealthLake からアクセスできます AWS SDKs。

1. AWS Management Console – アクセスに使用できるウェブインターフェイスを提供します HealthLake。

2. AWS Command Line Interface （ AWS CLI) – Windows、macOS HealthLake、Linux など、さまざまな AWS のサービス用のコマンドを提供します。のインストールの詳細については AWS CLI、「」を参照してくださいAWS Command Line Interface。
3. AWS SDKs – さまざまなプログラミング言語とプラットフォーム SDKs (Java、Python、Ruby、.、iOSNET、Android など) のライブラリとサンプルコードで構成される (ソフトウェア開発キット) AWS を提供します。SDKs は、 HealthLake と へのプログラムによるアクセスを作成するのに便利です AWS。詳細については、「 for AWS PythonSDK」を参照してください。

HIPAA 適格性とデータセキュリティ

これはHIPAA対象サービスです。1996 年 AWS米国の医療保険の相互運用性と説明責任に関する法律 (HIPAA) 、および AWS サービスを使用して保護対象の医療情報 (PHI) を処理、保存、および送信する方法の詳細については、HIPAA概要を参照してください。

個人を特定できる情報 (PII) HealthLake を含む への接続は暗号化する必要があります。デフォルトでは、 HTTPS経由で HealthLake 使用するすべての接続TLSは、暗号化された顧客コンテンツを HealthLake 格納し、 責任AWS共有の原則に従って動作します。

料金

HealthLake 料金の詳細については、AWS HealthLake 料金表ページを参照してください。に関連する潜在的なコストをより正確に見積もるには HealthLake、 HealthLake 料金計算ツールを使用できます。

AWS HealthLake の仕組み

AWS HealthLake は、ヘルスケア相互運用性 FHIR (R4) 仕様を利用してヘルスレコードを保存するデータストアを作成します。では HealthLake、次のタスクを実行できます。

> **Note**
>
> 2023 年 2 月 20 日以降、 HealthLake データストアはデフォルトでは統合自然言語処理 (NLP) を使用しません。データストアでこの機能を有効にする場合は、トラブルシューティングの章 HealthLake統合された自然言語処理機能を有効にするにはどうすればよいですか？の「」を参照してください。

- データストアを作成、モニタリング、削除します。
- を使用して`StartFHIRImportJob`、Amazon Simple Storage Service (Amazon S3) バケットからデータストアに医療データを一括インポートします。
- データストアに保存されているデータを管理するには、作成、読み取り、更新、削除 (CRUD) オペレーションを使用します。
- Amazon Athena SQLで を使用してデータストアをクエリします。
- FHIR REST API オペレーションで HTTPクライアントを使用してデータストアを検索します。
- Amazon Comprehend Medical APIオペレーションを有効にして、自然言語処理 () を使用してデータ内の医療インサイトを検索しますNLP。

データストアの作成とモニタリング

を使用すると HealthLake、高速ヘルスケア相互運用性リソース (FHIR) データを保存できるデータストアを作成およびモニタリングできます。

新しいデータストアを作成するには、CreateFHIRDatastore または HealthLake コンソールを使用できます。データストアのステータスを確認するには、DescribeFHIRDatastore を使用します。 複数のアクティブなデータストアのステータスを表示するには、ListFHIRDatastores を使用します。データストアを削除するには、DeleteFHIRDatastore を使用します。

FHIR REST API オペレーション

FHIR REST API オペレーションを使用して、 HealthLake データストアで作成、読み取り、更新、削除 (CRUD) オペレーションを実行できます。が FHIRRESTAPIオペレーション HealthLake をサポートする方法の詳細については、「」を参照してください HealthLake データストア FHIRRESTAPIとのやり取りの使用。

リソース拡張からのFHIR DocumentReference リソースの自動生成

> **Note**
>
> HealthLake データストアを作成し、 を含むデータを追加するとDocumentReference、AWS アカウントで料金が発生します。詳細については、「 のAWS HealthLake 料金」を参照してください。

HealthLake は、 DocumentReferenceリソースタイプで見つかったドキュメントNLPに対して を提供します。テキストを分析するために、 は次の Amazon Comprehend Medical APIオペレーション HealthLake を使用します。

- DetectEntitiesV2: 臨床テキストでさまざまな医療エンティティを検査し、エンティティのカテゴリ、場所、信頼スコアなど、それらに関する特定の情報を返します。
- InferICD10CM: 臨床テキストを検査し、病状を患者記録にリストされているエンティティとして検出し、それらのエンティティを Centers for Health Control の ICD-10-CM ナレッジベースの正規化された概念識別子にリンクします。
- InferRxNorm: 臨床テキストを検査し、患者記録にリストされているエンティティとして薬剤を検出し、国立医学図書記から RxNorm データベース内の正規化された概念識別子にリンクします。

HealthLake は、データストアに追加されると、 DocumentReferenceリソースタイプで見つかったデータを自動的に分析します。元のDocumentReferenceリソースファイルは変更されません。抽出された医療情報は、 FHIR準拠の拡張機能として自動的に追加されます。NLP の仕組みの詳細については HealthLake、「」を参照してくださいのリソースタイプの自然言語処理 (NLP) に基づく自動 FHIR DocumentReference リソース生成の使用 AWS HealthLake。

SQLベースのクエリで検索する

> **Note**
>
> 2022 年 11 月 14 日より前に作成されたデータストアの場合、検索は FHIRRESTAPIオペレーションに制限されます。 HealthLake データストア内のデータに SQLベースのクエリを使用するには、「」を参照してくださいAmazon Athena SQLで を使用して AWS HealthLake データストアをクエリする。

Amazon Athena はサーバーレス SQLベースのクエリサービスです。 HealthLake データストアは Apache Iceberg テーブルとして Athena に取り込まれます。これらのテーブルは、大規模な分析データセットをサポートするように設計されています。Athena では、各FHIRリソースタイプはテーブルとして表されます。Athena を使用すると、データストアに対してのみREADリクエストを行うことができます。SQLベースの検索の詳細については、「」を参照してくださいを使用して HealthLake データストアをクエリする SQL。

FHIR REST API オペレーションで検索する

データストアに保存されているヘルスレコードを検索するには、サポートされている検索パラメータでリソースタイプを指定するか、リソースタイプを指定せずにサーバーで見つかったリソース ID を使用します。FHIR REST API オペレーションを使用した検索の詳細については、「」を参照してください HealthLake データストアFHIRRESTAPIとのやり取りの使用。

データインポートのアクション

を使用して AWS HealthLake 、Amazon S3 バケットからファイルを一括インポートします。コンソールまたは S tartFHIRImportジョブを使用してインポートジョブを開始します。ファイルをインポートしたら、D escribeFHIRImportジョブを使用してジョブのステータスをモニタリングできます。インポートジョブが完了すると、データを Athena に追加したり、変換したり、分析してダウンストリームアプリケーションで使用したりできます。

データエクスポートのアクション

HealthLake を使用して、ファイルを Amazon S3 バケットに一括エクスポートします。コンソールまたは S tartFHIRExportジョブを使用してエクスポートジョブを開始します。ファイルをエクスポー

トした後、D escribeFHIRExportジョブを使用してジョブのステータスをモニタリングし、そのプロパティを表示できます。エクスポートジョブが完了したら、Amazon を使用してデータを視覚化 QuickSight するか、他の AWS サービスを使用してデータにアクセスできます。

AWS HealthLake サポートされているFHIRプロファイルの検証

HealthLake は、基本 FHIR R4 仕様をサポートしています。R4 仕様にはFHIRプロファイルが含まれています。プロファイルはFHIR、リソースタイプで使用され、基本リソースタイプの制約や拡張を使用して、より具体的なリソースタイプ定義を定義します。例えば、FHIRプロファイルは拡張子や値セットなどの必須フィールドを識別できます。リソースは複数のプロファイルをサポートできます。すべての HealthLake データストアは、 FHIR プロファイルの使用をサポートしています。

HealthLake データストアにデータを追加するときは、FHIRプロファイルを追加する必要はありません。リソースが追加または更新されたときにFHIRプロファイルが指定されていない場合、リソースは基本 FHIR R4 スキーマに対してのみ検証されます。

FHIR リソースが準拠するプロファイルは、 に取り込まれる前にリソースに含まれます HealthLake。 は、指定されたFHIRプロファイルを HealthLake データストアに追加したときに HealthLake 検証します。

FHIR プロファイルは実装ガイドで指定されています。 は、次の実装ガイドで定義されているFHIRプロファイル HealthLake を検証します。

でサポートされているFHIRプロファイル HealthLake

名前	Versi	実装ガイド	機能
米国コア	3.1.1	http://hl7.org/fhir/us/core/STU3.1.1/	デフォルト
米国コア	4.0.0	https://hl7.org/fhir/us/core/STU4/index.html	サポート
CARIN 青いボタン	1.1.0	http://hl7.org/fhir/us/carin-bb/STU1.1/	デフォルト
CARIN 青いボタン	1.0.0	https://hl7.org/fhir/us/carin-bb/STU1/	サポート
Da Vinci Payer Data Exchange	1.0.0	https://hl7.org/fhir/us/davinci-pdex/	デフォルト

名前	Versi	実装ガイド	機能
Da Vinci Health Record Exchange (HRex)	0.2.0	https://hl7.org/fhir/us/davinci-hrex/2020Sep/	デフォルト
DaVinci PDEX 計画ネット	1.1.0	https://hl7.org/fhir/us/davinci-pdex-plan-net/STU1.1/	デフォルト
DaVinci PDEX 計画ネット	1.0.0	https://hl7.org/fhir/us/davinci-pdex-plan-net/STU1/	サポート
DaVinci Payer Data Exchange (PDex) 米国薬物処方	1.1.0	https://hl7.org/fhir/us/davinci-drug-formulary/STU1.1/	デフォルト
DaVinci Payer Data Exchange (PDex) 米国薬物処方	1.0.1	https://hl7.org/fhir/us/davinci-drug-formulary/STU1.0.1/	サポート
National Health Authority の Ayushman Bharat Digital Mission (ABDM)	2.0	https://www.nrces.in/ndhm/fhir/r4/index.html	デフォルト

リソースで指定されたFHIRプロファイルの検証

FHIR プロファイルを検証するには、実装ガイドでURL指定されたプロファイルを使用して、個々のリソースの `profile`要素にプロファイルを追加します。

FHIR プロファイルは、データストアに新しいリソースを追加するときに検証されます。新しいリソースを追加するには、S tartFHIRImportジョブAPIオペレーションを使用するか、新しいリソースを追加するPOSTリクエストを行うか、既存のリソースを更新する PUT リクエストを行います。

Example – リソースで参照されているFHIRプロファイルを確認するには

プロファイルURLは、`"meta" : "profile"`キーと値のペアの `profile`要素に追加されます。このリソースはわかりやすくするために切り捨てられました。

```
            {
             "resourceType": "Patient",
             "id": "abcd1234efgh5678hijk9012",
             "meta": {
              "lastUpdated": "2023-05-30T00:48:07.8443764-07:00",
              "profile": [
               "http://hl7.org/fhir/us/core/StructureDefinition/us-core-patient"
              ]
             }
            }
```

Example - デフォルト以外のサポートされているFHIRプロファイルを参照する方法

サポートされているデフォルト以外のプロファイル (例： CarinBB 1.0.0) - バージョン (「|」で区切る) URLのプロファイルと `meta.profile`要素URLのベースプロファイルを追加します。このサンプルリソースはわかりやすくするために切り捨てられました。

```
            {
             "resourceType": "ExplanationOfBenefit",
             "id": "sample-EOB",
             "meta": {
              "lastUpdated": "2024-02-02T05:56:09.4+00:00",
              "profile": [
               "http://hl7.org/fhir/us/carin-bb/StructureDefinition/C4BB-
ExplanationOfBenefit-Pharmacy|1.0.0",
                "http://hl7.org/fhir/us/carin-bb/StructureDefinition/C4BB-
ExplanationOfBenefit-Pharmacy“
              ]
             }
            }
```

プリロードされたデータ型

HealthLake は、プリロードされたデータ型SYNTHEAとしてのみサポートします。Synthea は、モデルによって生成された患者の医療履歴をモデル化する合成患者ジェネレーターです。これは、HealthLake が FHIR R4-compliantのリソースバンドルを生成できるオープンソースの Git リポジトリです。これにより、ユーザーは実際の患者データを使用せずにモデルをテストできます。

事前ロードされたデータストアでは、次のリソースタイプを使用できます。

サポートされている Synthea リソースタイプ

AllergyIntolerance	場所
CarePlan	MedicationAdministration
CareTeam	MedicationRequest
Claim	監視結果
条件	組織
デバイス	患者
DiagnosticReport	プラクティショナー
エンカウンター	PractitionerRole
ExplanationofBenefit	手順
ImagingStudy	プロベナンス
イミュナイゼーション	

の使用を開始するためのアクセス許可の設定 AWS HealthLake

この章では、 を使用して、 の使用を開始 AWS HealthLake してデータストアを作成するために必要なアクセス許可 AWS Management Console を設定します。データストアを作成するアクセス許可を設定するには、データレイク管理者および管理者である IAM ユーザーまたはロールを作成します HealthLake 。このユーザーを AWS Lake Formation のデータレイク管理者にします。データレイク管理者は、Amazon Athena を使用してデータストアをクエリするために必要なリソースへのアクセス権を Lake Formation に付与します。

でデータストアを作成したら HealthLake、データストアにファイルをインポートしたりエクスポートしたりするためのアクセス許可を設定できます。ファイルをインポートするためのアクセス許可の設定については、「」を参照してくださいインポートジョブのアクセス許可の設定。ファイルをエクスポートするアクセス許可の設定については、「」を参照してくださいエクスポートジョブのアクセス許可の設定。

トピック

- にサインアップする AWS アカウント
- 管理アクセスを持つユーザーを作成する
- を使用するように IAM ユーザーまたはロールを設定する HealthLake （ IAM 管理者 ）
- Lake Formation で Data Lake 管理者としてユーザーまたはロールを追加する (IAM 管理者)

にサインアップする AWS アカウント

がない場合は AWS アカウント、次のステップを実行して作成します。

にサインアップするには AWS アカウント

1. https://portal.aws.amazon.com/billing/サインアップを開きます。
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 のサービス とリソースへのアクセス権があ

ります。セキュリティ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。

AWS サインアッププロセスが完了すると、 から確認メールが送信されます。https://aws.amazon.com/ に移動してマイアカウントを選択すると、いつでも現在のアカウントアクティビティを表示し、アカウントを管理できます。

管理アクセスを持つユーザーを作成する

にサインアップしたら AWS アカウント、日常的なタスクにルートユーザーを使用しないように AWS アカウントのルートユーザー、 を保護し AWS IAM Identity Center、 を有効にして管理ユーザーを作成します。

を保護する AWS アカウントのルートユーザー

1. ルートユーザーを選択し、 AWS アカウント E メールアドレスを入力して、アカウント所有者AWS Management Consoleとして にサインインします。次のページでパスワードを入力します。

   ルートユーザーを使用してサインインする方法については、AWS サインイン ユーザーガイドのルートユーザーとしてサインインするを参照してください。
2. ルートユーザーの多要素認証 (MFA) を有効にします。

   手順については、「 ユーザーガイド」の AWS アカウント 「ルートユーザーの仮想MFAデバイスを有効にする (コンソール)」を参照してください。 IAM

管理アクセスを持つユーザーを作成する

1. IAM Identity Center を有効にします。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「AWS IAM Identity Centerの有効化」を参照してください。
2. IAM Identity Center で、ユーザーに管理アクセス権を付与します。

   を ID ソース IAM アイデンティティセンターディレクトリ として使用する方法のチュートリアルについては、「 AWS IAM Identity Center ユーザーガイド」の「デフォルトを使用してユーザーアクセスを設定する IAM アイデンティティセンターディレクトリ」を参照してください。

管理アクセス権を持つユーザーとしてサインインする

- IAM Identity Center ユーザーでサインインするには、IAMIdentity Center ユーザーの作成時に E メールアドレスにURL送信されたサインインを使用します。

  IAM Identity Center ユーザーを使用してサインインする方法については、「 AWS サインイン ユーザーガイド」の AWS 「 アクセスポータルにサインインする」を参照してください。

追加のユーザーにアクセス権を割り当てる

1. IAM Identity Center で、最小特権のアクセス許可を適用するベストプラクティスに従うアクセス許可セットを作成します。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「権限設定を作成する」を参照してください。
2. グループにユーザーを割り当て、そのグループにシングルサインオンアクセス権を割り当てます。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「グループの結合」を参照してください。

を使用するように IAM ユーザーまたはロールを設定する HealthLake （ IAM 管理者 ）

> ペルソナ: IAM 管理者
>
> ユーザーとロールを作成でき、データレイク管理者を追加できるIAMユーザー。

このトピックのこれらのステップは、IAM管理者が実行する必要があります。

HealthLake データストアを Athena に接続するには、データレイク管理者および HealthLake 管理者である IAM ユーザーまたはロールを作成する必要があります。この新しいユーザーまたはロールは、 AWS Lake Formation を介してデータストアで見つかったリソースへのアクセスを許可し、`AmazonHealthLakeFullAccess` AWS 管理ポリシーをユーザーまたはロールに追加します。

> **Important**
>
> データレイク管理者である IAM ユーザーまたはロールは、新しいデータレイク管理者を作成できません。データレイク管理者を追加するには、AdministratorAccessアクセス権が付与されたIAMユーザーまたはロールを使用する必要があります。

管理者を作成するには

1. 組織のユーザーまたはロールに **AmazonHealthlakeFullAccess**IAM AWS 管理ポリシーを追加します。

   IAM ユーザーの作成に慣れていない場合は、「 IAMユーザーガイド」のIAM「 ユーザーの作成」および「 ポリシーの概要」を参照してください。 AWS IAM
2. IAM ユーザーまたはロールに AWS Lake Formation へのアクセス権を付与します。

   - 組織内のユーザーまたはロールに次のIAM AWS 管理ポリシーを追加します。 **AWSLakeFormationDataAdmin**

     > **Note**
     >
     > このAWSLakeFormationDataAdminポリシーは、すべての AWS Lake Formation リソースへのアクセスを許可します。常に必要最小限のアクセス許可を使用してタスクを達成してください。詳細については、「 IAMユーザーガイド」のIAM「ベストプラクティス」を参照してください。
3. ユーザーまたはロールに次のインラインポリシーを追加します。詳細については、「 IAMユーザーガイド」の「インラインポリシー」を参照してください。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-source-bucket/*",
```

```
                "arn:aws:s3:::amzn-s3-demo-logging-bucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ram:GetResourceShareInvitations",
                "ram:AcceptResourceShareInvitation",
                "glue:CreateDatabase",
                "glue:DeleteDatabase"
            ],
            "Resource": "*"
        }
    ]
}
```

AWSLakeFormationDataAdmin ポリシーの詳細については、「Lake Formation デベロッパーガイド」の「Lake Formation ペルソナとIAMアクセス許可のリファレンス」を参照してください。 AWS

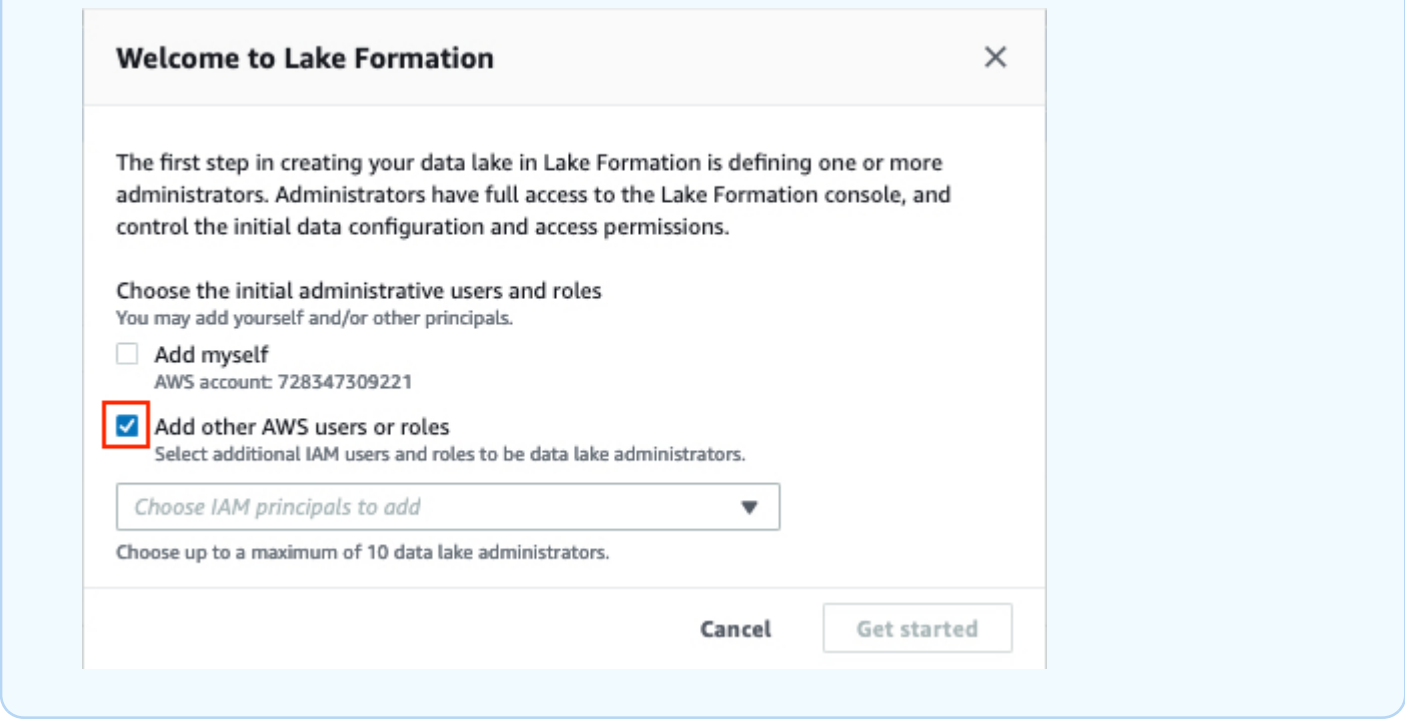
Lake Formation で Data Lake 管理者としてユーザーまたはロールを追加する (IAM 管理者)

次に、IAM管理者はステップ 1 で作成したユーザーまたはロールを Lake Formation のデータレイク管理者として追加する必要があります。

IAM ユーザーまたはロールをデータレイク管理者として追加するには

1. AWS Lake Formation コンソールを開きます。 https://console.aws.amazon.com/lakeformation/

> **Note**
>
> Lake Formation を初めて訪問する場合は、Lake Formation 管理者を定義するよう求める「Lake Formation へようこそ」ダイアログボックスが表示されます。



2. AWS Lake Formation データレイク管理者になる新しいユーザーまたはロールを割り当てます。
   - オプション 1:「Lake Formation へようこそ」ダイアログボックスが表示された場合。
     1. 他のユーザー AWS またはロールの追加 を選択します。
     2. 下矢印 (▼) を選択します。
     3. Lake Formation HealthLake 管理者にする管理者を選択します。
     4. [開始する] を選択します。
   - オプション 2: ナビゲーションペイン (☰) を使用します。
     1. ナビゲーションペイン (☰) を選択します。
     2. アクセス許可で、管理ロールとタスクを選択します。
     3. データレイク管理者セクションで、管理者の選択 を選択します。
     4. データレイク管理者の管理ダイアログボックスで、下矢印 (▼) を選択します。
     5. 次に、Lake Formation HealthLake 管理者にする管理者ユーザーまたはロールを選択または検索します。
     6. [Save] を選択します。
3. Lake Formation が管理するデフォルトのセキュリティ設定を変更します。 HealthLake データストアリソースは、 ではなく Lake Formation によって管理される必要がありますIAM。更新するには、AWS 「Lake Formation デベロッパーガイド」の「デフォルトのアクセス許可モデルを変更する」を参照してください。

でのデータストアの作成 AWS HealthLake

を完了するとの使用を開始するためのアクセス許可の設定 AWS HealthLake、データストアを作成する準備が整います。では AWS HealthLake、データストアを使用して HL7 FHIR (R4) 形式でデータを保存します。この章のトピックでは、データストアを作成する方法について説明します。

分析が有効なデータストアを作成し、Athena でデータストアへのアクセスを許可するには、 IAM ユーザー、グループ、またはロールに AWSLakeFormationDataAdmin管理ポリシーを追加します。このAWSLakeFormationDataAdminポリシーでは、データレイク管理者を作成し、Athena のデータストアへのアクセスを許可できます。アクセス許可の設定については、「」を参照してくださいの使用を開始するためのアクセス許可の設定 AWS HealthLake。

HealthLake は とも統合されています AWS CloudTrail。 CloudTrail を使用して、ユーザー、ロール、または AWS のサービスによって実行されたアクションの記録を提供できます HealthLake。は、 のすべてのAPI呼び出しとコンソールアクションをイベント HealthLake として CloudTrail キャプチャします。詳細については、「AWS CloudTrailを使用した AWS HealthLake APIコールのログ記録」を参照してください。

でサポートされる高速ヘルスケア相互運用性リソース (FHIR) リソースタイプの詳細については HealthLake、「」を参照してくださいでサポートされているFHIRリソースタイプ AWS HealthLake。

> ⚠ Amazon Athena の互換性
>
> HealthLake 2022 年 11 月 14 日より前に作成された 日付ストアは、Athena を使用してSQL クエリを実行できません。既存のデータストアで Athena 検索機能を使用するには、まずデータを新しいデータストアに移行します。既存のデータストアの移行の詳細については、「」を参照してくださいAmazon Athena を使用するための既存のデータストアの移行。

データストアを作成したら、 API_DescribeFHIRDatastoreまたは API_ListFHIRDatastores.html API オペレーションを使用して、ステータスを含むプロパティを取得できます。または、データストアのステータスやその他の詳細は、 HealthLake コンソールのデータストアページで確認できます。

HealthLake データストアのステータスは次のとおりです。

- 作成中 – データストアを作成中です。

- アクティブ — データストアがアクティブです。そこからデータをインポートおよびエクスポートできます。データストアに保存したFHIRリソースを管理および検索することもできます。
- 削除中 – データストアは削除中です。
- 削除済み – データストアは削除されました。

トピック

データストアの作成 (AWS Management Console)

> **⚠ HealthLake コンソールの違い**
>
> HealthLake コンソールは、FHIR有効なデータストアSMARTでの の作成をサポートしていません。FHIR 有効なデータストアSMARTで を作成するには、 AWS CLI または AWS サポートされている のいずれかを使用する必要がありますSDKS。詳細については、「SMART FHIR と の統合 AWS HealthLake」を参照してください。また、個々のデータストアの詳細ページを表示 HealthLake しても、 でサポートされている 2 種類のデータストアはコンソールで区別されません。

HealthLake データストアを作成するには

1. https://console.aws.amazon.com//healthlake/コンソールをホーム HealthLake で開きます。
2. ナビゲーションペイン (≡) を開きます。
3. 次に、データストアを選択します。
4. 次に、データストアの作成を選択します。
5. データストア設定セクションで、データストア名に名前を指定します。
6. (オプション) データストア設定セクションで、サンプルデータをプリロードするには、Synthea データをプリロードするチェックボックスをオンにします。
   - Synthea データはプリロードされたサンプルデータセットです。詳細については、「プリロードされたデータ型」を参照してください。
7. データストアの暗号化セクションで、 AWS所有キーを使用する (デフォルト) または別の AWSKMSキーを選択する (アドバンスト) を選択します。

8. タグ - オプションセクションでは、データストアにタグを追加できます。

   - データストアのタグ付けの詳細については、「」を参照してくださいデータストアへのタグの追加。

9. 次に、データストアの作成を選択します。データストアのステータスは、データストアページで確認できます。

データストアの作成 (AWS CLI および AWS SDKs)

次のコード例を使用して HealthLake データストアを作成できます。

AWS CLI

次の例は、 AWS CLIで CreateFHIRDatastore オペレーションを使用する方法を示しています。この例を実行するには、 AWS CLIをインストールする必要があります。データストアを作成すると、特に指定がない限り、保管時の暗号化はデフォルトで AWS所有KMSキーになります。での暗号化の詳細については、REST HealthLake 「」を参照してくださいfor RESTでの暗号化 AWS HealthLake。

例は、Unix、Linux、および macOS 用にフォーマットされています。Windows の場合は、各行の末尾にあるバックスラッシュ (\) UNIX 連結文字をキャレット () に置き換えます^。

```
aws healthlake create-fhir-datastore \
  --datastore-type-version R4 \
  --preload-data-config PreloadDataType="SYNTHEA" \
  --datastore-name "your-data-store-name"
```

成功すると、次のJSONレスポンスが返されます。データストアがデータを取り込む準備ができると、ステータスは に変わりますACTIVE。 HealthLake データストアへのデータのインポートの詳細については、「」を参照してください HealthLake データストアへのファイルのインポート。

```
{
    "DatastoreId": "eeb8005725ae22b35b4edbdc68cf2dfd",
 "DatastoreArn": "arn:aws:healthlake:us-west-2:111122223333:datastore/fhir/
eeb8005725ae22b35b4edbdc68cf2dfd",
    "DatastoreStatus": "CREATING",
    "DatastoreEndpoint": "https://healthlake.us-west-2.amazonaws.com/datastore/
eeb8005725ae22b35b4edbdc68cf2dfd/r4/"
```

```
}
```

すべてのデータストアデータストアのリストを表示するには、[ListFHIRDataStoreオペレーション]を使用できます。 HealthLake コンソールでアクティブデータストアのリストを表示することもできます。

Python (boto3)

次の例は、 `create_fhir_datastore`オペレーションを使用して HealthLake データストアを作成する方法を示しています。保管時のデータストア暗号化を作成する場合、特に指定がない限り、デフォルトで AWS所有 AWS KMS キーになります。での暗号化の詳細については、REST HealthLake 「」を参照してくださいfor RESTでの暗号化 AWS HealthLake。

```
import boto3
import logging #built in logging library
from botocore.exceptions import ClientError, ValidationError #specific exception
 ClientError from the boto3 library

def create_healthlake_datastore(DatastoreName=None):
    '''
    :param DatastoreName: the name of the data store, string
    :param:
    :return: True if the data store is created, else False
    '''

    # Create an Amazon Healthlake data store
    # Should we say something about region setting?
    # Should this example have some handling KMS keys

    try:
        if DatastoreName is None:
            healthlake_client = boto3.client('healthlake')
            healthlake_client.create_fhir_datastore(DatastoreTypeVersion='R4')

        else:
            healthlake_client = boto3.client('healthlake')
            healthlake_client.create_fhir_datastore(DatastoreTypeVersion='R4',
                                                    DatastoreName=DatastoreName)
    except (ClientError, ValidationError) as e:
        logging.error(e)
        return False

    return True
```

```
# Run the function above
create_healthlake_datastore(DatastoreName='test-datastore-delete-me-2')
```

データストアは、4 つのステータスのいずれかを持つことができます。`list_fhir_datastores` ステータスに関係なく HealthLake データストアのリストを表示するには、 を使用します。この例では、データストアのステータスに基づいてフィルタリングする方法を示します。

```
import boto3

healthlake_client = boto3.client('healthlake')
data_store_list = healthlake_client.list_fhir_datastores(Filter={'DatastoreStatus':
 'ACTIVE'})
print(data_store_list)
```

詳細については、Boto3 ドキュメント[list_fhir_datastore]の「」を参照してください。

HealthLake データストアへのファイルのインポート

を完了するとでのデータストアの作成 AWS HealthLake、Amazon Simple Storage Service (Amazon S3) バケットからデータストアにファイルをインポートできます。ファイルをインポートするには、HealthLake コンソールまたは `StartFHIRImportJob`APIオペレーションを使用してインポートジョブを開始します。

インポートジョブを作成するときは、Amazon S3 の入力データの場所、出力ログファイルの Amazon S3 バケットの場所、バケット HealthLake へのアクセスを許可する IAMロール、および顧客所有または AWS 所有 AWS Key Management Service キーを指定します。 はこのキー HealthLake を使用してソースの場所のデータを暗号化し、復号化してインポート HealthLake できるようにします。インポートジョブのアクセス許可の設定については、「」を参照してくださいインポートジョブのアクセス許可の設定。 AWS KMS キーの作成と使用の詳細については、「 Key Management Service https://docs.aws.amazon.com/kms/latest/developerguide/create-keys.htmlデベロッパーガイド」の「キーの作成」を参照してください。 AWS

HealthLake は、改行区切り JSON (`.ndjson`) 形式の入力ファイルを受け入れます。各行は有効な FHIRリソースで構成されます。API オペレーション `DescribeFHIRImportJob`および を使用して`ListFHIRImportJobs`、進行中のインポートジョブを記述および一覧表示できます。

インポートジョブごとに、 は `manifest.json` ファイル HealthLake を生成します。このログには、インポートジョブの成功と失敗の両方が示されます。 は、インポートジョブの作成時に指定した Amazon S3 バケットにファイルを HealthLake 出力します。詳細については、「マニフェストJSONファイル」を参照してください。

インポートジョブまたはエクスポートジョブをキューに入れることができます。これらの非同期インポートジョブまたはエクスポートジョブは、 FIFO (先入れ先出し) 方式で処理されます。インポートジョブまたはエクスポートジョブの進行中に、FHIRリソースを作成、読み取り、更新、または削除できます。

データストアに事前ロードされたデータを入力するか、データをインポートしたら、Amazon Athena SQLで を使用してデータストアのクエリを開始できます。詳細については、「Amazon Athena SQLで を使用して AWS HealthLake データストアをクエリする」を参照してください。

トピック

- インポートジョブのアクセス許可の設定
- でインポートジョブを開始する HealthLake
- マニフェストJSONファイル

- 例: を使用してインポートジョブを開始およびモニタリングする AWS CLI

インポートジョブのアクセス許可の設定

データストアにファイルをインポートする前に、Amazon S3 の入出力バケットにアクセスするためのアクセス HealthLake 許可を付与する必要があります。 HealthLake アクセスを許可するには、IAMのサービスロールを作成し HealthLake、ロールに信頼ポリシーを追加して HealthLake ロール継承アクセス許可を付与し、Amazon S3 バケットへのアクセスを許可するアクセス許可ポリシーをロールにアタッチします。

インポートジョブを作成するときは、 のこのロールの Amazon リソースネーム (ARN) を指定しますDataAccessRoleArn。IAM ロールと信頼ポリシーの詳細については、IAM「 ロール」を参照してください。

アクセス許可を設定したら、インポートジョブを使用してデータストアにファイルをインポートする準備が整います。詳細については、「でインポートジョブを開始する HealthLake」を参照してください。

インポート許可を設定するには

1. まだ作成していない場合は、出力ログファイルの送信先 Amazon S3 バケットを作成します。Amazon S3 バケットは サービスと同じ AWS リージョンにあり、すべてのオプションでブロックパブリックアクセスを有効にする必要があります。詳細については、Amazon S3のパブリックアクセスブロックの使用」を参照してください。Amazon 所有または顧客所有のKMS キーも暗号化に使用する必要があります。KMS キーの使用の詳細については、「Amazon Key Management Service」を参照してください。
2. 用のデータアクセスサービスロール HealthLake を作成し、次の信頼ポリシーで引き受けるアクセス許可を HealthLake サービスに付与します。 はこれ HealthLake を使用して出力 Amazon S3 バケットを書き込みます。

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

```
            "StringEquals": {
                "aws:SourceAccount": "your-account-id"
            },
            "ArnEquals": {
                "aws:SourceArn": "arn:aws:healthlake:us-west-2:account:datastore/
fhir/data store ID"
            }
        }
    }]
}
```

3. データアクセスロールにアクセス許可ポリシーを追加して、Amazon S3 バケットへのアクセスを許可します。をバケットの名前amzn-s3-demo-bucketに置き換えます。

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Action": [
            "s3:ListBucket",
            "s3:GetBucketPublicAccessBlock",
            "s3:GetEncryptionConfiguration"
        ],
        "Resource": [
            "arn:aws:s3:::amzn-s3-demo-source-bucket"
        ],
        "Effect": "Allow"
    },
    {
        "Action": [
            "s3:PutObject"
        ],
        "Resource": [
            "arn:aws:s3:::amzn-s3-demo-logging-bucket/*"
        ],
        "Effect": "Allow"
    },
    {
        "Action": [
            "kms:DescribeKey",
            "kms:GenerateDataKey*"
        ],
        "Resource": [
```

```
            "arn:aws:kms:us-east-1:012345678910:key/d330e7fc-b56c-4216-a250-
f4c43ef46e83"
        ],
        "Effect": "Allow"
    }]
}
```

でインポートジョブを開始する HealthLake

データストアを作成し、インポートジョブ (インポートジョブのアクセス許可の設定) のアクセス許可を設定したら、インポートジョブを使用してファイルのインポートを開始できます。インポートジョブは、 AWS HealthLake コンソールまたは AWS HealthLake インポート API、 start-fhir-import-job APIを使用して開始できます。

トピック

- API オペレーションを使用したファイルのインポート
- インポートジョブの開始 (コンソール)

API オペレーションを使用したファイルのインポート

前提条件

オペレーションを使用する場合は AWS HealthLake API、まず AWS Identity and Access Management （ IAM) ポリシーを作成し、 IAMロールにアタッチする必要があります。IAM ロールと信頼ポリシーの詳細については、IAM「 ポリシーとアクセス許可」を参照してください。また、暗号化にKMSキーを使用する必要があります。KMS キーの使用の詳細については、「Amazon Key Management Service」を参照してください。

ファイル (API) をインポートするには、次のステップを使用します。

1. Amazon S3 バケットにデータをアップロードします。
2. `start-fhir-import-job API` API オペレーションを使用します。ジョブを開始するときは、入力ファイルを含む Amazon S3 バケットの名前、暗号化に使用するKMSキー、および出力データ設定を指定します。
3. FHIR インポートジョブの詳細については、 describe-fhir-import-jobオペレーションを使用して、ジョブの ID、、名前ARN、開始時刻、終了時刻、および現在のステータスを取得します。を使用してlist-fhir-import-job、すべてのインポートジョブとそのステータスを表示します。

インポートジョブの開始 (コンソール)

コンソールでファイルをインポートするには、データを Amazon S3 バケットにアップロードします。

ファイルをインポートするには、次のステップを実行します。

1. Amazon S3 バケットにデータをアップロードします。
2. https://console.aws.amazon.com//healthlake/コンソールをホーム HealthLake で開きます。
3. データストアのデータストアの詳細ページに移動し、インポートを選択します。
4. Amazon S3 バケットを指定し、使用するIAMロールとKMSキーを作成または識別します。
5. [データをインポート] を選択します。

マニフェストJSONファイル

インポートジョブごとに、 は manifest.json file HealthLake を生成します。インポートジョブの作成時に指定した Amazon S3 バケットに ファイルが HealthLake 出力されます。

manifest.json ファイルは、インポートジョブの成功と失敗の両方を記述します。ログファイルは、 SUCCESSと という名前の 2 つのフォルダにまとめられていますFAILURE。出力ファイルには機密情報が含まれている可能性があるため、インポートジョブを作成するときは、出力 Amazon S3 バケットと暗号化用の AWS KMS キーの両方を指定する必要があります。

出力manifest.jsonファイルの例を次に示します。失敗したインポートジョブをトラブルシューティングする最初のステップとして、このファイルを使用することをお勧めします。各ファイルの詳細と、インポートジョブが失敗した原因について説明します。

```
  {
  "inputDataConfig": {
    "s3Uri": "s3://amzn-s3-demo-source-bucket/healthlake-input/invalidInput/"
  },
  "outputDataConfig": {
    "s3Uri": "s3://amzn-s3-demo-logging-bucket/32839038a2f47f17c2fe0f53f0c3a0ba-
FHIR_IMPORT-19dd7bb7bcc8ee12a09bf6d322744a3d/",
    "encryptionKeyID": "arn:aws:kms:us-west-2:123456789012:key/fbbbfee3-20b3-42a5-a99d-
c48c655ed545"
  },
  "successOutput": {
```

```
    "successOutputS3Uri": "s3://amzn-s3-demo-logging-
bucket/32839038a2f47f17c2fe0f53f0c3a0ba-FHIR_IMPORT-19dd7bb7bcc8ee12a09bf6d322744a3d/
SUCCESS/"
  },
  "failureOutput": {
    "failureOutputS3Uri": "s3://amzn-s3-demo-logging-
bucket/32839038a2f47f17c2fe0f53f0c3a0ba-FHIR_IMPORT-19dd7bb7bcc8ee12a09bf6d322744a3d/
FAILURE/"
  },
  "numberOfScannedFiles": 1,
  "numberOfFilesImported": 1,
  "sizeOfScannedFilesInMB": 0.023627,
  "sizeOfDataImportedSuccessfullyInMB": 0.011232,
  "numberOfResourcesScanned": 9,
  "numberOfResourcesImportedSuccessfully": 4,
  "numberOfResourcesWithCustomerError": 5,
  "numberOfResourcesWithServerError": 0
}
```

例: を使用してインポートジョブを開始およびモニタリングする AWS CLI

次の例は、 を使用してインポートジョブ AWS Command Line Interface を開始およびモニタリングする方法を示しています。また、start-fhir-import-job API を使用することもできます

```
aws healthlake start-fhir-import-job \
--input-data-config S3Uri=s3://amzn-s3-demo-source-bucket/inputFolder/ \
--datastore-id (Datastore ID) \
--data-access-role-arn "arn:aws:iam::012345678910:role/DataAccessRole" \
--job-output-data-config '{"S3Configuration": {"S3Uri":"s3://amzn-s3-demo-logging-
bucket/healthlake-output","KmsKeyId":"arn:aws:kms:us-east-1:012345678910:key/d330e7fc-
b56c-4216-a250-f4c43ef46e83"}}' \
--region us-east-1
```

インポートジョブが開始されると、次の確認が表示されます。

```
{
```

```
 "JobId": "8a4077553e9a485ad889c1a89c7541f0",
 "JobStatus": "SUBMITTED",
 "DatastoreId": "32839038a2f47f17c2fe0f53f0c3a0ba"
}
```

インポートジョブのステータスをモニタリングしたり、その設定プロパティを学習したりするには、次の例に示すように、 describe-fhir-import-jobAPIまたは AWS CLI コマンドを使用します。

```
    aws healthlake describe-fhir-import-job \
    --datastore-id (Datastore ID) \
    --job-id c145fbb27b192af392f8ce6e7838e34f \
    --region us-east-1
```

レスポンスには、次の情報が表示されます。

```
{
    "ImportJobProperties": {
    "InputDataConfig": {
    "S3Uri": "s3://amzn-s3-demo-source-bucket/(Prefix Name)/"
    },
    "DataAccessRoleArn": "arn:aws:iam::(AWS Account ID):role/(Role Name)",
    "JobStatus": "COMPLETED",
    "JobId": "c145fbb27b192af392f8ce6e7838e34f",
    "SubmitTime": 1606272542.161,
    "EndTime": 1606272609.497,
    "DatastoreId": "(Datastore ID)"
    }
}
```

すべてのインポートジョブのリストを表示するには、次の例に示すように、 list-fhir-import-jobsAPI または AWS CLI コマンドを使用します。結果を制限するために 1 つ以上のフィルターを追加できます。

```
    aws healthlake list-fhir-import-jobs\
    --datastore-id (Datastore ID) \
    --submitted-before (DATE like 2024-10-13T19:00:00Z)\
    --submitted-after (DATE like 2020-10-13T19:00:00Z )\
    --job-name "FHIR-IMPORT" \
```

```
    --job-status SUBMITTED  \
    --max-results (Integer between 1 and 500)
```

レスポンスには、次の情報が表示されます。

```
{
    "ImportJobProperties": {
    "OutputDataConfig": {
    "S3Uri": "s3://(Bucket Name)/(Prefix Name)/",
    "S3Configuration": {
     "S3Uri": "s3://(Bucket Name)/(Prefix Name)/",
     "KmsKeyId" : "(KmsKey Id)"
     },
    },
    "DataAccessRoleArn": "arn:aws:iam::(AWS Account ID):role/(Role Name)",
    "JobStatus": "COMPLETED",
    "JobId": "c145fbb27b192af392f8ce6e7838e34f",
    "JobName" "FHIR-IMPORT",
    "SubmitTime": 1606272542.161,
    "EndTime": 1606272609.497,
    "DatastoreId": "(Datastore ID)"
    }
}
"NextToken": String
```

HealthLake データストアからのファイルのエクスポート

データストアを作成してデータをインポートした後 (またはプリロードされたサンプルデータを使用する場合) 、データを Amazon S3 バケットにエクスポートできます。 HealthLake データストアからデータをエクスポートするには、次のオペレーションを使用します。

- および を使用して、 `StartFHIRExportJob`APIオペレーションを使用してエクスポートリクエストを行います AWS SDKs HealthLake。
  - このオペレーションは、システム全体のエクスポートリクエストのみをサポートします。
- を使用して `export`構文を使用して HealthLake FHIRエクスポートリクエストを行います RESTAPI。
  - このオペレーションは、システム全体、患者、およびグループのエクスポートリクエストの作成をサポートします。パラメータを適用して、エクスポートリクエスト内のデータをさらにフィルタリングすることもできます。

> **Important**
>
> HealthLake SDK `StartFHIRExportJob`APIオペレーションを使用したリクエストのエクスポートと `StartFHIRExportJobWithPost`APIオペレーションを使用したリクエストのFHIRRESTAPIエクスポートには、個別のIAMアクションがあります。を使用したSDKエクスポート`StartFHIRExportJob`と を使用したFHIRRESTAPIエクスポートの各IAMアクションでは`StartFHIRExportJobWithPost`、許可/拒否のアクセス許可を個別に処理できます。SDK と の両方のFHIRRESTAPIエクスポートを制限する場合は、各IAMアクションのアクセス許可を必ず拒否してください。

これらのオペレーションはどちらもAmazon S3バケットへのファイルのエクスポートのみをサポートします。データストアのすべての HealthLakeファイルは改行で区切られた JSON (`.ndjson`) ファイルとしてエクスポートされ、各行は有効なFHIRリソースで構成されます。

これらのオペレーションには、サービスロールが必要です。では、 HealthLake をサービスプリンシパルとして定義する必要があります。また、ファイルをエクスポートする の Amazon Simple Storage Service (S3) バケットを定義する必要があります。詳細については、「エクスポートジョブのアクセス許可の設定」を参照してください。

インポートジョブまたはエクスポートジョブをキューに入れることができます。これらの非同期インポートジョブまたはエクスポートジョブは、 FIFO (先入れ先出し) 方式で処理されます。インポートジョブまたはエクスポートジョブの進行中に、FHIRリソースを作成、読み取り、更新、または削除できます。

HealthLake データストアからファイルをエクスポートするには、以下のセクションを参照してください。

- エクスポートジョブのアクセス許可の設定
- HealthLake コンソールまたは を使用したデータストアからのファイルのエクスポート AWS SDKs
- FHIR REST API オペレーションを使用した HealthLake データストアからのデータのエクスポート

エクスポートジョブのアクセス許可の設定

データストアからファイルをエクスポートする前に、Amazon S3 の出力バケットにアクセスするためのアクセス HealthLake 許可を付与する必要があります。 HealthLake アクセスを許可するには、IAMのサービスロールを作成し HealthLake、ロールに信頼ポリシーを追加して HealthLake ロール継承アクセス許可を付与し、Amazon S3 バケットへのアクセスを許可するアクセス許可ポリシーをロールにアタッチします。

HealthLake で のロールを既に作成している場合はインポートジョブのアクセス許可の設定、そのロールを再利用して、このトピックにリストされている Amazon S3 バケットのエクスポートに追加のアクセス許可を付与できます。IAM ロールと信頼ポリシーの詳細については、IAM「 ポリシーとアクセス許可」を参照してください。

> **Important**
>
> HealthLake SDK `StartFHIRExportJob`APIオペレーションを使用したリクエストのエクスポートと `StartFHIRExportJobWithPost`APIオペレーションを使用したリクエストのFHIRRESTAPIエクスポートには、個別のIAMアクションがあります。を使用したSDKエクスポート`StartFHIRExportJob`と を使用したFHIRRESTAPIエクスポートの各IAMアクションでは`StartFHIRExportJobWithPost`、許可/拒否のアクセス許可を個別に処理できます。SDK と の両方のFHIRRESTAPIエクスポートを制限する場合は、各IAMアクションのアクセス許可を必ず拒否してください。ユーザーに へのフルアクセスを許可する場合 HealthLake、IAMユーザーアクセス許可の変更は必要ありません。

アクセス許可を設定するユーザーまたはロールには、ロールの作成、ポリシーの作成、ロールへのポリシーのアタッチを行うアクセス許可が必要です。次のIAMポリシーは、これらのアクセス許可を付与します。

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Action": ["iam:CreateRole", "iam:CreatePolicy", "iam:AttachRolePolicy"],
    "Effect": "Allow",
    "Resource": "*"
    }, {
    "Action": "iam:PassRole"
    "Effect": "Allow",
    "Resource": "*",
    "Condition": {
      "StringEquals": {
        "iam:PassedToService": "healthlake.amazonaws.com"
      }
    }
  }]
}
```

エクスポート許可を設定するには

1. まだ作成していない場合は、データストアからエクスポートするデータの送信先 Amazon S3 バケットを作成します。Amazon S3 バケットはサービスと同じAWSリージョンにあり、すべてのオプションでブロックパブリックアクセスを有効にする必要があります。詳細については、Amazon S3のパブリックアクセスブロックの使用」を参照してください。Amazon 所有または顧客所有のKMSキーも暗号化に使用する必要があります。KMS キーの使用の詳細については、「Amazon Key Management Service」を参照してください。
2. まだ作成していない場合は、 のデータアクセスサービスロールを作成し、次の信頼ポリシーで引き受けるアクセス許可を HealthLake サービスに HealthLake 付与します。 はこれ HealthLake を使用して出力 Amazon S3 バケットを書き込みます。で作成済みの場合はインポートジョブのアクセス許可の設定、再利用して、次のステップで Amazon S3 バケットのアクセス許可を付与できます。

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

```
        "Principal": {
            "Service": ["healthlake.amazonaws.com"]
        },
        "Action": "sts:AssumeRole",
        "Condition": {
            "StringEquals": {
                "aws:SourceAccount": "your-account-id"
            },
            "ArnEquals": {
                "aws:SourceArn": "arn:aws:healthlake:us-west-2:account:datastore/
fhir/data store ID"
            }
        }
    }]
}
```

3. 出力 Amazon S3 バケットへのアクセスを許可するアクセス許可ポリシーをデータアクセスロールに追加します。をバケットの名前amzn-s3-demo-bucketに置き換えます。

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Action": [
            "s3:ListBucket",
            "s3:GetBucketPublicAccessBlock",
            "s3:GetEncryptionConfiguration"
        ],
        "Resource": [
            "arn:aws:s3:::amzn-s3-demo-source-bucket"
        ],
        "Effect": "Allow"
    },
    {
        "Action": [
            "s3:PutObject"
        ],
        "Resource": [
            "arn:aws:s3:::amzn-s3-demo-logging-bucket/*"
        ],
        "Effect": "Allow"
    },
    {
        "Action": [
```

```
            "kms:DescribeKey",
            "kms:GenerateDataKey*"
        ],
        "Resource": [
            "arn:aws:kms:us-east-1:012345678910:key/d330e7fc-b56c-4216-a250-
f4c43ef46e83"
        ],
        "Effect": "Allow"
    }]
}
```

HealthLake コンソールまたは を使用したデータストアからのファイルのエクスポート AWS SDKs

を完了するとエクスポートジョブのアクセス許可の設定、データストアから Amazon Simple Storage Service (Amazon S3) バケットにファイルをエクスポートできます。データストアからファイルをエクスポートするには、 でエクスポートジョブを開始します HealthLake。エクスポートジョブは、データストアからファイルを改行区切り JSON (`.ndjson`) 形式でエクスポートします。各行は有効なFHIRリソースで構成されます。エクスポートジョブを開始するときは、暗号化用の AWS KMS キーを指定する必要があります。KMS キーの作成の詳細については、「 Key Management Service https://docs.aws.amazon.com/kms/latest/developerguide/create-keys.htmlデベロッパーガイド」の「キーの作成」を参照してください。 AWS

以下のトピックでは、 AWS HealthLake コンソールでエクスポートジョブを開始し、 start-fhir-export-jobAPIオペレーションで を開始する AWS SDKs方法について説明します。

トピック

- データストアからのファイルのエクスポート (コンソール)
- データストアからのファイルのエクスポート (AWS SDKs)

データストアからのファイルのエクスポート (コンソール)

ファイル (コンソール) をエクスポートするには、次のステップを使用します。

1. 同じリージョンに出力 S3 バケットを作成します HealthLake。
2. 新しいエクスポートジョブを開始するには、出力 Amazon S3 バケットを識別し、使用するIAMロールを作成または識別します。IAM ロールと信頼ポリシーの詳細については、「 IAMロー

ル」を参照してください。KMS キー暗号化も使用します。KMS キーの使用の詳細については、「Amazon Key Management Service」を参照してください。

3. エクスポートジョブのステータスを確認するには、 ListFHIRExportJobsAPIオペレーションを使用します。

データストアからのファイルのエクスポート (AWS SDKs)

を使用してデータストアからファイルをエクスポートするには AWS SDKs、 start-fhir-export-jobオペレーションを使用します。次のコードは、 SDK for Python (Boto3) を使用してエクスポートジョブを開始する方法を示しています。

```
import boto3

client = boto3.client('healthlake')

response = client.start_fhir_export_job(
    JobName='job name',
    OutputDataConfig={
        'S3Configuration': {
            'S3Uri': 's3://amzn-s3-demo-bucket/output-folder',
            'KmsKeyId': 'arn:aws:kms:us-west-2:account-number:key/AWS KMS key ID'
        }
    },
    DatastoreId='data store ID',
    DataAccessRoleArn='role ARN',
)
print(response['JobStatus'])
```

FHIR エクスポートジョブの ID、ARN、名前、開始時刻、終了時刻、および現在のステータスを取得するには、 を使用しますdescribe-fhir-export-job。を使用してlist-fhir-export-jobs、すべてのエクスポートジョブとそのステータスを一覧表示します。

次のコードは、 SDK for Python (Boto3) を使用して特定のエクスポートジョブのプロパティを取得する方法を示しています。

```
import boto3

client = boto3.client('healthlake')

describe_response = client.describe_fhir_export_job(
```

```
    DatastoreId=datastoreId,
    JobId=jobId
)
print(describe_response['ExportJobProperties'])
```

FHIR REST API オペレーションを使用した HealthLake データストアからのデータのエクスポート

を完了すると[エクスポートジョブのアクセス許可の設定]、 FHIRRESTAPIオペレーションを使用して HealthLake データストアからデータをエクスポートできます。を使用してエクスポートリクエストを行うにはFHIRRESTAPI、必要なアクセス許可を持つIAMユーザー、グループ、またはロールがあり、POSTリクエスト$exportの一部として を指定し、リクエストの本文にリクエストパラメータを含める必要があります。FHIR 仕様に従って、FHIRサーバーは GETリクエストをサポートする必要があります。また、 は POSTリクエストをサポートできます。追加のパラメータをサポートするには、エクスポートを開始するのに本文が必要です。したがって、 はPOSTリクエスト HealthLake をサポートします。

> **Important**
>
> HealthLake 2023 年 6 月 1 日より前に作成された データストアは、システム全体のエクスポートに対するFHIRRESTAPIエクスポートジョブリクエストのみをサポートします。HealthLake 2023 年 6 月 1 日より前に作成された データストアは、データストアのエンドポイントに対するGETリクエストを使用したエクスポートのステータスの取得をサポートしていません。

を使用して行うすべてのエクスポートリクエストFHIRRESTAPIは ndjson形式で返され、Amazon S3 バケットにエクスポートされます。各 S3 オブジェクトには、1 つのFHIRリソースタイプのみが含まれます。

AWS アカウントクォータに従ってエクスポートリクエストをキューに入れることができます。に関連付けられている Service Quotas の詳細については HealthLake、「」を参照してください[AWS HealthLake エンドポイントとクォータ]。

HealthLake では、次の 3 種類の一括エクスポートエンドポイントリクエストがサポートされています。

Type	説明	構文
システムエクスポート	サーバーからすべてのデータをエクスポートしますHealthLake FHIR。	`POST https://healthlake. your-region .amazonaws.com/datastore/ your-datastore-id /r4/$export`
すべての患者	患者リソースタイプに関連付けられたリソースタイプを含む、すべての患者に関連するすべてのデータをエクスポートします。	`POST https://healthlake. your-region .amazonaws.com/datastore/ your-datastore-id /r4/Patient/$export`
病棟のグループ	グループ ID で指定された患者のグループに関連するすべてのデータをエクスポートします。	`POST https://healthlake. your-region .amazonaws.com/datastore/ your-datastore-id /r4/Group/ ID/$export`

[開始する前に]

for を使用してエクスポートリクエストを行うには、次の要件を満たしたFHIRRESTAPIします HealthLake。

- エクスポートリクエストを行うために必要なアクセス許可を持つユーザー、グループ、またはロールを設定しておく必要があります。詳細については、「export リクエストの承認」を参照してください。
- データをエクスポートする Amazon S3 バケット HealthLake へのアクセスを許可するサービスロールを作成しておく必要があります。サービスロールでは、 をサービスプリンシパル HealthLake として指定する必要があります。アクセス許可の設定の詳細については、「」を参照してくださいエクスポートジョブのアクセス許可の設定。

export リクエストの承認

を使用してエクスポートリクエストを正常に実行するにはAPI、 IAM または FHIR REST OAuth2.0 を使用してユーザー、グループ、またはロールを承認します。サービスロールも必要です。

を使用したリクエストの承認 IAM

$export リクエストを行うとき、ユーザー、グループ、またはロールには、ポリシーに StartFHIRExportJobWithPost、DescribeFHIRExportJobWithGet、および CancelFHIRExportJobWithDeleteIAMアクションが含まれている必要があります。

> **Important**
>
> HealthLake SDK StartFHIRExportJobAPIオペレーションを使用したリクエストのエクスポートと StartFHIRExportJobWithPostAPIオペレーションを使用したリクエストのFHIRRESTAPIエクスポートには、個別のIAMアクションがあります。を使用したSDKエクスポートStartFHIRExportJobと を使用したFHIRRESTAPIエクスポートの各IAMアクションではStartFHIRExportJobWithPost、許可/拒否のアクセス許可を個別に処理できます。SDK と の両方のFHIRRESTAPIエクスポートを制限する場合は、各IAMアクションのアクセス許可を必ず拒否してください。

(2FHIR.0) SMARTでOAuth を使用してリクエストを承認する

FHIR 有効な HealthLake データストアSMARTで を$exportリクエストするときは、適切なスコープを割り当てる必要があります。サポートされているスコープの詳細については、「」を参照してくださいHealthLake データストアFHIRリソース固有のスコープ。

export リクエストの実行

このセクションでは、 FHIR REST を使用してエクスポートリクエストを行うときに必要な手順について説明しますAPI。

AWS アカウントで誤って課金されないように、 export構文を指定せずに リクエストを実行してPOSTリクエストをテストすることをお勧めします。

リクエストを行うには、以下を実行する必要があります。

1. サポートされているエンドポイントURLのexportPOSTリクエストで を指定します。
2. 必要なヘッダーパラメータを指定します。
3. 必要なパラメータを定義するリクエストボディを指定します。

ステップ 1: サポートされているエンドポイントURLの**POST**リクエスト**export**で を指定する

HealthLake は、3 種類の一括エクスポートエンドポイントリクエストをサポートしています。一括エクスポートリクエストを行うには、サポートされている 3 つのエンドポイントのいずれかに対して POSTベースのリクエストを行う必要があります。次の例は、リクエスト exportで を指定する方法を示していますURL。

- `POST https://healthlake.`**`your-region`**`.amazonaws.com/datastore/`**`your-datastore-id`**`/r4/$export`
- `POST https://healthlake.`**`your-region`**`.amazonaws.com/datastore/`**`your-datastore-id`**`/r4/Patient/$export`
- `POST https://healthlake.`**`your-region`**`.amazonaws.com/datastore/`**`your-datastore-id`**`/r4/Group/`**`ID`**`/$export`

そのPOSTリクエスト文字列では、以下のサポートされている検索パラメータを使用できます。

サポートされている検索パラメータ

HealthLake では、一括エクスポートリクエストで次の検索修飾子がサポートされています。

これらの例には、リクエストを送信する前にエンコードする必要がある特殊文字が含まれています。

名前	必須?	説明	例
_outputFormat	いいえ	リクエストされた一括データファイルを生成する形式。使用できる値は、application/fhir+ndjson 、application/ndjson 、 ですndjson。	
_type	いいえ	エクスポートジョブに含めるカンマ区切りのFHIRリソースタ	&_type=MedicationS

名前	必須?	説明	例
		イプの文字列。すべてのリソースをエクスポートするとコストがかかる_type可能性があるため、 を含めることをお勧めします。	tatement, Observation
_since	いいえ	日付タイムスタンプ以降に変更されたリソースタイプ。リソースタイプに最終更新時刻がない場合、レスポンスに含まれます。	&_since=2024-05-09T00%3A00%3A00Z

ステップ 2: 必要なヘッダーパラメータを指定する

を使用してエクスポートリクエストを行うにはAPI、次の 2 FHIR REST つのヘッダーパラメータを指定する必要があります。

- Content-Type : application/fhir+json
- 優先: respond-async

次に、リクエストボディに必要な要素を指定する必要があります。

ステップ 3: で必要なパラメータを定義するリクエスト本文を指定します。

エクスポートリクエストには、 JSON 形式の本文も必要です。本文には、次のパラメータを含めることができます。

キー	必須?	説明	値
DataAccessRoleArn	はい	HealthLake サービスロールARNの 。使用	arn:aws:iam:: **444455556**

キー	必須?	説明	値
		するサービスロールは、サービスプリンシパル HealthLake として を指定する必要があります。	**666** :role/**your-healthlake-service-role**
JobName	いいえ	エクスポートリクエストの名前。	**your-export-job-name**
S3Uri	はい	OutputDataConfig キーの一部。エクスポートされたデータがダウンロードされる送信先バケットURIの S3。	s3://DOC-EXAMPLE-DESTINATION-BUCKET/ **EXPORT-JOB** /
KmsKeyId	はい	OutputDataConfig キーの一部。Amazon S3 バケットの保護に使用される AWS KMS キーARNの 。Amazon S3	arn:aws:kms: **region-of-bucket:123456789012** :key/**1234abcd-12ab-34cd-56ef-1234567890ab**

Example – を使用して行われたエクスポートリクエストの本文 FHIR REST API

を使用してエクスポートリクエストを行うにはFHIRRESTAPI、次に示すように本文を指定する必要があります。

```
{
  "DataAccessRoleArn": "arn:aws:iam::444455556666:role/your-healthlake-service-role",
  "JobName": "your-export-job",
  "OutputDataConfig": {
    "S3Configuration": {
      "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/EXPORT-JOB",
      "KmsKeyId": "arn:aws:kms:region-of-
bucket:444455556666:key/1234abcd-12ab-34cd-56ef-1234567890ab"
```

```
    }
  }
}
```

リクエストが成功すると、次のレスポンスを受け取ります。

レスポンスヘッダー

```
content-location: https://healthlake.your-region.amazonaws.com/datastore/your-
datastore-id/r4/export/your-export-request-job-id
```

レスポンス本文

```
{
  "datastoreId": "your-data-store-id",
  "jobStatus": "SUBMITTED",
  "jobId": "your-export-request-job-id"
}
```

エクスポートリクエストの管理

エクスポートリクエストが成功したら、 exportを使用して現在のエクスポートリクエストのステータスを記述し、 exportを使用して現在のエクスポートリクエストをキャンセルすることで、そのリクエストを管理できます。

を使用してエクスポートリクエストをキャンセルするとRESTAPI、キャンセルリクエストを送信した時点までにエクスポートされたデータの一部に対してのみ請求されます。

以下のトピックでは、現在のエクスポートリクエストのステータスを取得またはキャンセルする方法について説明します。

エクスポートリクエストのキャンセル

エクスポートリクエストをキャンセルするには、DELETEリクエストを作成し、リクエスト でジョブ ID を指定しますURL。

```
DELETE https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
export/your-export-request-job-id
```

リクエストが成功すると、次の情報が表示されます。

```
{
  "exportJobProperties": {
    "jobId": "your-original-export-request-job-id",
    "jobStatus": "CANCEL_SUBMITTED",
    "datastoreId": "your-data-store-id"
  }
}
```

リクエストが成功しなかった場合は、次の情報が表示されます。

```
{
  "resourceType": "OperationOutcome",
  "issue": [
    {
      "severity": "error",
      "code": "not-supported",
      "diagnostics": "Interaction not supported."
    }
  ]
}
```

エクスポートリクエストの説明

エクスポートリクエストのステータスを取得するには、 exportと を使用してGETリクエストを行いますexport-request-job-id。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
export/your-export-request-id
```

JSON レスポンスには ExportJobProperties オブジェクトが含まれます。これには、次のキーと値のペアが含まれる場合があります。

名前	必須?	説明	値
DataAccessRoleArn	いいえ	HealthLake サービスロールARNの 。使用するサービスロールは、サービスプリンシパル HealthLake と	arn:aws:iam:: **444455556666** :role/**your-healthlake-service-role**

名前	必須?	説明	値
		して を指定する必要があります。	
SubmitTime	いいえ	エクスポートジョブが送信された日付。	`Apr 21, 2023 5:58:02`
EndTime	いいえ	エクスポートジョブが完了した時刻。	`Apr 21, 2023 6:00:08 PM`
JobName	いいえ	エクスポートリクエストの名前。	**`your-export-job-name`**
JobStatus	いいえ		有効な値は次のとおりです。 `SUBMITTED | IN_PROGRESS | COMPLETED_WITH_ERRORS | COMPLETED | FAILED`
S3Uri	はい	OutputDataConfig オブジェクトの一部。エクスポートされたデータがダウンロードされるレプリケート先バケットURIのAmazon S3。	`s3://DOC-EXAMPLE-DESTINATION-BUCKET/` **`EXPORT-JOB`** `/`
KmsKeyId	はい	OutputDataConfig オブジェクトの一部。Amazon S3 バケットの保護に使用される AWS KMS キー ARNの 。 Amazon S3	`arn:aws:kms:` **`region-of-bucket:123456789012`** `:key/`**`1234abcd-12ab-34cd-56ef-1234567890ab`**

Example : を使用して行われた describe エクスポートリクエストの本文 FHIR REST API

成功すると、次のJSONレスポンスが表示されます。

```
{
  "exportJobProperties": {
    "jobId": "your-export-request-id",
    "JobName": "your-export-job",
    "jobStatus": "SUBMITTED",
    "submitTime": "Apr 21, 2023 5:58:02 PM",
    "endTime": "Apr 21, 2023 6:00:08 PM",
    "datastoreId": "your-data-store-id",
    "outputDataConfig": {
      "s3Configuration": {
        "S3Uri": "s3://DOC-EXAMPLE-DESTINATION-BUCKET/EXPORT-JOB",
        "KmsKeyId": "arn:aws:kms:region-of-
bucket:444455556666:key/1234abcd-12ab-34cd-56ef-1234567890ab""
      }
    },
    "DataAccessRoleArn": "arn:aws:iam::444455556666:role/your-healthlake-service-role",
  }
}
```

でのデータストアの削除 HealthLake

データストアの削除は非同期オペレーションです。開始すると、ステータスは Delete に変わります。データストアは、日付ストアのすべてのFHIRデータと、必要な基盤となるインフラストラクチャも削除されるまで、削除のステータスを維持します。

データおよびインフラストラクチャが削除されると、 HealthLake データストアのステータスは削除済みに変わります。削除後、データストアに関する詳細は、 `DescribeFHIRDataStore` および `ListFHIRDataStores`オペレーションを 7 日間使用することによってのみ利用できます。7 日後、削除されたデータストアは結果に表示されません。

データストアを正常に削除するには、リクエストを行うユーザー、グループ、またはロールに IAM アクションがIAMポリシー`glue:DeleteDatabase`に追加されている必要があります。このIAMアクションは、 管理ポリシー AWS の一部として含まれません`AmazonHealthLakeFullAccess`。

データストアは AWS Management Console、、 AWS SDKs、または を使用して削除できます AWS CLI。

トピック

- データストアの削除 (コンソール)
- データストアの削除 (AWS SDKs および AWS CLI)

データストアの削除 (コンソール)

コンソールでデータストアを削除するには、データストアページでデータストアを選択し、削除を選択します。

HealthLake データストアを削除するには

1. https://console.aws.amazon.com//healthlake/コンソールをホーム HealthLake で開きます。
2. ナビゲーションペイン (≡) を開きます。
3. 次に、データストアを選択します。
4. データストアページで、削除するデータストアの横にあるオプションを選択します。
5. 次に、削除を選択します。
6. ダイアログボックスに「」と入力**delete**して、選択したデータストアを削除することを確認します。

7. その後、[削除] をクリックします。その後、データストアのステータスがアクティブから削除に変わります。

データストアの削除 (AWS SDKs および AWS CLI)

以下のコードサンプルを使用して、 HealthLake データストアを削除できます。

AWS CLI

以下の例は、 で DeleteFHIRDatastoreオペレーションを使用する方法を示しています AWS CLI。この例を実行するには、 AWS CLIをインストールする必要があります。

```
aws healthlake delete-fhir-datastore --datastore-id
 'eeb8005725ae22b35b4edbdc68cf2dfd'
```

成功すると、次のJSONレスポンスが返されます。

```
{
    "DatastoreProperties": {
  "DatastoreId": "eeb8005725ae22b35b4edbdc68cf2dfd",
        "DatastoreArn": "arn:aws:healthlake:us-west-2:728347309221:datastore/fhir/",
        "DatastoreName": "delete-me",
        "DatastoreStatus": "ACTIVE",
        "CreatedAt": "2022-10-03T10:53:45.020000-07:00",
        "DatastoreTypeVersion": "R4",
        "DatastoreEndpoint": "https://healthlake.us-west-2.amazonaws.com/
datastore/5b6e4cd798289a4ab8dad6c1002dd731/r4/",
        "SseConfiguration": {
            "KmsEncryptionConfig": {
                "CmkType": "AWS_OWNED_KMS_KEY"
            }
        },
        "PreloadDataConfig": {
            "PreloadDataType": "SYNTHEA"
        }
    }
}
```

Python (boto3)

for AWS SDKPython は、単一のパラメータ を取り込む describe_fhir_datastoreメソッドをサポートしていますDatastoreId。

```
import boto3

#Create a Healthlake client
healthlake_client = boto3.client('healthlake')

#Call the describe_fhir_datastore method
data_store_details =
 healthlake_client.describe_fhir_datastore(DatastoreId='cdf8f1557e57c543bdc627fb8f12b7fd')

print(data_store_details)
```

成功すると、Python ディクショナリが返されます。

```
{'DatastoreProperties': {'DatastoreId': 'cdf8f1557e57c543bdc627fb8f12b7fd',
 'DatastoreArn': 'arn:aws:healthlake:us-west-2:728347309221:datastore/fhir/
cdf8f1557e57c543bdc627fb8f12b7fd', 'DatastoreName': '08-24-2022-test-data-
store', 'DatastoreStatus': 'ACTIVE', 'CreatedAt': datetime.datetime(2022,
 8, 23, 22, 12, 14, 359000, tzinfo=tzlocal()), 'DatastoreTypeVersion': 'R4',
 'DatastoreEndpoint': 'https://healthlake.us-west-2.amazonaws.com/datastore/
cdf8f1557e57c543bdc627fb8f12b7fd/r4/', 'SseConfiguration': {'KmsEncryptionConfig':
 {'CmkType': 'AWS_OWNED_KMS_KEY'}}, 'PreloadDataConfig': {'PreloadDataType':
 'SYNTHEA'}}, 'ResponseMetadata': {'RequestId': 'aef4b268-ad4b-4b57-
bc97-2da956356835', 'HTTPStatusCode': 200, 'HTTPHeaders': {'date': 'Wed, 05 Oct
 2022 01:21:44 GMT', 'content-type': 'application/x-amz-json-1.0', 'content-
length': '547', 'connection': 'keep-alive', 'x-amzn-requestid': 'aef4b268-ad4b-4b57-
bc97-2da956356835'}, 'RetryAttempts': 0}}
```

一度に複数のデータストアに関する詳細を返すには、 を使用します。 ListFHIRDatastore

次の例 AWS CLI に示すように、 を使用して DeleteFHIRDataStore コマンドを使用します。delete-fhir-datastore API または コンソールを使用してデータストアを削除することもできます。データストアを削除すると、データストアと基盤となるインフラストラクチャに含まれるすべてのFHIRリソースバージョンが削除されます。削除されたデータストアに関連するログは、HIPAAガイドラインに従ってサービスアカウント内に保持されます。

```
aws healthlake delete-fhir-datastore
```

```
    --datastore-id (Data Store ID)
```

次のJSONレスポンスの例に示すように、ステータスはDELETING「」に変わり、データストアとそのコンテンツが削除されていることを確認します。

```
{
  "DatastoreEndpoint": "https://healthlake.us-east-1.amazonaws.com/
datastore/eeb8005725ae22b35b4edbdc68cf2dfd/r4/",
    "DatastoreArn": "arn:aws:healthlake:us-east-1:(AWS Account ID):datastore/(Datastore
 ID)",
    "DatastoreStatus": "DELETING",
    "DatastoreId": "(Datastore ID)"
}
```

HealthLake データストアFHIRRESTAPIとのやり取りの使用

では AWS HealthLake、高速ヘルスケア相互運用性リソース (FHIR) RESTAPIインタラクションを使用して、データストア内のFHIRリソースを管理および検索します。 FHIR REST API インタラクションは、データストア内のリソースに対して作成、読み取り、更新、削除 (CRUD) インタラクションを実行するために使用されます。は FHIRがサポートする検索オペレーションのサブセットをサポートするため HealthLake、 GETまたは POSTHTTPリクエストを使用して複雑な検索文字列を作成することもできます。

適合のために、FHIRリソースタイプは HL7 FHIR R4 StructureDefinitionリソースに従って検証されます。アクティブな HealthLake データストアの FHIR関連の機能を見つけるには、次のように URL で metadataが指定されている をGETリクエストします。

```
GET https://healthlake.region.amazonaws.com/datastore/datastore-id/r4/metadata
```

成功すると、 HealthLake データストアの200HTTPレスポンスコードと機能ステートメントを受け取ります。詳細については、HL7FHIRR4 ドキュメントCapabilityStatementの「」を参照してください。

次の表は、 でサポートされているFHIRインタラクションの一覧です AWS HealthLake。

FHIR でサポートされているインタラクション AWS HealthLake

FHIR インタラクション	説明
システム全体のやり取り	
capabilities	システムの機能ステートメントを取得する
batch/transaction	1 回の操作で一連のリソースを更新、作成、または削除します。
タイプレベルのインタラクション	
create	サーバーに割り当てられた ID を使用して新しいリソースを作成する
search	一部のフィルター条件に基づいてリソースタイプを検索する
history	特定のリソースタイプの変更履歴を取得する

FHIR インタラクション	説明
インスタンスレベルのインタラクション	
read	リソースの現在の状態の読み取り
history	特定のリソースの変更履歴を読み取る
vread	リソースの特定のバージョンの状態を読み取る
update	ID でリソースを更新する (または新しい場合は作成)
delete	リソースを削除する

トピック

- でサポートされているFHIRリソースタイプ AWS HealthLake
- HealthLake データストアでの作成、読み取り、更新、削除 (CRUD) オペレーションの実行
- FHIR REST API オペレーションを使用した HealthLake データストアの検索
- FHIR リソース履歴の読み取り
- 患者 $everything FHIRRESTAPIオペレーションを使用した患者データの取得
- $export を使用した HealthLake データストアからのデータのエクスポート

でサポートされているFHIRリソースタイプ AWS HealthLake

次の表に、 でサポートされている FHIR R4 リソースタイプを示します AWS HealthLake。詳細については、HL7FHIRR4 ドキュメントの「リソースインデックス」を参照してください。

FHIR でサポートされている R4 リソースタイプ HealthLake

アカウント	DetectedIssue	Invoice	プラクティショナー
ActivityDefinition	デバイス	[Library] (ライブラリ)	PractitionerRole
AdverseEvent	DeviceDefinition	リンク	手順
AllergyIntolerance	DeviceMetric	リスト	利点

予約	DeviceUseStatement	場所	アンケート
AppointmentResponse	DeviceRequest	メジャー	QuestionnaireResponse
AuditEvent- メモを参照	DiagnosticReport	MeasureReport	RelatedPerson
バイナリ	DocumentManifest	メディア	RequestGroup
BodyStructure	DocumentReference	薬剤	ResearchStudy
バンドル - 注を参照	EffectEvidenceSynthesis	MedicationAdministration	ResearchSubject
CapabilityStatement	エンカウンター	MedicationDispense	RiskAssessment
CarePlan	エンドポイント	MedicationKnowledge	RiskEvidenceSynthesis
CareTeam	EpisodeOfCare	MedicationRequest	スケジュール
ChargeItem	EnrollmentRequest	MedicationStatement	ServiceRequest
ChargeItemDefinition	EnrollmentResponse	MessageHeader	スロット
Claim	ExplanationOfBenefit	MolecularSequence	舌
ClaimResponse	FamilyMemberHistory	NutritionOrder	StructureDefinition
コミュニケーション	フラグ	監視結果	StructureMap
CommunicationRequest	目標	OperationOutcome	Substance
コンポジション	グループ	組織	SupplyDelivery
ConceptMap	GuidanceResponse	OrganizationAffiliation	SupplyRequest
条件	HealthcareService	パラメータ	タスク

同意	ImagingStudy	患者	ValueSet
Contract	イミュナイゼーション	PaymentNotice	VisionPrescription
カバレッジ	ImmunizationEvaluation	PaymentReconciliation	VerificationResult
CoverageEligibilityRequest	ImmunizationRecommendation	個人	
CoverageEligibilityResponse	InsurancePlan	PlanDefinition	

⚠ FHIR 仕様と HealthLake

- バイナリ、バンドル、パラメータのリソースタイプを使用して GETまたは OperationOutcomeP0STリクエストを行うことはできません。
- AuditEvent — AuditEvent リソースは作成または削除できますが、更新または削除することはできません。
- バンドル — がバンドルリクエスト HealthLake を管理する方法は複数あります。詳細については、Bundle を使用した複数のFHIRリソースの管理を参照してください。
- VerificationResult — このリソースタイプは、2023 年 12 月 9 日以降に作成されたデータストアでのみサポートされます。

HealthLake データストアでの作成、読み取り、更新、削除 (CRUD) オペレーションの実行

データストアの管理、データのインポート、データのエクスポートにはネイティブ AWS アクションを使用しますが、4 つの主なFHIRHTTPオペレーションを使用して HealthLake 、データストア内のFHIRリソースの作成 (P0ST) 、読み取り (GET) 、更新 (PUT) 、削除 (DELETE) を行います。以下のトピックでは、 FHIRRESTAPIサービスを使用して HealthLake データストアで作成、読み取り、更新、削除 (CRUD) オペレーションを実行する方法について説明します。HTTP クライアントを介して送信されるリクエストを認証 HealthLake APIするには、署名バージョン 4 の署名プロセスを使用す

る必要があります。詳細については、『』の「署名バージョン 4 の署名プロセス」を参照してくださいAWS 全般のリファレンス。

トピック

- を使用したリソースの作成 POST
- を使用したリソースの読み取り GET
- を使用したリソースの更新 PUT
- を使用したリソースの削除 DELETE
- Bundle を使用した複数のFHIRリソースの管理

を使用したリソースの作成 POST

POST リクエストを使用して HealthLake データストアに新しいリソースを作成します。 POSTリクエストでは、 id要素を指定する必要はありません。リソースが正常に作成されると、 HealthLake サーバーは201作成済みHTTPステータスコードを返します。

> **Note**
>
> DocumentReference リソースタイプに対してPOSTリクエストを行うと、既存の拡張機能は変更されません。代わりに、 は既存の拡張機能を含む新しい拡張機能をデータストア AWS HealthLake に追加します。がDocumentReferenceリソースタイプで自然言語処理 (NLP) HealthLake を使用して貴重な医療データを抽出する方法の詳細については、「」を参照してくださいのリソースタイプの自然言語処理 (NLP) に基づく自動FHIR DocumentReference リソース生成の使用 AWS HealthLake。

Example **POST** リクエストを使用した**Patient**リソースの作成。

HealthLake データストアPOSTリクエストを作成するには、データストアのエンドポイントを使用してJSONリクエストボディを指定します。データストアのエンドポイントを見つけるには、HealthLake コンソールの データストアを参照するか、 AWS HealthLake APIリファレンスの DescribeFHIRDatastore オペレーションを使用します。

POST Request

```
POST https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient
```

JSON Request Body

```
{
  "resourceType": "Patient",
  "identifier": [ { "system": "urn:oid:1.2.36.146.595.217.0.1", "value":
 "12345" } ],
  "name": [ {
      "family": "Silva",
      "given": ["Ana", "Carolina"]
  } ],
  "gender": "female",
  "birthDate": "1992-02-10"
}
```

JSON 応答

患者リソースの作成を確認するには、201作成済みHTTPステータスコードと次のJSONレスポンスを受け取ります。

```
{
  "resourceType": "Patient",
  "identifier": [
    {
      "system": "urn:oid:1.2.36.146.595.217.0.1",
      "value": "12345"
    }
  ],
  "name": [
    {
      "family": "Silva",
      "given": [
        "Ana",
        "Carolina"
      ]
    }
  ],
  "gender": "female",
  "birthDate": "1992-02-10",
  "id": "274b408a-1201-4e9f-a621-1df937f1a26d",
  "meta": {
    "lastUpdated": "2022-06-13T23:31:24.427Z"
  }
```

```
}
```

を使用したリソースの読み取り GET

この例では、 GETリクエストを使用して患者FHIRリソースを読み取る方法を示します。

Example **GET** リクエストを使用して特定の**Patient**リソースを読み取る。

HealthLake データストアGETリクエストを作成するには、データストアのエンドポイントを使用します。データストアのエンドポイントを見つけるには、 HealthLake コンソールの データストアを参照するか、 AWS HealthLake APIリファレンスの DescribeFHIRDatastore オペレーションを使用します。

また、 リソースタイプ**Patient**と有効な識別子 を含める必要があります**2de04858-ba65-44c1-8af1-f2fe69a977d9**。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient/2de04858-ba65-44c1-8af1-f2fe69a977d9
```

JSON 応答

成功すると、200HTTPステータスコードと次のJSONレスポンスを受け取ります。

```
{
    "resourceType": "Patient",
    "active": true,
    "name": [
        {
            "use": "official",
            "family": "Doe",
            "given": [
                "Jane"
            ]
        },
        {
            "use": "usual",
            "given": [
                "Jane"
            ]
        }
    ],
    "gender": "female",
```

```
    "birthDate": "1966-09-01",
    "meta": {
        "lastUpdated": "2020-11-23T06:24:13.202Z"
    },
    "id": "2de04858-ba65-44c1-8af1-f2fe69a977d9"
}
```

を使用したリソースの更新 **PUT**

次の例は、 PUTを使用して、患者FHIRリソースタイプの患者に関する詳細を更新する方法を示しています。さらに、まだ作成されていないリソースに対してPUTリクエストを行うと、初期バージョンが作成されます。

リクエストは、リソースが更新された場合は200HTTPステータスコードを返します。新しいリソースが作成された場合は201HTTPステータスコードを返します。

> **Note**
>
> DocumentReference リソースタイプに対してPUTリクエストを行うと、既存の拡張機能は変更されません。代わりに、 は既存の拡張機能を含む新しい拡張機能をデータストア AWS HealthLake に追加します。がDocumentReferenceリソースタイプで自然言語処理 (NLP) HealthLake を使用して貴重な医療データを抽出する方法の詳細については、「」を参照してくださいのリソースタイプの自然言語処理 (NLP) に基づく自動FHIR DocumentReference リソース生成の使用 AWS HealthLake。

Example **PUT** リクエストを使用した**Patient**リソースタイプの更新

PUT リクエストを行うときは、データストアのエンドポイント、更新するリソースタイプの名前、識別子、リクエストJSONボディが必要です。

PUT を使用して新しいリソースを作成する場合、指定された識別子を使用して新しいリソースを作成します。

PUT Request

有効なPUTリクエストの構造例：

```
PUT https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient/2de04858-ba65-44c1-8af1-f2fe69a977d9
```

JSON Request Body

指定された患者リソースを更新するために使用されるJSON本文の例。

```
{
  "id": "2de04858-ba65-44c1-8af1-f2fe69a977d9",
  "resourceType": "Patient",
  "active": true,
  "name": [
    {
      "use": "official",
      "family": "Doe",
      "given": [
        "Jane"
      ]
    },
    {
      "use": "usual",
      "given": [
        "Jane"
      ]
    }
  ],
  "gender": "female",
  "birthDate": "1985-12-31"
}
```

JSONレスポンス

変更を確認する応答JSONとして、次の内容が表示されます。

```
{
 "id": "2de04858-ba65-44c1-8af1-f2fe69a977d9",
 "resourceType": "Patient",
 "active": true,
 "name": [{
   "use": "official",
   "family": "Doe",
   "given": [
    "Jane"
   ]
  },
```

```
  {
   "use": "usual",
   "given": [
    "Jane"
   ]
  }
 ],
 "gender": "female",
 "birthDate": "1985-12-31",
 "meta": {
  "lastUpdated": "2020-11-23T06:43:45.133Z"
 }
}
```

条件付き更新

条件付き更新では、論理 ID ではなく、一部の識別検索条件に基づいて既存のリソースを更新できます。サーバーはこの更新を処理するときに、このリクエストの 1 つの論理 ID を解決するために、リソースタイプの標準検索機能を使用して検索を実行します。

実行するアクションは、見つかった一致の数によって異なります。

- 一致なし、リクエストボディに ID が指定されていません: サーバーはリソースを作成します。
- 一致なし、提供された ID、リソースが ID にまだ存在しない: サーバーはインタラクションを「Create interaction」として更新として扱います。
- 一致なし、ID が指定され、既に存在: サーバーは`409 Conflict`エラーで更新を拒否します。
- 1 つの一致、リソース ID が指定されていない OR (リソース ID が指定され、見つかったリソースと一致する) : サーバーは上記のように、一致するリソースに対して更新を実行します。リソースが更新されると、サーバーは SHALLを返します`200 OK`。
- 1 つの一致、リソース ID が指定されましたが、リソースと一致しません: サーバーは、クライアント ID の仕様ができれば の問題であることを示す`409 Conflict`エラーを返します。 OperationOutcome
- 複数の一致: サーバーは、クライアントの基準が で十分に選択されていないことを示す`412 Precondition Failed`エラーを返します。 OperationOutcome

Example – 名前がペター、生年月日が 2000 年 1 月 1 日、電話番号 1234567890 である患者リソースを更新します。

```
PUT https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient?name=peter&birthdate=2000-01-01&phone=1234567890
```

を使用したリソースの削除 **DELETE**

HealthLake データストア内のリソースを削除するには、 DELETEHTTPリクエストを行う必要があります。

Example **DELETE** リクエストを使用して特定の**Patient**リソースタイプを削除する。

DELETE リクエストを作成するには、データストアのエンドポイントを使用します。データストアのエンドポイントを検索するには、 HealthLake コンソールの データストアで を参照するか、「 リファレンス」にある DescribeFHIRDatastore オペレーションを使用します。 AWS HealthLake API

また、リソースタイプと有効な識別子を含める必要があります。

```
DELETE https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient/2de04858-ba65-44c1-8af1-f2fe69a977d9
```

HTTPレスポンス

成功すると、リソースがデータストアになくなったことを確認する204HTTPステータスコードを受け取ります。削除リクエストが失敗すると、DELETEリクエストが失敗した理由を示す 400 シリーズのHTTPステータスコードを受け取ります。

Bundle を使用した複数のFHIRリソースの管理

HL7 FHIR R4 仕様では、バンドルは単なるリソースのコレクションです。 は、FHIRRESTAPI リクエストでの Bundle リソースタイプの作成と、バンドルトランザクションを使用した 1 回の FHIRRESTAPIリクエストで複数のCRUDオペレーションの実行 HealthLake をサポートしています。バンドルトランザクションでは、FHIRRESTAPIリクエストbatchでバンドルタイプを として指定する必要があります。

すべてのバンドルリクエストは によって記録されます AWS CloudTrail。 CloudTrail での の使用の詳細については HealthLake、「」を参照してくださいAWS CloudTrailを使用した AWS HealthLake APIコールのログ記録。

HL7 FHIR R4 リソース (外部)

- 完全な仕様については、 FHIRドキュメントインデックスの「リソースタイプ: バンドル」を参照してください。
- を使用したバッチインタラクションの詳細についてはAPI、 FHIRドキュメントインデックスのFHIRREST「 を使用したバッチインタラクション」を参照してください。 FHIR REST API

以下のセクションでは、新しい Bundle リソースを作成するか、バンドルトランザクションを使用してリソースを個別に処理するために、FHIRRESTAPIリクエストを構築する方法について説明します。

> ⚠ HealthLake コンソール、、 AWS CLIの違い AWS SDKs
>
> HealthLake コンソールは、リクエスト で Bundle リソースタイプが指定されている Bundle タイプのオペレーションのみをサポートしますFHIRRESTAPIURL。

FHIR バンドルを使用した複数のCRUDオペレーションの実行

リクエスト でリソースタイプが指定されていない場合URL、FHIRRESTAPIリクエストは個々のデータストアトランザクションとして解析されます。JSON 本文で提供される各CRUDオペレーションが評価され、特定のHTTPステータスコードが返されます。 はバンドルタイプ HealthLake をサポートしますbatch。

1 つのFHIRRESTAPIリクエストで複数のCRUDオペレーションを実行するには、次の手順を実行します。

次のリストは、バンドルリクエストで使用されるFHIRRESTAPIリクエストボディの切り捨てられた部分を示しています。リクエスト本文全体については、「複数のCRUDオペレーションを含むバンドルリクエストの作成」を参照してください。

1. POST リクエストでリソースタイプを指定しないでください。

```
POST https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
```

2. リクエストボディで、バンドルタイプを として指定します。 `"type": "batch"`
3. リクエストボディで、 `resource`キーで始まる各CRUDインタラクションのリソース固有のデータを指定します。

4. 各CRUDオペレーションは、次のようにリクエストボディrequestで として指定されます。

```
{ ...
  "request" : {
    "method" : "HTTP-VERB",
    "url" : "FHIR-RESOURCE-TYPE-URL"
  }
  ...
}
```

JSON レスポンスでは、リクエストで指定された各CRUDオペレーションHTTPのステータスコードを取得します。

HealthLake バンドルトランザクションの制限

- バンドル HealthLake の制限の詳細については、「」を参照してくださいAWS HealthLake エンドポイントとクォータ。

以下は、複数のオペレーションを含む Bundle CRUDオペレーションの例です。

Example – 複数のCRUDオペレーションを含む Bundle リクエストの作成。

複数のCRUDオペレーションを実行するFHIRRESTAPIリクエストを行うには、データストアエンドポイントを使用してPOSTリクエストを行い、JSONリクエスト本文を指定する必要があります。

データストアのエンドポイントは、 HealthLake コンソールの データストアで確認できます。または、 AWS HealthLake APIリファレンスの DescribeFHIRDatastore オペレーションを使用します。

POST Request

データストアのエンドポイントを使用してPOSTリクエストを行います。次のタブJSON「リクエスト本文」を使用して、リクエスト本文の必須要素を確認します。

```
POST https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
```

JSON Request Body

リクエストボディでは、次のキーと値のペアを、個々のCRUDリクエストに関する他のFHIRリソース固有のデータとともに指定する必要があります。最初の例は、必要な要素を強調した切り

捨てられたJSONリクエスト本文を示しています。2 番目の例は、JSONリクエスト本文全体を示しています。

```
{
  "resourceType": "Bundle",
  "id": "bundle-batch-operation",
  "meta": {
    "lastUpdated": "2014-08-18T01:43:30Z"
  },
  "type": "batch", ## Required
  "entry": [
    {
      ## CRUD Transaction - 1
      "resource": {
        "resourceType": "Patient",
        ...
      },
      "request": { ## Required
        "method": "POST",
        "url": "Patient"
      }
    },
    {
      ## CRUD Transaction - 2
      "resource": {
        "resourceType": "Medication",
        ...
      },
      "request": { ## Required
        "method": "POST",
        "url": "Medication"
      }
    }
  ]
}
```

新しい Patientおよび Medicationリソースタイプの作成を示す完全な例を次に示します。

```
{
  "resourceType": "Bundle",
  "id": "bundle-transaction",
  "meta": {
    "lastUpdated": "2014-08-18T01:43:30Z"
```

```
  },
  "type": "batch",
  "entry": [
    {
      "resource": {
        "resourceType": "Patient",
        "meta": {
          "lastUpdated": "2022-06-03T17:53:36.724Z"
        },
        "text": {
          "status": "generated",
          "div": "Some narrative"
        },
        "active": true,
        "name": [
          {
            "use": "official",
            "family": "Jackson",
            "given": [
              "Mateo",
              "James"
            ]
          }
        ],
        "gender": "male",
        "birthDate": "1974-12-25"
      },
      "request": {
        "method": "POST",
        "url": "Patient"
      }
    },
    {
      "resource": {
        "resourceType": "Medication",
        "id": "med0310",
        "contained": [
          {
            "resourceType": "Substance",
            "id": "sub03",
            "code": {
              "coding": [
                {
                  "system": "http://snomed.info/sct",
```

```
                  "code": "55452001",
                  "display": "Oxycodone (substance)"
                }
              ]
            }
          }
        ],
        "code": {
          "coding": [
            {
              "system": "http://snomed.info/sct",
              "code": "430127000",
              "display": "Oral Form Oxycodone (product)"
            }
          ]
        },
        "form": {
          "coding": [
            {
              "system": "http://snomed.info/sct",
              "code": "385055001",
              "display": "Tablet dose form (qualifier value)"
            }
          ]
        },
        "ingredient": [
          {
            "itemReference": {
              "reference": "#sub03"
            },
            "strength": {
              "numerator": {
                "value": 5,
                "system": "http://unitsofmeasure.org",
                "code": "mg"
              },
              "denominator": {
                "value": 1,
                "system": "http://terminology.hl7.org/CodeSystem/v3-
orderableDrugForm",
                "code": "TAB"
              }
            }
          }
```

```
        ]
      },
      "request": {
        "method": "POST",
        "url": "Medication"
      }
    }
  ]
}
```

JSON 応答

サンプルバンドルトランザクションで指定されたリソースの作成を確認するには、含まれるCRUDオペレーションごとに201作成済みHTTPステータスコードを取得します。CRUD オペレーションが失敗すると、個々のリクエストが失敗した理由を示す 400 シリーズHTTPのステータスになります。

```
{
  "resourceType": "Bundle",
  "type": "batch-response",
  "timestamp": "2022-06-15T01:31:34.300+00:00",
  "entry": [
    {
      "response": {
        "status": "201",
        "location": "Patient/fd68ce38-ba30-4459-9eeb-476ad9f4f4ca",
        "lastModified": "2022-06-15T01:31:34.180+00:00"
      }
    },
    {
      "response": {
        "status": "201",
        "location": "Medication/5bf3b8cc-4076-4219-aba1-e2c53d7916f4",
        "lastModified": "2022-06-15T01:31:34.180+00:00"
      }
    }
  ]
}
```

バンドルリソースタイプとしてのリソースのグループ化

新しい Bundle リソースタイプを作成するには、FHIRRESTAPIリクエストBundleで を指定し、グループ化するリソースを含む有効なJSON本文を指定する必要があります。

リクエスト で Bundle を指定するとURL、JSONリクエスト本文の内容はそのまま HealthLake データストアに保存されます。したがって、個々のリソースタイプに対してCRUDオペレーションを実行することはできません。このタイプのバンドルには、単一の新しいリソース ID が割り当てられます。リソースはそのまま保存されるため、 Bundle リソースタイプに保存されている個々のリソースに対して GETまたは POSTリクエストを行うことはできません。

> **Note**
>
> HL7 FHIR R4 仕様では、グループ、コンポジション、リストを使用したリソースのグループ化もサポートされています。これらのリソースタイプを作成する場合、個々のリソースは直接含まれません。代わりに、 Reference要素を使用して個々のリソースをポイントします。したがって、これらのリソースタイプを使用すると、それらに含まれる個々のリソースを変更できます。

Bundle リソースタイプを作成するには、POSTリクエストで指定し、含めるリソースを列挙する JSON を指定する必要があります。

Example – **POST**リクエストを使用した Bundle リソースの作成

bundle リソースを作成するには、次の手順を実行します。

1. FHIR REST API リクエストを次のようにフォーマットします。

   ```
   POST https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/Bundle
   ```

2. グループ化するリソースを指定するJSON本文を指定します。この例では、2 つの患者リソースをグループ化します。

```
{
  "resourceType": "Bundle",
  "id": "bundle-transaction",
  "meta": {
    "lastUpdated": "2018-03-11T11:22:16Z"
  },
  "type": "document",
  "entry": [
    {
      "resource": {
        "resourceType": "Patient",
```

```
        "name": [
          {
            "family": "Smith",
            "given": [
              "Jane"
            ]
          }
        ],
        "gender": "female",
        "address": [
          {
            "line": [
              "123 Main St."
            ],
            "city": "Anycity",
            "state": "Any State",
            "postalCode": "12345"
          }
        ]
      }
    },
    {
      "resource": {
        "resourceType": "Patient",
        "name": [
          {
            "family": "Jackson",
            "given": [
              "Mateo"
            ]
          }
        ],
        "gender": "male",
        "address": [
          {
            "line": [
              "1234 Main St."
            ],
            "city": "Anycity",
            "state": "Any State",
            "postalCode": "12345"
          }
        ]
      }
```

```
    }
  ]
}
```

FHIR REST API オペレーションを使用した HealthLake データストアの検索

HealthLake では、FHIR標準の一部として提供されるRESTAPIオペレーションを使用してデータストアを検索できます。このセクションでは、複数の異なるリソースタイプで GETおよび POSTリクエストを行う方法の例を示します。

> **Note**
>
> 個人を特定できる情報 (PII) または保護対象の医療情報 (PHI) を含むクエリでは、 POSTリクエストを使用することをお勧めします。POST リクエストでは、 PIIまたは PHIはリクエストボディの一部として追加され、転送中に暗号化されます。

このFHIR仕様は複数の検索パラメータタイプをサポートしていますが、サブセットのみ HealthLake をサポートしています。詳細については、サポートされている検索パラメータタイプおよびでサポートされている高度な検索パラメータ HealthLakeを参照してください。

FHIR REST API オペレーションを使用したデータストアの検索。

- サポートされている検索パラメータタイプ
- でサポートされている高度な検索パラメータ HealthLake
  - _include
  - _revinclude
  - _summary
  - _elements
  - _total
  - _sort
  - _count
  - Chaining and Reverse Chaining(_has)
- サポートされている検索修飾子

- サポートされている検索コンパレータ
- でサポートされていない検索パラメータ HealthLake
- POST 例で検索する
- GET 例で検索する

サポートされている検索パラメータタイプ

次の表は、 でサポートされている検索パラメータタイプを示しています HealthLake。

サポートされている検索パラメータタイプ

検索パラメータ	説明
_id	リソース ID (完全な ではないURL)
_lastUpdated	最終更新日。サーバーは境界精度に裁量があります。
タグ	リソースタグで検索します。
プロファイル	プロファイルでタグ付けされたすべてのリソースを検索します。
セキュリティ	このリソースに適用されているセキュリティラベルを検索します。
_ソース	リソースのソースを検索します。
テキスト	リソースの説明を検索します。
createdAt	カスタム拡張機能 - で検索しますcreatedAt。

> **Note**
>
> 次の検索パラメータは、2023 年 12 月 9 日以降に作成されたデータストアでのみサポートされます: _security、_source、_text、createdAt。

次の表は、特定のリソースタイプの指定されたデータ型に基づいてクエリ文字列を変更する方法の例を示しています。わかりやすくするために、例列の特殊文字はエンコードされていません。クエリを正常に実行するには、クエリ文字列が適切にエンコードされていることを確認します。

検索パラメータタイプ	詳細	例

数値	指定されたリソース内の数値を検索します。重要な数字が観察されます。
	
	有効桁数は、先頭の 0 を除く検索パラメータ値で に固有です。
	
	比較プレフィックスは許可されています。
		`[parameter]=100`
		
		`[parameter]=1e2`
		
		`[parameter]=lt100`

日付/DateTime	特定の日付または時刻を検索します。想定される形式は `yyyy-mm-ddThh:mm:ss[Z|(+|-)hh:mm]` ですが、異なる場合があります。
	
	、date、dateTime、instant よび のデータ型を受け入れますTiming。検索でこれらのデータ型を使用する詳細については、 FHIRドキュメントインデックスの「日付」を参照してください。
	
	比較プレフィックスは許可されています。
		`[parameter]=eq2013-01-14`
		
		`[parameter]=gt2013-01-14T10:00`
		
		`[parameter]=ne2013-01-14`

文字列	大文字と小文字を区別して一連の文字を検索します。 `HumanName` と の両方Addressのタイプをサ	`[base]/Patient?given=eve`

検索パラメータタイプ	詳細	例
	ポートします。詳細については、 FHIRドキュメントインデックスのHumanName「 データ型エントリ」とAddress「 データ型エントリ」を参照してください。 高度な検索は:text修飾子を使用してサポートされています。	[base]/Patient?given:contains=eve
トークン	多くの場合、 close-to-exact 医療コード値のペアと比較して、文字列に対する一致を検索します。 大文字と小文字の区別は、クエリの作成時に使用するコードシステムにリンクされます。サブミテーションベースのクエリは、大文字と小文字の区別に関連する問題を減らすのに役立ちます。わかりやすくするために、 |はエンコードされていません。	[parameter]=[system]|[code] ：ここでは、コーディングシステム[system]を参照し、その特定のシステム内で見つかったコード値[code]を参照します。 [parameter]=[code] ：ここで、入力はコードまたはシステムのいずれかに一致します。 [parameter]=|[code] ：ここで入力はコードと一致し、システムプロパティには識別子がありません。

検索パラメータタイプ	詳細	例
複合	修飾子$と ,オペレーションを使用して、1 つのリソースタイプ内の複数のパラメータを検索します。 比較プレフィックスは許可されています。	/Patient?language=FR,NL&language=EN Observation?component-code-value-quantity=http://loinc.org|8480-6$lt60 [base]/Group?characteristic-value=gender$mixed
数量	数値、システム、コードを値として検索します。番号は必須ですが、システムとコードはオプションです。数量データ型に基づきます。詳細については、FHIR「 ドキュメントインデックス」の「数量」を参照してください。 次の想定構文を使用します。[parameter]=[prefix][number]|[system]|[code]	[base]/Observation?value-quantity=5.4|http://unitsofmeasure.org|mg [base]/Observation?value-quantity=5.4|http://unitsofmeasure.org|mg [base]/Observation?value-quantity=5.4|http://unitsofmeasure.org|mg [base]/Observation?value-quantity=le5.4|http://unitsofmeasure.org|mg
リファレンス	他のリソースへの参照を検索します。	[base]/Observation?subject=Patient/23 test

検索パラメータタイプ	詳細	例
URI	特定のリソースを明確に識別する文字列を検索します。	`[base]/ValueSet?url=http://acme.org/fhir/ValueSet/123`
スペシャル	統合された医療NLP拡張機能に基づいて検索します。	

でサポートされている高度な検索パラメータ HealthLake

HealthLake では、次の高度な検索パラメータがサポートされています。

名前	説明	例	機能
`_include`	検索リクエストで追加のリソースが返されるようにリクエストするために使用されます。ターゲットリソースインスタンスによって参照されるリソースを返します。	`Encounter?_include=Encounter:subject`	
`_revinclude`	検索リクエストで追加のリソースが返されるようにリクエストするために使用されます。プライマリリソースインスタンスを参照するリソースが返されます。	`Patient?_id=` **`patient-identifier`** `&_revinclude=Encounter:patient`	
`_summary`	概要は、リソースのサブセットをリクエストするために使用できます。	`Patient?_summary=text`	次の概要パラメータがサポートされています: `_summary=true` 、`_summary=false` 、`_summary=text` 、`_summary=data` 。

名前	説明	例	機能
_elemen	検索結果のリソースの一部として返される特定の要素のセットをリクエストします。	Patient?_elements=identifier,active,link	
_total	検索パラメータに一致するリソースの数を返します。	Patient?_total=accurate	_total=accurate 、 をサポートします_total=none 。
_sort	カンマ区切りリストを使用して、返された検索結果のソート順を示します。- プレフィックスは、カンマ区切りリスト内の任意のソートルールに使用して、降順を示すことができます。	Observation?_sort=status,-date	タイプ のフィールドによるソートをサポートしますNumber, String, Quantity, Token, URI, Reference 。によるソートDateは、2023 年 12 月 9 日以降に作成されたデータストアでのみサポートされます。最大 5 つのソートルールをサポートします。
_count	検索バンドルのページごとに返されるリソースの数を制御します。	Patient?_count=100	最大ページサイズは 100 です。
chainin	参照されるリソースの要素を検索します。は、連鎖検索を、参照されるリソース内の要素.に送信します。	DiagnosticReport?subject:Patient.name=peter	

名前	説明	例	機能
reverse chainin (_has)	リソースを参照するリソースの要素に基づいてリソースを検索します。	Patient?_has:Observation:patient:code=1234-5	

_include

検索クエリ_includeで を使用すると、指定された追加のFHIRリソースも返されます。を使用して_include、前方にリンクされているリソースを含めます。

Example – **_include**を使用して、離脱症状がある患者または患者のグループを検索するには

リソースConditionタイプを検索し、その診断subjectの も返す_includeように指定します。Condition リソースタイプでは、患者リソースタイプまたはグループリソースタイプのいずれかsubjectを指します。

わかりやすくするために、この例の特殊文字はエンコードされていません。クエリを正常に実行するには、クエリ文字列が適切にエンコードされていることを確認します。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Condition?code=49727002&_include=Condition:subject
```

_revinclude

検索クエリ_revincludeで を使用すると、指定された追加のFHIRリソースも返されます。を使用して_revinclude、後方にリンクされたリソースを含めます。

Example - 特定の患者**_revinclude**にリンクされた関連する Encounter および Observation リソースタイプを含めるために を使用するには

この検索を行うには、まず_id検索パラメータで識別子を指定Patientして個人を定義します。次に、 構造Encounter:patientと を使用して追加のFHIRリソースを指定しますObservation:patient。

わかりやすくするために、この例の特殊文字はエンコードされていません。クエリを正常に実行するには、クエリ文字列が適切にエンコードされていることを確認します。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient?_id=patient-
identifier&_revinclude=Encounter:patient&_revinclude=Observation:patient
```

_summary

検索クエリ_summaryで を使用すると、ユーザーはFHIRリソースのサブセットをリクエストできます。次のいずれかの値を含めることができます: true, text, data, false。その他の値は無効として扱われます。返されたリソースは meta.tag 'SUBSETTED' で とマークされ、リソースが不完全であることを示します。

- true: リソースの基本定義で「概要」としてマークされているサポートされているすべての要素を返します (複数可) 。
- text:「text」、「id」、「meta」要素のみ、および最上位の必須要素のみを返します。
- data:「text」要素を除くすべての部分を返します。
- false: リソースのすべての部分を返します (複数可)

単一の検索リクエストでは、 を _includeまたは _revinclude 検索パラメータと組み合わせる_summary=textことはできません。

Example – データストア内の患者リソースの「テキスト」要素を取得します。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient?_summary=text
```

_elements

検索クエリ_elementsで を使用すると、特定のFHIRリソース要素をリクエストできます。返されたリソースは meta.tag 'SUBSETTED' で とマークされ、リソースが不完全であることを示します。

_elements パラメータは、リソースのルートレベルで定義された要素など、基本要素名のカンマ区切りリストで構成されます。リストされている要素のみが返されます。_elements パラメータ値に無効な要素が含まれている場合、サーバーはそれらを無視し、必須要素と有効な要素を返します。

_elements は、含まれているリソース (検索モードが の返されたリソースinclude) には適用されません。

単一の検索リクエストでは、 を検索_summaryパラメータと組み合わせる_elementsことはできません。

Example – HealthLake データストア内の患者リソースの「識別子」、「アクティブ」、「リンク」要素を取得します。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient?_elements=identifier,active,link
```

_total

検索クエリ_totalで を使用すると、リクエストされた検索パラメータに一致するリソースの数が返されます。 HealthLake は、検索レスポンスBundle.totalの で一致したリソース (検索モードが である返されたリソースmatch) の総数を返します。

_total は、accurate、 noneパラメータ値をサポートしています。 _total=estimateはサポートされていません。その他の値は無効として扱われます。 _totalは、含まれるリソース (検索モードが の返されたリソースinclude) には適用されません。

Example – データストア内の患者リソースの総数を取得します。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient?_total=accurate
```

_sort

検索クエリ_sortで を使用すると、結果が特定の順序で配置されます。結果は、ソートルールのカンマ区切りリストに基づいて優先順位が付けられます。ソートルールは有効な検索パラメータである必要があります。その他の値は無効として扱われます。

1 つの検索リクエストで、最大 5 つのソート検索パラメータを使用できます。オプションで、-プレフィックスを使用して降順を指定できます。デフォルトでは、サーバーは昇順でソートされます。

サポートされているソート検索パラメータタイプは、 ですNumber, String, Date, Quantity, Token, URI, Reference。検索パラメータがネストされた要素を参照している場合、この検索パラメータはソートではサポートされていません。例えば、リソースタイプの「名前」を検索する患者 は HumanName データ型の Patient.name 要素を参照し、ネストされたと見なされます。したがって、患者リソースを「名前」でソートすることはサポートされていません。

Example – データストアで患者リソースを取得し、生年月日で昇順にソートします。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient?_sort=birthdate
```

_count

パラメータ_countは、1 ページで返されるリソースの数に関するサーバーへの指示として定義されます。

最大ページサイズは 100 です。100 より大きい値は無効です。 _count=0はサポートされていません。

Example – 患者リソースを検索し、検索ページのサイズを 25 に設定します。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient?_count=25
```

Chaining and Reverse Chaining(_has)

での連鎖と逆連鎖FHIRにより、相互接続されたデータを取得するためのより効率的でコンパクトな方法が提供されるため、複数の個別のクエリが不要になり、デベロッパーやユーザーにとってデータの取得がより便利になります。

いずれかのレベルの再帰が 100 を超える結果を返した場合、 HealthLake は 4xx を返し、データストアが過負荷になり、複数のページ分割が発生するのを防ぎます。

Example – 連鎖 - 患者名がペターである患者を参照 DiagnosticReport するすべての を取得します。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
DiagnosticReport?subject:Patient.name=peter
```

Example — リバースチェーニング - 患者リソースを取得します。ここで、患者リソースは少なくとも 1 つの観測値によって参照され、観測値には 1234 のコードがあり、観測値は患者検索パラメータ内の患者リソースを指します。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient?_has:Observation:patient:code=1234
```

サポートされている検索修飾子

検索修飾子は文字列ベースのフィールドで使用されます。のすべての検索修飾子は、ブールベースのロジック HealthLake を使用します。たとえば、大きな文字列フィールドを検索結果に含めるために小さな文字列を含めるように:containsを指定できます。

サポートされている検索修飾子

検索修飾子	Type
: 欠落	を除くすべてのパラメータ `Composite`
:exact	文字列
:contains	文字列
: なし	トークン
:text	トークン
: 識別子	リファレンス

サポートされている検索コンパレータ

検索コンパレータを使用して、検索の一致の性質を制御できます。数値、日付、数量フィールドを検索するときにコンパレータを使用できます。次の表に、 でサポートされている検索コンパレータとその定義を示します HealthLake。

サポートされている検索コンパレータ

検索コンパレータ	説明
eq	リソースの パラメータの値は、指定された値と等しくなります。
ne	リソースの パラメータの値が、指定された値と等しくありません。
gt	リソースの パラメータの値が、指定された値を超えています。
lt	リソースの パラメータの値が、指定された値未満です。
G	リソースの パラメータの値は、指定された値以上です。

検索コンパレータ	説明
le	リソースの パラメータの値が、指定された値以下です。
sa	リソースの パラメータの値は、指定された値の後に開始されます。
EBS	リソースの パラメータの値は、指定された値より前に終了します。

でサポートされていない検索パラメータ HealthLake

サポートされている検索パラメータの完全なリストについては、FHIR「検索パラメータレジストリ」を参照してください。 HealthLake は、表に記載されているものを除くすべての検索パラメータをサポートしています。

サポートされていない検索パラメータ

バンドル構成	近くの場所
バンドル識別子	Consent-source-reference
バンドルメッセージ	契約患者
バンドルタイプ	リソースコンテンツ
バンドルタイムスタンプ	リソースクエリ

POST 例で検索する

POST リクエストを行うことで HealthLake データストアを検索できます。クエリパラメータは、URIまたはリクエストボディで指定できますが、両方を 1 つのリクエストで使用することはできません。

このトピックの例では、このベストプラクティスに従います。

> **Note**
>
> 個人を特定できる情報 (PII) または保護対象の医療情報 (PHI) を含むクエリでは、 POSTリクエストを使用することをお勧めします。POST リクエストでは、 PIIまたは PHIはリクエストボディの一部として追加され、転送中に暗号化されます。

POST リクエスト本文に パラメータを指定してリクエストを行う場合は、 ヘッダーContent-Type: application/x-www-form-urlencodedの一部として を使用します。

このトピックでは、次のリソースタイプを使用して POSTで検索する方法の例を示します。

- Age: Age は で定義されたリソースタイプではありませんFHIR。代わりに、経過時間は患者リソースタイプの一部としてキャプチャされます。特定の年齢または年齢範囲に基づいて患者のグループを検索するには、 を使用しますthe section called “サポートされている検索コンパレータ”。詳細については、 FHIRドキュメントインデックスの「リソースタイプ: 患者」を参照してください。
- 条件: このリソースタイプは、診断、状況、臨床状態、問題など、懸念のレベルに上昇した臨床概念に関連する詳細を保存します。詳細については、「 FHIRドキュメントインデックス」の「リソースタイプ: 条件」を参照してください。 は、 にあるドキュメントに基づいて新しい条件 HealthLake を作成します DocumentReference。これらの追加は、POSTリクエストを行うときにデフォルトで除外されます。これらを含めるには、検索で条件リソースに有効な識別子を指定する必要があります。
- DocumentReference: このリソースタイプは でサポートされています HealthLake。このリソースタイプは、任意のタイプのドキュメントの参照をサポートします。詳細については、「 FHIRドキュメントインデックス」の「リソースタイプ： DocumentReference」を参照してください。 は、 にあるドキュメントの統合自然言語処理 (NLP) HealthLake も提供します DocumentReference。詳細については、「のリソースタイプの自然言語処理 (NLP) に基づく自動FHIR DocumentReference リソース生成の使用 AWS HealthLake」を参照してください。
- 場所: このリソースタイプには、付随的な場所 (事前の指定や認可なしで医療に使用される場所) と正式に指定された専用の場所の両方が含まれます。詳細については、 FHIRドキュメントインデックスの「リソースタイプ: 場所」を参照してください。
- 観察: 患者、デバイス、またはその他のサブジェクトに関する測定と簡単なアサーション。は、リソースで見つかったドキュメントに基づいて新しい観察 DocumentReference リソース HealthLake を作成します。が新しいリソース HealthLake を作成する方法の詳細については、「」を参照してくださいのリソースタイプの自然言語処理 (NLP) に基づく自動FHIR DocumentReference リソース生成の使用 AWS HealthLake。これらの追加は、POSTリクエストを行うときにデフォルトで除外されます。これらを含めるには、検索で観測リソースの有効な識別子

を指定する必要があります。詳細については、 FHIRドキュメントインデックスの「リソースタイプ: 観測」を参照してください。

各タブには、指定されたリソースタイプで検索する方法の例が表示されます。これには、リクエスト本文でリクエストを指定する方法の例が含まれています。

Age

以下を使用して、Patientリソースタイプに対して POSTベースの検索リクエストを作成します。この検索では、 eq検索コンパレータを使用して、1997 年生まれの個人を検索します。

リクエストURLとリクエスト本文を指定する必要があります。リクエスト の例を次に示しますURL。

```
POST https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/Patient/_search
```

検索で 1997 年を指定するには、リクエストボディに次の要素を追加します。

```
birthdate=eq1997
```

JSON 応答

成功すると、200HTTPレスポンスコードと同様のJSONレスポンスが返されます。

Condition

以下を使用して、 Conditionリソースタイプに対してPOSTリクエストを行います。この検索では、医療コード を含む HealthLake データストア内の場所を検索します72892002。

リクエストURLとリクエスト本文を指定する必要があります。リクエスト の例を次に示しますURL。

```
POST https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/Condition/_search
```

検索する医療コードを指定するには、このJSON要素をリクエスト本文に追加します。

```
code=72892002
```

JSON 応答

成功すると、200HTTPレスポンスコードが返されます。わかりやすくするために、次のJSONレスポンスは切り捨てられています。


```
{
 "resourceType": "Bundle",
 "type": "searchset",
 "entry": [{
   "resource": {
    "resourceType": "Condition",
    "id": "0063326c-6b42-4d13-af2f-1efe0a65f016",
    "meta": {
     "lastUpdated": "2022-08-23T00:22:49.681Z"
    },
    "clinicalStatus": {
     "coding": [{
      "system": "http://terminology.hl7.org/CodeSystem/condition-clinical",
      "code": "resolved"
     }]
    },
    "verificationStatus": {
     "coding": [{
      "system": "http://terminology.hl7.org/CodeSystem/condition-ver-status",
      "code": "confirmed"
     }]
    },
    "code": {
     "coding": [{
      "system": "http://snomed.info/sct",
      "code": "72892002",
      "display": "Normal pregnancy"
     }],
     "text": "Normal pregnancy"
    },
    "subject": {
     "reference": "Patient/5fc0070a-696a-4855-94a9-175f1c641a33"
    },
    "encounter": {
     "reference": "Encounter/44078ab9-7ac7-4731-9ac8-4b3ff21a7bdb"
    },
    "onsetDateTime": "2019-08-15T01:19:17-07:00",
    "abatementDateTime": "2020-03-26T01:19:17-07:00",
    "recordedDate": "2019-08-15T01:19:17-07:00"
   },
```

```
  "search": {
   "mode": "match"
  }
 },
 {
  "resource": {
   "resourceType": "Condition",
   "id": "d00afdb2-1d2c-44fe-9f3b-033c0fe751a3",
   "meta": {
    "lastUpdated": "2022-08-23T00:20:47.100Z"
   },
   "clinicalStatus": {
    "coding": [{
     "system": "http://terminology.hl7.org/CodeSystem/condition-clinical",
     "code": "resolved"
    }]
   },
   "verificationStatus": {
    "coding": [{
     "system": "http://terminology.hl7.org/CodeSystem/condition-ver-status",
     "code": "confirmed"
    }]
   },
   "code": {
    "coding": [{
     "system": "http://snomed.info/sct",
     "code": "72892002",
     "display": "Normal pregnancy"
    }],
    "text": "Normal pregnancy"
   },
   "subject": {
    "reference": "Patient/d0a5cd1e-8da7-41bd-9b2f-41eef45246e5"
   },
   "encounter": {
    "reference": "Encounter/73758e67-4aaf-4e80-982b-8821f0b6fdfb"
   },
   "onsetDateTime": "2019-06-13T20:37:40-07:00",
   "abatementDateTime": "2020-01-23T19:37:40-08:00",
   "recordedDate": "2019-06-13T20:37:40-07:00"
  },
  "search": {
   "mode": "match"
  }
```

```
  }
 ]
}
```

DocumentReference

DocumentReference リソースタイプでPOSTリクエストを行うときに HealthLake、 の統合自然言語処理 (NLP) の結果を表示するには、次のようにリクエストをフォーマットします。

```
POST https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
DocumentReference/_search
```

参照する DocumentReference 要素を指定するには、「」を参照してください検索パラメータ。リクエストボディの を として指定しますJSON。

```
_lastUpdated=le2021-12-19&infer-icd10cm-entity-text-concept-score;=streptococcal|
0.6&infer-rxnorm-entity-text-concept-score=Amoxicillin|0.8
```

このクエリ文字列は、複数の検索パラメータを使用して、統合された医療NLP結果の生成に使用される Amazon Comprehend Medical APIオペレーションを検索します。

Location

Location リソースタイプに対してPOSTリクエストを行うには、以下を使用します。この検索では、住所の一部としてボストンという都市名を含む HealthLake データストア内の場所を検索します。

リクエストURLとリクエスト本文を指定する必要があります。リクエスト の例を次に示しますURL。

```
POST https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Location/_search
```

検索でボストンを指定するには、リクエストボディに次の要素を追加します。

```
address=Boston
```

JSON 応答

成功すると、200HTTPレスポンスコードが返されます。わかりやすくするために、JSONレスポンスは切り捨てられました。

```
{
 "resourceType": "Bundle",
 "type": "searchset",
 "entry": [{
   "resource": {
    "resourceType": "Location",
    "id": "0a6903c7-25c5-4ae4-8354-be88f9c5f2ee",
    "meta": {
     "lastUpdated": "2022-08-23T00:24:24.570Z"
    },
    "status": "active",
    "name": "BRIGHAM AND WOMEN'S HOSPITAL",
    "telecom": [{
     "system": "phone",
     "value": "6177325500"
    }],
    "address": {
     "line": [
      "75 FRANCIS STREET"
     ],
     "city": "BOSTON",
     "state": "MA",
     "postalCode": "02115",
     "country": "US"
    },
    "position": {
     "longitude": -71.020173,
     "latitude": 42.33196
    },
    "managingOrganization": {
     "reference": "Organization/27379046-608b-32f0-9df7-8c833cf5d11d",
     "display": "BRIGHAM AND WOMEN'S HOSPITAL"
    }
   },
   "search": {
    "mode": "match"
   }
  },

  {
   "resource": {
    "resourceType": "Location",
    "id": "ca5e7f65-4eb5-4bff-9a6f-07bc80acf8d0",
```

```
    "meta": {
     "lastUpdated": "2022-08-23T00:20:47.100Z"
    },
    "status": "active",
    "name": "BETH ISRAEL DEACONESS MEDICAL CENTER",
    "telecom": [{
     "system": "phone",
     "value": "6176677000"
    }],
    "address": {
     "line": [
      "330 BROOKLINE AVENUE"
     ],
     "city": "BOSTON",
     "state": "MA",
     "postalCode": "02215",
     "country": "US"
    },
    "position": {
     "longitude": -71.020173,
     "latitude": 42.33196
    },
    "managingOrganization": {
     "reference": "Organization/cb6a50e0-af76-3758-99ad-3200ede03fff",
     "display": "BETH ISRAEL DEACONESS MEDICAL CENTER"
    }
   },
   "search": {
    "mode": "match"
   }
  }
 ]
}
```

Observation

以下を使用して、Observationリソースタイプに対して POSTベースの検索リクエストを行います。この検索では、value-concept検索パラメータ を使用して医療コード を検索します266919005。このステータスは を示しますNever smoker。

リクエストURLとリクエスト本文を指定する必要があります。リクエスト の例を次に示しますURL。

```
POST https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Observation/_search
```

ステータスを指定するには、 Never smoker の本文value-concept=266919005に を設定しますJSON。

```
value-concept=266919005
```

JSON 応答

成功すると、200HTTPレスポンスコードが返されます。わかりやすくするために、次のJSONレスポンスは切り捨てられています。

```
{
 "resourceType": "Bundle",
 "type": "searchset",
 "link": [{
  "relation": "next",
  "url": "https://healthlake.us-west-2.amazonaws.com/
datastore/3651c6d3c1e81e785adba06b710b52a9/r4/Observation?value-
concept=266919005&=AAMA-
EFRSURBSGlpcGIyN250ZG9WRXVnTTFOdmtxQk9Bb3Y0YjhVcVdUMGV0eVozNmdjQU9nRjRNUUtscjhCZ1NMUG84VGNqN
 }],
 "entry": [{
   "resource": {
    "resourceType": "Observation",
    "id": "000038e0-71c6-4cc0-9c6c-50c8b1c53309",
    "meta": {
     "lastUpdated": "2022-11-03T01:02:38.981Z"
    },
    "status": "final",
    "category": [{
     "coding": [{
      "system": "http://terminology.hl7.org/CodeSystem/observation-category",
      "code": "survey",
      "display": "survey"
     }]
    }],
    "code": {
     "coding": [{
      "system": "http://loinc.org",
      "code": "72166-2",
```

```
      "display": "Tobacco smoking status NHIS"
     }],
     "text": "Tobacco smoking status NHIS"
    },
    "subject": {
     "reference": "Patient/598c9d7a-0494-448e-a81e-d50e3606e8db"
    },
    "encounter": {
     "reference": "Encounter/86bdee4a-2aa9-474a-b43f-6237cd68e512"
    },
    "effectiveDateTime": "2019-12-11T19:44:57-08:00",
    "issued": "2019-12-11T19:44:57.438-08:00",
    "valueCodeableConcept": {
     "coding": [{
      "system": "http://snomed.info/sct",
      "code": "266919005",
      "display": "Never smoker"
     }],
     "text": "Never smoker"
    }
   },
   "search": {
    "mode": "match"
   }
  },

  {
   "resource": {
    "resourceType": "Observation",
    "id": "0c2f6260-e671-4cfd-ac3d-e75f073fa3cd",
    "meta": {
     "lastUpdated": "2022-11-03T01:05:21.488Z"
    },
    "status": "final",
    "category": [{
     "coding": [{
      "system": "http://terminology.hl7.org/CodeSystem/observation-category",
      "code": "survey",
      "display": "survey"
     }]
    }],
    "code": {
     "coding": [{
      "system": "http://loinc.org",
```

```
      "code": "72166-2",
      "display": "Tobacco smoking status NHIS"
     }],
     "text": "Tobacco smoking status NHIS"
    },
    "subject": {
     "reference": "Patient/89d9a9b7-9720-4881-a2ab-d7907544b26f"
    },
    "encounter": {
     "reference": "Encounter/8ebba7b0-fdfc-4ec1-a9aa-907cccf60925"
    },
    "effectiveDateTime": "2018-11-17T03:59:36-08:00",
    "issued": "2018-11-17T03:59:36.550-08:00",
    "valueCodeableConcept": {
     "coding": [{
      "system": "http://snomed.info/sct",
      "code": "266919005",
      "display": "Never smoker"
     }],
     "text": "Never smoker"
    }
   },
   "search": {
    "mode": "match"
   }
  }
 ]
}
```

GET 例で検索する

データ HealthLake ストアを検索するには、 GET リクエストを実行します。 は、リクエスト本文の一部としてではなくURI、 の一部として HealthLake のみクエリパラメータを提供します。

> **Note**
>
> 個人を特定できる情報 (PII) または保護対象の医療情報 (PHI) を含むクエリでは、 POSTリクエストを使用することをお勧めします。POST リクエストでは、 PIIまたは PHIはリクエストボディの一部として追加され、転送中に暗号化されます。

このトピックでは、 でサポートされているリソースタイプGETを使用して で検索する方法の例を示します HealthLake。

- Age: Age は で定義されたリソースタイプではありませんFHIR。代わりに、経過時間は患者リソースタイプの一部としてキャプチャされます。特定の年齢または年齢範囲に基づいて患者のグループを検索するには、 を使用する必要がありますthe section called “サポートされている検索コンパレータ”。詳細については、 FHIRドキュメントインデックスの「リソースタイプ: 患者」を参照してください。
- 条件: このリソースタイプは、診断、状況、臨床状態、問題など、懸念のレベルに上昇した臨床概念に関連する詳細を保存します。詳細については、「 FHIRドキュメントインデックス」の「リソースタイプ: 条件」を参照してください。 は、 にあるドキュメントに基づいて新しい条件 HealthLake を作成します DocumentReference。これらの追加は、POSTリクエストを行うときにデフォルトで除外されます。これらを含めるには、検索で条件リソースに有効な識別子を指定する必要があります。
- DocumentReference: このリソースタイプは でサポートされています HealthLake。このリソースタイプは、任意のタイプのドキュメントの参照をサポートします。詳細については、「 FHIRドキュメントインデックス」の「リソースタイプ： DocumentReference」を参照してください。 は、 にあるドキュメントの統合自然言語処理 (NLP) HealthLake も提供します DocumentReference。詳細については、「のリソースタイプの自然言語処理 (NLP) に基づく自動 FHIR DocumentReference リソース生成の使用 AWS HealthLake」を参照してください。
- 場所: このリソースタイプには、付随的な場所 (事前の指定や認可なしで医療に使用される場所) と、正式に指定された専用の場所の両方が含まれます。詳細については、「 FHIRドキュメントインデックス」の「リソースタイプ: 場所」を参照してください。
- 観察: 患者、デバイス、またはその他のサブジェクトに関する測定と簡単なアサーション。は、リソースで見つかったドキュメントに基づいて新しい観察 DocumentReference リソース HealthLake を作成します。で新しいリソース HealthLake を作成する方法の詳細については、「」を参照してくださいのリソースタイプの自然言語処理 (NLP) に基づく自動FHIR DocumentReference リソース生成の使用 AWS HealthLake。これらの追加は、POSTリクエストを行うときにデフォルトで除外されます。これらを含めるには、検索で観測リソースの有効な識別子を指定する必要があります。詳細については、 FHIRドキュメントインデックスの「リソースタイプ: 観測」を参照してください。

各タブには、指定されたリソースタイプで検索する方法の例が表示されます。これには、 でリクエストを指定する方法の例とURI、関連するJSONレスポンスが含まれています。

Age

以下を使用して、Patientリソースタイプに対して GETベースの検索リクエストを作成します。この検索では、 eq検索コンパレータを使用して、1997 年生まれの個人を検索します。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4//
Patient?birthdate=eq1997
```

JSON 応答

成功すると、200HTTPレスポンスコードが返されます。

Condition

Condition リソースタイプに対してGETリクエストを行うには、以下を使用します。この検索では、医療コード を含む HealthLake データストア内の場所を検索します72892002。

リクエストURLとリクエスト本文を指定する必要があります。リクエスト の例を次に示しますURL。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Condition?code=72892002
```

JSON 応答

成功すると、200HTTPレスポンスコードが返されます。わかりやすくするために、次のJSONレスポンスは切り捨てられています。

```
{
 "resourceType": "Bundle",
 "type": "searchset",
 "entry": [{
   "resource": {
    "resourceType": "Condition",
    "id": "0063326c-6b42-4d13-af2f-1efe0a65f016",
    "meta": {
     "lastUpdated": "2022-08-23T00:22:49.681Z"
    },
    "clinicalStatus": {
     "coding": [{
      "system": "http://terminology.hl7.org/CodeSystem/condition-clinical",
      "code": "resolved"
     }]
```

```
    },
    "verificationStatus": {
     "coding": [{
      "system": "http://terminology.hl7.org/CodeSystem/condition-ver-status",
      "code": "confirmed"
     }]
    },
    "code": {
     "coding": [{
      "system": "http://snomed.info/sct",
      "code": "72892002",
      "display": "Normal pregnancy"
     }],
     "text": "Normal pregnancy"
    },
    "subject": {
     "reference": "Patient/5fc0070a-696a-4855-94a9-175f1c641a33"
    },
    "encounter": {
     "reference": "Encounter/44078ab9-7ac7-4731-9ac8-4b3ff21a7bdb"
    },
    "onsetDateTime": "2019-08-15T01:19:17-07:00",
    "abatementDateTime": "2020-03-26T01:19:17-07:00",
    "recordedDate": "2019-08-15T01:19:17-07:00"
   },
   "search": {
    "mode": "match"
   }
  },
  {
   "resource": {
    "resourceType": "Condition",
    "id": "d00afdb2-1d2c-44fe-9f3b-033c0fe751a3",
    "meta": {
     "lastUpdated": "2022-08-23T00:20:47.100Z"
    },
    "clinicalStatus": {
     "coding": [{
      "system": "http://terminology.hl7.org/CodeSystem/condition-clinical",
      "code": "resolved"
     }]
    },
    "verificationStatus": {
     "coding": [{
```

```
      "system": "http://terminology.hl7.org/CodeSystem/condition-ver-status",
      "code": "confirmed"
     }]
    },
    "code": {
     "coding": [{
      "system": "http://snomed.info/sct",
      "code": "72892002",
      "display": "Normal pregnancy"
     }],
     "text": "Normal pregnancy"
    },
    "subject": {
     "reference": "Patient/d0a5cd1e-8da7-41bd-9b2f-41eef45246e5"
    },
    "encounter": {
     "reference": "Encounter/73758e67-4aaf-4e80-982b-8821f0b6fdfb"
    },
    "onsetDateTime": "2019-06-13T20:37:40-07:00",
    "abatementDateTime": "2020-01-23T19:37:40-08:00",
    "recordedDate": "2019-06-13T20:37:40-07:00"
   },
   "search": {
    "mode": "match"
   }
  }
 ]
}
```

DocumentationReference

この例は、レンサ球状の診断を受け、アモキシアプレッションも処方された患者のDocumentReferenceリソースタイプで検索リクエストを作成する方法を示しています。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
DocumentReference?_lastUpdated=le2021-12-19&infer-icd10cm-entity-text-concept-
score;=streptococcal|0.6&infer-rxnorm-entity-text-concept-score=Amoxicillin|0.8
```

成功すると、次のJSONレスポンスが返されます。

```
{
    "resourceType": "Bundle",
    "type": "searchset",
```

```
    "entry": [
    {
       "resource": {
       "resourceType": "DocumentReference",
       "id": "985c3e94-4219-4c79-97a1-c94694525e24",
       "meta": {
          "lastUpdated": "2020-11-23T06:09:10.719Z"
        },
        "extension": [
          {
            "url": "http://healthlake.amazonaws.com/aws-cm/",
            "extension": [
              {
                "url": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/",
                "extension": [
                  {
                    "url": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/raw-
response",
                    "valueString": "{Entities: [{Id: 0,Text: otitis media,Category:
 MEDICAL_CONDITION,Type: DX_NAME,Score: 0.9815994,BeginOffset: 151,EndOffset:
 163,Attributes: [],Traits: [{Name: DIAGNOSIS,Score: 0.95042425}],ICD10CMConcepts:
 [{Description: Otitis media, unspecified, unspecified ear,Code: H66.90,Score:
 0.7176407}, {Description: Otitis media, unspecified,Code: H66.9,Score:
 0.6930445}, {Description: Otitis media, unspecified, left ear,Code: H66.92,Score:
 0.688161}, {Description: Otitis media, unspecified, bilateral,Code: H66.93,Score:
 0.6748094}, {Description: Otitis media, unspecified, right ear,Code:
 H66.91,Score: 0.6645618}]}, {Id: 1,Text: streptococcal sore throat,Category:
 MEDICAL_CONDITION,Type: DX_NAME,Score: 0.92208487,BeginOffset: 461,EndOffset:
 486,Attributes: [],Traits: [],ICD10CMConcepts: [{Description: Streptococcal
 pharyngitis,Code: J02.0,Score: 0.55638546}, {Description: Acute streptococcal
 tonsillitis, unspecified,Code: J03.00,Score: 0.53159785}, {Description:
 Streptococcal sepsis, unspecified,Code: A40.9,Score: 0.51865804}, {Description:
 Acute pharyngitis, unspecified,Code: J02.9,Score: 0.45085955}, {Description:
 Streptococcal infection, unspecified site,Code: A49.1,Score: 0.41550553}]},
 {Id: 3,Text: disorder,Category: MEDICAL_CONDITION,Type: DX_NAME,Score:
 0.9191257,BeginOffset: 488,EndOffset: 496,Attributes: [],Traits: [{Name:
 DIAGNOSIS,Score: 0.93372077}],ICD10CMConcepts: [{Description: Parkinson's
 disease,Code: G20,Score: 0.6959145}, {Description: Illness, unspecified,Code:
 R69,Score: 0.68428487}, {Description: Disorder of bone, unspecified,Code:
 M89.9,Score: 0.6542605}, {Description: Unspecified mental disorder due to known
 physiological condition,Code: F09,Score: 0.6240179}, {Description: Mental disorder,
 not otherwise specified,Code: F99,Score: 0.61046}]}],ModelVersion: 0.1.0}"
                  },
                  {
```

```
                    "url": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/
model-version",
                    "valueString": "0.1.0"
                  },
                  {
                    "url": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/aws-
cm-icd10-entity",
                    "extension": [
                      {
                        "url": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/
aws-cm-icd10-entity-id",
                        "valueInteger": 0
                      },
                      {
                        "url": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/
aws-cm-icd10-entity-text",
                        "valueString": "otitis media"
                      },
                      {
                        "url": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/
aws-cm-icd10-entity-begin-offset",
                        "valueInteger": 151
                      },
                      {
                        "url": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/
aws-cm-icd10-entity-end-offset",
                        "valueInteger": 163
                      },
                      {
                        "url": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/
aws-cm-icd10-entity-score",
                        "valueDecimal": 0.9815994
                      },
                      {
                        "url": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/
aws-cm-icd10-entity-ConceptList",
                        "extension": [
                          {
                            "url": "http://healthlake.amazonaws.com/aws-cm/infer-
icd10/aws-cm-icd10-entity-Concept",
                            "extension": [
                              {
                                "url": "http://healthlake.amazonaws.com/aws-cm/
infer-icd10/aws-cm-icd10-entity-Concept-Code",
```

```
                                "valueString": "H66.90"
                              },
                              {
                                "url": "http://healthlake.amazonaws.com/aws-cm/
infer-icd10/aws-cm-icd10-entity-Concept-Description",
                                "valueString": "Otitis media, unspecified,
 unspecified ear"
                              },
                              {
                                "url": "http://healthlake.amazonaws.com/aws-cm/
infer-icd10/aws-cm-icd10-entity-Concept-Score",
                                "valueDecimal": 0.7176407
                              }
                            ]
                          },
                          {
                            "url": "http://healthlake.amazonaws.com/aws-cm/infer-
icd10/aws-cm-icd10-entity-Concept",
                            "extension": [
                              {
                                "url": "http://healthlake.amazonaws.com/aws-cm/
infer-icd10/aws-cm-icd10-entity-Concept-Code",
                                "valueString": "H66.9"
                              },
                              {
                                "url": "http://healthlake.amazonaws.com/aws-cm/
infer-icd10/aws-cm-icd10-entity-Concept-Description",
                                "valueString": "Otitis media, unspecified"
                              },
                              {
                                "url": "http://healthlake.amazonaws.com/aws-cm/
infer-icd10/aws-cm-icd10-entity-Concept-Score",
                                "valueDecimal": 0.6930445
                              }
                            ]
                          },
                          {
                            "url": "http://healthlake.amazonaws.com/aws-cm/infer-
icd10/aws-cm-icd10-entity-Concept",
                            "extension": [
                              {
                                "url": "http://healthlake.amazonaws.com/aws-cm/
infer-icd10/aws-cm-icd10-entity-Concept-Code",
                                "valueString": "H66.92"
```

```
                    }
                ]
            }
```

Location

Location リソースタイプに対してGETリクエストを行うには、以下を使用します。この検索では、住所の一部としてボストンという都市名を含む HealthLake データストア内の場所を検索します。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4//
Location?address=boston
```

JSON 応答

成功すると、200HTTPレスポンスコードが返されます。わかりやすくするために、JSONレスポンスは切り捨てられました。

```
{
    "resourceType": "Bundle",
    "type": "searchset",
    "entry": [
        {
            "resource": {
                "resourceType": "Location",
                "id": "0a6903c7-25c5-4ae4-8354-be88f9c5f2ee",
                "meta": {
                    "lastUpdated": "2022-08-23T00:24:24.570Z"
                },
                "status": "active",
                "name": "BRIGHAM AND WOMEN'S HOSPITAL",
                "telecom": [
                    {
                        "system": "phone",
                        "value": "6177325500"
                    }
                ],
                "address": {
                    "line": [
                        "75 FRANCIS STREET"
                    ],
                    "city": "BOSTON",
```

```
                    "state": "MA",
                    "postalCode": "02115",
                    "country": "US"
                },
                "position": {
                    "longitude": -71.020173,
                    "latitude": 42.33196
                },
                "managingOrganization": {
                    "reference":
"Organization/27379046-608b-32f0-9df7-8c833cf5d11d",
                    "display": "BRIGHAM AND WOMEN'S HOSPITAL"
                }
            },
            "search": {
                "mode": "match"
            }
        },
        {
            "resource": {
                "resourceType": "Location",
                "id": "3cc3ad99-e0ff-48b4-b277-052abfc41058",
                "meta": {
                    "lastUpdated": "2022-08-23T00:19:37.029Z"
                },
                "status": "active",
                "name": "NEW ENGLAND BAPTIST HOSPITAL",
                "telecom": [
                    {
                        "system": "phone",
                        "value": "6177545800"
                    }
                ],
                "address": {
                    "line": [
                        "125 PARKER HILL AVENUE"
                    ],
                    "city": "BOSTON",
                    "state": "MA",
                    "postalCode": "02120",
                    "country": "US"
                },
                "position": {
                    "longitude": -71.020173,
```

```
                    "latitude": 42.33196
                },
                "managingOrganization": {
                    "reference": "Organization/9a7149fa-49fc-3c87-b935-
d29c55808717",
                    "display": "NEW ENGLAND BAPTIST HOSPITAL"
                }
            },
            "search": {
                "mode": "match"
            }
        },
        {
            "resource": {
                "resourceType": "Location",
                "id": "3f956715-3890-4235-85be-3fba5e3488ee",
                "meta": {
                    "lastUpdated": "2022-08-23T00:23:38.981Z"
                },
                "status": "active",
                "name": "MASSACHUSETTS GENERAL HOSPITAL",
                "telecom": [
                    {
                        "system": "phone",
                        "value": "6177262000"
                    }
                ],
                "address": {
                    "line": [
                        "55 FRUIT STREET"
                    ],
                    "city": "BOSTON",
                    "state": "MA",
                    "postalCode": "02114",
                    "country": "US"
                },
                "position": {
                    "longitude": -71.020173,
                    "latitude": 42.33196
                },
                "managingOrganization": {
                    "reference": "Organization/d78e84ec-30aa-3bba-a33a-
f29a3a454662",
                    "display": "MASSACHUSETTS GENERAL HOSPITAL"
```

```
                }
            },
            "search": {
                "mode": "match"
            }
        },
        {
            "resource": {
                "resourceType": "Location",
                "id": "6cc07b51-7287-443c-b772-c864f7831e13",
                "meta": {
                    "lastUpdated": "2022-08-23T00:21:11.045Z"
                },
                "status": "active",
                "name": "TUFTS MEDICAL CENTER",
                "telecom": [
                    {
                        "system": "phone",
                        "value": "6176365000"
                    }
                ],
                "address": {
                    "line": [
                        "800 WASHINGTON STREET"
                    ],
                    "city": "BOSTON",
                    "state": "MA",
                    "postalCode": "02111",
                    "country": "US"
                },
                "position": {
                    "longitude": -71.020173,
                    "latitude": 42.33196
                },
                "managingOrganization": {
                    "reference": "Organization/b7175ab4-
bde5-3848-891b-579bccb77c7c",
                    "display": "TUFTS MEDICAL CENTER"
                }
            },
            "search": {
                "mode": "match"
            }
        },
```

```
        {
            "resource": {
                "resourceType": "Location",
                "id": "8101300f-f685-49e7-b428-43b7855c39ee",
                "meta": {
                    "lastUpdated": "2022-08-23T00:22:06.474Z"
                },
                "status": "active",
                "name": "BOSTON CHILDREN'S HOSPITAL",
                "telecom": [
                    {
                        "system": "phone",
                        "value": "6177356000"
                    }
                ],
                "address": {
                    "line": [
                        "300 LONGWOOD AVENUE"
                    ],
                    "city": "BOSTON",
                    "state": "MA",
                    "postalCode": "02115",
                    "country": "US"
                },
                "position": {
                    "longitude": -71.020173,
                    "latitude": 42.33196
                },
                "managingOrganization": {
                    "reference": "Organization/d7b11827-25f2-350b-
bcd8-939fc59851b0",
                    "display": "BOSTON CHILDREN'S HOSPITAL"
                }
            },
            "search": {
                "mode": "match"
            }
        },
        {
            "resource": {
                "resourceType": "Location",
                "id": "8b7641d3-6997-48bb-bd60-23e35dfaae9d",
                "meta": {
                    "lastUpdated": "2022-08-23T00:20:47.099Z"
```

```
                },
                "status": "active",
                "name": "BRIGHAM AND WOMEN'S FAULKNER HOSPITAL",
                "telecom": [
                    {
                        "system": "phone",
                        "value": "6179837000"
                    }
                ],
                "address": {
                    "line": [
                        "1153 CENTRE STREET"
                    ],
                    "city": "BOSTON",
                    "state": "MA",
                    "postalCode": "02130",
                    "country": "US"
                },
                "position": {
                    "longitude": -71.020173,
                    "latitude": 42.33196
                },
                "managingOrganization": {
                    "reference": "Organization/d733d4a9-080d-3593-
b910-2366e652b7ea",
                    "display": "BRIGHAM AND WOMEN'S FAULKNER HOSPITAL"
                }
            },
            "search": {
                "mode": "match"
            }
        },
        {
            "resource": {
                "resourceType": "Location",
                "id": "998ef80b-7b58-4dc3-99ac-c440ec9e282d",
                "meta": {
                    "lastUpdated": "2022-08-23T00:21:11.046Z"
                },
                "status": "active",
                "name": "BRIGHAM AND WOMEN'S FAULKNER HOSPITAL",
                "telecom": [
                    {
                        "system": "phone",
```

```
                        "value": "6179837000"
                    }
                ],
                "address": {
                    "line": [
                        "1153 CENTRE STREET"
                    ],
                    "city": "BOSTON",
                    "state": "MA",
                    "postalCode": "02130",
                    "country": "US"
                },
                "position": {
                    "longitude": -71.020173,
                    "latitude": 42.33196
                },
                "managingOrganization": {
                    "reference": "Organization/d733d4a9-080d-3593-
b910-2366e652b7ea",
                    "display": "BRIGHAM AND WOMEN'S FAULKNER HOSPITAL"
                }
            },
            "search": {
                "mode": "match"
            }
        },
        {
            "resource": {
                "resourceType": "Location",
                "id": "c454bed3-7013-4376-81cf-4f49342f1402",
                "meta": {
                    "lastUpdated": "2022-08-23T00:24:24.573Z"
                },
                "status": "active",
                "name": "MASSACHUSETTS GENERAL HOSPITAL",
                "telecom": [
                    {
                        "system": "phone",
                        "value": "6177262000"
                    }
                ],
                "address": {
                    "line": [
                        "55 FRUIT STREET"
```

```
                    ],
                    "city": "BOSTON",
                    "state": "MA",
                    "postalCode": "02114",
                    "country": "US"
                },
                "position": {
                    "longitude": -71.020173,
                    "latitude": 42.33196
                },
                "managingOrganization": {
                    "reference": "Organization/d78e84ec-30aa-3bba-a33a-
f29a3a454662",
                    "display": "MASSACHUSETTS GENERAL HOSPITAL"
                }
            },
            "search": {
                "mode": "match"
            }
        },
        {
            "resource": {
                "resourceType": "Location",
                "id": "ca5e7f65-4eb5-4bff-9a6f-07bc80acf8d0",
                "meta": {
                    "lastUpdated": "2022-08-23T00:20:47.100Z"
                },
                "status": "active",
                "name": "BETH ISRAEL DEACONESS MEDICAL CENTER",
                "telecom": [
                    {
                        "system": "phone",
                        "value": "6176677000"
                    }
                ],
                "address": {
                    "line": [
                        "330 BROOKLINE AVENUE"
                    ],
                    "city": "BOSTON",
                    "state": "MA",
                    "postalCode": "02215",
                    "country": "US"
                },
```

```
                "position": {
                    "longitude": -71.020173,
                    "latitude": 42.33196
                },
                "managingOrganization": {
                    "reference": "Organization/cb6a50e0-
af76-3758-99ad-3200ede03fff",
                    "display": "BETH ISRAEL DEACONESS MEDICAL CENTER"
                }
            },
            "search": {
                "mode": "match"
            }
        }
    ]
}
```

Observation

以下を使用して、Observationリソースタイプに対して GETベースの検索リクエストを作成します。この検索では、value-concept検索パラメータ を使用して医療コード を検索します266919005。このステータスは を示しますNever smoker。

リクエストURLとクエリ文字列を指定する必要があります。リクエスト の例を次に示しますURL。

```
POST https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Observation?value-concept=266919005
```

ステータスを指定するには、 をクエリ文字列value-concept=266919005としてNever smoker設定します。

JSON 応答

成功すると、200HTTPレスポンスコードが返されます。わかりやすくするために、次のJSONレスポンスは切り捨てられています。

```
{
 "resourceType": "Bundle",
 "type": "searchset",
 "link": [{
  "relation": "next",
```

```
  "url": "https://healthlake.us-west-2.amazonaws.com/
datastore/3651c6d3c1e81e785adba06b710b52a9/r4/Observation?value-
concept=266919005&=AAMA-
EFRSURBSGlpcGIyN250ZG9WRXVnTTFOdmtxQk9Bb3Y0YjhVcVdUMGV0eVozNmdjQU9nRjRNUUtscjhCZ1NMUG84VGNqN
 }],
 "entry": [{
   "resource": {
    "resourceType": "Observation",
    "id": "000038e0-71c6-4cc0-9c6c-50c8b1c53309",
    "meta": {
     "lastUpdated": "2022-11-03T01:02:38.981Z"
    },
    "status": "final",
    "category": [{
     "coding": [{
      "system": "http://terminology.hl7.org/CodeSystem/observation-category",
      "code": "survey",
      "display": "survey"
     }]
    }],
    "code": {
     "coding": [{
      "system": "http://loinc.org",
      "code": "72166-2",
      "display": "Tobacco smoking status NHIS"
     }],
     "text": "Tobacco smoking status NHIS"
    },
    "subject": {
     "reference": "Patient/598c9d7a-0494-448e-a81e-d50e3606e8db"
    },
    "encounter": {
     "reference": "Encounter/86bdee4a-2aa9-474a-b43f-6237cd68e512"
    },
    "effectiveDateTime": "2019-12-11T19:44:57-08:00",
    "issued": "2019-12-11T19:44:57.438-08:00",
    "valueCodeableConcept": {
     "coding": [{
      "system": "http://snomed.info/sct",
      "code": "266919005",
      "display": "Never smoker"
     }],
     "text": "Never smoker"
    }
```

```
  },
  "search": {
   "mode": "match"
  }
 },

 {
  "resource": {
   "resourceType": "Observation",
   "id": "0c2f6260-e671-4cfd-ac3d-e75f073fa3cd",
   "meta": {
    "lastUpdated": "2022-11-03T01:05:21.488Z"
   },
   "status": "final",
   "category": [{
    "coding": [{
     "system": "http://terminology.hl7.org/CodeSystem/observation-category",
     "code": "survey",
     "display": "survey"
    }]
   }],
   "code": {
    "coding": [{
     "system": "http://loinc.org",
     "code": "72166-2",
     "display": "Tobacco smoking status NHIS"
    }],
    "text": "Tobacco smoking status NHIS"
   },
   "subject": {
    "reference": "Patient/89d9a9b7-9720-4881-a2ab-d7907544b26f"
   },
   "encounter": {
    "reference": "Encounter/8ebba7b0-fdfc-4ec1-a9aa-907cccf60925"
   },
   "effectiveDateTime": "2018-11-17T03:59:36-08:00",
   "issued": "2018-11-17T03:59:36.550-08:00",
   "valueCodeableConcept": {
    "coding": [{
     "system": "http://snomed.info/sct",
     "code": "266919005",
     "display": "Never smoker"
    }],
    "text": "Never smoker"
```

```
    }
   },
   "search": {
    "mode": "match"
   }
  }
 ]
}
```

FHIR リソース履歴の読み取り

FHIR history インタラクションは、 HealthLake データストア内の特定のFHIRリソースの履歴を取得します。このインタラクションを使用して、FHIRリソースのコンテンツが時間の経過とともにどのように変化したかを判断できます。また、監査ログと連携して、変更前後のリソースの状態を確認するのにも役立ちます。

> **Note**
>
> FHIR リソースhistoryは、10/25/2024 以降に作成されたすべての HealthLake データストアに対してデフォルトで有効になっています。データストアがこの日付より前に作成された場合は、サポートチケットを送信してFHIRhistoryインタラクションを有効にできます。を使用してケースを作成しますAWS Support Center Console。ケースを作成するには、 にログイン AWS アカウント し、ケースの作成を選択します。

history インタラクションは HTTP GET コマンドを使用して実行されます。FHIR インタラクション create、update、および は、保存されるリソースの履歴バージョンdeleteになります。は、FHIRhistoryインタラクションに対して次の検索パラメータ HealthLake をサポートします。

HealthLake がサポートするFHIR**history**インタラクションの検索パラメータ

検索パラメータ	説明
_count : integer	ページ上の検索結果の最大数。サーバーは、リクエストされた数またはデータストアでデフォルトで許可される検索結果の最大数のいずれか低い方を返します。

検索パラメータ	説明
_since : instant	特定の時点以降に作成されたリソースバージョンのみを含めます。
_at : date(Time)	日時値で指定された期間のある時点で最新であったリソースバージョンのみを含めます。詳細については、 HL7FHIRRESTfulAPIドキュメントdateの「」を参照してください。

次の例では、 のFHIRPatientリソースの 1 ページあたり 100 件の履歴検索結果を返します HealthLake。URL パス全体を表示するには、コピーボタンをスクロールします。URL の形式は次のとおりです。

```
GET https://healthlake.region.amazonaws.com/datastore/datastore-id/r4/Patient/id/
_history?_count=100
```

履歴インタラクションの戻りコンテンツはFHIRリソース に含まれBundle、タイプは に設定されますhistory。これには、指定されたバージョン履歴が含まれ、最後に最も古いバージョンでソートされ、削除されたリソースが含まれます。history インタラクションの詳細については、HL7FHIRRESTfulAPIドキュメントのhistory「」を参照してください。

> **Note**
>
> 特定のFHIRリソースタイプhistoryに対して をオプトアウトできます。オプトアウトするには、 を使用してケースを作成しますAWS Support Center Console。ケースを作成するには、 にログイン AWS アカウント し、ケースの作成を選択します。

バージョン固有のFHIRリソース履歴の読み取り

FHIR vread インタラクションは、 HealthLake データストア内のリソースのバージョン固有の読み取りを実行します。このインタラクションを使用すると、FHIRリソースの内容を過去の特定の時点と同じように表示できます。

HealthLake は、サポートされている各リソースの でのバージョニングCapabilityStatement.rest.resource.versioningのサポートを宣言します。すべての

HealthLake データストアには、すべてのリソースに Resource.meta.versionId (vid) が含まれます。

FHIR history インタラクションが有効になっている場合 (10/25/2024 以降に作成されたデータストアの場合はデフォルトで、古いデータストアの場合はリクエストによって)、Bundleレスポンスには vidの一部として が含まれますlocation。次の例では、 は番号 としてvid表示されます1。完全な例を確認するには、「バンドル/バンドルレスポンスの例 (JSON)」を参照してください。

```
"response" : {
    "status" : "201 Created",
    "location" : "Patient/12423/_history/1",
    ...}
```

vread インタラクションは HTTP GET コマンドを使用して実行されます。次のvreadインタラクションは、 で指定されたFHIRPatientリソースメタデータのバージョンについて、リソースに指定されたコンテンツを含む 1 つのインスタンスを返しますvid。次の例のURLパス全体を表示するには、コピーボタンをスクロールします。URL は 形式です。

```
GET https://healthlake.region.amazonaws.com/datastore/datastore-id/r4/Patient/id/
_history/vid
```

> **Note**
>
> FHIR リソースを読み取るvreadときに なしでhistoryインタラクションを使用する場合、HealthLake は常にリソースのメタデータの最新バージョンを返します。

vread インタラクションの詳細については、HL7FHIRRestful APIドキュメントvreadの「」を参照してください。

患者 $everything FHIRRESTAPIオペレーションを使用した患者データの取得

患者 $everything オペレーションは、FHIR患者リソースと、その患者に関連する他のリソースをクエリするために使用されます。このオペレーションは、患者に対してレコード全体へのアクセスを許可したり、患者に関連する一括データダウンロードをプロバイダーが実行したりするために使用できます。 は、特定の患者 ID に対してすべて $ HealthLake をサポートします。

> **Note**
>
> 現在、患者 $everything オペレーションは、2024 年 2 月 27 日以降に作成されたデータストアでサポートされています。

患者に関連するすべてのリソースを取得する

患者 $everything は、以下の例に示すように呼び出すことができるRESTAPIオペレーションです。

GET Request

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient/patient-id/$everything
```

> **Note**
>
> レスポンスのリソースは、リソースタイプとリソース ID でソートされます。
> レスポンスには常に Bundle.total が入力されます。

患者 $everything パラメータ

HealthLake は、次のクエリパラメータをサポートします。

パラメータ	詳細
start	指定された開始日以降にすべての患者データを取得します。
end	指定された終了日より前にすべての患者データを取得します。
since	指定された日付以降に更新されたすべての患者データを取得します。
_タイプ	特定のリソースタイプの患者データを取得します。
_カウント	患者データを取得し、ページサイズを指定します。

Example - 指定された開始日以降にすべての患者データを取得する

患者 $everything はstart、フィルターを使用して、特定の日付より後のデータのみをクエリできます。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient/patient-id/$everything?start=2024-03-15T00:00:00.000Z
```

Example - 指定された終了日より前にすべての患者データを取得する

患者 $everything はend、フィルターを使用して、特定の日付より前のデータのみをクエリできます。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient/patient-id/$everything?end=2024-03-15T00:00:00.000Z
```

Example - 指定された日付以降に更新されたすべての患者データを取得する

患者 $everything はsince、フィルターを使用して、特定の日付以降に更新されたデータのみをクエリできます。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient/patient-id/$everything?since=2024-03-15T00:00:00.000Z
```

Example - 特定のリソースタイプの患者データを取得する

患者 $everything は_typeフィルターを使用して、レスポンスに含める特定のリソースタイプを指定できます。複数のリソースタイプをカンマ区切りリストで指定できます。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient/patient-id/$everything?_type=Observation,Condition
```

Example - 患者データを取得し、ページサイズを指定する

患者 $すべて は _count を使用してページサイズを設定できます。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient/patient-id/$everything?_count=15
```

患者 $everything **start**と **end** 属性

HealthLake は、開始クエリパラメータと終了クエリパラメータに対して次のリソース属性をサポートします。

リソース	リソースエレメント
アカウント	Account.servicePeriod.start
AdverseEvent	AdverseEvent.date
AllergyIntolerance	AllergyIntolerance.recordedDate
予約	予約.開始
AppointmentResponse	AppointmentResponse.start
AuditEvent	AuditEvent.period.start
Basic (ベーシック)	Basic.created
BodyStructure	NO_DATE
CarePlan	CarePlan.period.start
CareTeam	CareTeam.period.start
ChargeItem	ChargeItem.occurrenceDateTime， ChargeItem.occurrencePeriod.start, ChargeItem.occurrenceTiming.event

リソース	リソースエレメント
Claim	Claim.billablePeriod.start
ClaimResponse	ClaimResponse.created
ClinicalImpression	ClinicalImpression.date
コミュニケーション	Communication.sent
CommunicationRequest	CommunicationRequest.occurrenceDateTime， CommunicationRequest.occurrencePeriod.start
コンポジション	Composition.date
条件	条件。recordedDate
同意	同意。dateTime
カバレッジ	Coverage.period.start
CoverageEligibilityRequest	CoverageEligibilityRequest.created
CoverageEligibilityResponse	CoverageEligibilityResponse.created
DetectedIssue	DetectedIssue.identified
DeviceRequest	DeviceRequest.authoredOn

リソース	リソースエレメント
DeviceUseStatement	DeviceUseStatement.recordedOn
DiagnosticReport	DiagnosticReport.effective
DocumentManifest	DocumentManifest.created
DocumentReference	DocumentReference.context.period.start
エンカウンター	Encounter.period.start
EnrollmentRequest	EnrollmentRequest.created
EpisodeOfCare	EpisodeOfCare.period.start
ExplanationOfBenefit	ExplanationOfBenefit.billablePeriod.start
FamilyMemberHistory	NO_DATE
フラグ	Flag.period.start
目標	目標。statusDate
グループ	NO_DATE
ImagingStudy	ImagingStudy.started

リソース	リソースエレメント
イミュナイゼーション	Immunization.recorded
ImmunizationEvaluation	ImmunizationEvaluation.date
ImmunizationRecommendation	ImmunizationRecommendation.date
Invoice	請求書.日付
リスト	リスト.日付
MeasureReport	MeasureReport.period.start
メディア	Media.issued
MedicationAdministration	MedicationAdministration.effective
MedicationDispense	MedicationDispense.whenPrepared
MedicationRequest	MedicationRequest.authoredOn
MedicationStatement	MedicationStatement.dateAsserted
MolecularSequence	NO_DATE
NutritionOrder	NutritionOrder.dateTime

リソース	リソースエレメント
監視結果	観察効果
患者	NO_DATE
個人	NO_DATE
手順	Procedure.performed
利点	Provenance.occurredPeriod.start、Provenance。occurredDateTime
QuestionnaireResponse	QuestionnaireResponse.authored
RelatedPerson	NO_DATE
RequestGroup	RequestGroup.authoredOn
ResearchSubject	ResearchSubject.period
RiskAssessment	RiskAssessment.occurrenceDateTime， RiskAssessment.occurrencePeriod.start
スケジュール	スケジュール。planningHorizon
ServiceRequest	ServiceRequest.authoredOn
舌	舌。receivedTime
SupplyDelivery	SupplyDelivery.occurrenceDateTime， SupplyDelivery.occurrencePeriod.start, SupplyDelivery.occurrenceTiming.event

リソース	リソースエレメント
SupplyRequest	SupplyRequest.authoredOn
VisionPrescription	VisionPrescription.dateWritten

$export を使用した HealthLake データストアからのデータのエクスポート

リクエスト$exportの一部として FHIR REST API 指定を使用してエクスポートPOSTリクエストを行い、リクエストの本文にリクエストパラメータを含めるには。FHIR 仕様に従って、FHIRサーバーは GETリクエストをサポートする必要があります。また、 は POSTリクエストをサポートできます。追加のパラメータをサポートするには、エクスポートを開始するのに本文が必要であるため、はPOSTリクエスト HealthLake をサポートします。

> **Important**
>
> HealthLake 2023 年 6 月 1 日より前に作成された データストアは、システム全体のエクスポートに対するFHIRRESTAPIエクスポートジョブリクエストのみをサポートします。HealthLake 2023 年 6 月 1 日より前に作成された データストアは、データストアのエンドポイントに対するGETリクエストを使用したエクスポートのステータスの取得をサポートしていません。

を使用して行うすべてのエクスポートリクエストFHIRRESTAPIは ndjson形式で返され、Amazon S3 バケットにエクスポートされます。各 S3 オブジェクトには、1 つのFHIRリソースタイプのみが含まれます。

AWS アカウントごとに一度に 1 つのエクスポートリクエストを行うことができます。に関連付けられている Service Quotas の詳細については HealthLake、「」を参照してくださいAWS HealthLake エンドポイントとクォータ。

を使用して をリクエストするエクスポートの作成の詳細についてはAPI、FHIRREST「」を参照してくださいFHIR REST API オペレーションを使用した HealthLake データストアからのデータのエクスポート。

Amazon Athena SQLで を使用して AWS HealthLake データストアをクエリする

HealthLake データストアを作成すると、高度にネストされたFHIRデータ構造が Amazon Athena に取り込まれ、 でクエリ可能な Iceberg テーブルに自動的に変換されますSQL。この新しいリソースへのアクセスの許可は、 AWS Lake Formation を使用して管理されます。各FHIRリソースタイプは、Athena の個々のテーブルとして表されます。

> **Important**
>
> 2022 年 11 月 14 日より前に作成されたデータストアの場合は、既存のデータストアを新しいデータストアに移行して、 を使用してクエリを実行する必要がありますSQL。ヘルプについては、「Amazon Athena を使用するための既存のデータストアの移行」を参照してください。

> **Note**
>
> 2023 年 2 月 20 日以降、 HealthLake データストアはデフォルトでは統合自然言語処理 (NLP) を使用しません。データストアでこの機能を有効にする場合は、トラブルシューティングの章 HealthLake統合された自然言語処理機能を有効にするにはどうすればよいですか？の「」を参照してください。

HealthLake データストアを作成するには、 HealthLake 管理者であるIAMユーザーまたはロールにIAMポリシーとサービスロールを追加する必要があります。アクセス許可の設定の詳細については、「」を参照してくださいの使用を開始するためのアクセス許可の設定 AWS HealthLake。

HealthLake データストアは Iceberg テーブルとして Athena に取り込まれます。Athena での Iceberg テーブルの機能の詳細については、「Athena ユーザーガイド」の「Iceberg テーブルの使用」を参照してください。

HealthLake は、Athena HealthLake のデータストアのデータストアのREADオペレーションをサポートします。オペレーションを使用した作成、読み取り、更新、削除 (CRUD) FHIRRESTAPIオペレーションの詳細については、「」を参照して、CRUDオペレーションが Athena のデータにどのように影響するか HealthLake データストアFHIRRESTAPIとのやり取りの使用を確認してください。

この章のトピックでは、 HealthLake データストアを Athena に接続する方法、 を使用してデータストアをクエリする方法SQL、結果を他の AWS サービスに接続してさらに分析する方法について説明します。

目次

データストアを Amazon Athena に接続する

> **Important**
>
> 2022 年 11 月 14 日以降、 へのアクセスIAM要件 HealthLake が変更されました。データストアを作成し、Athena でデータストアへのアクセスを許可するには、IAMユーザー、グループ、またはロールに `AWSLakeFormationDataAdmin`管理ポリシーを追加する必要があります。`AWSLakeFormationDataAdmin` ポリシーを使用してデータレイク管理者を作成し、Athena のデータストアへのアクセスを許可できます。

このトピックでは、Athena ユーザー、グループ、またはロールを作成し、 HealthLake データストアにあるFHIRリソースへのアクセス権を付与するために必要な手順の概要を説明します。

- ユーザー、グループ、またはロールに HealthLake データストアへのアクセスを許可する (AWS Lake Formation コンソール)
- Athena アカウントのセットアップ

ユーザー、グループ、またはロールに HealthLake データストアへのアクセスを許可する (AWS Lake Formation コンソール)

> ペルソナ： HealthLake 管理者
>
> HealthLake 管理者ペルソナは AWS Lake Formation のデータレイク管理者です。Lake Formation HealthLake のデータストアへのアクセスを許可します。

作成されたデータストアごとに、 AWS Lake Formation コンソールに 2 つのエントリが表示されます。1 つのエントリはリソースリンクです。リソースリンク名は常に斜体で表示されます。各リソースリンクは、リンクされた共有リソースの名前と所有者とともに表示されます。すべての HealthLake データストアで、共有リソース所有者が HealthLake サービスアカウントです。もう 1 つのエントリは、 HealthLake サービスアカウントの HealthLake データストアです。この手順のステップでは、リソースリンクであるデータストアを使用します。

リソースリンクの詳細については、Lake Formation デベロッパーガイドの「Lake Formation でのリソースリンクの仕組みAWS 」を参照してください。

ユーザー、グループ、またはロールが Athena でデータをクエリできるようにするには、リソースデータベースに対する Describe アクセス許可を付与する必要があります。次に、テーブルで Select と Describe を付与する必要があります。

STEP 1: HealthLake データストアリソースリンクデータベースに対するDESCRIBEアクセス許可を付与するには

1. AWS Lake Formation コンソールを開きます。 https://console.aws.amazon.com/lakeformation/
2. プライマリナビゲーションバーで、データベースを選択します。
3. データベースページで、斜体で表示されているデータストアの名前の横にあるラジオボタンを選択します。
4. アクション (▾) を選択します。
5. [Grant] (付与) を選択します。
6. データ許可の付与ページのプリンシパルで、IAMユーザーまたはロールを選択します。
7. IAM ユーザーまたはロールで、下矢印 (▾) を使用するか、Athena でクエリを実行できるようにするIAMユーザー、ロール、またはグループを検索します。
8. LF タグまたはカタログリソースカードで、名前付きデータカタログリソースオプションを選択します。

9. データベースで、下矢印 (▾) を使用して、アクセスを共有する HealthLake データストアデータベースを選択します。
10. リソースリンクのアクセス許可カードで、リソースリンクのアクセス許可で、説明を選択します。

許可が成功すると、許可を付与する成功バナーが表示されます。先ほど付与したアクセス許可を表示するには、データレイクのアクセス許可を選択します。テーブルでユーザー、グループ、ロールを見つけます。アクセス許可列の下に、説明が一覧表示されます。

次に、ターゲットでグラントを使用して、データベース内のすべてのテーブルで Select と Describe を付与する必要があります。

STEP 2: HealthLake データストアリソースリンク内のすべてのテーブルへのアクセスを許可する

1. AWS Lake Formation コンソールを開きます。 https://console.aws.amazon.com/lakeformation/
2. プライマリナビゲーションバーで、データベースを選択します。
3. データベースページで、斜体のデータストアの名前の横にあるラジオボタンを選択します。
4. アクション (▾) を選択します。
5. ターゲットの付与を選択します。
6. データ許可の付与ページのプリンシパルで、IAMユーザーまたはロールを選択します。
7. IAM ユーザーまたはロールで、下矢印 (▾) を使用するか、Athena でクエリを実行できるようにするIAMユーザー、グループ、またはロールを検索します。
8. LF タグまたはカタログリソースカードで、名前付きデータカタログリソースオプションを選択します。
9. データベースで、下矢印 (▾) を使用して、アクセスを許可する HealthLake データストアデータベースを選択します。
10. テーブルで、すべてのテーブルを選択して、すべてのテーブルを HealthLake ユーザーと共有します。
11. テーブルのアクセス許可カードで、テーブルのアクセス許可で、説明と選択を選択します。
12. [Grant] (付与) を選択します。

grant を選択すると、Grant permissions success バナーが表示されます。指定されたユーザーは、Athena HealthLake のデータストアに対してクエリを実行できるようになりました。

Athena の開始方法

> **HealthLake ユーザー**
>
> HealthLake ユーザーは Athena コンソール、または AWS SDKs を使用して AWS CLI、HealthLake 管理者によって共有された HealthLake データストアをクエリします。

Athena を使用してデータストアをクエリするには、次の 3 つの操作を行う必要があります。

- Lake Formation 経由で HealthLake データストアへのアクセスをIAMユーザーまたはロールに許可します。詳細については、「ユーザー、グループ、またはロールに HealthLake データストアへのアクセスを許可する (AWS Lake Formation コンソール) 」を参照してください。
- HealthLake データストアのワークグループを作成します。
- クエリ結果を保存する Amazon S3 バケットを指定します。

Athena の使用を開始するには、 ユーザー、グループ、またはロールに AmazonAthenaFullAccessおよび AmazonS3FullAccess AWS 管理ポリシーを追加します。 AWS マネージドポリシーの使用は、新しいサービスの使用を開始するのに最適な方法です。AWS マネージドポリシーは、すべての AWS のユーザーが使用できるため、特定のユースケースに対して最小特権のアクセス許可が付与されない場合があることに留意してください。IAM ポリシーでアクセス許可を設定する場合は、タスクの実行に必要なアクセス許可のみを付与します。IAM および最小特権の適用の詳細については、「 IAMユーザーガイド」の「最小特権のアクセス許可の適用」を参照してください。

> **Important**
>
> Athena で HealthLake データストアをクエリするには、Athena エンジンバージョン 3 を使用する必要があります。

ワークグループはリソースであるため、 IAMベースのポリシーを使用して特定のワークグループへのアクセスを制御できます。詳細については、「Athena ユーザーガイド」の「ワークグループを使用してクエリのアクセスとコストを制御する」を参照してください。

ワークグループの設定の詳細については、Athena ユーザーガイドhttps://docs.aws.amazon.com/athena/latest/ug/workgroups-procedure.htmlの「」を参照してください。

> **Note**
>
> Amazon S3 バケットがあるリージョンで、Athena コンソールが一致している必要があります。

クエリを実行する前に、Amazon S3 のクエリ結果バケットの場所を指定しておく、または指定されたバケットがあり、その設定がクライアント設定を上書きするワークグループを使用する必要があります。出力ファイルは、実行されるすべてのクエリに対して自動的に保存されます。

Athena コンソールでのクエリ結果の場所の指定の詳細については、「Amazon Athena ユーザーガイド」の「Athena コンソールを使用したクエリ結果の場所の指定」を参照してください。 Amazon Athena

Athena で HealthLake データストアをクエリする方法の例については、「」を参照してくださいを使用して HealthLake データストアをクエリする SQL。

を使用して HealthLake データストアをクエリする SQL

> **Note**
>
> 2023 年 2 月 20 日以降、 HealthLake データストアはデフォルトでは統合自然言語処理 (NLP) を使用しません。データストアでこの機能を有効にする場合は、トラブルシューティングの章 HealthLake統合された自然言語処理機能を有効にするにはどうすればよいですか？の「」を参照してください。
> このトピックのすべての例では、Synthea を使用して作成された架空のデータを使用します。Synthea データでプリロードされたデータストアの作成の詳細については、「」を参照してくださいでのデータストアの作成 AWS HealthLake。

HealthLake データストアを Athena にインポートすると、 HealthLake データストアの各リソースタイプがテーブルに変換されます。これらのテーブルは、個別にクエリすることも、 SQLベースのクエリを使用してグループとしてクエリすることもできます。データストアの構造のため、データは複数の異なるデータ型として Athena にインポートされます。これらのデータ型にアクセスできるSQLクエリの作成の詳細については、「Amazon Athena ユーザーガイド」の「複雑な型とネストされた構造を持つ配列のクエリ」を参照してください。 Amazon Athena

リソースタイプの各要素について、FHIR仕様はカーディナリティを定義します。要素の基数によって、この要素を表示できる回数の下限と上限が定義されます。SQL クエリを構築するときは、これを考慮する必要があります。例えば、リソースタイプ: 患者 のいくつかの要素を見てみましょう。

- 要素: Name FHIR仕様はカーディナリティを に設定します`0..*`。

  要素は配列としてキャプチャされます。

```
[{
 id = null,
 extension = null,
 use = official,
 _use = null,
 text = null,
 _text = null,
 family = Wolf938,
 _family = null,
 given = [Noel608],
 _given = null,
 prefix = null,
 _prefix = null,
 suffix = null,
 _suffix = null,
 period = null
}]
```

  Athena では、リソースタイプがどのように取り込まれたかを確認するには、テーブルとビューでリソースタイプを検索します。この配列の要素にアクセスするには、ドット表記を使用できます。`given` と の値にアクセスする簡単な例を次に示します`family`。

```
SELECT
    name[1].given as FirstName,
    name[1].family as LastName
FROM Patient
```

- 要素： MaritalStatus FHIR仕様はカーディナリティを に設定します`0..1`。

  この要素は としてキャプチャされますJSON。

```
{
 id = null,
 extension = null,
```

```
 coding = [
  {
   id = null,
   extension = null,
   system = http: //terminology.hl7.org/CodeSystem/v3-MaritalStatus,
    _system = null,
   version = null,
   _version = null,
   code = S,
   _code = null,
   display = Never Married,
   _display = null,
   userSelected = null,
   _userSelected = null
  }

 ],
 text = Never Married,
 _text = null
}
```

Athena では、リソースタイプがどのように取り込まれたかを確認するには、テーブルとビューでリソースタイプを検索します。のキーと値のペアにアクセスするにはJSON、ドット表記を使用できます。配列ではないため、配列インデックスは必要ありません。の値にアクセスする簡単な例を次に示しますtext。

```
SELECT
    maritalstatus.text as MaritalStatus
FROM Patient
```

へのアクセスと検索の詳細についてはJSON、「Athena ユーザーガイド」の「クエリJSON」を参照してください。

Athena データ操作言語 (DML) クエリステートメントは Trino に基づいています。Athena は Trino のすべての機能をサポートしているわけではなく、大きな違いがあります。詳細については、Amazon Athena ユーザーガイド」のDML「クエリ、関数、演算子」を参照してください。

さらに、Athena は、データストアのクエリを作成するときに発生する可能性のある複数の HealthLake データ型をサポートしています。Athena のデータ型の詳細については、Amazon Athena」の「Amazon Athena のデータ型」を参照してください。 Amazon Athena

Athena でのSQLクエリの動作の詳細については、SQLAmazon Athena」の「Amazon Athena のリファレンス」を参照してください。 Amazon Athena

各タブには、Athena を使用して指定されたリソースタイプと関連する要素を検索する方法の例が表示されます。

Element: Extension

要素は、データストアにカスタムフィールドを作成するextensionために使用されます。

この例では、 Patientリソースタイプにある extension要素の機能にアクセスする方法を示します。

HealthLake データストアが Athena にインポートされると、リソースタイプの要素の解析は異なります。の構造elementは可変であるため、スキーマで完全に指定することはできません。この変動性を処理するために、配列内の要素は文字列として渡されます。

のテーブルの説明にはPatient、 とextension記述されている 要素が表示されます。つまりarray<string>、インデックス値を使用して配列の要素にアクセスできます。ただし、文字列の要素にアクセスするには、 を使用する必要がありますjson_extract。

以下は、患者テーブルにある extension要素からの 1 つのエントリです。

```
[{
  "valueString": "Kerry175 Cummerata161",
  "url": "http://hl7.org/fhir/StructureDefinition/patient-mothersMaidenName"
 },
 {
  "valueAddress": {
   "country": "DE",
   "city": "Hamburg",
   "state": "Hamburg"
  },
  "url": "http://hl7.org/fhir/StructureDefinition/patient-birthPlace"
 },
 {
  "valueDecimal": 0.0,
  "url": "http://synthetichealth.github.io/synthea/disability-adjusted-life-years"
 },
 {
  "valueDecimal": 5.0,
  "url": "http://synthetichealth.github.io/synthea/quality-adjusted-life-years"
 }
```

```
]
```

これは有効な ですがJSON、Athena はそれを文字列として扱います。

このSQLクエリ例は、 要素patient-mothersMaidenNameと patient-birthPlace要素を含むテーブルを作成する方法を示しています。これらの要素にアクセスするには、異なる配列インデックスと json_extract.

```
SELECT
    extension[1],
    json_extract(extension[1], '$.valueString') AS MothersMaidenName,
    extension[2],
    json_extract(extension[2], '$.valueAddress.city') AS birthPlace
FROM patient
```

に関連するクエリの詳細についてはJSON、「Amazon Athena ユーザーガイド」の「 からのデータの抽出JSON」を参照してください。 Amazon Athena

Element: birthDate (Age)

Age は、 の患者リソースタイプの要素ではありませんFHIR。経過時間に基づいてフィルタリングする検索の 2 つの例を次に示します。

age は要素ではないため、SQLクエリbirthDateには を使用します。要素が にどのように取り込まれたかを確認するにはFHIR、テーブルとビューでテーブル名を検索します。文字列型であることがわかります。

例 1: 年齢の値の計算

このサンプルSQLクエリでは、組み込みSQLツールである current_dateと を使用して、これらのコンポーネントyearを抽出します。次に、これらを減算して、患者の実際の年齢を という列として返しますage。

```
SELECT
 (year(current_date) - year(date(birthdate))) as age
FROM patient
```

例 2: 以前に生まれ、 2019-01-01である患者のフィルタリングmale。

このSQLクエリでは、 CAST関数を使用してbirthDate要素をタイプ としてキャストする方法とDATE、 WHERE句の 2 つの条件に基づいてフィルタリングする方法を示します。要素はデフォ

ルトで型文字列として取り込まれるCASTため、型として取り込む必要がありますDATE。その後、 <演算子を使用して、別の日付 と比較できます2019-01-01。を使用するとAND、 WHERE句に 2 番目の条件を追加できます。

```
SELECT birthdate
FROM patient
-- we convert birthdate (varchar) to date  > cast that as date too
WHERE CAST(birthdate AS DATE) < CAST('2019-01-01' AS DATE) AND gender = 'male'
```

Resource type: Location

この例では、都市名が Attle™ である Location リソースタイプ内の場所を検索します。

```
SELECT *
FROM Location
WHERE address.city='ATTLEBORO'
LIMIT 10;
```

Element: Age

```
SELECT birthdate
FROM patient
-- we convert birthdate (varchar) to date  > cast that as date too
WHERE CAST(birthdate AS DATE) < CAST('2019-01-01' AS DATE) AND gender = 'male'
```

Resource type: Condition

リソースタイプ条件は、懸念レベルにまで達した問題に関連する診断データを保存します。HealthLakeの統合された医療自然言語処理 (NLP) はCondition、リソースタイプで見つかった詳細に基づいて新しい DocumentReference リソースを生成します。新しいリソースが生成されると、 HealthLake は タグを meta要素SYSTEM_GENERATEDに追加します。このサンプルSQLクエリは、条件テーブルを検索し、結果が削除されたSYSTEM_GENERATED場合に結果を返す方法を示しています。

HealthLakeの統合自然言語処理 (NLP) の詳細については、「」を参照してくださいのリソースタイプの自然言語処理 (NLP) に基づく自動FHIR DocumentReference リソース生成の使用 AWS HealthLake。

```
SELECT *
FROM condition
```

```
WHERE meta.tag[1] is NULL
```

指定された文字列要素内で検索して、クエリをさらにフィルタリングすることもできます。modifierextension 要素には、一連の条件の生成に使用されたDocumentReferenceリソースに関する詳細が含まれます。ここでも、 json_extractを使用して、文字列として Athena に取り込まれるネストされたJSON要素にアクセスする必要があります。

このサンプルSQLクエリは、特定の に基づいてCondition生成されたすべての を検索する方法を示していますDocumentReference。CAST を使用して JSON要素を文字列として設定し、LIKEを使用して比較できるようにします。

```
SELECT
    meta.tag[1].display as SystemGenerated,
    json_extract(modifierextension[4], '$.valueReference.reference') as
 DocumentReference
FROM condition
WHERE meta.tag[1].display = 'SYSTEM_GENERATED'

AND CAST(json_extract(modifierextension[4], '$.valueReference.reference') as
 VARCHAR) LIKE '%DocumentReference/67aa0278-8111-40d0-8adc-43055eb9d18d%'
```

Resource type: Observation

リソースタイプであるオブザベーションは、患者、デバイス、またはその他のサブジェクトに関する測定値と簡単なアサーションを保存します。 HealthLakeの統合自然言語処理 (NLP) は、Observationリソースで見つかった詳細に基づいて新しいDocumentReferenceリソースを生成します。このサンプルSQLクエリにはWHERE meta.tag[1] is NULL、コメントアウトされた が含まれます。つまり、SYSTEM_GENERATED結果が含まれます。

```
SELECT valueCodeableConcept.coding[1].code
FROM Observation
WHERE  valueCodeableConcept.coding[1].code = '266919005'
-- WHERE meta.tag[1] is NULL
```

この列は としてインポートされましたstruct。したがって、ドット表記を使用してその中の要素にアクセスできます。

Resource type: MedicationStatement

MedicationStatement は、患者が使用した、使用している、または今後使用する予定の薬剤に関する詳細を保存するために使用できるFHIRリソースタイプです。 HealthLakeの統合医療自然言

語処理 (NLP) は、 MedicationStatement リソースタイプで見つかったドキュメントに基づいて新しい DocumentReference リソースを生成します。新しいリソースが生成されると、 HealthLake は タグを meta要素SYSTEM_GENERATEDに追加します。このサンプルSQLクエリは、識別子を使用して単一の患者に基づいてフィルタリングし、 HealthLakeの統合 によって追加されたリソースを検索するクエリを作成する方法を示していますNLP。

```
SELECT *
FROM medicationstatement
WHERE meta.tag[1].display = 'SYSTEM_GENERATED' AND subject.reference =
 'Patient/0679b7b7-937d-488a-b48d-6315b8e7003b';
```

の統合医療 の詳細についてはNLP、 HealthLake「」を参照してくださいのリソースタイプの自然言語処理 (NLP) に基づく自動FHIR DocumentReference リソース生成の使用 AWS HealthLake。

複雑なフィルタリングを使用したSQLクエリの例

> **Note**
>
> 2023 年 2 月 20 日以降、 HealthLake データストアはデフォルトでは統合自然言語処理 (NLP) を使用しません。データストアでこの機能を有効にする場合は、トラブルシューティングの章 HealthLake統合された自然言語処理機能を有効にするにはどうすればよいですか？の「」を参照してください。

このトピックの例には、複雑なフィルタリングを使用する Athena と HealthLakeの統合に関するSQLクエリが含まれています。

Example 人口統計データに基づくフィルタリング条件の作成

患者コホートを作成するときは、正しい患者属性を特定することが重要です。このサンプルクエリは、Trino ドット表記 と json_extract を使用して HealthLake データストア内のデータをフィルタリングする方法を示しています。

```
SELECT
    id
    , CONCAT(name[1].family, ' ', name[1].given[1]) as name
    , (year(current_date) - year(date(birthdate))) as age
```

```
    , gender as gender
    , json_extract(extension[1], '$.valueString') as MothersMaidenName
    , json_extract(extension[2], '$.valueAddress.city') as birthPlace
    , maritalstatus.coding[1].display as maritalstatus
    , address[1].line[1] as addressline
    , address[1].city as city
    , address[1].district as district
    , address[1].state as state
    , address[1].postalcode as postalcode
    , address[1].country as country
    , json_extract(address[1].extension[1], '$.extension[0].valueDecimal') as latitude
    , json_extract(address[1].extension[1], '$.extension[1].valueDecimal') as longitude
    , telecom[1].value as telNumber
    , deceasedboolean as deceasedIndicator
    , deceaseddatetime
FROM database.patient;
```

Athena コンソールを使用すると、結果をさらにソートしてダウンロードできます。

Example 患者とその関連条件のフィルターの作成

このサンプルクエリは、 HealthLake データストアで見つかった患者のすべての関連条件を検索してソートする方法を示しています。

```
SELECT
 patient.id as patientId
    , condition.id  as conditionId
    , CONCAT(name[1].family, ' ', name[1].given[1]) as name
    , condition.meta.tag[1].display
    , json_extract(condition.modifierextension[1], '$.valueDecimal') AS confidenceScore
    , category[1].coding[1].code as categoryCode
    , category[1].coding[1].display as categoryDescription
    , code.coding[1].code as diagnosisCode
    , code.coding[1].display as diagnosisDescription
    , onsetdatetime
    , severity.coding[1].code as severityCode
    , severity.coding[1].display as severityDescription
    , verificationstatus.coding[1].display as verificationStatus
    , clinicalstatus.coding[1].display as clinicalStatus
    , encounter.reference as encounterId
    , encounter.type as encountertype
FROM database.patient, condition
WHERE CONCAT('Patient/', patient.id) = condition.subject.reference
```

```
ORDER BY name;
```

Athena コンソールを使用して、これらの結果をさらにソートしたり、ダウンロードしてさらに分析したりできます。

Example 患者とその関連する観察結果のフィルターの作成

このサンプルクエリは、 HealthLake データストアで見つかった患者の関連するすべての観測値を検索してソートする方法を示しています。

```
SELECT
 patient.id as patientId
    , observation.id as observationId
    , CONCAT(name[1].family, ' ', name[1].given[1]) as name
    , meta.tag[1].display
    , json_extract(modifierextension[1], '$.valueDecimal') AS confidenceScore
    , status
    , category[1].coding[1].code as categoryCode
    , category[1].coding[1].display as categoryDescription
    , code.coding[1].code as observationCode
    , code.coding[1].display as observationDescription
    , effectivedatetime
    , CASE
  WHEN valuequantity.value IS NOT NULL THEN CONCAT(CAST(valuequantity.value AS
 VARCHAR),' ',valuequantity.unit)
       WHEN valueCodeableConcept.coding [ 1 ].code IS NOT NULL THEN
 CAST(valueCodeableConcept.coding [ 1 ].code AS VARCHAR)
       WHEN valuestring IS NOT NULL THEN CAST(valuestring AS VARCHAR)
       WHEN valueboolean IS NOT NULL THEN CAST(valueboolean AS VARCHAR)
       WHEN valueinteger IS NOT NULL THEN CAST(valueinteger AS VARCHAR)
       WHEN valueratio IS NOT NULL THEN CONCAT(CAST(valueratio.numerator.value AS
 VARCHAR),'/',CAST(valueratio.denominator.value AS VARCHAR))
       WHEN valuerange IS NOT NULL THEN CONCAT(CAST(valuerange.low.value AS
 VARCHAR),'-',CAST(valuerange.high.value AS VARCHAR))
       WHEN valueSampledData IS NOT NULL THEN CAST(valueSampledData.data AS VARCHAR)
       WHEN valueTime IS NOT NULL THEN CAST(valueTime AS VARCHAR)
       WHEN valueDateTime IS NOT NULL THEN CAST(valueDateTime AS VARCHAR)
       WHEN valuePeriod IS NOT NULL THEN valuePeriod.start
       WHEN component[1] IS NOT NULL THEN CONCAT(CAST(component[2].valuequantity.value
 AS VARCHAR),' ',CAST(component[2].valuequantity.unit AS VARCHAR),
 '/', CAST(component[1].valuequantity.value AS VARCHAR),'
 ',CAST(component[1].valuequantity.unit AS VARCHAR))
    END AS observationvalue
```

```
 , encounter.reference as encounterId
    , encounter.type as encountertype
FROM database.patient, observation
WHERE CONCAT('Patient/', patient.id) = observation.subject.reference
ORDER BY name;
```

Example 患者のフィルタリング条件とその関連手順の作成

処置を患者に接続することは、医療の重要な側面です。このSQLクエリは、患者と処置のリソースタイプを使用して Athena でこれを行う方法を示しています。このSQLクエリは、 HealthLake データストアにあるすべての患者とその関連手順を返します。

```
SELECT
 patient.id  as patientId
 , PROCEDURE.id as procedureId
 , CONCAT(name[1].family, ' ', name[1].given[1]) as name
 , status
 , category.coding[1].code as categoryCode
 , category.coding[1].display as categoryDescription
 , code.coding[1].code as procedureCode
 , code.coding[1].display as procedureDescription
 , performeddatetime
 , performer[1]
 , encounter.reference as encounterId
 , encounter.type as encountertype
FROM database.patient, procedure
WHERE CONCAT('Patient/', patient.id) = procedure.subject.reference
ORDER BY name;
```

これで、Athena コンソールを使用して、結果をダウンロードしてさらに分析したり、結果をよりよく理解するためにソートしたりできます。

Example 患者とその関連する処方のフィルタリング条件の作成

患者が現在使用している薬剤のリストを確認することは重要です。Athena を使用すると、HealthLake データストアにある患者タイプと MedicationRequest リソースタイプの両方を使用するSQLクエリを記述できます。

このSQLクエリは、Athena にインポートされた患者と MedicationRequest テーブルを結合します。また、ドット表記を使用して、処方を個々のエントリに整理します。

```
SELECT
```

```
 patient.id  as patientId
 , medicationrequest.id  as medicationrequestid
 , CONCAT(name[1].family, ' ', name[1].given[1]) as name
 , status
 , statusreason.coding[1].code as categoryCode
 , statusreason.coding[1].display as categoryDescription
 , category[1].coding[1].code as categoryCode
 , category[1].coding[1].display as categoryDescription
 , priority
 , donotperform
 , encounter.reference as encounterId
 , encounter.type as encountertype
 , medicationcodeableconcept.coding[1].code as medicationCode
 , medicationcodeableconcept.coding[1].display as medicationDescription
 , dosageinstruction[1].text as dosage
FROM database.patient, medicationrequest
WHERE CONCAT('Patient/', patient.id ) = medicationrequest.subject.reference
ORDER BY name
```

Athena コンソールを使用して結果をソートしたり、ダウンロードして詳細な分析を行うことができます。

Example MedicationStatement リソースタイプで見つかった薬剤の表示

このクエリ例は、 を使用して Athena にJSONインポートされたネストされた を整理する方法を示していますSQL。クエリは、 meta要素を使用して、薬剤が の統合自然言語処理 () HealthLakeによっていつ追加されたかを示しますNLP。 HealthLakeと Amazon Comprehend Medical の統合の詳細については、「」を参照してくださいのリソースタイプの自然言語処理 (NLP) に基づく自動 FHIR DocumentReference リソース生成の使用 AWS HealthLake。また、 json_extractを使用してJSON文字列の配列内のデータを検索します。

```
SELECT
 medicationcodeableconcept.coding[1].code as medicationCode
 , medicationcodeableconcept.coding[1].display as medicationDescription
 , meta.tag[1].display
 , json_extract(modifierextension[1], '$.valueDecimal') AS confidenceScore
FROM medicationstatement;
```

Athena コンソールを使用して、これらの結果をダウンロードしたり、ソートしたりできます。

Example 特定の疾患タイプのフィルタリング

この例では、18 ~ 75 歳で、疾患にかかっている患者のグループを見つける方法を示します。

```
SELECT patient.id as patientId,
 condition.id as conditionId,
 CONCAT(name [ 1 ].family, ' ', name [ 1 ].given [ 1 ]) as name,
 (year(current_date) - year(date(birthdate))) AS age,
 CASE
  WHEN condition.encounter.reference IS NOT NULL THEN condition.encounter.reference
  WHEN observation.encounter.reference IS NOT NULL THEN observation.encounter.reference
 END as encounterId,
 CASE
  WHEN condition.encounter.type IS NOT NULL THEN observation.encounter.type
  WHEN observation.encounter.type IS NOT NULL THEN observation.encounter.type
 END AS encountertype,
 condition.code.coding [ 1 ].code as diagnosisCode,
 condition.code.coding [ 1 ].display as diagnosisDescription,
 observation.category [ 1 ].coding [ 1 ].code as categoryCode,
 observation.category [ 1 ].coding [ 1 ].display as categoryDescription,
 observation.code.coding [ 1 ].code as observationCode,
 observation.code.coding [ 1 ].display as observationDescription,
 effectivedatetime AS observationDateTime,
 CASE
      WHEN valuequantity.value IS NOT NULL THEN CONCAT(CAST(valuequantity.value AS
 VARCHAR),' ',valuequantity.unit)
      WHEN valueCodeableConcept.coding [ 1 ].code IS NOT NULL THEN
 CAST(valueCodeableConcept.coding [ 1 ].code AS VARCHAR)
      WHEN valuestring IS NOT NULL THEN CAST(valuestring AS VARCHAR)
      WHEN valueboolean IS NOT NULL THEN CAST(valueboolean AS VARCHAR)
      WHEN valueinteger IS NOT NULL THEN CAST(valueinteger AS VARCHAR)
      WHEN valueratio IS NOT NULL THEN CONCAT(CAST(valueratio.numerator.value AS
 VARCHAR),'/',CAST(valueratio.denominator.value AS VARCHAR))
      WHEN valuerange IS NOT NULL THEN CONCAT(CAST(valuerange.low.value AS
 VARCHAR),'-',CAST(valuerange.high.value AS VARCHAR))
      WHEN valueSampledData IS NOT NULL THEN CAST(valueSampledData.data AS VARCHAR)
      WHEN valueTime IS NOT NULL THEN CAST(valueTime AS VARCHAR)
      WHEN valueDateTime IS NOT NULL THEN CAST(valueDateTime AS VARCHAR)
      WHEN valuePeriod IS NOT NULL THEN valuePeriod.start
      WHEN component[1] IS NOT NULL THEN CONCAT(CAST(component[2].valuequantity.value
 AS VARCHAR),' ',CAST(component[2].valuequantity.unit AS VARCHAR),
 '/', CAST(component[1].valuequantity.value AS VARCHAR),'
 ',CAST(component[1].valuequantity.unit AS VARCHAR))
    END AS observationvalue,
```

```
 CASE
  WHEN condition.meta.tag [ 1 ].display = 'SYSTEM GENERATED' THEN 'YES'
  WHEN condition.meta.tag [ 1 ].display IS NULL THEN 'NO'
  WHEN observation.meta.tag [ 1 ].display = 'SYSTEM GENERATED' THEN 'YES'
  WHEN observation.meta.tag [ 1 ].display IS NULL THEN 'NO'
   END AS IsSystemGenerated,
  CAST(
    json_extract(
      condition.modifierextension [ 1 ],
      '$.valueDecimal'
    ) AS int
  ) AS confidenceScore
FROM database.patient,
 database.condition,
 database.observation
WHERE CONCAT('Patient/', patient.id) = condition.subject.reference
 AND CONCAT('Patient/', patient.id) = observation.subject.reference
   AND (year(current_date) - year(date(birthdate))) >= 18
   AND (year(current_date) - year(date(birthdate))) <= 75
   AND condition.code.coding [ 1 ].display like ('%diabetes%');
```

これで、Athena コンソールを使用して結果をソートしたり、ダウンロードしてさらに分析したりできます。

AWS HealthLake およびインターフェイスVPCエンドポイント (AWS PrivateLink)

インターフェイスVPCエンドポイントを作成 AWS HealthLake することで、 VPCと の間にプライベート接続を確立できます。インターフェイスVPCエンドポイントは、インターネットゲートウェイAWS PrivateLink、NATデバイス、VPN接続、または AWS Direct Connect 接続 HealthLakeAPIsなしで、プライベートアクセスに使用できるテクノロジーである を利用しています。のインスタンスは HealthLake、パブリック IP アドレスVPCがなくても と通信できますAPIs。VPC と の間のトラフィック HealthLake。 は Amazon ネットワークを離れません。

各インターフェースエンドポイントは、サブネット内の 1 つ以上の Elastic Network Interface によって表されます。

詳細については、「Amazon VPCユーザーガイド」の「インターフェイスVPCエンドポイント (AWS PrivateLink)」を参照してください。

エンドポイントに関する HealthLake VPC考慮事項

のインターフェイスVPCエンドポイントを設定する前に HealthLake、「Amazon VPCユーザーガイド」の「インターフェイスエンドポイントのプロパティと制限」を確認してください。

HealthLake は、 からのすべてのAPIアクションの呼び出しをサポートしますVPC。

のインターフェイスVPCエンドポイントの作成 HealthLake

Amazon VPCコンソールまたは AWS Command Line Interface () を使用して、 HealthLakeサービス用のVPCエンドポイントを作成できますAWS CLI。詳細については、「Amazon VPC ユーザーガイド」の「インターフェイスエンドポイントの作成」を参照してください。

次のサービス名を使用して HealthLake、 のVPCエンドポイントを作成します。

- com.amazonaws.*region*.healthlake

エンドポイントDNSのプライベートを有効にすると、リージョンのデフォルトDNS名を使用して にAPI HealthLakeリクエストを行うことができます。例えば、*healthlake.us-east-1.amazonaws.com* と指定します。

詳細については、「Amazon VPCユーザーガイド」の「インターフェイスエンドポイントを介したサービスへのアクセス」を参照してください。

のVPCエンドポイントポリシーの作成 HealthLake

へのアクセスを制御するエンドポイントポリシーをVPCエンドポイントにアタッチできます HealthLake。このポリシーでは、以下の情報を指定します。

- アクションを実行できるプリンシパル。
- 実行可能なアクション。
- アクションを実行できるリソース。

詳細については、「Amazon VPCユーザーガイド」のVPC「エンドポイントを使用した サービスへのアクセスの制御」を参照してください。

例: HealthLake アクションのVPCエンドポイントポリシー

以下は、 のエンドポイントポリシーの例です HealthLake。このポリシーは、エンドポイントにアタッチされると、すべてのリソースのすべてのプリンシパルに HealthLake`CreateFHIRDatastore`アクションへのアクセスを許可します。

```
{
   "Statement":[
      {
         "Principal":"*",
         "Effect":"Allow",
         "Action":[
            "healthlake:create-fhir-datastore"
         ],
         "Resource":"*"
      }
   ]
}
```

でのリソースのタグ付け AWS HealthLake

AWS のリソースにメタデータをタグ形式で割り当てることができます。各タグは、ユーザー定義のキーと値で構成されるラベルです。タグは、リソースの管理、識別、整理、検索、フィルタリングに役立ちます。

このトピックでは、一貫性のある効果的なタグ付け戦略を実装する目的で広く使用されているタグ付けカテゴリと戦略について説明します。以下のセクションでは、 AWSリソース、タグ付け、詳細な請求、AWSIdentity and Access Management () に関する基本的な知識を前提としていますIAM。

各 タグは 2 つの部分で構成されます:

- タグキー (環境 CostCenter、プロジェクトなど) 。タグキーでは、大文字と小文字が区別されます。
- タグ値 (111122223333 や Production など) 。タグキーと同様に、タグ値では大文字と小文字が区別されます。

タグを使用し、リソースを目的、所有者、環境などの基準別に分類できます。詳細については、「AWSタグ付け戦略」を参照してください。

タグは、各リソースのサービスコンソール、サービス 、または AWS から、一度に 1 つのリソースを追加、変更API、または削除できますCLI。

タグ付けを有効にするには、 TagResources が承認されていることを確認します。次の例のようなIAMポリシーをアタッチ TagResources することで、 を承認できます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "healthlake:CreateFHIRDatastore",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "healthlake:TagResource",
            "Resource": "*"
        }
    ]
```

```
}
```

重要な注意点

AWS HealthLake は、 責任AWS共有モデルポリシーに基づいて顧客データを保護します。つまり、すべての顧客データは移行中と保管中の両方で暗号化されます。ただし、データストアまたはジョブベースのオペレーションに顧客が入力したすべての名前が暗号化されるわけではありません。個人を特定できる情報や保護対象の医療情報を含めないでください。詳細については、 AWS HealthLake「セキュリティ」の章を参照してください。

ベストプラクティス

AWS リソースのタグ付け戦略を作成するときは、次のベストプラクティスに従ってください。

- 個人を特定できる情報 (PII) 、個人健康情報 (PHI) 、またはその他の機密情報をタグに保存しないでください。
- タグには、標準化された、大文字と小文字の区別がある形式を使用し、すべてのリソースタイプに一貫して適用します。
- リソースアクセスコントロールの管理、コスト追跡、オートメーション、整理など、複数の目的に対応したタグガイドラインを考慮します。
- 自動ツールを使用してリソースタグを管理します。 AWSResource Groups と Resource Groups Tagging API は、タグのプログラムによる制御を可能にし、タグとリソースを自動的に管理、検索、フィルタリングできるようにします。
- タグ付けは、より多くのタグを使用する場合により効果的です。
- タグはユーザーのニーズに応じて編集または変更できますが、アクセスコントロールタグを更新するには、それらのタグを参照するポリシーを更新してリソースへのアクセスを制御する必要があります。

タグ付け要件

タグには、次の要件があります。

- キーの前に aws: を付けることはできません。
- キーはタグセットごとに一意であることが必要です。
- キーに使用できる文字数は 1 ~ 128 文字です。

- 値に使用できる文字数は 0～256 文字です。
- 値は、タグセットごとに一意にする必要はありません。
- キーと値に使用できる文字は、Unicode 文字、数字、空白、および _ . : / = + - @ の記号です。
- キーと値は大文字と小文字が区別されます。

データストアへのタグの追加

データストアにタグを追加すると、AWSリソースを識別して整理し、リソースへのアクセスを管理するのに役立ちます。まず、データストアに 1 つ以上のタグ (キーと値のペア) を追加します。ユーザーごとに最大 50 個のタグを使用できます。キーフィールドと値フィールドで使用できる文字にも制限があります。

タグを取得したら、これらのタグに基づいてデータストアへのアクセスを管理するIAMポリシーを作成できます。 HealthLake コンソールまたは を使用して AWS CLI 、データストアにタグを追加できます。リポジトリにタグを追加すると、追加したリポジトリに影響が生じる場合があります。データストアにタグを追加する前に、タグを使用してデータストアなどのリソースへのアクセスを制御する可能性のあるIAMポリシーを必ず確認してください。

を使用して HealthLake データストアにタグ AWS CLI を追加するには、次の手順に従います。データストアの作成時にタグを追加するには、「」を参照してくださいでのデータストアの作成 AWS HealthLake。

ターミナルまたはコマンドラインで、tag-resource コマンドを実行し、タグを追加するデータストアの Amazon リソースネーム (ARN) と、追加するタグのキーと値を指定します。データストアには複数のタグを追加できます。キーフィールドと値フィールドで使用できる文字にも制限がありますタグ付け要件。たとえば、作成中にデータストアにタグを追加するには、 で次のコマンドを使用します AWS CLI。データストアの名前は Test_Data_Store で、キーで追加された 2 つのタグは key1 と key2 で、それぞれ value1 と value2 として値を持ちます。
:

```
aws healthlake create-fhir-datastore --datastore-type-version R4 --preload-data-config
 PreloadDataType="SYNTHEA" --datastore-name "Test_Data_Store" --tags '[{"Key": "key1",
 "Value": "value1"}, {"Key": "key2", "Value": "value2"}]' --region us-east-1
```

既存のデータストアにタグを追加するには、次のコマンド例を実行します。

```
aws healthlake tag-resource --resource-arn "arn:aws:healthlake:us-
east-1:691207106566:datastore/fhir/0725c83f4307f263e16fd56b6d8ebdbe" --tags '[{"Key":
 "key1", "Value": "value1"}]' --region us-east-1
```

成功した場合、このコマンドはレスポンスを返しません。

データストアのタグの一覧表示

を使用して HealthLake データストアのAWSタグのリスト AWS CLI を表示するには、次の手順に従います。タグが追加されていない場合、返されるリストは空になります。

ターミナルまたはコマンドラインで、次の例に示すように コマンドを実行します list-tags-for-resource。

```
aws healthlake-test list-tags-for-resource --resource-arn "arn:aws:healthlake:us-
east-1:674914422125:datastore/fhir/0725c83f4307f263e16fd56b6d8ebdbe" --region us-
east-1
```

```
{
    "tags": {
        "key": "value",
        "key1": "value1"
    }
}
```

データストアからのタグの削除

データストアに関連付けられた 1 つ以上のタグを削除できます。タグを削除しても、そのタグに関連付けられた他の AWS リソースからタグを削除することにはなりません。

ターミナルまたはコマンドラインで、untag-resource コマンドを実行し、タグを削除するデータストアの Amazon リソースネーム (ARN) と、削除するタグのタグキーを指定します。

```
aws healthlake untag-resource --resource-arn "arn:aws:healthlake:us-
east-1:674914422125:datastore/fhir/b91723d65c6fdeb1d26543a49d2ed1fa" --tag-keys
 '["key1"]' --region us-east-1
```

成功した場合、このコマンドはレスポンスを返しません。データストアに関連付けられているタグを確認するには、 コマンドを実行します list-tags-for-resource。

モニタリング HealthLake

モニタリングは、 およびその他の HealthLake AWSソリューションの信頼性、可用性、パフォーマンスを維持する上で重要な部分です。 AWSには、監視 HealthLake、問題発生時の報告、および必要に応じて自動アクションを実行するための以下のモニタリングツールが用意されています。

- Amazon CloudWatch は、 AWS リソースと、 で実行しているアプリケーションを AWS リアルタイムでモニタリングします。メトリクスの収集と追跡、カスタマイズされたダッシュボードの作成、指定したメトリクスが特定のしきい値 に達したときに通知したりアクションを実行したりするアラームの設定を行うことができます。例えば、 で Amazon EC2インスタンスのCPU使用状況やその他のメトリクス CloudWatch を追跡し、必要に応じて新しいインスタンスを自動的に起動できます。詳細については、「Amazon CloudWatch ユーザーガイド」を参照してください。
- AWS CloudTrail は、 アカウントによって、またはアカウントに代わって行われたAPI呼び出しおよび関連イベントをキャプチャします AWS 。次に、指定した Amazon S3 バケットにログファイルが渡されます。が呼び出したユーザーとアカウント AWS、それらの呼び出しの送信元 IP アドレス、およびいつ発生したかを特定できます。詳細については、AWS CloudTrail ユーザーガイドをご参照ください。

トピック

- Amazon HealthLake によるモニタリング CloudWatch

Amazon HealthLake によるモニタリング CloudWatch

HealthLake を使用して をモニタリングすることで CloudWatch、raw データを収集し、読み取り可能なほぼリアルタイムのメトリクスに加工できます。これらの統計は 15 か月間保持されるため、その履歴情報を利用して、ウェブアプリケーションまたはサービスの動作をより的確に把握できます。また、特定のしきい値を監視するアラームを設定し、これらのしきい値に達したときに通知を送信したりアクションを実行したりできます。詳細については、「Amazon CloudWatch ユーザーガイド」を参照してください。

メトリクスは HealthLake APIs、以下を含むすべての についてレポートされます。

- データストア管理 APIs — CreateFHIRDatastore、DeleteFHIRDatastore、DescribeFHIRDatastore、ListFHIRDatastores

- インポートとエクスポート APIs — S tartFHIRImportジョブ、L istFHIRImportジョブ、D escribeFHIRImportジョブ、S tartFHIRExportジョブ、L istFHIRExportジョブ、D escribeFHIRExportジョブ
- HTTP REST クライアントとリソースの管理 APIs — CreateResource、 DeleteResource、GetCapabilities ReadResource、 SearchAll SearchWithGet、 SearchWithPost、UpdateResource。
- タグ付け APIs — ListTagsForResource， TagResource， UntagResource

以下の表は、 HealthLake のメトリクスとディメンションの一覧です。

以下のメトリクスが報告されます。各 は、ユーザーが指定したデータ範囲の頻度カウントとして表示されます。

メトリクス

メトリクス	説明
コールカウント	への呼び出しの数APIs。これはアカウントまたは指定したデータストアについて報告できます。 単位: カウント 有効な統計: Sum、Count ディメンション: オペレーション、データストア ID、データストアタイプ
成功したリクエスト	成功したAPIリクエストの数。 単位: カウント 有効な統計: Sum、Average ディメンション: オペレーション、データストア、データストアタイプ
ユーザーエラー	ユーザーエラーが原因で失敗したリクエストの数。

メトリクス	説明
	単位: カウント 有効な統計: Sum、Average ディメンション: オペレーション、データストア ID、データストアタイプ
サーバーエラー	サーバーエラーのために失敗したリクエストの数。 単位: カウント 有効な統計: Sum、Average ディメンション: オペレーション、データストア ID、データストアタイプ
スロットルリクエスト	スロットリングされたリクエストの数。このメトリクスは、ユーザーまたはサーバーのエラー数に含まれません。 単位: カウント 有効な統計: Sum、Average ディメンション: オペレーション、データストア ID、データストアタイプ
レイテンシー	ユーザーリクエストの処理にかかったミリ秒単位の時間。 単位: ミリ秒 有効な統計: Minimum、Maximum、Average ディメンション: オペレーション、データストア ID、データストアタイプ

次のディメンションが報告されます。

ディメンション

ディメンション	説明
操作	使用されたAPIオペレーション
DataStoreID	API リクエストに含まれるデータストア
DataStoreType	データストアのタイプ (現在は FHIR R4 のみがサポートされています)

のメトリクスは、 AWSマネジメントコンソール、、 AWS CLIまたは HealthLake を使用して取得できます CloudWatch API。は、Amazon AWS Software Development Kit (SDKs) または CloudWatch API ツールのいずれかから使用できます CloudWatch API。 HealthLake コンソールには、 の raw データに基づくグラフが表示されます CloudWatch API。

HealthLake でモニタリングするには、適切な CloudWatch アクセス許可が必要です CloudWatch。詳細については、「Amazon ユーザーガイド」の「Amazon の認証とアクセスコントロール CloudWatch 」を参照してください。 CloudWatch

HealthLake メトリクスの表示

メトリクスを表示するには (CloudWatch コンソール)

1. AWS マネジメントコンソールにサインインし、CloudWatch コンソールを開きます。
2. メトリクスを選択し、すべてのメトリクスを選択し、AWS/HealthLake を選択します。
3. ディメンションを選択してメトリクスの名前を選んだら、グラフに追加 を選択します。
4. 日付範囲の値を選択します。選択した日付範囲のメトリクスカウントがグラフに表示されます。

を使用したアラームの作成 CloudWatch

CloudWatch アラームは、指定された期間にわたって単一のメトリクスを監視し、1 つ以上のアクションを実行します。Amazon Simple Notification Service (Amazon SNS) トピックまたは Auto Scaling ポリシーに通知を送信するアクションです。アクションは、指定した複数の期間における特定のしきい値に対するメトリクスの値に基づいています。 CloudWatch は、アラームの状態が変わったときに Amazon SNS メッセージを送信することもできます。

CloudWatch アラームは、状態が変わり、指定した期間保持された場合にのみアクションを呼び出します。

メトリクスを表示するには (CloudWatch コンソール)

1. AWS マネジメントコンソールにサインインし、CloudWatch コンソールを開きます。
2. [Alarms]、[Create Alarm] の順に選択します。
3. AWS/HealthLake を選択し、メトリクスを選択します。
4. [Time Range] (時間の範囲) で、モニタリングする期間を選択し、[Next] (次へ) を選択します。
5. [Name] (名前) と [Description] (説明) を入力します。
6. 常に >= を選択し、最大値を入力します。
7. アラーム状態に達したときに E メールを送信 CloudWatch する場合は、「アクション」セクションの「このアラームが発生するたびに状態が 」を選択しますALARM。通知の送信では、メーリングリストを選択するか、新しいリストを選択して新しいメーリングリストを作成します。
8. [Alarm Preview] (アラームの確認) セクションでアラームをプレビューします。アラームに問題がなければ、[Create Alarm] (アラームの作成) を選択します。

SMART FHIR と の統合 AWS HealthLake

FHIR 有効な HealthLake データストアの代替医療アプリケーションと再利用可能なテクノロジー (SMART) FHIR を使用すると、準拠アプリケーションSMARTは HealthLake データストアに保存されているデータにアクセスできます。 HealthLake データにアクセスするには、サードパーティーの認可サーバーを使用してリクエストを認証および承認し、 で追加のリソースを設定します AWS。

HealthLake データストアFHIRで SMARTで を使用するには、CreateFHIRDatastore APIリクエストで以下を指定する必要があります。

- を AuthorizationStrategyに設定しますSMART_ON_FHIR_V1。
- 認可サーバーでトークンデコードを管理するために AWS Lambda 作成した ARNのとIdpLambdaArn 等しい を設定します。
- 認可サーバーで指定されたメタデータ要素を定義します。これらのメタデータ要素は、 検出ドキュメントで返されます。詳細については、「FHIR 有効な HealthLake データストアの検出ドキュメントSMARTの の取得」を参照してください。
- オプション: 認可サーバーできめ細かな認可を設定FineGrainedAuthorizationEnabledしている場合は、 を有効にします。

FHIR 有効なデータストアSMARTで を作成するには、 AWS Command Line Interface （ AWS CLI) を使用するか、 AWS サポートされている のいずれかを使用しますSDKs。有効FHIRになっているHealthLake データストアSMARTでの の作成は、 HealthLake コンソールではサポートされていません。詳細については、「有効FHIRになっているデータストアSMARTで を作成する」を参照してください。

リクエストでこれらのパラメータを指定するには、他の AWS サービス (AWS Secrets Manager および AWS Lambda) でリソースをセットアップし、新しいIAMサービスロールを作成し、準拠した認可サーバーSMARTで FHIR を設定する必要があります。セクション「必要なリソースのセットアップ」を使用して、FHIR準拠データストアSMARTに を実装し、必要なリソースの設定の詳細とSMART、 on FHIR Application での のやり取りの概要を確認します HealthLake。

つまり、 経由でユーザー認証情報を管理する代わりに、準拠した認可サーバーSMARTで FHIR を使用して管理 AWS Identity and Access Management します。

HealthLake は 1.0 FHIR SMARTで をサポートしています。このフレームワークの詳細については、SMART「Application Launch Framework 実装ガイドリリース 1.0」を参照してください。

SMART で を使用してデータストアのリクエストを認可および認証するにはFHIR、 は以下の使用 HealthLake をサポートします。

- OpenID (AuthN) 統合: そのユーザーまたはクライアントアプリケーションが誰 (または何) であると主張するかを認証するために使用されます。
- OAuth 2.0 (AuthZ) 統合: 認証されたリクエストがデータを読み書きできる HealthLake データストア内のFHIRリソースを承認するために使用します。これは、認可サーバーで設定されたスコープによって定義されます。

目次

- FHIR 準拠のデータストアSMARTに を実装するために必要なリソースの設定
  - クライアントアプリケーションが SMARTでデータを起動し、 からデータをリクエストして HealthLake データストアFHIRを有効にする方法

SMART での の認証要件 FHIR

FHIR HealthLake データストアSMART上の のFHIRリソースにアクセスするには、クライアントアプリケーションが OAuth 2.0 準拠の認可サーバーによって認可され、FHIRRESTAPIリクエストの一部としてOAuthベアラートークンを提示する必要があります。認可サーバーのエンドポイントを見つけるには、 HealthLake SMARTWell-Known Uniform Resource Identifier を介して FHIR Discovery Document の を使用します。このプロセスについては、「FHIR 有効な HealthLake データストアの検出ドキュメントSMARTの の取得」を参照してください。

FHIR HealthLake データストアSMARTで を作成するときは、C reateFHIRDatastore リクエストの `metadata`要素で認可サーバーのエンドポイントとトークンエンドポイントを定義する必要があります。`metadata` 要素の定義の詳細については、「」を参照してください有効FHIRになっている HealthLake データストアSMARTでの の作成。

認可サーバーエンドポイントを使用して、クライアントアプリケーションは認可サービスでユーザーを認証します。承認および認証されると、JSONウェブトークン (JWT) が認可サービスによって生成され、クライアントアプリケーションに渡されます。このトークンには、クライアントアプリケーションが使用できるFHIRリソーススコープが含まれており、ユーザーがアクセスできるデータが制限されます。オプションで、起動スコープが指定されている場合、レスポンスにはそれらの詳細が含まれます。でサポートされているFHIRスコープSMARTの の詳細については HealthLake、「」を参照してくださいでサポートされるSMARTFHIROAuthスコープ HealthLake。

認可サーバーによってJWT付与された を使用して、クライアントアプリケーションはFHIR有効な HealthLake データストアSMARTで をFHIRRESTAPI呼び出します。を検証してデコードするには JWT、Lambda 関数を作成する必要があります。 は、FHIRRESTAPIリクエストを受信したときに、ユーザーに代わってこの Lambda 関数を HealthLake 呼び出します。スターター Lambda 関数の例については、「」を参照してください有効FHIRになっている HealthLake データストアSMARTでの でのトークン検証 AWS Lambda のための の使用。

FHIR 有効な HealthLake データストアSMARTで を作成するために必要な認可サーバー要素

C reateFHIRDatastore リクエストでは、 `IdentityProviderConfiguration` オブジェクトの `metadata`要素の一部として認可エンドポイントとトークンエンドポイントを指定する

必要があります。認可エンドポイントとトークンエンドポイントの両方が必要です。これを CreateFHIRDatastore リクエストで指定する方法の例については、「」を参照してください有効FHIRになっている HealthLake データストアSMARTでの の作成。

FHIR 有効な HealthLake データストアで FHIR REST API SMARTリクエストを完了するために必要なクレーム

FHIR 有効な HealthLake データストアSMARTの で有効なFHIRRESTAPIリクエストとなるには、AWS Lambda 関数に次のクレームが含まれている必要があります。

- nbf: (Not Before) Claim — 「nbf」(not Before) クレームJWTMUSTNOTは、 が処理のために受け入れられるまでの時間を識別します。「nbf」クレームの処理には、「nbf」クレームにdate/time MUST be after or equal to the not-before date/timeリストされている現在の が必要です。提供されているサンプル Lambda 関数は、サーバーレスポンスiatから に変換しますnbf。
- exp: (有効期限) クレーム - 「有効期限」 (有効期限) クレームは、 を処理に受け入れJWTることができない有効期限を識別します。
- isAuthorized: ブール値を に設定しますTrue。認可サーバーでリクエストが承認されたことを示します。
- aud: (対象者) クレーム — 「音声」 (対象者) クレームJWTは、 が対象となる受信者を識別します。これは、FHIR有効になっている HealthLake データストアエンドポイントSMARTの である必要があります。
- scope: これは、少なくとも 1 つのFHIRリソース関連のスコープである必要があります。このスコープは認可サーバーで定義されます。で受け入れられるFHIRリソース関連のスコープの詳細については HealthLake、「」を参照してくださいHealthLake データストアFHIRリソース固有のスコープ。

でサポートされるSMARTFHIROAuthスコープ HealthLake

HealthLake は認可プロトコルとして OAuth 2.0 を使用します。認可サーバーでこのプロトコルを使用すると、クライアントアプリケーションが読み書きアクセスを持つことができる HealthLake データストア内のFHIRリソースを定義できます。

on SMARTFHIRフレームワークは、認可サーバーからリクエストできる一連のスコープを定義します。FHIR フレームワークSMARTの でスコープ定義を表示するには、「 HL7FHIRリソースガイド」のSMARTFHIR「 スコープ」を参照してください。

例えば、患者によるテスト結果の表示や連絡先の詳細の表示のみを許可するように設計されたクライアントアプリケーションは、readスコープのリクエスト (FHIRRESTリクエスト経由) のみを許可する必要があります。これらをスコープとして定義するには、次のような文字列を指定しますpatient/Observation.read。これにより、クライアントアプリケーションは、Observationリソースタイプで読み取り専用方式で Patientリソースタイプへのアクセスをリクエストできます。

スタンドアロン起動スコープ

HealthLake はスタンドアロン起動モードのスコープ をサポートしていますlaunch/patient。

スタンドアロン起動モードでは、ユーザーと患者がクライアントアプリケーションに認識されていないため、クライアントアプリケーションは患者の臨床データへのアクセスをリクエストします。したがって、クライアントアプリケーションの認可リクエストでは、患者スコープが返されることを明示的に要求します。認証に成功すると、認可サーバーはリクエストされた起動患者スコープを含むアクセストークンを発行します。必要な患者コンテキストは、認可サーバーのレスポンスでアクセストークンとともに提供されます。

サポートされている起動モードスコープ

スコープ	説明
launch/patient	認可レスポンスで患者データが返されるように要求する OAuth 2.0 認可リクエストのパラメータ。

HealthLake データストアFHIRリソース固有のスコープ

HealthLake は 3 つのレベルのスコープを定義します。

- 患者固有のスコープは、1 人の患者に関する特定のデータへのアクセスを許可します。起動コンテキストで指定された患者。
- ユーザーレベルのスコープは、ユーザーがアクセスできる特定のデータへのアクセスを許可します。
- システムレベルのスコープは、 HealthLake データストアにあるすべてのFHIRリソースへの読み取り/書き込みアクセスを許可します。

次の表は、 でサポートされているFHIRリソース関連のスコープを構築するための構文を示しています HealthLake。一般的な形式は次のとおりです。

```
( 'patient' | 'user' | 'system' ) '/' ( fhir-resource | '*' ) '.' ( 'read' | 'write' |
 '*' )
```

HealthLake データストアでサポートされている認可スコープ

スコープ構文	スコープの例	結果
`patient/(fhir-resource | '*').('read' | 'write' | '*')`	`patient/AllergyIntolerance.*`	クライアントアプリケーションには、競合に対する読み取り/書き込みアクセス権があります。
`user/(fhir-resource | '*').('read' | 'write' | '*')`	`user/Observation.read`	クライアントアプリケーションは、記録されたすべての観測値への読み取りアクセス権を持ちます。
`system/('read' | 'write' | *)`	`system/*.*`	クライアントアプリケーションは、すべてのデータへの読み取り/書き込みアクセス権を持ちます。

有効FHIRになっている HealthLake データストアSMARTでの でのトークン検証 AWS Lambda のための の使用

FHIR 有効な HealthLake データストアSMARTで を作成するときは、 `CreateFHIRDatastore` リクエストで AWS Lambda 関数ARNの を指定する必要があります。Lambda 関数の ARNは、`IdpLambdaArn`パラメータを使用して `IdentityProviderConfiguration` オブジェクトで指定されます。

FHIR 有効な HealthLake データストアSMARTで を作成する前に、Lambda 関数を作成する必要があります。データストアを作成すると、Lambda を変更ARNすることはできません。データストアの作成時にARN指定した Lambda を表示するには、 `DescribeFHIRDatastore`APIオペレーションを使用します。

FHIR 有効な HealthLake データストアで SMARTのFHIRRESTリクエストが成功するには、Lambda 関数が以下を実行する必要があります。

- Lambda 関数は、 HealthLake データストアエンドポイントに 1 秒未満のレスポンスを返す必要があります。
- クライアントアプリケーションによって送信されたRESTAPIリクエストの認可ヘッダーで提供されるアクセストークンをデコードします。
- FHIR REST API リクエストを実行するのに十分なアクセス許可を持つ IAMサービスロールを割り当てます。
- FHIR REST API リクエストを完了するには、次のクレームが必要です。詳細については、「必要なクレーム」を参照してください。
  - `nbf`
  - `exp`
  - `isAuthorized`
  - `aud`
  - `scope`

Lambda を使用する場合は、Lambda 関数に加えて実行ロールとリソースベースのポリシーを作成する必要があります。Lambda 関数の実行ロールは、実行時に必要な AWS サービスおよびリソースにアクセスするためのアクセス許可を関数に付与する IAMロールです。指定するリソースベースのポリシーでは、 HealthLake がユーザーに代わって関数を呼び出すことを に許可する必要があります。

このトピックのセクションでは、クライアントアプリケーションからのリクエスト例とデコードされたレスポンス、 AWS Lambda 関数の作成に必要なステップ、および が引き受け HealthLake ることができるリソースベースのポリシーの作成方法について説明します。

- パート 1: Lambda 関数の作成
- パート 2: AWS Lambda 関数で使用される HealthLake サービスロールの作成
- パート 3: Lambda 関数の実行ロールの更新
- パート 4: Lambda 関数へのリソースポリシーの追加
- パート 5: Lambda 関数の同時実行のプロビジョニング

AWS Lambda 関数の作成

このトピックで作成された Lambda 関数は、 がFHIR有効な HealthLake データストアSMARTで へのリクエスト HealthLake を受信したときにトリガーされます。クライアントアプリケーションからのリクエストには、 REST API 呼び出しと、アクセストークンを含む認可ヘッダーが含まれています。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Authorization: Bearer i8hweunweunweofiwweoijewiwe
```

このトピックの Lambda 関数の例では AWS Secrets Manager 、 を使用して認可サーバーに関連する認証情報を隠します。Lambda 関数で認可サーバーのログインの詳細を直接指定しないことを強くお勧めします。

Example 認可ベアラートークンを含むFHIRRESTリクエストの検証

Lambda 関数の例は、FHIR有効な HealthLake データストアSMARTで に送信された FHIR REST リクエストを検証する方法を示しています。この Lambda step-by-steps関数の実装方法については、「」を参照してください[を使用した Lambda 関数の作成 AWS Management Console]。

FHIR REST API リクエストに有効なデータストアエンドポイント、アクセストークン、RESTオペレーションが含まれていない場合、Lambda 関数は失敗します。必要な認可サーバー要素の詳細については、「」を参照してください[必要なクレーム]。

```
import base64
import boto3
import logging
import json
import os
from urllib import request, parse

logger = logging.getLogger()
logger.setLevel(logging.INFO)

## Uses Secrets manager to gain access to the access key ID and secret access key for
 the authorization server
client = boto3.client('secretsmanager', region_name="region-of-datastore")
response = client.get_secret_value(SecretId='name-specified-by-customer-in-
secretsmanager')
secret = json.loads(response['SecretString'])
client_id = secret['client_id']
client_secret = secret['client_secret']
```

```
unencoded_auth = f'{client_id}:{client_secret}'
headers = {
  'Authorization': f'Basic {base64.b64encode(unencoded_auth.encode()).decode()}',
  'Content-Type': 'application/x-www-form-urlencoded'
}

auth_endpoint = os.environ['auth-server-base-url'] # Base URL of the Authorization
 server
user_role_arn = os.environ['iam-role-arn'] # The IAM role client application will use
 to complete the HTTP request on the datastore

def lambda_handler(event, context):
    if 'datastoreEndpoint' not in event or 'operationName' not in event or
 'bearerToken' not in event:
    return {}

    datastore_endpoint = event['datastoreEndpoint']
    operation_name = event['operationName']
    bearer_token = event['bearerToken']
    logger.info('Datastore Endpoint [{}], Operation Name:
 [{}]'.format(datastore_endpoint, operation_name))

    ## To validate the token
    auth_response = auth_with_provider(bearer_token)
    logger.info('Auth response: [{}]'.format(auth_response))
    auth_payload = json.loads(auth_response)
    ## Required parameters needed to be sent to the datastore endpoint for the HTTP
 request to go through
    auth_payload["isAuthorized"] = bool(auth_payload["active"])
    auth_payload["nbf"] = auth_payload["iat"]
    return {"authPayload": auth_payload, "iamRoleARN": user_role_arn}

## access the server
def auth_with_provider(token):
    data = {'token': token, 'token_type_hint': 'access_token'}
    req = request.Request(url=auth_endpoint + '/v1/introspect',
 data=parse.urlencode(data).encode(), headers=headers)
    with request.urlopen(req) as resp:
    return resp.read().decode()
```

を使用した Lambda 関数の作成 AWS Management Console

この手順では、FHIR有効な HealthLake データストアで のFHIRRESTAPIリクエストを処理するときに引き受けるサービスロールが既に作成されていること HealthLake を前提SMARTとしています。サービスロールを作成していない場合でも、Lambda 関数を作成できます。Lambda 関数が機能する前に、サービスロールARNの を追加する必要があります。サービスロールの作成と Lambda 関数での指定の詳細については、「」を参照してください。 のデコードに使用される AWS Lambda 関数で使用する HealthLake サービスロールの作成 JWT

Lambda 関数を作成するには (AWS Management Console)

1. Lambda コンソールの [関数ページ] を開きます。
2. [Create function (関数の作成)] を選択します。
3. [ゼロから作る] を選択します。
4. 基本情報 に関数名を入力します。Runtime で、Python ベースのランタイムを選択します。
5. [Execution role] (実行ロール) で [Create a new role with basic Lambda permissions] (基本的な Lambda アクセス権限で新しいロールを作成) を選択します。

   Lambda は、Amazon にログをアップロードするアクセス許可を関数に付与する実行ロールを作成します CloudWatch。Lambda 関数は、関数を呼び出すときに実行ロールを引き受け、実行ロールを使用して の認証情報を作成します AWS SDK。
6. コードタブを選択し、サンプルの Lambda 関数を追加します。

   Lambda 関数が使用するサービスロールをまだ作成していない場合は、サンプルの Lambda 関数が機能する前に作成する必要があります。Lambda 関数のサービスロールの作成の詳細については、「」を参照してくださいのデコードに使用される AWS Lambda 関数で使用する HealthLake サービスロールの作成 JWT。

```
import base64
import boto3
import logging
import json
import os
from urllib import request, parse

logger = logging.getLogger()
logger.setLevel(logging.INFO)
```

```
## Uses Secrets manager to gain access to the access key ID and secret access key
 for the authorization server
client = boto3.client('secretsmanager', region_name="region-of-datastore")
response = client.get_secret_value(SecretId='name-specified-by-customer-in-
secretsmanager')
secret = json.loads(response['SecretString'])
client_id = secret['client_id']
client_secret = secret['client_secret']


unencoded_auth = f'{client_id}:{client_secret}'
headers = {
  'Authorization': f'Basic {base64.b64encode(unencoded_auth.encode()).decode()}',
  'Content-Type': 'application/x-www-form-urlencoded'
}

auth_endpoint = os.environ['auth-server-base-url'] # Base URL of the Authorization
 server
user_role_arn = os.environ['iam-role-arn'] # The IAM role client application will
 use to complete the HTTP request on the datastore

def lambda_handler(event, context):
    if 'datastoreEndpoint' not in event or 'operationName' not in event or
 'bearerToken' not in event:
    return {}

    datastore_endpoint = event['datastoreEndpoint']
    operation_name = event['operationName']
    bearer_token = event['bearerToken']
    logger.info('Datastore Endpoint [{}], Operation Name:
 [{}]'.format(datastore_endpoint, operation_name))

    ## To validate the token
    auth_response = auth_with_provider(bearer_token)
    logger.info('Auth response: [{}]'.format(auth_response))
    auth_payload = json.loads(auth_response)
    ## Required parameters needed to be sent to the datastore endpoint for the HTTP
 request to go through
    auth_payload["isAuthorized"] = bool(auth_payload["active"])
    auth_payload["nbf"] = auth_payload["iat"]
    return {"authPayload": auth_payload, "iamRoleARN": user_role_arn}

## Access the server
def auth_with_provider(token):
```

```
    data = {'token': token, 'token_type_hint': 'access_token'}
    req = request.Request(url=auth_endpoint + '/v1/introspect',
data=parse.urlencode(data).encode(), headers=headers)
    with request.urlopen(req) as resp:
    return resp.read().decode()
```

Lambda 関数の実行ロールの変更

Lambda 関数を作成したら、Secrets Manager を呼び出すために必要なアクセス許可を含めるように実行ロールを更新する必要があります。Secrets Manager では、作成する各シークレットに があります ARN。最小権限を適用するには、実行ロールが Lambda 関数の実行に必要なリソースにのみアクセスできる必要があります。

Lambda 関数の実行ロールを変更するには、IAMコンソールで検索するか、Lambda コンソールで設定を選択します。Lambda 関数の実行ロールの管理の詳細については、「」を参照してください Lambda 実行ロール。

Example へのアクセスを許可する Lambda 関数実行ロール **GetSecretValue**

アクションを実行ロール IAM GetSecretValue に追加すると、サンプルの Lambda 関数が機能するために必要なアクセス許可が付与されます。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "arn:aws:secretsmanager:your-region:your-aws-account-
id:secret:secret-name-DKodTA"
    }
  ]
}
```

この時点で、FHIR有効な HealthLake データストア SMART で に送信された FHIR REST リクエストの一部として提供されるアクセストークンを検証するために使用できる Lambda 関数を作成しました。

のデコードに使用される AWS Lambda 関数で使用する HealthLake サービスロールの作成 JWT

ペルソナ: IAM 管理者

IAM ポリシーを追加または削除し、新しい ID IAM を作成できるユーザー。

サービスロール

サービスロールとは、サービスがユーザーに代わってアクションを実行するために引き受ける IAMロール です。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細については、「 IAMユーザーガイド」の「 にアクセス許可を委任するロールを作成する AWS のサービス」を参照してください。

JSON ウェブトークン (JWT) がデコードされると、認可 Lambda は IAMロール も返す必要がありますARN。このロールには、RESTAPIリクエストを実行するために必要なアクセス許可が必要です。そうしないと、アクセス許可が不十分であるために失敗します。

を使用してカスタムポリシーを設定するIAM場合は、必要な最小限のアクセス許可を付与することをお勧めします。詳細については、「 IAMユーザーガイド」の「最小特権のアクセス許可を適用する」を参照してください。

認可 Lambda 関数で指定する HealthLake サービスロールを作成するには、2 つのステップが必要です。

- まず、IAMポリシーを作成する必要があります。ポリシーは、認可サーバーでスコープを指定したFHIRリソースへのアクセスを指定する必要があります。
- 次に、サービスロールを作成する必要があります。ロールを作成するときは、信頼関係を指定し、ステップ 1 で作成したポリシーをアタッチします。信頼関係は、 をサービスプリンシパル HealthLake として指定します。このステップでは、 HealthLake データストアARNと AWS アカウント ID を指定する必要があります。

新しいIAMポリシーの作成

認可サーバーで定義するスコープによって、認証されたユーザーが HealthLake データストア内のどのFHIRリソースにアクセスできるかが決まります。

作成したIAMポリシーは、定義したスコープに合わせて調整できます。

IAM ポリシーステートメントの Action要素で、次のアクションを定義できます。テーブルAction内の各 に対して、 を定義できますResource types。データストア HealthLake では、アクセスIAM許可ポリシーステートメントの Resource要素で定義できる唯一のサポートされているリソースタイプです。

個々のFHIRリソースは、IAMアクセス許可ポリシーの要素として定義できるリソースではありません。

で定義されるアクション HealthLake

アクション	説明	アクセスレベル	リソースタイプ (必須)
CreateResource	リソースの作成にアクセス許可を付与	書き込み	データストア ARN: arn:aws:healthlake:**your-region** : **111122223333** :datastore/fhir/**your-datastore-id**
DeleteResource	リソースを削除する許可を付与	書き込み	データストア ARN: arn:aws:healthlake:**your-region** : **111122223333** :datastore/fhir/**your-datastore-id**
ReadResource	リソースを読み取るアクセス許可を付与します	読み取り	データストア ARN: arn:aws:healthlake:**your-region** : **111122223333** :datastore/fhir/**your-datastore-id**
SearchWithGet	GET メソッドでリソースを検索する許可を付与	読み取り	データストア ARN: arn:aws:healthlake:**your-region** : **111122223333** :datastore/fhir/**your-datastore-id**

アクション	説明	アクセスレベル	リソースタイプ (必須)
SearchWithPost	POST メソッドでリソースを検索する許可を付与	読み取り	データストア ARN: arn:aws:healthlake:**your-region** :**111122223333** :datastore/fhir/**your-datastore-id**
StartFHIRExportJobWithPost	でFHIRエクスポートジョブを開始する許可を付与GET	書き込み	データストア ARN: arn:aws:healthlake:**your-region** :**111122223333** :datastore/fhir/**your-datastore-id**
UpdateResource	リソースを更新する許可を付与	書き込み	データストア ARN: arn:aws:healthlake:**your-region** :**111122223333** :datastore/fhir/**your-datastore-id**

開始するには、 を使用できますAmazonHealthLakeFullAccess。このポリシーは、データストアで見つかったすべてのFHIRリソースの読み取り、書き込み、検索、エクスポートを許可します。データストアに対する読み取り専用アクセス許可を付与するには、 を使用しますAmazonHealthLakeReadOnlyAccess。

AWS Management Console、、または を使用してカスタムポリシーを作成する方法の詳細については AWS CLI、「 IAMユーザーガイドIAMSDKs」の「ポリシーの作成IAM」を参照してください。

のサービスロールの作成 HealthLake（ IAM コンソール）

この手順を使用して、サービスロールを作成します。サービスを作成するときは、 IAMポリシーも指定する必要があります。

のサービスロールを作成するには HealthLake（ IAM コンソール ）

1. にサインイン AWS Management Console し、 で IAMコンソールを開きますhttps://console.aws.amazon.com/iam/。
2. IAM コンソールのナビゲーションペインで [ロール] を選択します。
3. 続いて、[ロールの作成] を選択します。
4. 信頼エンティティの選択ページで、カスタム信頼ポリシーを選択します。
5. 次に、カスタム信頼ポリシーで、次のようにサンプルポリシーを更新します。をアカウント番号**your-account-id**に置き換え、インポートまたはエクスポートジョブで使用するデータストアARNの を追加します。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Principal": {
            "Service": "healthlake.amazonaws.com"
        },
        "Condition": {
          "StringEquals": {
            "aws:SourceAccount": "your-account-id"
            },
            "ArnEquals": {
              "aws:SourceArn": "arn:aws:healthlake:your-region:your-account-id:datastore/fhir/your-datastore-id"
            }
          }
        }
  ]
}
```

6. [次へ] を選択します。
7. アクセス許可の追加ページで、 HealthLake サービスが引き受けるポリシーを選択します。ポリシーを検索するには、 アクセス許可ポリシーで検索します。
8. 次に、ポリシーのアタッチを選択します。
9. 次に、名前、レビュー、作成のページで、ロール名に名前を入力します。
10. （ オプション) 説明 の で、ロールの簡単な説明を追加します。

11. 可能な場合は、このロールの目的を識別するのに役立つロール名またはロール名サフィックスを入力します。ロール名は 内で一意である必要があります AWS アカウント。大文字と小文字は区別されません。例えば、**PRODROLE** と **prodrole** というロール名を両方作成することはできません。多くのエンティティによりロールが参照されるため、作成後にロール名を変更することはできません。
12. ロールの詳細を確認し、ロールの作成を選択します。

サンプル Lambda 関数ARNでロールを指定する方法については、「」を参照してくださいAWS Lambda 関数の作成。

Lambda 実行ロール

Lambda 関数の実行ロールは、 AWS サービスとリソースにアクセスするためのアクセス許可を関数に付与する IAMロールです。このページでは、Lambda 関数の実行ロールを作成、表示、および管理する方法について説明します。

デフォルトでは、 を使用して新しい Lambda 関数を作成すると、Lambda は最小限のアクセス許可で実行ロールを作成します AWS Management Console。実行ロールで付与されたアクセス許可を管理するには、「Lambda デベロッパーガイド」のIAM「コンソールでの実行ロールの作成」を参照してください。

このトピックで提供されるサンプル Lambda 関数は、Secrets Manager を使用して認可サーバーの認証情報を隠します。

作成したIAMロールと同様に、最小特権のベストプラクティスに従うことが重要です。開発フレーズ中に、必要以上のアクセス許可を付与することがあります。ベストプラクティスとしては、本番環境に関数を公開する前に、ポリシーを調整して必要なアクセス許可のみを含めるようにします。詳細については、「 IAMユーザーガイド」の「最小特権の適用」を参照してください。

Lambda 関数のトリガーを HealthLake に許可する

そのため HealthLake 、 はユーザーに代わって Lambda 関数を呼び出すことができるため、以下を実行する必要があります。

- `CreateFHIRDatastore` リクエストで HealthLake 呼び出す Lambda 関数ARNのと`IdpLambdaArn`等しく設定する必要があります。
- がユーザーに代わって Lambda 関数を呼び出す HealthLake ことを許可するリソースベースのポリシーが必要です。

は、FHIR有効な HealthLake データストアSMARTで に対するFHIRRESTAPIリクエスト HealthLake を受け取ると、ユーザーに代わってデータストアの作成時に指定された Lambda 関数を呼び出すアクセス許可が必要です。 HealthLake アクセスを許可するには、リソースベースのポリシーを使用します。Lambda 関数のリソースベースのポリシーの作成の詳細については、「 AWS Lambda デベロッパーガイド」の「 AWS サービスによる Lambda 関数の呼び出しを許可する」を参照してください。

Lambda 関数の同時実行のプロビジョニング

> **Important**
>
> HealthLake では、Lambda 関数の最大実行時間が 1 秒 (1000 ミリ秒) 未満である必要があります。
> Lambda 関数が実行時間の制限を超えると、TimeOut例外が発生します。

この例外が発生しないように、プロビジョニングされた同時実行を設定することをお勧めします。呼び出しの増加前にプロビジョニングされた同時実行数を割り当てることにより、すべてのリクエストが、短いレイテンシーで、初期化されたインスタンスによって処理されるようにできます。プロビジョニングされた同時実行の設定の詳細については、「Lambda デベロッパーガイド」の「プロビジョニングされた同時実行の設定」を参照してください。

Lambda 関数の平均実行時間を確認するには、Lambda コンソールの Lambda 関数のモニタリングページを使用します。デフォルトでは、Lambda コンソールには、関数コードがイベントの処理に費やす平均、最小、最大時間を示す期間グラフが表示されます。Lambda 関数のモニタリングの詳細については、Lambda デベロッパーガイドの「Lambda コンソールでの関数のモニタリング」を参照してください。

Lambda 関数の同時実行をプロビジョニング済みで、それをモニタリングする場合は、「Lambda デベロッパーガイド」の「同時実行のモニタリング」を参照してください。

有効FHIRになっている HealthLake データストアSMARTでの の作成

でFHIRフレームワークSMARTの を使用するには HealthLake、C reateFHIRDatastore リクエストで指定された `IdentityProviderConfiguration`パラメータを使用して HealthLake データストアを作成します。`IdentityProviderConfiguration` パラメータで、次の情報を指定します。

- を AuthorizationStrategyに設定しますSMART_ON_FHIR_V1。
- 認可サーバーでトークンデコードを管理するために AWS Lambda 作成した ARNの とIdpLambdaArn等しい を設定します。
- 認可サーバーで指定されたメタデータ要素をJSONブロックとして定義します。これらのメタデータ要素は、 検出ドキュメントで返されます。
- オプション: を有効にしますFineGrainedAuthorizationEnabled。が提供するきめ細かな認可Trueを使用するには、 を指定します。 HealthLake

FHIR 有効なデータストアSMARTで を作成するには、 AWS Command Line Interface （ AWS CLI) を使用するか、 AWS サポートされている のいずれかを使用しますSDKs。有効FHIRになっている HealthLake データストアSMARTでの の作成は、 HealthLake コンソールではサポートされていません。

を使用してFHIR有効になっている HealthLake データストアSMARTで AWS CLI を作成する

次のコード例を使用して、 を使用してFHIR有効な HealthLake データストアSMARTに を作成できます AWS CLI。有効FHIRになっている HealthLake データストアSMARTで を作成するときは、identity-provider-configurationパラメータを指定する必要があります。

identity-provider-configuration パラメータでは、オプションで をにFineGrainedAuthorizationEnabled等しく設定することで、きめ細かな認可を有効にできますTrue。きめ細かな認可の詳細については、「」を参照してくださいFHIR 有効化された HealthLake データストアSMARTでの でのきめ細かな認可の使用。以下の例には、改行\を示す特殊文字、またはエスケープ文字として が含まれています。これはわかりやすくするためです。

```
aws healthlake create-fhir-datastore \
  --region us-east-1 \
  --datastore-name "your-data-store-name" \
  --datastore-type-version R4 \
  --preload-data-config PreloadDataType="SYNTHEA" \
  --sse-configuration '{ "KmsEncryptionConfig": { \
    "CmkType": "customer-managed-kms-key1",
    "KmsKeyId": "arn:aws:kms:us-east-1:your-account-id:key/your-key-id" } }' \
  --identity-provider-configuration  \
      '{"AuthorizationStrategy": "SMART_ON_FHIR_V1", \
      "FineGrainedAuthorizationEnabled": boolean-false-by-default, \
```

```
      "IdpLambdaArn": "arn:aws:lambda:your-region:your-account-id:function:your-lambda-
name" \
      "Metadata": "{\"issuer\":\"https://ehr.example.com\",\"jwks_uri\":\"https://
ehr.example.com/.well-known/jwks.json\",\"authorization_endpoint\":\"https://
ehr.example.com/auth/authorize\",\"token_endpoint\":\"https://ehr.token.com/auth/
token\",\"token_endpoint_auth_methods_supported\":[\"client_secret_basic\",\"foo\"],
\"grant_types_supported\":[\"client_credential\",\"foo\"],\"registration_endpoint\":
\"https://ehr.example.com/auth/register\",\"scopes_supported\":[\"openId\",\"profile\",
\"launch\"],\"response_types_supported\":[\"code\"],\"management_endpoint\":\"https://
ehr.example.com/user/manage\",\"introspection_endpoint\":\"https://ehr.example.com/
user/introspect\",\"revocation_endpoint\":\"https://ehr.example.com/user/revoke\",
\"code_challenge_methods_supported\":[\"S256\"],\"capabilities\":[\"launch-ehr\",\"sso-
openid-connect\",\"client-public\"]}"}'
```

成功すると、次のJSONレスポンスが表示されます。

```
{
  "DatastoreArn": "arn:aws:healthlake:your-region:111122223333:datastore/fhir/your-
datastore-id",
  "DatastoreEndpoint": "https://healthlake.your-region.amazonaws.com/datastore/your-
datastore-id/r4/",
  "DatastoreId": "your-data-store-id",
  "DatastoreStatus": "data-store-creation-status"
}
```

FHIR 有効化された HealthLake データストアSMARTでの でのきめ細かな認可の使用

スコープだけでは、リクエスタがデータストア内のどのデータにアクセスすることを許可されているかについて、必要な具体性が得られません。きめ細かな認可を使用すると、FHIR有効な HealthLake データストアSMARTで へのアクセスを許可するときに、より高いレベルの特異度が可能になります。きめ細かな認可を使用するには、CreateFHIRDatastore リクエストの IdentityProviderConfigurationパラメータTrueで をにFineGrainedAuthorizationEnabled等しく設定します。

きめ細かな認可を有効にした場合、認可サーバーはアクセストークンid_tokenとともにでfhirUserスコープを返します。これにより、ユーザーに関する情報がクライアントアプリケーションによって取得されます。クライアントアプリケーションは、fhirUserクレームを現在のユーザーを表すFHIRリソースURIの として扱う必要があります。これは、Patient、Practitioner、または RelatedPerson です。認可サーバーのレスポンスには、ユーザーがアクセスできるデータ

を定義するuser/スコープも含まれています。これは、FHIRリソース固有のスコープに関連するスコープに定義された構文を使用します。

```
user/(fhir-resource | '*').('read' | 'write' | '*')
```

以下は、きめ細かな認可を使用してデータアクセス関連のFHIRリソースタイプをさらに指定する方法の例です。

- fhirUser が の場合Practitioner、きめ細かな認可によって、ユーザーがアクセスできる患者のコレクションが決まります。へのアクセスfhirUserは、患者がfhirUser一般提供者として を参照している患者に対してのみ許可されます。

  ```
  Patient.generalPractitioner : [{Reference(Practitioner)}]
  ```

- fhirUser が Patientまたは RelatedPersonで、リクエストで参照される患者が と異なる場合fhirUser、きめ細かな認可によって、リクエストされた患者の fhirUser へのアクセスが決まります。リクエストされたPatientリソースで関係が指定されている場合、アクセスが許可されます。

  ```
  Patient.link.other : {Reference(Patient|RelatedPerson)}
  ```

FHIR 有効な HealthLake データストアの検出ドキュメントSMART の の取得

クライアントアプリケーションがFHIRRESTリクエストを正常に実行するには、 HealthLake データストアで定義されている認可要件を収集する必要があります。このリクエストが成功するために認可 (ベアラートークン) は必要ありません。

これを行うには、GETリクエストを行い、データストアのエンドポイント/.well-known/smart-configurationに を追加します。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/.well-known/smart-configuration
```

これにより、 HealthLake データストアの検出ドキュメントが BLOB JSON として返されます。データ HealthLake ストアに定義されている仕様authorization_endpointと機能token_endpointとともに、 と があります。

```
{
  "authorization_endpoint": "https://oidc.example.com/authorize",
  "token_endpoint": "https://oidc.example.com/oauth/token",
  "capabilities": [
    "launch-ehr",
    "client-public"
  ]
}
```

URLs クライアントアプリケーションを正常に起動するために が必要

- 認可エンドポイント: クライアントアプリケーションまたはユーザーを認可URLするために必要な 。
- トークンエンドポイント: クライアントアプリケーションが通信に使用する認可サーバーのエンドポイント。

SMART 有効な HealthLake データストアでのFHIRRESTAPIリクエストの実行

FHIRが有効な HealthLake データストアで SMARTに対してFHIRRESTAPIリクエストを行うことができます。次の例は、認可ヘッダーJWTに を含むクライアントアプリケーションからのリクエストと、Lambda がレスポンスをデコードする方法を示しています。クライアントアプリケーションリクエストが承認され、認証されたら、認可サーバーからベアラートークンを受け取る必要があります。FHIRが有効な HealthLake データストアで にFHIRRESTAPIリクエストを送信するときは、認可ヘッダーSMARTのベアラートークンを使用します。

```
GET https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/
Patient/[ID]
Authorization: Bearer auth-server-provided-bearer-token
```

ベアラートークンが認可ヘッダーで検出され、ID が検出されなかった AWS IAMため、FHIR有効になっている HealthLake データストアSMARTの が作成されたときに指定された Lambda 関数が HealthLake 呼び出されます。トークンが Lambda 関数によって正常にデコードされたとき、 が に送信したレスポンスの例を次に示します HealthLake。

```
{
  "authPayload": {
```

```
    "iss": "https://authorization-server-endpoint/oauth2/token", # The issuer
 identifier of the authorization server
    "aud": "https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/
r4/", # Required, data store endpoint
    "iat": 1677115637,  # Identifies the time at which the token was issued
    "nbf": 1677115637,  # Required, the earliest time the JWT would be valid
    "exp": 1997877061,  # Required, the time at which the JWT is no longer valid
    "isAuthorized": "true",  # Required, boolean indicating the request has been
 authorized
    "uid": "100101",  # Unique identifier returned by the auth server
    "scope": "system/*.*" # Required, the scope of the request
  },
  "iamRoleARN": "iam-role-arn" #Required, IAM role to complete the request
}
```

FHIR 準拠のデータストアSMARTに を実装するために必要なリソースの設定

このトピックでは、 の外部で AWS アカウントでプロビジョニングする必要があるリソース HealthLake、FHIR有効な HealthLake データストアSMARTで を作成する方法、およびSMARTFHIR クライアントアプリケーションの が認可サーバーと HealthLake データストアとやり取りする方法について説明します。

このワークフローのステップでは、SMARTFHIRリクエストの処理方法と、リクエストを成功させるために必要なリソースの基本的なステップを定義します。

SMART オンFHIRリクエストプロセスでは、3 つのアプリケーションが連携します。

- エンドユーザー: 一般的に、患者または臨床医はFHIR、アプリケーションSMART上のサードパーティーを使用してデータストア内の HealthLake データにアクセスします。
- SMART FHIR アプリケーションの (クライアントアプリケーションと呼ばれる) : HealthLake データストアで見つかったデータにアクセスしたいアプリケーション。
- 認可サーバー: ユーザーを認証し、アクセストークンを発行できる OpenID Connect 準拠サーバー。
- HealthLake データストア: Lambda 関数を使用してベアラートークンを提供するFHIRRESTリクエストに応答する、FHIR有効な HealthLake データストアSMART上の 。

これらのアプリケーションが連携するには、次のリソースを作成する必要があります。

認可サーバーをセットアップし、必要なスコープを定義し、トークンのイントロスペクションを処理する AWS Lambda 関数を作成したら、FHIR有効な HealthLake データストアSMARTで を作成することをお勧めします。

1. 認可サーバーエンドポイントのセットアップ — 認可サーバー

FHIR フレームワークSMARTで を使用するには、データストアで行われたFHIRRESTリクエストを検証できるサードパーティーの認可サーバーを設定する必要があります。が動作する認可サーバーエンドポイントの設定の詳細については HealthLake、「」を参照してくださいSMART での の認証要件 FHIR。

2. 認可サーバー上のデータ HealthLake ストア内のデータにアクセスできるユーザーを制御するスコープを定義する — 認可サーバー

on SMARTFHIRフレームワークでは、OAuthスコープを使用して、認証されたリクエストがアクセスできるFHIRリソースとその範囲を決定します。スコープの定義は、最小特権を設計する方法です。FHIR フレームワークSMARTの で定義され、 でサポートされているスコープの詳細については、 HealthLake 「」を参照してくださいでサポートされるSMARTFHIROAuthスコープ HealthLake。

3. トークンイントロスペクションを実行できる AWS Lambda 関数のセットアップ — AWS アカウント

SMART FHIR 有効なデータストアでクライアントアプリケーションから送信されたFHIRRESTリクエストには、JSONウェブトークン () が含まれますJWT。デコードおよび検証可能な Lambda 関数の設定の詳細については、「 のデコードJWT」を参照してください。

4. 有効FHIRになっている HealthLake データストアSMARTで を作成する — AWS アカウント

FHIR HealthLake データストアSMARTで を作成するには、 を指定する必要がありますIdentityProviderConfiguration。C reateFHIRDatastore リクエストに必要なIdentityProviderConfigurationパラメータの詳細については、FHIR「有効になっている HealthLake データストアでの の作成SMART」を参照してください。

クライアントアプリケーションが SMARTでデータを起動し、 からデータをリクエストして HealthLake データストアFHIRを有効にする方法

このセクションでは、クライアントアプリケーションがSMARTオンFHIRコンテキストで を使用して起動し、 HealthLake データストアで正常にFHIRRESTリクエストできる方法について説明します。

1. クライアントアプリケーションが Well-Known Uniform Resource Identifier に**GET**リクエストを行う

SMART 有効なクライアントアプリケーションは、 HealthLake データストアの認可エンドポイントを検索するGETリクエストを行う必要があります。これは、Well-Known Uniform Resource Identifier (URI) リクエストを介して行われます。詳細については、FHIR「有効になっている HealthLake データストアの Discovery Document SMARTで を取得する」を参照してください。

2. アクセスとスコープのリクエスト

クライアントアプリケーションは認可サーバーの認可エンドポイントを使用して、ユーザーがログインできるようにします。このプロセスはユーザーを認証します。スコープは、クライアントアプリケーションがアクセスできる HealthLake データストア内のFHIRリソースを定義するために使用されます。スコープの定義の詳細については、「」を参照してくださいでサポートされるSMARTFHIROAuthスコープ HealthLake。

3. アクセストークン

ユーザーが認証されたので、クライアントアプリケーションは認可サーバーからJWTアクセストークンを受け取ります。このトークンは、クライアントアプリケーションがFHIRRESTリクエストを送信するときに提供されます HealthLake。Lambda 関数を使用して をJWTデコードする方法の詳細については、「」を参照してくださいトークン検証の実行。

4. FHIR 有効な HealthLake データストアSMARTで をFHIRRESTリクエストする

これで、クライアントアプリケーションは認可サーバーによって提供されるアクセストークンを使用して、 HealthLake データストアエンドポイントにFHIRRESTリクエストを送信できるようになりました。FHIR REST リクエストの例については、「」を参照してくださいSMART 有効な HealthLake データストアでのFHIRRESTAPIリクエストの実行。

5. JWT アクセストークンの検証

FHIR REST リクエストで送信されたアクセストークンを検証するには、Lambda 関数を使用します。トークンの詳細分析を実行できる Lambda 関数を作成する方法については、「」を参照してくださいAWS Lambda 関数の作成。

のリソースタイプの自然言語処理 (NLP) に基づく自動FHIR DocumentReference リソース生成の使用 AWS HealthLake

> **Note**
>
> 2023 年 2 月 20 日以降、 HealthLake データストアはデフォルトでは統合自然言語処理 (NLP) を使用しません。データストアでこの機能を有効にする場合は、トラブルシューティングの章 HealthLake統合された自然言語処理機能を有効にするにはどうすればよいですか？の「」を参照してください。
> Amazon Comprehend Medical の統合 を有効にしている場合NLP、DocumentReferenceリソースを作成または更新すると、 AWS アカウントで料金が発生します。詳細については、「 のAWS HealthLake 料金」を参照してください。
> Amazon Comprehend Medical は、アジアパシフィック (ムンバイ) では利用できません。アジアパシフィック (ムンバイ) リージョンで作成された HealthLake データストアは、統合された自然言語処理 () をサポートしていませんNLP。

HealthLake は、 DocumentReferenceリソースタイプに保存されているデータの非構造化データ処理に Amazon Comprehend Medical を使用して、統合された自然言語処理 (NLP) を自動的に提供します。これを行うには、 は Amazon Comprehend Medical の DetectEntities-V2、InferICD10-CM、および InferRxNormAPIオペレーションを HealthLake 呼び出します。結果は、拡張機能としてDocumentReferenceリソースに自動的に追加されます。Amazon Comprehend Medical APIオペレーションが SIGN、SYMPTOM、および である特性を検出するとDIAGNOSIS、Linkageリソースタイプが自動的に生成されます。新しい条件および観測リソースは、SIGN、、SYMPTOMまたは の特性で識別されるエンティティから作成されDIAGNOSIS、このリンクリソースを使用してソースドキュメントにリンクされます。

統合された によって生成されたリソースの場合NLP、GETリクエストを行うことができますが、これらの新しいリソースの検索はサポートされていません。

HealthLakeと Athena の統合を使用してこれらの拡張機能を検索する方法の詳細については、「」を参照してくださいを使用して HealthLake データストアをクエリする SQL。

目次

- Amazon Comprehend Medical APIオペレーションを に統合する方法の例 HealthLake
- 検索パラメータ

Amazon Comprehend Medical と の統合方法 HealthLake

HealthLake は、Amazon Comprehend Medical を使用してDocumentReferenceリソースタイプで見つかったデータを推測します。Amazon Comprehend Medical APIオペレーション DetectEntities-V2、InferICD10-CM、 は病状を特性としてInferRxNorm検出します。オペレーションごとに異なるインサイトが得られます。

> **⚠ 言語サポート**
>
> Amazon Comprehend Medical APIのオペレーションでは、英語のテキストで医療エンティティのみが検出されます。

- DetectEntities-V2: 臨床テキストでさまざまな医療エンティティを検査し、エンティティカテゴリ、場所、信頼スコアなど、それらに関する特定の情報を返します。
- InferICD10-CM: 患者レコード内の病状をエンティティとして検出し、それらのエンティティを、世界保健機関 () の認可を受けている CDCの国立保健統計センターの ICD-10-CM ナレッジベースの正規化された概念識別子にリンクしますWHO。
- InferRxNorm: 薬剤を患者記録にリストされているエンティティとして検出し、国立医学図書館から RxNorm データベース内の正規化された概念識別子にリンクします。

各APIオペレーションでサポートされている特性は、SIGN、SYMPTOM、および ですDIAGNOSIS。特性が検出されると、 HealthLake データストア内のさまざまな場所に FHIR準拠の拡張機能として追加されます。

拡張機能が追加される場所。

- DocumentReference: Amazon Comprehend Medical APIオペレーションの結果は、DocumentReference リソースタイプ内の各ドキュメントextensionに として追加されます。拡張機能の結果は 2 つのグループに分割されます。これらは、 に基づいて結果で確認できますURL。
  - http://healthlake.amazonaws.com/system-generated-resources/
    - これらは、 によって作成または追加されたリソースタイプです HealthLake。

  - `http://healthlake.amazonaws.com/aws-cm/`
    - Amazon Comprehend Medical APIオペレーションの raw 出力が HealthLake データストアに追加される場所。
- `Linkage`: このリソースタイプは、統合された の結果として追加または作成されますNLP。特定の に対するGETリクエストは、リンクされたリソースのリスト`Linkage`を返します。`Linkage` がによって追加されたかどうかを確認するには HealthLake、追加された`"tag": [{"display": "SYSTEM_GENERATED"}]`キーと値のペアを探します。Linkage のFHIR仕様の詳細については、FHIRドキュメントインデックスの「リソースタイプ: Linkage」を参照してください。
- FHIR Amazon Comprehend Medical APIオペレーションの結果として生成された リソースタイプ。
  - `Observation`: 特性が `SIGN`または の場合、Amazon Comprehend Medical operationsAPI DetectEntities-V2 と InferICD10-CM の結果が追加されます`SYMPTOM`。
  - `Condition`: 特性が の場合、Amazon Comprehend Medical APIオペレーション DetectEntities-V2 と InferICD10-CM の結果が追加されます`DIAGNOSIS`。
  - `MedicationStatement`: Amazon Comprehend Medical APIオペレーションの結果 InferRxNorm が追加されました。

FHIR REST API オペレーションとの統合

デフォルトでは、Amazon Comprehend Medical APIオペレーションによって検出された特性は、`GET`リクエストの実行時に返されません。

これらのリソースタイプの統合NLPオペレーションの結果を表示するには、既知の を指定する必要がありま`ID`す。

- `Linkage`
- `Observation`
- `Condition`
- `MedicationStatement`

DocumentReference リソースタイプ以外の統合NLPオペレーションの結果は、指定された に Amazon Comprehend Medical APIオペレーションの結果が含まれていることが`ID`わかっているGETリクエストでのみ使用できます。

Amazon Comprehend Medical APIオペレーションを に統合する方法の例 HealthLake

例 1: HealthLake データストアに取り込まれた患者レコード

患者の医療専門家との出会いに基づく臨床記録の例を次に示します。

> ⚠ 合成データ
>
> この例のテキストは合成コンテンツであり、個人の健康情報 () は含まれていませんPHI。

```
1991-08-31

# Chief Complaint
- Headache
- Sinus Pain
- Nasal Congestion
- Sore Throat
- Pain with Bright Lights
- Nasal Discharge
- Cough


# History of Present Illness
Jerónimo599
 is a 4 month-old non-hispanic white male.

# Social History
 Patient has never smoked.


Patient comes from a middle socioeconomic background.

Patient currently has Aetna.

# Allergies
No Known Allergies.

# Medications
No Active Medications.
```

```
# Assessment and Plan
Patient is presenting with bee venom (substance), mold (organism), house dust
 mite (organism), animal dander (substance), grass pollen (substance), tree pollen
 (substance), lisinopril, sulfamethoxazole / trimethoprim, fish (substance).

## Plan

The patient was prescribed the following medications:
- astemizole 10 mg oral tablet
- nda020800 0.3 ml epinephrine 1 mg/ml auto-injector
The patient was placed on a careplan:
- self-care interventions (procedure)
```

この情報は、 DocumentReference リソースで base64 形式でエンコードされます。このドキュメントが に取り込まれ HealthLake 、Amazon Comprehend Medical APIオペレーションが完了すると、結果を表示するには、DocumentReferenceリソースタイプの GETリクエストから開始できます。

```
 GET https://https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/
r4/eeb8005725ae22b35b4edbdc68cf2dfd/r4/DocumentReference
```

Amazon Comprehend Medical APIオペレーションが成功したら、以下extensionにリンクされている 内でこれらのキーと値のペアを探します。 "url": "http://healthlake.amazonaws.com/aws-cm/"

```
{
  "url": "http://healthlake.amazonaws.com/aws-cm/status/",
  "valueString": "SUCCESS"
 },
 {
  "url": "http://healthlake.amazonaws.com/aws-cm/message/",
  "valueString": "The Amazon HealthLake integrated medical NLP operation was
 successful."
 }
```

次のタブは、リソースタイプに基づいて、取り込まれた医療記録が HealthLake データストアにどのように報告されるかを示しています。

DocumentReference

単一のDocumentReferenceリソースタイプの結果を表示するには、特定のリソースidの が指定されている をGETリクエストします。

```
GET https://https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/eeb8005725ae22b35b4edbdc68cf2dfd/r4/DocumentReference/0e938f03-da7f-4178-acd8-eea9586c46ed
```

成功すると、200HTTPレスポンスコードと次のJSONレスポンス (わかりやすくするために切り捨てられています) が返されます。

以下にそのhttp://healthlake.amazonaws.com/system-generated-resources/部分を示します。新しい Linkage/*e366d29f-2c22-4c19-866e-09603937935a* が追加されたことを確認できます。また、 HealthLake が推論ベースの検出結果を特定の Observationおよびconditionリソースタイプに追加した場所も確認できます。

これらのリソースタイプがどのように修正されたかを確認するには、関連するタブを選択します。

```
{
  "extension": [
   {
    "url": "http://healthlake.amazonaws.com/linkage",
    "valueReference": {
     "reference": "Linkage/e366d29f-2c22-4c19-866e-09603937935a"
    }
   },
   {
    "url": "http://healthlake.amazonaws.com/nlp-entity",
    "valueReference": {
     "reference": "Observation/c6e0a3ff-7a17-4d8b-bfd0-d02d7da090c5"
    }
   },
   {
    "url": "http://healthlake.amazonaws.com/nlp-entity",
    "valueReference": {
     "reference": "Condition/0854e1f3-894d-448e-a8d9-3af5b9902baf"
    }
   }
  ],
  "url": "http://healthlake.amazonaws.com/system-generated-resources/"
 }
```

Linkage

単一のLinkageリソースタイプの結果を表示するには、特定のリソースIDの が指定されているをGETリクエストします。

```
GET https://https://healthlake.your-region.amazonaws.com/
datastore/your-datastore-id/r4/eeb8005725ae22b35b4edbdc68cf2dfd/r4/
Linkage/e366d29f-2c22-4c19-866e-09603937935a
```

成功すると、200HTTPレスポンスコードと、次の切り捨てられたJSONレスポンスが返されます。

レスポンスには item要素が含まれています。キーと値のペアは、 の変更に使用され、"type": "alternate"キーConditionと値のペアの下にObservationsリストされている特定のDocumentReferenceエントリ"type": "source"を示します。

また、 *meta*要素と*"tag": [{"display": "SYSTEM_GENERATED"}]*、これらのリソースがによって作成されたことを示す対応するキーと値のペア も表示されます HealthLake。

```
{
  "resourceType": "Linkage",
  "id": "e366d29f-2c22-4c19-866e-09603937935a",
  "active": true,
  "item":
  [
   {
    "type": "alternate",
    "resource": {
     "reference": "Observation/c6e0a3ff-7a17-4d8b-bfd0-d02d7da090c5",
     "type": "Observation"
    }
   },
   {
    "type": "alternate",
    "resource": {
     "reference": "Condition/9d5c1ef6-f822-4faf-b55f-7c70f2a4aa8d",
     "type": "Condition"
    }
   },
   {
    "type": "source",
    "resource": {
```

```
    "reference": "DocumentReference/0e938f03-da7f-4178-acd8-eea9586c46ed",
    "type": "DocumentReference"
   }
  }
 ],
 "meta": {
  "lastUpdated": "2022-10-21T19:38:31.327Z",
  "tag": [{
   "display": "SYSTEM_GENERATED"
  }]
 }
}
```

Resource type: Observation

単一のObservationリソースタイプの結果を表示するには、特定のリソースIDの が指定されているGETリクエストを実行します。

```
GET https://https://healthlake.your-region.amazonaws.com/
datastore/your-datastore-id/r4/eeb8005725ae22b35b4edbdc68cf2dfd/r4/
Observation/e366d29f-2c22-4c19-866e-09603937935a
```

Amazon Comprehend Medical APIオペレーションの結果はcode、、meta、 の各要素に変更されますmodifierExtension。

code

タイプの要素CodeableConcept。詳細については、 FHIRドキュメントインデックスCodeableConceptの「」を参照してください。

HealthLake は、次の 3 つのキーと値のペアを追加します。

- "system": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/": ここで、 は特定の Amazon Comprehend Medical APIオペレーションURLを指します。この場合、InferICD10CM。
- "code": "A52.06": ここで、 A52.06は、Centers for Cypress Control のナレッジベースにある概念を識別する ICD-10-CM コードです。
- "display": "Other syphilitic heart involvement": ここで、 "Other syphilitic heart involvement"はオントロジー内の ICD-10-CM コードの長い説明です。

次の切り捨てられたJSONレスポンスには、 code要素のみが含まれます。

```
"code": {
  "coding":
  [
    {
   "system": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/",
   "code": "A52.06",
   "display": "Other syphilitic heart involvement"
    }
  ],
  "text": "Other syphilitic heart involvement"
 }
```

割り当てられた ICD-10-CM コードが正しいというモデルの信頼度を理解するには、 modifierExtension要素を使用します。

meta

meta 要素には、Amazon Comprehend Medical APIオペレーションによって追加された詳細がcode要素に含まれているかどうかを示すメタデータが含まれています。

次の切り捨てられたJSONレスポンスには、 meta要素のみが含まれます。

```
"meta": {
  "lastUpdated": "2022-10-21T19:38:30.879Z",
  "tag": [{
   "display": "SYSTEM_GENERATED"
  }]
 }
```

modifierExtension

modifierExtension 要素には、 code要素で見つかった割り当てられたコードの信頼度に関する詳細が含まれています。また、結果の生成に DocumentReference 使用された元の および関連する Linkage リソースタイプへのリンクを提供するキーと値のペアもあります。

追加されるcoding要素ごとに、 entity-scoreと が entity-Concept-Scoreに追加されますmodifierExtension。キーと値のペアの値ごとに、スコアが表示されます。の場合entity-score、このスコアは Amazon Comprehend Medical が検出の精度に対して持っている信頼度です。の場合entity-Concept-Score、このスコアは、エンティティが正確に ICD-10-CM 概念にリンクされているという Amazon Comprehend Medical の信頼度です。

次の切り捨てられたJSONレスポンスには、 modifierExtension要素のみが含まれます。

```
"modifierExtension": [{
   "url": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/aws-cm-icd10-entity-
score",
   "valueDecimal": 0.45005733
  },
  {
   "url": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/aws-cm-icd10-entity-
Concept-Score",
   "valueDecimal": 0.1111792
  },
  {
   "url": "http://healthlake.amazonaws.com/system-generated-linkage",
   "valueReference": {
    "reference": "Linkage/e366d29f-2c22-4c19-866e-09603937935a"
   }
  },
  {
   "url": "http://healthlake.amazonaws.com/source-document-reference",
   "valueReference": {
    "reference": "DocumentReference/0e938f03-da7f-4178-acd8-eea9586c46ed"
   }
  }
 ]
```

フルJSONレスポンス

```
{
  "subject": {
   "reference": "Patient/0679b7b7-937d-488a-b48d-6315b8e7003b"
  },
  "resourceType": "Observation",
  "status": "unknown",
  "code": {
   "coding": [{
    "system": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/",
    "code": "A52.06",
    "display": "Other syphilitic heart involvement"
   }],
   "text": "Other syphilitic heart involvement"
  },
  "meta": {
```

```
   "lastUpdated": "2022-10-21T19:38:30.879Z",
   "tag": [{
    "display": "SYSTEM_GENERATED"
   }]
  },
  "modifierExtension": [{
    "url": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/aws-cm-icd10-entity-
score",
    "valueDecimal": 0.45005733
   },
   {
    "url": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/aws-cm-icd10-entity-
Concept-Score",
    "valueDecimal": 0.1111792
   },
   {
    "url": "http://healthlake.amazonaws.com/system-generated-linkage",
    "valueReference": {
     "reference": "Linkage/e366d29f-2c22-4c19-866e-09603937935a"
    }
   },
   {
    "url": "http://healthlake.amazonaws.com/source-document-reference",
    "valueReference": {
     "reference": "DocumentReference/0e938f03-da7f-4178-acd8-eea9586c46ed"
    }
   }
  ],
  "id": "7e88c7c5-21a5-4dd7-8fc2-a02474fba583"
 }
```

Condition

単一のConditionリソースタイプの結果を表示するには、特定のリソースIDの が指定されている をGETリクエストします。

```
GET https://https://healthlake.your-region.amazonaws.com/datastore/your-
datastore-id/r4/eeb8005725ae22b35b4edbdc68cf2dfd/r4/Condition/b06d343d-
ddb8-4f36-82cb-853fcd434dfd
```

Amazon Comprehend Medical APIオペレーションの結果はcode、、meta、 の各要素に変更されますmodifierExtension。

code

タイプの要素CodeableConcept。詳細については、 FHIRドキュメントインデックスCodeableConceptの「」を参照してください。

HealthLake は、次の 3 つのキーと値のペアを追加します。

- "system": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/": ここで、 は特定の Amazon Comprehend Medical APIオペレーションURLを指します。この場合、InferICD10CM。
- "code": "I70.0": ここで、 A52.06は、Centers for Cypress Control のナレッジベースにある概念を識別する ICD-10-CM コードです。
- "display": "Atherosclerosis of aorta": ここで、 "Other syphilitic heart involvement"はオントロジーの ICD-10-CM コードの長い説明です。

次の切り捨てられたJSONレスポンスには、 code要素のみが含まれます。

```
"code": {
  "coding":
  [
    {
   "system": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/",
   "code": "I70.0",
   "display": "Atherosclerosis of aorta"
    }
  ],
  "text": "Atherosclerosis of aorta"
 }
```

割り当てられた ICD-10-CM コードが正しいというモデルの信頼度を理解するには、modifierExtension要素を使用します。

meta

meta 要素には、Amazon Comprehend Medical APIオペレーションによって追加された詳細がcode要素に含まれているかどうかを示すメタデータが含まれています。

次の切り捨てられたJSONレスポンスには、 meta要素のみが含まれます。

```
"meta": {
  "lastUpdated": "2022-10-21T19:38:30.877Z",
```

```
  "tag": [{
   "display": "SYSTEM_GENERATED"
  }]
 }
```

modifierExtension

modifierExtension 要素には、 code要素で見つかった割り当てられたコードの信頼度に関する詳細が含まれています。また、結果の生成に DocumentReference 使用された元の および関連する Linkage リソースタイプへのリンクを提供するキーと値のペアもあります。

追加されるcoding要素ごとに、 entity-scoreと が entity-Concept-Scoreに追加されますmodifierExtension。キーと値のペアの値ごとに、スコアが表示されます。の場合entity-score、このスコアは Amazon Comprehend Medical が検出の精度に対して持っている信頼度です。の場合entity-Concept-Score、このスコアは、エンティティが正確に ICD-10-CM 概念にリンクされているという Amazon Comprehend Medical の信頼度です。

次の切り捨てられたJSONレスポンスには、 modifierExtension要素のみが含まれます。

```
"modifierExtension": [{
   "url": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/aws-cm-icd10-entity-
score",
   "valueDecimal": 0.94417894
  },
  {
   "url": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/aws-cm-icd10-entity-
Concept-Score",
   "valueDecimal": 0.8458298
  },
  {
   "url": "http://healthlake.amazonaws.com/system-generated-linkage",
   "valueReference": {
    "reference": "Linkage/e366d29f-2c22-4c19-866e-09603937935a"
   }
  },
  {
   "url": "http://healthlake.amazonaws.com/source-document-reference",
   "valueReference": {
    "reference": "DocumentReference/0e938f03-da7f-4178-acd8-eea9586c46ed"
   }
  }
 ]
```

フルJSONレスポンス


```
{
  "subject": {
   "reference": "Patient/0679b7b7-937d-488a-b48d-6315b8e7003b"
  },
  "resourceType": "Condition",
  "code": {
   "coding": [{
    "system": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/",
    "code": "I70.0",
    "display": "Atherosclerosis of aorta"
   }],
   "text": "Atherosclerosis of aorta"
  },
  "meta": {
   "lastUpdated": "2022-10-21T19:38:30.877Z",
   "tag": [{
    "display": "SYSTEM_GENERATED"
   }]
  },
  "modifierExtension": [{
    "url": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/aws-cm-icd10-entity-
score",
    "valueDecimal": 0.94417894
   },
   {
    "url": "http://healthlake.amazonaws.com/aws-cm/infer-icd10/aws-cm-icd10-entity-
Concept-Score",
    "valueDecimal": 0.8458298
   },
   {
    "url": "http://healthlake.amazonaws.com/system-generated-linkage",
    "valueReference": {
     "reference": "Linkage/e366d29f-2c22-4c19-866e-09603937935a"
    }
   },
   {
    "url": "http://healthlake.amazonaws.com/source-document-reference",
    "valueReference": {
     "reference": "DocumentReference/0e938f03-da7f-4178-acd8-eea9586c46ed"
    }
   }
  ],
```

```
  "id": "b06d343d-ddb8-4f36-82cb-853fcd434dfd"
 }
```

例 2: MedicationStatement リソースタイプ**DocumentReference**を含む

患者の医療専門家との出会いに基づく臨床記録の例を次に示します。

> ⚠ 合成データ
>
> この例のテキストは合成コンテンツであり、個人の健康情報 () は含まれていませんPHI。

```
Tom is not prescribed Advil
```

次のタブは、リソースタイプに基づいて、取り込まれた医療記録が HealthLake データストアにどのように報告されるかを示しています。

DocumentReference

単一のDocumentReferenceリソースタイプの結果を表示するには、特定のリソースIDの が指定されている をGETリクエストします。

```
GET https://https://healthlake.your-region.amazonaws.com/datastore/your-datastore-id/r4/eeb8005725ae22b35b4edbdc68cf2dfd/r4/DocumentReference/c549125d-a218-421f-b8bf-23614c5e796c
```

成功すると、200HTTPレスポンスコードと次の切り捨てられたJSONレスポンスが返されます。

キーと値のペア は"url": "http://healthlake.amazonaws.com/system-generated-resources/"、この中のリソースタイプextensionが Amazon Comprehend Medical API オペレーションによって追加されたことを示します。新しいLinkageリソースタイプと複数のMedicationStatementリソースを表示できます。

```
"extension": [{
   "extension": [{
     "url": "http://healthlake.amazonaws.com/linkage",
     "valueReference": {
      "reference": "Linkage/394bb244-177b-4409-8657-26b20ed56dd7"
     }
```

```
    },
    {
     "url": "http://healthlake.amazonaws.com/nlp-entity",
     "valueReference": {
      "reference": "MedicationStatement/cbf6af10-b0b9-451c-bdde-99611e3498a8"
     }
    },
    {
     "url": "http://healthlake.amazonaws.com/nlp-entity",
     "valueReference": {
      "reference": "MedicationStatement/9a89b0d3-6681-45ca-9926-27951edce5c7"
     }
    },
    {
     "url": "http://healthlake.amazonaws.com/nlp-entity",
     "valueReference": {
      "reference": "MedicationStatement/4a01f6c8-5f3a-4122-80ab-405312f96aa2"
     }
    },
    {
     "url": "http://healthlake.amazonaws.com/nlp-entity",
     "valueReference": {
      "reference": "MedicationStatement/fbfb77d8-70cf-4579-b4c0-d6fe3c01656b"
     }
    },
    {
     "url": "http://healthlake.amazonaws.com/nlp-entity",
     "valueReference": {
      "reference": "MedicationStatement/1340c9ce-9c48-4bf9-9b2f-d0ab027f5e0b"
     }
    }
   ],
   "url": "http://healthlake.amazonaws.com/system-generated-resources/"
  }
```

Linkage

単一のLinkageリソースタイプの結果を表示するには、特定のリソースIDの が指定されているをGETリクエストします。

```
GET https://https://healthlake.your-region.amazonaws.com/
datastore/your-datastore-id/r4/eeb8005725ae22b35b4edbdc68cf2dfd/r4/
Linkage/394bb244-177b-4409-8657-26b20ed56dd7
```

成功すると、200HTTPレスポンスコードと次のJSONレスポンスが返されます。

レスポンスには item要素が含まれます。キーと値のペアは、MedicationStatementリソースタイプの変更に使用される特定のDocumentReferenceエントリ"type": "source"を示します。

また、 meta要素と"tag": [{"display": "SYSTEM_GENERATED"}]、これらのリソースがによって作成されたことを示す対応するキーと値のペア も確認できます HealthLake。


```
{
  "resourceType": "Linkage",
  "id": "394bb244-177b-4409-8657-26b20ed56dd7",
  "active": true,
  "item": [{
    "type": "alternate",
    "resource": {
     "reference": "MedicationStatement/cbf6af10-b0b9-451c-bdde-99611e3498a8",
     "type": "MedicationStatement"
    }
   },
   {
    "type": "alternate",
    "resource": {
     "reference": "MedicationStatement/9a89b0d3-6681-45ca-9926-27951edce5c7",
     "type": "MedicationStatement"
    }
   },
   {
    "type": "alternate",
    "resource": {
     "reference": "MedicationStatement/4a01f6c8-5f3a-4122-80ab-405312f96aa2",
     "type": "MedicationStatement"
    }
   },
   {
    "type": "alternate",
    "resource": {
     "reference": "MedicationStatement/fbfb77d8-70cf-4579-b4c0-d6fe3c01656b",
     "type": "MedicationStatement"
    }
   },
   {
    "type": "alternate",
```

```
    "resource": {
     "reference": "MedicationStatement/1340c9ce-9c48-4bf9-9b2f-d0ab027f5e0b",
     "type": "MedicationStatement"
    }
   },
   {
    "type": "source",
    "resource": {
     "reference": "DocumentReference/c549125d-a218-421f-b8bf-23614c5e796c",
     "type": "DocumentReference"
    }
   }
  ],
  "meta": {
   "lastUpdated": "2022-10-24T20:05:03.501Z",
   "tag": [{
    "display": "SYSTEM_GENERATED"
   }]
  }
 }
```

MedicationStatement

単一のMedicationStatementリソースタイプの結果を表示するには、特定のリソースIDの が指定されている をGETリクエストします。

```
GET https://https://healthlake.your-region.amazonaws.com/
datastore/your-datastore-id/r4/eeb8005725ae22b35b4edbdc68cf2dfd/r4/
MedicationStatement/9a89b0d3-6681-45ca-9926-27951edce5c7
```

MedicationStatement リソースタイプは、Amazon Comprehend Medical InferRxNorm API オペレーションの結果が見つかった場所です。結果は、、、medicationCodeableConcept、metaの各要素に修正されますmodifierExtension。

medicationCodeableConcept

タイプの要素CodeableConcept。詳細については、 FHIRドキュメントインデックスCodeableConceptの「」を参照してください。

HealthLake は、次の 3 つのキーと値のペアを追加します。

- "system": ""http://healthlake.amazonaws.com/aws-cm/infer-rxnorm/: ここで、 は特定の Amazon Comprehend Medical APIオペレーションURLを指します。この場合、InferRxNorm。
- "code": "731533": ここで、 731533は Rx とも呼ばれる RxNorm 概念 ID ですCUI。
- "display": "ibuprofen 200 MG Oral Capsule [Advil]": ibuprofen 200 MG Oral Capsule [Advil]は RxNorm 概念の説明です。

次の切り捨てられたJSONレスポンスには、 MedicationStatement要素のみが含まれます。

```
"medicationCodeableConcept": {
  "coding": [
   {
    "system": "http://healthlake.amazonaws.com/aws-cm/infer-rxnorm/",
    "code": "731533",
    "display": "ibuprofen 200 MG Oral Capsule [Advil]"
   }
  ]
 }
```

meta

meta 要素には、Amazon Comprehend Medical APIオペレーションによって追加された詳細がcode要素に含まれているかどうかを示すメタデータが含まれています。

次の切り捨てられたJSONレスポンスには、 meta要素のみが含まれます。

```
"meta": {
  "lastUpdated": "2022-10-24T20:05:02.800Z",
  "tag": [
   {
    "display": "SYSTEM_GENERATED"
   }
  ]
 }
```

modifierExtension

modifierExtension 要素には、結果の生成に DocumentReference 使用された元の および関連する Linkage リソースタイプへのリンクを提供するキーと値のペアが含まれています。

```
"modifierExtension": [
   {
    "url": "http://healthlake.amazonaws.com/system-generated-linkage",
    "valueReference": {
     "reference": "Linkage/394bb244-177b-4409-8657-26b20ed56dd7"
    }
   },
   {
    "url": "http://healthlake.amazonaws.com/source-document-reference",
    "valueReference": {
     "reference": "DocumentReference/c549125d-a218-421f-b8bf-23614c5e796c"
    }
   }
 ]
```

検索パラメータ

次の表に、統合医療 の検索可能な属性を示しますNLP。

検索パラメータ

検索パラメータ	の一致を検索します。
detectEntities-エンティティカテゴリ	AWS CM 拡張機能内の DetectEntities サブ拡張機能内のエンティティカテゴリ
detectEntitiesエンティティテキスト	AWS CM 拡張機能内の DetectEntities サブ拡張機能内のエンティティテキスト
detectEntities-エンティティタイプ	AWS CM 拡張機能内の DetectEntities サブ拡張機能内のエンティティタイプ
detectEntities-エンティティスコア	AWS CM 拡張機能内の DetectEntities サブ拡張機能内のエンティティスコア
infer-icd10cm-entity-text	ICD1CM 拡張機能内の Infer0CM サブ拡張機能内のエンティティテキスト AWS
infer-icd10cm-entity-score	ICD1CM 拡張機能内の Infer0CM サブ拡張機能内のエンティティスコア AWS

検索パラメータ	の一致を検索します。
infer-icd10cm-entity-concept-code	CM 拡張機能内の InferICD10CM サブ拡張機能内のエンティティ概念コード AWS
infer-icd10cm-entity-concept-description	CM 拡張機能内の InferICD10CM サブ拡張機能内のエンティティ概念の説明 AWS
infer-icd10cm-entity-concept-score	CM 拡張機能内の InferICD10CM サブ拡張機能内のエンティティ概念スコア AWS
infer-rxnorm-entity-score	AWS CM 拡張機能内の InferRxNorm サブ拡張機能内のエンティティスコア
infer-rxnorm-entity-text	AWS CM 拡張機能内の InferRxNorm サブ拡張機能内のエンティティテキスト
infer-rxnorm-entity-conceptコード	AWS CM 拡張機能内の InferRxNorm サブ拡張機能内のエンティティ概念コード
infer-rxnorm-entity-concept- 説明	AWS CM 拡張機能内の InferRxNorm サブ拡張機能内のエンティティ概念の説明
infer-rxnorm-entity-conceptスコア	AWS CM 拡張機能内の InferRxNorm サブ拡張機能内のエンティティ概念スコア

`EntityText` と が同じエンティティの一部`EntityCategory`である条件に一致するように、 は特別な検索 HealthLake を提供します。次の表は、 でサポートされている特別な検索パラメータを示しています HealthLake。

検索パラメータ

検索パラメータ	返された一致
detectEntities-entity-text-category	DetectEntities サブ拡張機能に、 entityText と の両方に一致するエンティティが少なくとも 1 つある場合 entityCategory。

検索パラメータ	返された一致
detectEntities-entity-type-score	DetectEntities サブ拡張機能に、 entityType と の両方に一致するエンティティが少なくとも 1 つある場合 entityScore。
detectEntities-entity-text-score	DetectEntities サブ拡張機能に、 entityText と の両方に一致するエンティティが少なくとも 1 つある場合 entityScore。
detectEntities-entity-text-type	DetectEntities サブ拡張機能に、 entityText と の両方に一致するエンティティが少なくとも 1 つある場合 entityType。
detectEntities-entity-category-score	entityCategory と の両方に一致するエンティティが少なくとも 1 つある場合entityScore。
infer-icd10cm-entity-text-concept-code	InferICD10CM サブ拡張に に一致するエンティティが少なくとも 1 つあり entityText 、そのエンティティ conceptCode がコードに一致するエンティティが少なくとも 1 つある場合。
infer-icd10cm-entity-text-concept-score	InferICD10CM サブ拡張機能に、 に一致するエンティティが少なくとも 1 つあり entityText 、そのエンティティ conceptScore がスコアに一致するエンティティが少なくとも 1 つある場合。
infer-icd10cm-entity-concept-description-concept-score	InferICD10CM サブ拡張のエンティティ内に、概念の説明と に一致する概念が少なくとも 1 つある場合 conceptScore。
infer-rxnorm-entity-text-概念コード	InferRxNorm サブ拡張に に一致するエンティティが 1 つ以上あり entityText 、そのエンティティ conceptCode がコードに一致するエンティティが 1 つ以上ある場合。

検索パラメータ	返された一致
infer-rxnorm-entity-text-concept-score	InferRxNorm サブ拡張機能に、 に一致するエンティティが少なくとも 1 つあり entityText 、そのエンティティ conceptScore がスコアに一致するエンティティが少なくとも 1 つある場合。
infer-rxnorm-entity-concept-description-concept-score	InferRxNorm サブ拡張機能のエンティティ内に、概念の説明と に一致する概念が少なくとも 1 つある場合conceptScore。

のセキュリティ AWS HealthLake

でのクラウドセキュリティが最優先事項 AWS です。 AWS のお客様は、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 とお客様の間での責任共有です。責任共有モデルでは、これをクラウドのセキュリティおよびクラウド内のセキュリティと説明しています。

- クラウドのセキュリティ AWS は、 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。AWS コンプライアンスプログラム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監査者は定期的にセキュリティの有効性をテストおよび検証。が適用されるコンプライアンスプログラムの詳細については HealthLake、「コンプライアンスプログラムAWSによる対象範囲内のサービスコンプライアンスプログラム」を参照してください。
- クラウド内のセキュリティ — お客様の責任は、使用する AWS サービスによって決まります。また、お客様は、データの機密性、会社の要件、適用される法律や規制など、その他の要因についても責任を負います。

このドキュメントは、 を使用する際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 HealthLake。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 HealthLake ように を設定する方法について説明します。また、 HealthLake リソースのモニタリングや保護に役立つ他の AWSサービスの使用方法についても説明します。

トピック

- でのデータ保護 AWS HealthLake
- for RESTでの暗号化 AWS HealthLake
- の転送中の暗号化 AWS HealthLake
- の Identity and Access Management AWS HealthLake
- AWS CloudTrailを使用した AWS HealthLake APIコールのログ記録
- のコンプライアンス検証 AWS HealthLake
- の耐障害性 AWS HealthLake
- AWS HealthLake のインフラストラクチャセキュリティ
- AWS HealthLakeでのセキュリティのベストプラクティス

でのデータ保護 AWS HealthLake

責任 AWS 共有モデル、 でのデータ保護に適用されます AWS HealthLake。このモデルで説明されているように、 AWS はすべての を実行するグローバルインフラストラクチャを保護する責任があります AWS クラウド。お客様は、このインフラストラクチャでホストされているコンテンツに対する管理を維持する責任があります。また、使用する「 AWS のサービス 」のセキュリティ設定と管理タスクもユーザーの責任となります。データプライバシーの詳細については、「データプライバシー FAQ」を参照してください。欧州におけるデータ保護の詳細については、 AWS セキュリティブログの AWS 責任共有モデルとGDPRブログ記事を参照してください。

データ保護の目的で、認証情報を保護し AWS アカウント 、 AWS IAM Identity Center または AWS Identity and Access Management () を使用して個々のユーザーを設定することをお勧めしますIAM。この方法により、それぞれのジョブを遂行するために必要な権限のみが各ユーザーに付与されます。また、次の方法でデータを保護することもお勧めします:

- 各アカウントで多要素認証 (MFA) を使用します。
- SSL/TLS を使用して AWS リソースと通信します。1TLS.2 が必要で、1.3 TLS をお勧めします。
- API とユーザーアクティビティのログ記録をセットアップします AWS CloudTrail。 CloudTrail 証跡を使用して AWS アクティビティをキャプチャする方法については、「 AWS CloudTrail ユーザーガイド」の CloudTrail 「証跡の使用」を参照してください。
- AWS 暗号化ソリューションと、その中のすべてのデフォルトのセキュリティコントロールを使用します AWS のサービス。
- Amazon Macie などの高度なマネージドセキュリティサービスを使用します。これらは、Amazon S3 に保存されている機密データの検出と保護を支援します。
- コマンドラインインターフェイスまたは AWS を介して にアクセスするときに FIPS 140-3 検証済みの暗号化モジュールが必要な場合はAPI、FIPSエンドポイントを使用します。利用可能なFIPSエンドポイントの詳細については、「連邦情報処理規格 (FIPS) 140-3」を参照してください。

顧客の E メールアドレスなどの機密情報やセンシティブ情報は、タグや [名前] フィールドなどの自由形式のフィールドに配置しないことを強くお勧めします。これは、コンソール、、 HealthLake または を使用して または他の AWS のサービス を操作する場合も同様ですAPI AWS CLI AWS SDKs。名前に使用する自由記述のテキストフィールドやタグに入力したデータは、課金や診断ログに使用される場合があります。URL を外部サーバーに提供する場合は、そのサーバーへのリクエストを検証URLするために認証情報を に含めないことを強くお勧めします。

for RESTでの暗号化 AWS HealthLake

HealthLake は、サービス所有の AWS Key Management Service (AWS KMS) キーを使用して、保管中の顧客の機密データを保護するための暗号化をデフォルトで提供します。カスタマーマネージドKMSキーもサポートされており、データストアからのファイルのインポートとエクスポートの両方に必要です。カスタマーマネージドKMSキーの詳細については、「Amazon Key Management Service」を参照してください。お客様は、データストアの作成時に、 AWS所有KMSキーまたはカスタマーマネージドKMSキーを選択できます。データストアの作成後に暗号化設定を変更することはできません。データストアが AWS所有のKMSキーを使用している場合は、 と表示 AWS_OWNED_KMS_KEY され、保管時の暗号化に使用される特定のキーは表示されません。

AWS 所有KMSキー

HealthLake は、デフォルトでこれらのキーを使用して、保管中の個人を特定できるデータや Private Health Information(PHI) データなどの機密情報を自動的に暗号化します。 AWSが所有するKMSキーはアカウントに保存されません。これらは、複数のAWSアカウントで使用するために がAWS所有および管理するKMSキーのコレクションの一部です。 AWSサービスは、 AWS所有KMSキーを使用してデータを保護することができます。AWS 所有KMSキーを表示、管理、使用したり、その使用を監査したりすることはできません。ただし、データを暗号化するキーを保護するための作業やプログラムを操作したり変更したりする必要はありません。

AWS 所有KMSキーを使用する場合、月額料金や使用料金は請求されません。また、アカウントのAWSKMSクォータにもカウントされません。詳細については、「 AWS所有キー」を参照してください。

カスタマーマネージド KMS キー

HealthLake は、作成、所有、管理する対称カスタマーマネージドKMSキーの使用をサポートし、既存のAWS所有暗号化に 2 番目の暗号化レイヤーを追加します。この暗号化層はユーザーが完全に制御できるため、次のようなタスクを実行できます。

- キーポリシー、IAMポリシー、グラントの確立と維持
- キー暗号化マテリアルのローテーション
- キーポリシーの有効化と無効化
- タグの追加
- キーエイリアスの作成

- キー削除のスケジュール設定

CloudTrail を使用して、 が AWS KMS ユーザーに代わって HealthLake に送信するリクエストを追跡することもできます。 AWS KMS 追加料金が適用されます。詳細については、「 カスタマー所有キー」を参照してください。

カスタマーマネージドキーを作成する

対称カスタマーマネージドキーは、 AWSマネジメントコンソール、または を使用して作成できます AWS KMS APIs。

AWS 「 Key Management Service デベロッパーガイド」の「対称カスタマーマネージドキーの作成」の手順に従います。

キーポリシーは、カスタマーマネージドキーへのアクセスを制御します。すべてのカスタマーマネージドキーには、キーポリシーが 1 つだけ必要です。このポリシーには、そのキーを使用できるユーザーとその使用方法を決定するステートメントが含まれています。カスタマーマネージドキーを作成する際に、キーポリシーを指定することができます。詳細については、「AWS Key Management Service デベロッパーガイド」の「カスタマーマネージドキーへのアクセスを管理する」を参照してください。

HealthLake リソースでカスタマーマネージドキーを使用するには、kms:CreateGrant キーポリシーでオペレーションを許可する必要があります。これにより、指定されたキーへのアクセスを制御するカスタマーマネージドKMSキーに許可が追加され、ユーザーは kms:grant オペレーション HealthLake に必要なアクセス許可を付与されます。詳細については、「許可の使用」を参照してください。

HealthLake リソースでカスタマーマネージドKMSキーを使用するには、キーポリシーで次のAPIオペレーションを許可する必要があります。

- kms: 特定のカスタマーマネージドKMSキーにグラントCreateGrant を追加し、グラントオペレーションへのアクセスを許可します。
- kms: キーの検証に必要なカスタマーマネージドキーの詳細DescribeKey を提供します。これはすべてのオペレーションに必要です。
- kms: すべての書き込みオペレーションで保管中のリソースを暗号化するためのアクセス GenerateDataKey を提供します。
- kms:Decrypt は、暗号化されたリソースの読み取りまたは検索オペレーションへのアクセスを提供します。

以下は、ユーザーがそのキーで暗号化 AWS HealthLake されたデータストアを作成して操作できるようにするポリシーステートメントの例です。

```
"Statement": [
    {
        "Sid": "Allow access to create data stores and do CRUD/search in AWS
 HealthLake",
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::111122223333:HealthLakeFullAccessRole"
        },
        "Action": [
            "kms:DescribeKey",
            "kms:CreateGrant",
            "kms:GenerateDataKey",
            "kms:Decrypt"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "kms:ViaService": "healthlake.amazonaws.com",
                "kms:CallerAccount": "111122223333"
            }
        }
    }
]
```

カスタマーマネージドKMSキーを使用するために必要なIAMアクセス許可

カスタマーマネージドKMSキーを使用して AWS KMS 暗号化を有効にしてデータストアを作成する場合、キーポリシーと HealthLake データストアを作成するユーザーまたはロールのIAMポリシーの両方に必要なアクセス許可があります。

kms:ViaService condition キーを使用して、KMSキーの使用を、送信元のリクエストのみに制限できます HealthLake。

キーポリシーの詳細については、AWS「 Key Management Service デベロッパーガイド」のIAM「ポリシーの有効化」を参照してください。

リポジトリを作成するIAMユーザー、IAMロール、またはAWSアカウントには、kms:CreateGrant、kms:GenerateDataKey、および アクセス kms:DescribeKey 許可と必要なHealthLakeアクセス許可が必要です。

が で許可 HealthLake を使用する方法 AWS KMS

HealthLake には、カスタマーマネージドKMSキーを使用するための許可が必要です。カスタマーマネージドKMSキーで暗号化されたデータストアを作成すると、 は にCreateGrantリクエストを送信して、ユーザーに代わって許可 HealthLake を作成しますAWSKMS。の許可AWSKMSは、顧客アカウントのKMSキー HealthLake へのアクセスを許可するために使用されます。

がユーザーに代わって HealthLake 作成する許可は、取り消したり、廃止したりしないでください。アカウントのAWSKMSキーを使用するアクセス HealthLake 許可を付与する許可を取り消したり廃止したりする場合、このデータにアクセス HealthLake したり、データストアにプッシュされた新しいFHIRリソースを暗号化したり、プル時に復号したりすることはできません。の許可を取り消しまたは廃止すると HealthLake、変更はすぐに行われます。アクセス権限を取り消すには、権限を取り消すのではなく、データストアを削除する必要があります。データストアが削除されると、 はユーザーに代わって許可を HealthLake 廃止します。

HealthLake の暗号化キーのモニタリング

CloudTrail カスタマーマネージドKMSキーを使用するときに、 を使用して が AWS KMS ユーザーに代わって HealthLake に送信するリクエストを追跡できます。ログの CloudTrail ログエントリは、userAgent フィールドで healthlake.amazonaws.com を表示し、 によって行われたリクエストを明確に区別します HealthLake。

次の例は CreateGrant、カスタマーマネージドキーによって暗号化されたデータにアクセス HealthLake するために によって呼び出される AWS KMS オペレーションをモニタリング DescribeKey するための GenerateDataKey、、、Decrypt、および の CloudTrail イベントです。

以下は、 CreateGrant を使用して HealthLake が顧客提供のKMSキーにアクセスできるようにし、がそのKMSキー HealthLake を使用して保管中のすべての顧客データを暗号化できるようにする方法を示しています。

ユーザーは、独自の grant を作成する必要はありません。 は、 に CreateGrant リクエストを送信することで、ユーザーに代わって grant AWS HealthLake を作成しますKMS。の許可 AWS KMS は、顧客アカウントの AWS KMS キー HealthLake へのアクセスを許可するために使用されます。

```
{
```

```
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EXAMPLEROLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Sampleuser01,
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLEKEYID",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "EXAMPLEROLE",
                "arn": "arn:aws:iam::111122223333:role/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Sampleuser01"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2021-06-30T19:33:37Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "healthlake.amazonaws.com"
    },
    "eventTime": "2021-06-30T20:31:15Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "healthlake.amazonaws.com",
    "userAgent": "healthlake.amazonaws.com",
    "requestParameters": {
        "operations": [
            "CreateGrant",
            "Decrypt",
            "DescribeKey",
            "Encrypt",
            "GenerateDataKey",
            "GenerateDataKeyWithoutPlaintext",
            "ReEncryptFrom",
            "ReEncryptTo",
            "RetireGrant"
        ],
        "granteePrincipal": "healthlake.us-east-1.amazonaws.com",
        "keyId": "arn:aws:kms:us-east-1:111122223333:key/EXAMPLE_KEY_ARN",
        "retiringPrincipal": "healthlake.us-east-1.amazonaws.com"
```

```
    },
    "responseElements": {
        "grantId": "EXAMPLE_ID_01"
    },
    "requestID": "EXAMPLE_ID_02",
    "eventID": "EXAMPLE_ID_03",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-east-1:111122223333:key/EXAMPLE_KEY_ARN"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

次の例は、 GenerateDataKey を使用して、ユーザーが保存する前にデータを暗号化するために必要なアクセス許可を持っていることを確認する方法を示しています。

```
          {
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EXAMPLEUSER",
        "arn": "arn:aws:sts::111122223333:assumed-role/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLEKEYID",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "EXAMPLEROLE",
                "arn": "arn:aws:iam::111122223333:role/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Sampleuser01"
            },
            "webIdFederationData": {},
            "attributes": {
```

```
                "creationDate": "2021-06-30T21:17:06Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "healthlake.amazonaws.com"
    },
    "eventTime": "2021-06-30T21:17:37Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "healthlake.amazonaws.com",
    "userAgent": "healthlake.amazonaws.com",
    "requestParameters": {
        "keySpec": "AES_256",
        "keyId": "arn:aws:kms:us-east-1:111122223333:key/EXAMPLE_KEY_ARN"
    },
    "responseElements": null,
    "requestID": "EXAMPLE_ID_01",
    "eventID": "EXAMPLE_ID_02",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-east-1:111122223333:key/EXAMPLE_KEY_ARN"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

次の例は、 が Decrypt オペレーションを HealthLake 呼び出して、保存された暗号化されたデータキーを使用して暗号化されたデータにアクセスする方法を示しています。

```
            {
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EXAMPLEUSER",
```

```
        "arn": "arn:aws:sts::111122223333:assumed-role/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLEKEYID",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "EXAMPLEROLE",
                "arn": "arn:aws:iam::111122223333:role/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Sampleuser01"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2021-06-30T21:17:06Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "healthlake.amazonaws.com"
    },
    "eventTime": "2021-06-30T21:21:59Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "healthlake.amazonaws.com",
    "userAgent": "healthlake.amazonaws.com",
    "requestParameters": {
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "keyId": "arn:aws:kms:us-east-1:111122223333:key/EXAMPLE_KEY_ARN"
    },
    "responseElements": null,
    "requestID": "EXAMPLE_ID_01",
    "eventID": "EXAMPLE_ID_02",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-east-1:111122223333:key/EXAMPLE_KEY_ARN"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
```

```
}
```

次の例は、 DescribeKey オペレーション HealthLake を使用して、 AWS KMS カスタマー所有 AWS KMS キーが使用可能な状態であるかどうかを確認し、機能していない場合のユーザーのトラブルシューティングを支援する方法を示しています。

```
         {
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EXAMPLEUSER",
        "arn": "arn:aws:sts::111122223333:assumed-role/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLEKEYID",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "EXAMPLEROLE",
                "arn": "arn:aws:iam::111122223333:role/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Sampleuser01"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2021-07-01T18:36:14Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "healthlake.amazonaws.com"
    },
    "eventTime": "2021-07-01T18:36:36Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DescribeKey",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "healthlake.amazonaws.com",
    "userAgent": "healthlake.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-east-1:111122223333:key/EXAMPLE_KEY_ARN"
    },
    "responseElements": null,
    "requestID": "EXAMPLE_ID_01",
```

```
    "eventID": "EXAMPLE_ID_02",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-east-1:111122223333:key/EXAMPLE_KEY_ARN"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

詳細

以下のリソースは、保管時のデータの暗号化に関する詳細情報を提供します。

AWS Key Management Service の基本概念の詳細については、 AWS KMS ドキュメントを参照してください。

AWS KMS ドキュメントのセキュリティのベストプラクティスの詳細については、「」を参照してください。

の転送中の暗号化 AWS HealthLake

AWS HealthLake は TLS 1.2 を使用して、パブリックエンドポイントおよびバックエンドサービスを介して転送中のデータを暗号化します。

の Identity and Access Management AWS HealthLake

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 AWS のサービス するのに役立つ です。 IAM管理者は、誰を認証 (サインイン) し、誰に HealthLake リソースの使用を承認する (アクセス許可を付与 AWS のサービス する) かを制御します。 IAMは、追加料金なしで使用できる です。

トピック

- 対象者

- アイデンティティを使用した認証
- ポリシーを使用したアクセスの管理
- と AWS HealthLake の連携方法 IAM
- のアイデンティティベースのポリシーの例 AWS HealthLake
- AWS の マネージドポリシー AWS HealthLake
- AWS HealthLake ID とアクセスのトラブルシューティング

対象者

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

サービスユーザー – HealthLake 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 HealthLake 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。 HealthLake 機能にアクセスできない場合は、「AWS HealthLake 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 HealthLake 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 HealthLake。サービスユーザーがどの HealthLake 機能やリソースにアクセスするかを決めるのは管理者の仕事です。その後で、サービスユーザーのアクセス許可を変更するために、IAM 管理者にリクエストを送信する必要があります。IAM の基本概念については、このページの情報を確認します。会社IAMで を使用する方法の詳細については HealthLake、「」を参照してくださいと AWS HealthLake の連携方法 IAM。

IAM 管理者 – IAM管理者は、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 HealthLake。で使用できる HealthLake 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のアイデンティティベースのポリシーの例 AWS HealthLake。

アイデンティティを使用した認証

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

ID ソースを介して提供された認証情報を使用して、フェデレーティッド ID AWS として にサインインできます。 AWS IAM Identity Center （ IAM Identity Center) ユーザー、会社のシングルサインオン

認証、Google または Facebook 認証情報は、フェデレーティッド ID の例です。フェデレーテッドアイデンティティ としてサインインする場合、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設定しています。フェデレーション AWS を使用して にアクセスすると、間接的にロールを引き受けることになります。

ユーザーのタイプに応じて、 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、「 AWS サインイン ユーザーガイド」の「 にサインインする方法 AWS アカウント」を参照してください。

AWS プログラムで にアクセスする場合、 はソフトウェア開発キット (SDK) とコマンドラインインターフェイス (CLI) AWS を提供し、認証情報を使用してリクエスト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に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「リクエストの署名バージョン 4」を参照してください。

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

AWS アカウント ルートユーザー

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

フェデレーテッドアイデンティティ

ベストプラクティスとして、管理者アクセスを必要とするユーザーを含む人間のユーザーに、ID プロバイダーとのフェデレーションを使用して一時的な認証情報 AWS のサービス を使用して にアクセスすることを要求します。

フェデレーティッド ID は、エンタープライズユーザーディレクトリ、ウェブ ID プロバイダー、、AWS Directory Serviceアイデンティティセンターディレクトリのユーザー、または ID ソースを通じて提供された認証情報 AWS のサービス を使用して にアクセスするユーザーです。フェデレー

ティッド ID がアクセスすると AWS アカウント、ロールを引き受け、ロールは一時的な認証情報を提供します。

アクセスを一元管理する場合は、 AWS IAM Identity Centerを使用することをお勧めします。IAM Identity Center でユーザーとグループを作成することも、独自の ID ソースのユーザーとグループのセットに接続して同期し、すべての AWS アカウント とアプリケーションで使用できるようにすることもできます。IAM Identity Center の詳細については、「 ユーザーガイド」のIAM「Identity Center とは」を参照してください。 AWS IAM Identity Center

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。可能な場合は、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、IAMユーザーガイドの「長期的な認証情報を必要とするユースケースのためにアクセスキーを定期的にローテーションする」を参照してください。

IAM グループは、IAM ユーザーの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、「 IAMユーザーガイド」のIAM「 ユーザーのユースケース」を参照してください。

IAM ロール

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

IAM ロールと一時認証情報は、次の状況で役立ちます。

- フェデレーションユーザーアクセス – フェデレーティッド ID に許可を割り当てるには、ロールを作成してそのロールの許可を定義します。フェデレーティッド ID が認証されると、その ID はロールに関連付けられ、ロールで定義されている許可が付与されます。フェデレーションのロールの詳細については、「 ユーザーガイド」の「サードパーティー ID プロバイダー (フェデレーション) のロールを作成する」を参照してください。 IAM 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、IAMIdentity 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 User Guide」の「Permission sets」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、自分のアカウントのリソースにアクセスすることを別のアカウントの信頼済みプリンシパルに許可できます。クロスアカウントアクセス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 のサービス、 (ロールをプロキシとして使用する代わりに) リソースに直接ポリシーをアタッチできます。クロスアカウントアクセス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、「 IAM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IAM」を参照してください。
- クロスサービスアクセス — 一部の は他の の機能 AWS のサービス を使用します AWS のサービス。例えば、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と、別のサービスで別のアクションを開始するアクションを実行できます。 FASは、 を呼び出すプリンシパルのアクセス許可をリクエストと組み合わせて使用し AWS のサービス、ダウンストリームサービス AWS のサービス にリクエストを送信します。 FAS リクエストは、他の AWS のサービス またはリソースとのやり取りを完了する必要があるリクエストをサービスが受信し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、IAM 内からサービスロールを作成、変更、削除できます。詳細については、「 IAMユーザーガイド」の「 にアクセス許可を委任するロールを作成する AWS のサービス」を参照してください。
  - サービスにリンクされたロール – サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行する

ロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント 、 サービスによって所有されます。IAM 管理者は、サービス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 インスタンス内でのアクセスキーの保存に推奨されます。 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、EC2 インスタンスで実行されるプログラムは一時認証情報を取得することができます。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」を参照してください。IAM

ポリシーを使用したアクセスの管理

でアクセスを制御する AWS には、ポリシーを作成し、ID AWS またはリソースにアタッチします。ポリシーは のオブジェクト AWS であり、アイデンティティまたはリソースに関連付けられると、そのアクセス許可を定義します。 は、プリンシパル (ユーザー、ルートユーザー、またはロールセッション) がリクエストを行うときに、これらのポリシー AWS を評価します。ポリシーでの権限により、リクエストが許可されるか拒否されるかが決まります。ほとんどのポリシーはJSON、ドキュメント AWS として に保存されます。JSON ポリシードキュメントの構造と内容の詳細については、「 IAMユーザーガイド」のJSON「ポリシーの概要」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。IAM管理者は、リソースで必要なアクションを実行するためのアクセス許可をユーザーに付与する IAMポリシーを作成できます。その後、管理者はロールに IAMポリシーを追加し、ユーザーはロールを引き受けることができます。

IAM ポリシーは、オペレーションの実行方法を問わず、アクションの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。ID ベースのポリシーを作成する方法については、「 IAMユーザーガイド」の「カスタマー管理ポリシーを使用してカスタムIAMアクセス許可を定義する」を参照してください。

アイデンティティベースポリシーは、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。管理ポリシーまたはインラインポリシーを選択する方法については、「 IAMユーザーガイド」の「管理ポリシーとインラインポリシーの選択」を参照してください。

リソースベースのポリシー

リソースベースのポリシーは、リソースにアタッチするJSONポリシードキュメントです。リソースベースポリシーの例としては、IAMロール信頼ポリシーや Amazon S3 バケットポリシーなどがあります。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスをコントロールできます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、または を含めることができます AWS のサービス。

リソースベースのポリシーは、そのサービス内にあるインラインポリシーです。リソースベースのポリシーIAMでは、 から AWS 管理ポリシーを使用することはできません。

アクセスコントロールリスト (ACLs)

アクセスコントロールリスト (ACLs) は、リソースへのアクセス許可を持つプリンシパル (アカウントメンバー、ユーザー、またはロール) を制御します。 ACLsは、ポリシードキュメント形式を使用しませんが、リソースベースのJSONポリシーに似ています。

Amazon S3、 AWS WAF、および Amazon VPCは、 をサポートするサービスの例ですACLs。の詳細についてはACLs、Amazon Simple Storage Service デベロッパーガイドの「アクセスコントロールリスト (ACL) の概要」を参照してください。

その他のポリシータイプ

AWS は、一般的で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) の最大アクセス許可を指定するJSONポリシーです AWS Organizations。 AWS Organizations は、ビジネスが所有する複数の AWS アカウント をグループ化して一元管理するためのサービスです。組織内のすべての機能を有効にすると、サービスコントロールポリシー (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。
- リソースコントロールポリシー (RCPs) – 所有する各リソースにアタッチされたJSONポリシーを更新することなく、アカウント内のリソースに使用可能なアクセス許可の上限を設定するために使用できるIAMポリシーRCPsです。は、メンバーアカウントのリソースのアクセス許可RCPを制限し、組織に属しているかどうかにかかわらず AWS アカウントのルートユーザー、 を含む ID の有効なアクセス許可に影響を与える可能性があります。をサポートする のリストを含む Organizations と の詳細についてはRCPs、「 AWS Organizations ユーザーガイドRCPs」の AWS のサービス「リソースコントロールポリシー (RCPs) 」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もあります。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。詳細については、IAM ユーザーガイドの「セッションポリシー」を参照してください。

複数のポリシータイプ

1 つのリクエストに複数のタイプのポリシーが適用されると、結果として作成される権限を理解するのがさらに難しくなります。複数のポリシータイプが関係する場合にリクエストを許可するかどうかを AWS が判断する方法については、「 IAMユーザーガイド」の「ポリシーの評価ロジック」を参照してください。

と AWS HealthLake の連携方法 IAM

IAM を使用して へのアクセスを管理する前に HealthLake、 で使用できるIAM機能を確認してください HealthLake。

IAM で使用できる の機能 AWS HealthLake

IAM 機能	HealthLake サポート
アイデンティティベースポリシー	はい
リソースベースのポリシー	いいえ
ポリシーアクション	はい
ポリシーリソース	あり
ポリシー条件キー	はい
ACLs	不可
ABAC (ポリシー内のタグ)	はい
一時的な認証情報	はい
プリンシパル権限	はい
サービスロール	はい
サービスリンクロール	いいえ

HealthLake および他の AWS のサービスがほとんどの IAM機能とどのように連携するかの概要を把握するには、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。

のアイデンティティベースのポリシー AWS HealthLake

アイデンティティベースのポリシーのサポート: あり

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、「 IAMユーザーガイド」の「カスタマー管理ポリシーを使用したカスタムIAMアクセス許可の定義」を参照してください。

IAM のアイデンティティベースのポリシーでは、許可または拒否するアクションとリソース、またアクションが許可または拒否される条件を指定できます。プリンシパルは、それが添付されているユーザーまたはロールに適用されるため、アイデンティティベースのポリシーでは指定できません。JSON ポリシーで使用できるすべての要素については、「 IAMユーザーガイド」の「 IAMJSONポリシー要素リファレンス」を参照してください。

のアイデンティティベースのポリシーの例 AWS HealthLake

HealthLake 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のアイデンティティベースのポリシーの例 AWS HealthLake。

内のリソースベースのポリシー AWS HealthLake

リソースベースのポリシーのサポート: なし

リソースベースのポリシーは、リソースにアタッチするJSONポリシードキュメントです。リソースベースポリシーの例としては、IAMロール信頼ポリシーや Amazon S3 バケットポリシーなどがあります。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスをコントロールできます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、または を含めることができます AWS のサービス。

クロスアカウントアクセスを有効にするには、全体のアカウント、または別のアカウントの IAM エンティティを、リソースベースのポリシーのプリンシパルとして指定します。リソースベースのポリシーにクロスアカウントのプリンシパルを追加しても、信頼関係は半分しか確立されない点に注意してください。プリンシパルとリソースが異なる場合 AWS アカウント、信頼されたアカウントのIAM管理者は、リソースへのアクセス許可をプリンシパルエンティティ (ユーザーまたはロール) に付与

する必要もあります。IAM 管理者は、アイデンティティベースのポリシーをエンティティにアタッチすることで権限を付与します。ただし、リソースベースのポリシーで、同じアカウントのプリンシパルへのアクセス権が付与されている場合は、アイデンティティベースのポリシーをさらに付与する必要はありません。詳細については、「 IAM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IAM」を参照してください。

のポリシーアクション AWS HealthLake

ポリシーアクションのサポート:あり

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。

JSON ポリシーの `Action`要素は、ポリシーでアクセスを許可または拒否するために使用できる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。一致するAPIオペレーションがないアクセス許可のみのアクションなど、いくつかの例外があります。また、ポリシーに複数のアクションが必要なオペレーションもあります。これらの追加アクションは、依存アクションと呼ばれます。

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権限を付与するポリシーで使用されます。

HealthLake アクションのリストを確認するには、「サービス認可リファレンス」の「 で定義されるアクション AWS HealthLake」を参照してください。

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 HealthLake を使用します。

```
healthlake
```

単一のステートメントで複数のアクションを指定するには、各アクションをカンマで区切ります。

```
"Action": [
      "healthlake:action1",
      "healthlake:action2"
         ]
```

HealthLake 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のアイデンティティベースのポリシーの例 AWS HealthLake。

のポリシーリソース AWS HealthLake

ポリシーリソースのサポート: あり

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。

Policy ResourceJSON要素は、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 要素を含める必要があります。ベストプラクティスとして、Amazon リソースネーム (ARN) を使用してリソースを指定します。これは、リソースレベルの許可と呼ばれる特定のリソースタイプをサポートするアクションに対して実行できます。

オペレーションのリスト化など、リソースレベルの権限をサポートしないアクションの場合は、ステートメントがすべてのリソースに適用されることを示すために、ワイルドカード (*) を使用します。

```
"Resource": "*"
```

HealthLake リソースタイプとその のリストを確認するにはARNs、「サービス認可リファレンス」の「 で定義されるリソース AWS HealthLake」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、「 で定義されるアクション AWS HealthLake」を参照してください。

HealthLake 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のアイデンティティベースのポリシーの例 AWS HealthLake。

のポリシー条件キー AWS HealthLake

サービス固有のポリシー条件キーのサポート: あり

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。

Condition 要素 (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 要素はオプションです。イコールや未満などの条件演算子を使用して条件式を作成することで、ポリシーの条件とリクエスト内の値を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 要素を指定する場合、または 1 つの Condition 要素に複数のキーを指定する場合、 AWS では AND 論理演算子を使用してそれらを評価します。1 つの条

件キーに複数の値を指定すると、 は論理ORオペレーションを使用して条件 AWS を評価します。ステートメントの権限が付与される前にすべての条件が満たされる必要があります。

条件を指定する際にプレースホルダー変数も使用できます。たとえば、IAM 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 IAM ユーザーアクセス許可を付与できます。詳細については、IAMユーザーガイドの「 IAMポリシーエレメント: 変数およびタグ」を参照してください。

AWS は、グローバル条件キーとサービス固有の条件キーをサポートしています。すべての AWS グローバル条件キーを確認するには、「 IAMユーザーガイド」のAWS 「 グローバル条件コンテキストキー」を参照してください。

HealthLake 条件キーのリストを確認するには、「サービス認可リファレンス」の「 の条件キー AWS HealthLake」を参照してください。条件キーを使用できるアクションとリソースについては、「 で定義されるアクション AWS HealthLake」を参照してください。

HealthLake 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のアイデンティティベースのポリシーの例 AWS HealthLake。

のアクセスコントロールリスト (ACLs) AWS HealthLake

をサポートACLs： いいえ

アクセスコントロールリスト (ACLs) は、リソースへのアクセス許可を持つプリンシパル (アカウントメンバー、ユーザー、またはロール) を制御します。 ACLsは、ポリシードキュメント形式を使用しませんが、リソースベースのJSONポリシーに似ています。

を使用した属性ベースのアクセスコントロール (ABAC ） AWS HealthLake

サポート ABAC (ポリシー内のタグ): はい

属性ベースのアクセスコントロール (ABAC) は、属性に基づいてアクセス許可を定義する認可戦略です。では AWS、これらの属性はタグと呼ばれます。タグは、IAMエンティティ (ユーザーまたはロール) および多くの AWS リソースにアタッチできます。エンティティとリソースのタグ付けは、 の最初のステップですABAC。次に、プリンシパルのタグがアクセスしようとしているリソースのタグと一致する場合に、オペレーションを許可するABACポリシーを設計します。

ABAC は、急速に成長している環境や、ポリシー管理が煩雑になる状況で役立ちます。

タグに基づいてアクセスを管理するには、`aws: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提供します。

サービスがすべてのリソースタイプに対して 3 つの条件キーすべてをサポートする場合、そのサービスの値はありです。サービスが一部のリソースタイプに対してのみ 3 つの条件キーのすべてをサポートする場合、値は「部分的」になります。

の詳細についてはABAC、「 IAMユーザーガイド」のABAC「認可によるアクセス許可の定義」を参照してください。をセットアップする手順を含むチュートリアルを表示するにはABAC、「 ユーザーガイド」の「属性ベースのアクセスコントロール (ABAC) を使用する」を参照してください。 IAM

での一時的な認証情報の使用 AWS HealthLake

一時的な認証情報のサポート: あり

一部の AWS のサービス は、一時的な認証情報を使用してサインインすると機能しません。一時的な認証情報と AWS のサービス 連携する などの詳細については、「 IAMユーザーガイド」のAWS のサービス「 と連携する IAM 」を参照してください。

ユーザー名とパスワード以外の AWS Management Console 方法で にサインインする場合は、一時的な認証情報を使用します。たとえば、会社のシングルサインオン (SSO) リンク AWS を使用してにアクセスすると、そのプロセスによって一時的な認証情報が自動的に作成されます。また、ユーザーとしてコンソールにサインインしてからロールを切り替える場合も、一時的な認証情報が自動的に作成されます。ロールの切り替えの詳細については、「 IAMユーザーガイド」の「ユーザーからIAMロールへの切り替え (コンソール)」を参照してください。

一時的な認証情報は、 AWS CLI または を使用して手動で作成できます AWS API。その後、これらの一時的な認証情報を使用してアクセスすることができます AWS。 AWS では、長期的なアクセスキーを使用する代わりに、一時的な認証情報を動的に生成することをお勧めします。詳細については、「」の「一時的なセキュリティ認証情報IAM」を参照してください。

のクロスサービスプリンシパル許可 AWS HealthLake

転送アクセスセッションをサポート (FAS): はい

ユーザーIAM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と、別のサービスで別のアクションを開始するアクションを実行できます。 FASは、 を呼び出すプリンシパルのアクセス許可と AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストのリクエストを使用します。 FAS リクエストは、他の AWS のサービス またはリソースとのやり取りを完了する必要があるリクエストをサービスが受信し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。

AWS HealthLake のサービスロール

サービスロールのサポート: あり

サービスロールとは、サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロール です。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細については、「 IAMユーザーガイド」の「 にアクセス許可を委任するロールを作成する AWS のサービス」を参照してください。

へのフルアクセスに必要なサービスロールとインラインポリシーについては AWS HealthLake、「」を参照してください の使用を開始するためのアクセス許可の設定 AWS HealthLake。

> **Warning**
>
> サービスロールのアクセス許可を変更すると、 HealthLake 機能が破損する可能性があります。が指示する場合以外 HealthLake は、サービスロールを編集しないでください。

のサービスにリンクされたロール AWS HealthLake

サービスにリンクされたロールのサポート: なし

サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント 、 サービスによって所有されます。IAM 管理者は、サービス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。

サービスにリンクされたロールの作成または管理の詳細については、「IAMと連携するAWS サービス」を参照してください。表の中から、[サービスリンクロール] 列に `Yes` と記載されたサービスを見つけます。サービスにリンクされたロールに関するドキュメントをサービスで表示するには、[はい] リンクを選択します。

のアイデンティティベースのポリシーの例 AWS HealthLake

デフォルトでは、 ユーザーおよびロールには、 HealthLake 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 Command Line Interface （ AWS CLI ） AWS Management Console、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、リソースで必要なアクションを実行するためのアクセス許可をユーザーに付与する IAMポリシーを作成できます。その後、管理者はロールに IAMポリシーを追加し、ユーザーはロールを引き受けることができます。

これらのJSON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IAM「ポリシーの作成 (コンソール) 」を参照してください。

で定義されるアクションとリソースタイプの詳細については HealthLake、「サービス認可リファレンスARNs」の「 のアクション、リソース、および条件キー AWS HealthLake」を参照してください。

トピック

- ポリシーのベストプラクティス
- AWS HealthLake コンソールを使用する
- のデータ AWS HealthLake ストアへのアクセス Amazon Athena
- ユーザー自身のアクセス許可を表示することをユーザーに許可する

ポリシーのベストプラクティス

ID ベースのポリシーは、ユーザーのアカウントで誰かが HealthLake リソースを作成、アクセス、または削除できる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 管理ポリシーを開始し、最小特権のアクセス許可に移行する – ユーザーとワークロードにアクセス許可を付与するには、多くの一般的なユースケースにアクセス許可を付与する AWS 管理ポリシーを使用します。これらは で使用できます AWS アカウント。ユースケースに固有の AWS カスタマー管理ポリシーを定義して、アクセス許可をさらに減らすことをお勧めします。詳細については、「IAMユーザーガイド」の「AWS マネージドポリシー」または「AWS ジョブ機能の管理ポリシー」を参照してください。
- 最小特権を適用する – IAM ポリシーでアクセス許可を設定する場合は、タスクの実行に必要なアクセス許可のみを付与します。これを行うには、特定の条件下で特定のリソースに対して実行できるアクションを定義します。これは、最小特権アクセス許可とも呼ばれています。IAM を使用してアクセス許可を適用する方法の詳細については、IAM ユーザー ガイドの「IAM のポリシーとアクセス許可」を参照してください。
- IAMポリシーで条件を使用してアクセスをさらに制限する – ポリシーに条件を追加して、アクションやリソースへのアクセスを制限できます。たとえば、ポリシー条件を記述して、すべてのリクエストを を使用して送信するように指定できますSSL。条件を使用して、サービスアクションが などの特定の を通じて使用される場合に AWS のサービス、サービスアクションへのアクセ

スを許可することもできます AWS CloudFormation。詳細については、「 IAMユーザーガイド」のIAMJSON「ポリシー要素: 条件」を参照してください。

- IAM Access Analyzer を使用してIAMポリシーを検証し、安全で機能的なアクセス許可を確保する – IAM Access Analyzer は、ポリシーがポリシー言語 (JSON) とIAMベストプラクティスに準拠するように、新規および既存のIAMポリシーを検証します。 IAMAccess Analyzer には、安全で機能的なポリシーの作成に役立つ 100 を超えるポリシーチェックと実用的な推奨事項が用意されています。詳細については、「 IAMユーザーガイド」のIAM「Access Analyzer によるポリシーの検証」を参照してください。
- 多要素認証を要求する (MFA) – でIAMユーザーまたはルートユーザーを必要とするシナリオがある場合は AWS アカウント、セキュリティを強化MFAするために をオンにします。API オペレーションが呼び出されるMFAタイミングを要求するには、ポリシーにMFA条件を追加します。詳細については、「 IAMユーザーガイド」の「 を使用した安全なAPIアクセスMFA」を参照してください。

IAM でのベストプラクティスの詳細については、「IAMユーザーガイド」の「IAM でのセキュリティのベストプラクティス」を参照してください。

AWS HealthLake コンソールを使用する

AWS HealthLake コンソールにアクセスするには、一連の最小限のアクセス許可が必要です。これらのアクセス許可により、 内の HealthLake 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。

AWS CLI または のみを呼び出すユーザーには、最小限のコンソールアクセス許可を付与する必要はありません 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。

へのフルアクセスについては HealthLake、 IAM ユーザーまたはロールに次のポリシーをアタッチします: `AmazonHealthLakeFullAccess`および `AWSLakeFormationDataAdmin`。また、サービスロールである HealthLake インラインポリシーをアタッチする必要があります。サービスロールとは、サービスがユーザーに代わってアクションを実行するために引き受ける IAMロール です。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細については、「 IAMユーザーガイド」の「 にアクセス許可を委任するロールを作成する AWS のサービス」を参照してください。必要なサービスロールを作成するインラインポリシーについては、「」を参照してくださいの使用を開始するためのアクセス許可の設定 AWS HealthLake。また、 AWS Lake Formation コ

ンソールまたは を使用してCLI、 AWS Lake Formation Data Lake HealthLake 管理者に管理者を割り当てる必要があります。詳細については、「の使用を開始するためのアクセス許可の設定 AWS HealthLake」を参照してください。

のデータ AWS HealthLake ストアへのアクセス Amazon Athena

のデータ HealthLake ストアへのアクセス権をユーザーとロールに付与する場合は Amazon Athena、ロールまたはユーザーに次のIAMポリシーをアタッチします。 `AmazonAthenaFullAccess`および`AmazonS3FullAccess`。 `Select` および アクセス`Describe`許可は、 AWS Lake Formation コンソールまたは を介して AWS Lake Formation 管理者によって付与 AWS Lake Formation される AWS Lake Formation、 によって管理されるテーブルにも必要ですCLI。詳細については、「の使用を開始するためのアクセス許可の設定 AWS HealthLake」を参照してください

ユーザー自身のアクセス許可を表示すること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 IAM 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 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 AWS API。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
```

```
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS の マネージドポリシー AWS HealthLake

AWS 管理ポリシーは、 によって作成および管理されるスタンドアロンポリシーです AWS。 AWS 管理ポリシーは、多くの一般的なユースケースにアクセス許可を付与するように設計されているため、ユーザー、グループ、ロールにアクセス許可の割り当てを開始できます。

AWS 管理ポリシーは、すべての AWS お客様が使用できるため、特定のユースケースに対して最小特権のアクセス許可を付与しない場合があることに注意してください。ユースケース別にカスタマーマネージドポリシーを定義して、マネージドポリシーを絞り込むことをお勧めします。

AWS 管理ポリシーで定義されているアクセス許可は変更できません。が AWS 管理ポリシーで定義されたアクセス許可 AWS を更新すると、ポリシーがアタッチされているすべてのプリンシパル ID (ユーザー、グループ、ロール) が更新されます。 AWS は、新しい AWS のサービス が起動されたとき、または既存のサービスで新しいAPIオペレーションが利用可能になったときに、 AWS 管理ポリシーを更新する可能性が最も高くなります。

詳細については、「IAM ユーザーガイド」の「AWS 管理ポリシー」を参照してください。

AWS 管理ポリシー： AmazonHealthLakeFullAccess

このAmazonHealthLakeFullAccessポリシーは、 へのフルアクセスを提供します HealthLake。このポリシーをユーザーまたはロールにアタッチすると、ユーザーは HealthLake を使用して のデータへのアクセス、クエリ、インポート、エクスポートを行うことができます HealthLake。で多くの

一般的なアクションを実行するには HealthLake、ユーザーまたはロールにポリシーを追加する必要があります。詳細については、の使用を開始するためのアクセス許可の設定 AWS HealthLake「」および「 HealthLake オペレーションとアクセス許可」を参照してください。

AmazonHealthLakeFullAccess ポリシーを IAM ID にアタッチできます。

このポリシーは、ユーザーとロールが でクエリ、検索、インポート、エクスポートを実行できるようにする*administrative and contributor*アクセス許可を付与します。また HealthLake、これらのアクセス許可を持つユーザーとロールに代わって が HealthLake アクションを実行できるようにします。

許可の詳細

このポリシーには、次のステートメントが含まれます。

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Action": [
    "healthlake:*",
    "s3:ListAllMyBuckets",
    "s3:ListBucket",
    "s3:GetBucketLocation",
    "iam:ListRoles"
   ],
   "Resource": "*",
   "Effect": "Allow"
  },
  {
   "Effect": "Allow",
   "Action": "iam:PassRole",
   "Resource": "*",
   "Condition": {
    "StringEquals": {
     "iam:PassedToService": "healthlake.amazonaws.com"
    }
   }
  }
 ]
```

```
}
```

AWS 管理ポリシー： AmazonHealthLakeReadOnlyAccess

AmazonHealthLakeReadOnlyAccess ポリシーは、他の AWS サービスの および関連リソースへの読み取り専用アクセス HealthLake とアクセス許可を付与します。このポリシーは、 HealthLake データストアをクエリおよび表示する権限を付与するユーザーに適用しますが、データストアを作成または変更する権限は付与しません。

AmazonHealthLakeReadOnlyAccess ポリシーを IAM ID にアタッチできます。

このポリシーは、ユーザーとロールにクエリを許可する*read-only*アクセス許可を付与します HealthLake。

許可の詳細

このポリシーには、次のステートメントが含まれます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "healthlake:ListFHIRDatastores",
                "healthlake:DescribeFHIRDatastore",
                "healthlake:DescribeFHIRImportJob",
                "healthlake:DescribeFHIRExportJob",
                "healthlake:GetCapabilities",
                "healthlake:ReadResource",
                "healthlake:SearchWithGet",
                "healthlake:SearchWithPost",
                "healthlake:SearchEverything"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

HealthLake オペレーションとアクセス許可

次の表に、 の一般的なオペレーション HealthLake と、それらを実行するために必要なアクセス許可を示します。

HealthLake オペレーション	必要なアクセス許可
でデータストアを作成する HealthLake	`AmazonHealthLakeFullAccess` によって管理される `AmazonLakeFormationDataAdmin` 、、インラインポリシー、AWS Lake Formation 管理者権限 AWS Lake Formation
でデータストアを削除する HealthLake	`AmazonHealthLakeFullAccess` によって管理される 、`AmazonLakeFormationDataAdmin` 、インラインポリシー、AWS Lake Formation 管理者権限 AWS Lake Formation
のデータストアを一覧表示、検索、またはクエリする HealthLake	`AmazonHealthLakeReadOnlyAccess`
を使用してデータストアをクエリする Amazon Athena	`AmazonAthenaFullAccess` によって管理されるテーブルに対する 、`AmazonS3FullAccess` 、 AWS Lake Formation `Select`および アクセス`Describe`許可 AWS Lake Formation
からデータをインポートする HealthLake	「インポートジョブのアクセス許可の設定」を参照してください。
からのデータのエクスポート HealthLake	「データストアからのファイルのエクスポート (AWS SDKs) 」を参照してください。

HealthLake AWS 管理ポリシーの更新

このサービスがこれらの変更の追跡を開始した時点 HealthLake からの の AWS マネージドポリシーの更新に関する詳細を表示します。このページの変更に関する自動アラートについては、HealthLake ドキュメント履歴ページのRSSフィードにサブスクライブしてください。

変更	説明	日付
AmazonHealthLakeFullAccess	`AmazonHealthLakeFullAccess` へのフルアクセスを許可するために必要な ポリシー HealthLake。	2022 年 11 月 14 日
AmazonHealthLakeReadOnlyAccess	`AmazonHealthLakeReadOnlyAccess` への読み取り専用アクセスに必要な ポリシー HealthLake。	2022 年 11 月 14 日
HealthLake が変更の追跡を開始しました	HealthLake が AWS マネージドポリシーの変更の追跡を開始しました。	2022 年 11 月 14 日

AWS HealthLake ID とアクセスのトラブルシューティング

次の情報は、 と の使用時に発生する可能性がある一般的な問題の診断 HealthLake と修正に役立ちますIAM。

トピック

- でアクションを実行する権限がない AWS HealthLake
- iam を実行する権限がありません。PassRole
- AWS アカウント外のユーザーに AWS HealthLake 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 HealthLake

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。担当の管理者はお客様のユーザー名とパスワードを発行した人です。

次の例のエラーは、mateojacksonIAMユーザーがコンソールを使用して架空の*my-example-widget*リソースの詳細を表示しようとしているが、架空のhealthlake:*GetWidget*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:
 healthlake:GetWidget on resource: my-example-widget
```

この場合、Mateo は、healthlake:*GetWidget* アクションを使用して *my-example-widget* リソースへのアクセスが許可されるように、管理者にポリシーの更新を依頼します。

iam を実行する権限がありません。PassRole

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 HealthLake にロールを渡すことができるようにする必要があります。

一部の AWS のサービス では、新しいサービスロールまたはサービスにリンクされたロールを作成する代わりに、既存のロールをそのサービスに渡すことができます。そのためには、サービスにロールを渡す権限が必要です。

以下の例のエラーは、marymajor という IAM ユーザーがコンソールを使用して HealthLake でアクションを実行しようする場合に発生します。ただし、このアクションをサービスが実行するには、サービスロールから付与された権限が必要です。メアリーには、ロールをサービスに渡す許可がありません。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

この場合、Mary のポリシーを更新してメアリーに iam:PassRole アクションの実行を許可する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン認証情報を提供した担当者が管理者です。

AWS アカウント外のユーザーに AWS HealthLake 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 (ACLs) をサポートするサービスでは、そ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 HealthLake をサポートしているかどうかを確認するには、「」を参照してくださいと AWS HealthLake の連携方法 IAM。
- 所有 AWS アカウント する 全体のリソースへのアクセスを提供する方法については、「 IAMユーザーガイド」の「所有 AWS アカウント する別の の IAM ユーザーへのアクセスを提供する」を参照してください。
- リソースへのアクセスをサードパーティーに提供する方法については AWS アカウント、「 IAM ユーザーガイド」の「サードパーティー AWS アカウント が所有する へのアクセスを提供する」を参照してください。
- ID フェデレーションを介してアクセスを提供する方法については、「IAM ユーザーガイド」の「外部で認証されたユーザー (ID フェデレーション) へのアクセスの許可」を参照してください。
- クロスアカウントアクセスでのロールとリソースベースのポリシーの使用の違いについては、「 IAM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IAM」を参照してください。

AWS CloudTrailを使用した AWS HealthLake APIコールのログ記録

AWS HealthLake は、ユーザー AWS CloudTrail、ロール、または のサービスによって実行されたアクションを記録する AWS サービスである と統合されています HealthLake。 は、 のすべてのAPI呼び出しをイベント HealthLake として CloudTrail キャプチャします。キャプチャされた呼び出しには、 HealthLake コンソールからの呼び出しと、 HealthLake APIオペレーションへのコード呼び出しが含まれます。証跡を作成する場合は、 イベントなど、Amazon S3 バケットへのイベントの継続的な配信 CloudTrailを有効にすることができます HealthLake。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。によって収集された情報を使用して CloudTrail、 に対するリクエスト HealthLake、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

詳細については CloudTrail、「 AWS CloudTrail ユーザーガイド」を参照してください。

AWS HealthLake CloudTrail 内の情報

CloudTrail AWS アカウントを作成すると、 がアカウントで有効になります。でアクティビティが発生すると HealthLake、そのアクティビティは CloudTrail イベント履歴の他の AWS サービスイベントとともにイベントに記録されます。 AWS アカウントで最近のイベントを表示、検索、ダウンロードできます。詳細については、「イベント履歴を使用した CloudTrail イベントの表示」を参照してください。

のイベントなど、 AWS アカウントのイベントの継続的な記録については HealthLake、証跡を作成します。証跡により 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、コンソールで証跡を作成すると、すべての AWS リージョンに証跡が適用されます。証跡は、 AWS パーティション内のすべてのリージョンからのイベントをログに記録し、指定した Amazon S3 バケットにログファイルを配信します。さらに、 CloudTrail ログで収集されたイベントデータをより詳細に分析し、それに基づいて行動するように、他の AWS サービスを設定できます。詳細については、次を参照してください:

- 証跡の作成のための概要
- CloudTrail サポートされているサービスと統合
- の Amazon SNS 通知の設定 CloudTrail
- 複数のリージョンからの CloudTrail ログファイルの受信と複数のアカウントからのログファイルの受信 CloudTrail

すべての HealthLake アクションは によってログに記録 CloudTrail され、「 HealthLake APIリファレンス」と「このデベロッパーガイド」にFHIRREST、 を使用して実行されるアクションが記載されていますAPI。たとえば、次のアクションを呼び出すと、 CloudTrail ログファイルにエントリが生成されます。

- `DescribeFHIRImportJob`
- `DescribeFHIRExportJob`
- `StartFHIRImportJob`
- `ListFHIRImportJobs`
- `StartFHIRExportJob`
- `ListFHIRExportJobs`
- `CreateFHIRDatastore`
- `ListFHIRDatastores`
- `DeleteFHIRDatastore`
- `DescribeFHIRDatastore`
- `UpdateResource`
- `CreateResource`
- `DeleteResource`
- `ReadResource`

- GetCapabilities
- SearchWithGet
- SearchWithPost

各イベントまたはログエントリには、誰がリクエストを生成したかという情報が含まれます。アイデンティティ情報は、以下を判別するのに役立ちます:

- リクエストが root または AWS Identity and Access Management（ IAM) ユーザー認証情報を使用して行われたかどうか。
- リクエストがロールまたはフェデレーションユーザーのテンポラリなセキュリティ認証情報を使用して行われたかどうか。
- リクエストが別の AWS サービスによって行われたかどうか。

詳細については、CloudTrail userIdentity「 要素」を参照してください。

AWS HealthLake ログファイルエントリについて

証跡は、指定した Amazon S3 バケットにイベントをログファイルとして配信できるようにする設定です。 CloudTrail ログファイルには、1 つ以上のログエントリが含まれます。イベントは任意のソースからの単一のリクエストを表し、リクエストされたアクション、アクションの日時、リクエストパラメータなどに関する情報が含まれます。 CloudTrail ログファイルはパブリック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。

次の例は、 CreateFHIRDatastoreアクションを示す CloudTrail ログエントリを示しています。

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA2B3ZHOADD2OJ4AHJX:git
 full_access_iam_role580074395690222150",
        "arn": "arn:aws:sts::691207106566:assumed-role/
colossusfrontend_full_access_iam_role/_iam_role580074395690222150",
        "accountId": "AccountID",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
```

```
                "principalId": "AROA2B3ZHOADD2OJ4AHJX",
                "arn": "arn:aws:iam::691207106566:role/full_access_iam_role",
                "accountId": "AccountID",
                "userName": "full_access_iam_role"
            },
            "webIdFederationData": {

            },
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2020-11-20T00:08:15Z"
            }
        }
    },
    "eventTime": "2020-11-20T00:08:16Z",
    "eventSource": "healthlake.amazonaws.com",
    "eventName": "CreateFHIRDatastore",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "3.213.247.1",
    "userAgent": "Coral/Netty4",
    "requestParameters": {
        "datastoreName":
 "testCreateFHIRDatastore_GBYAZFCLLBLSUTOYYFQZRLBLQJNFOYQVRPZBOJAIIUAHICAEAGIWLNVQEYAMSXVWMBLXC
        "datastoreTypeVersion": "R4",
        "clientToken": "d737ffe0-14dd-44cc-9f0a-fdf59b26c66b"
    },
    "responseElements": {
        "datastoreId": "datastoreID",
        "datastoreArn": "arn:aws:healthlake:us-
east-1:691207106566:datastore/55576c487ff4975262b10d1d65eb4509",
        "datastoreStatus": "CREATING",
        "datastoreEndpoint": "datastore_endpoint/"
    },
    "requestID": "68e62bdd-d2d4-44c1-af69-e6f055a69f99",
    "eventID": "7ef483dc-5dca-469e-823a-7d9e3a7fe924",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "691207106566"
}
```

のコンプライアンス検証 AWS HealthLake

サードパーティーの監査者は、複数の コンプライアンスプログラム AWS HealthLake の一環として のセキュリティと AWS コンプライアンスを評価します。これには HealthLake が含まれます HIPAA。

AWS のサービス が特定のコンプライアンスプログラムの範囲内にあるかどうかを確認するには、コンプライアンスAWS のサービス プログラムによる範囲内コンプライアンスを参照し、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、「Downloading AWS Artifact Reports 」を参照してください。

を使用する際のお客様のコンプライアンス責任 AWS のサービス は、お客様のデータの機密性、貴社のコンプライアンス目的、適用される法律および規制によって決まります。 は、コンプライアンスに役立つ以下のリソース AWS を提供します。

- セキュリティのコンプライアンスとガバナンス – これらのソリューション実装ガイドでは、アーキテクチャ上の考慮事項について説明し、セキュリティとコンプライアンスの機能をデプロイする手順を示します。
- 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 のサービス 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。このガイドは、ガイダンスを保護し AWS のサービス 、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) を含む) 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たものです。

- 「 デベロッパーガイド」の「 ルールによるリソースの評価」 – この AWS Config サービスは、リソース設定が社内プラクティス、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 のサービス 、セキュリティ状態を包括的に把握できます AWS。Security Hub では、セキュリティコントロールを使用して AWS リソースを評価し、セキュリティ業界標準とベストプラクティスに対するコンプライアンスをチェックします。サポートされているサービスとコントロールのリストについては、「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、不審なアクティビティや悪意のあるアクティビティがないか環境を監視することで、、ワークロード、コンテナ、データに対する潜在的な脅威 AWS のサービス 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 のサービス 、 AWS 使用状況を継続的に監査し、リスクの管理方法と規制や業界標準への準拠を簡素化できます。

の耐障害性 AWS HealthLake

AWS グローバルインフラストラクチャは、 AWS リージョンとアベイラビリティーゾーンを中心に構築されています。 AWS リージョンは、低レイテンシー、高スループット、および高度に冗長なネットワークで接続された、物理的に分離および分離された複数のアベイラビリティーゾーンを提供します。アベイラビリティーゾーンでは、ゾーン間で中断することなく自動的にフェイルオーバーするアプリケーションとデータベースを設計および運用することができます。アベイラビリティーゾーンは、従来の単一または複数のデータセンターインフラストラクチャよりも可用性が高く、フォールトトレラントで、スケーラブルです。

AWS リージョンとアベイラビリティーゾーンの詳細については、AWS 「 グローバルインフラストラクチャ」を参照してください。

グローバル AWS インフラストラクチャに加えて、 HealthLake には、データの耐障害性とバックアップのニーズに対応できるように、いくつかの機能が用意されています。

AWS HealthLake のインフラストラクチャセキュリティ

マネージドサービスである AWS HealthLake は、ホワイトペーパー「Amazon Web Services: セキュリティプロセスの概要」に記載されている AWS グローバルネットワークセキュリティの手順で保護されています。

が AWS 公開したAPI呼び出しを使用して、ネットワーク HealthLake 経由で にアクセスします。クライアントは Transport Layer Security (TLS) 1.0 以降をサポートしている必要があります。1.2 TLS 以降をお勧めします。クライアントは、一時 Diffie-Hellman (PFS) や楕円曲線一時 Diffie-Hellman () など、完全な前方秘匿性 (DHE) を持つ暗号スイートもサポートする必要がありますECDHE。これらのモードは、Java 7 以降など、ほとんどの最新システムでサポートされています。

また、リクエストは、アクセスキー ID と、IAM プリンシパルに関連付けられているシークレットの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、一時的なセキュリティ認証情報を生成し、リクエストに署名することもできます。

AWS HealthLakeでのセキュリティのベストプラクティス

AWS HealthLake には、独自のセキュリティポリシーを開発および実装する際に考慮すべきさまざまなセキュリティ機能が用意されています。以下のベストプラクティスは一般的なガイドラインであり、完全なセキュリティソリューションを説明するものではありません。これらのベストプラクティスはお客様の環境に必ずしも適切または十分でない可能性があるので、処方箋ではなく、あくまで有用な考慮事項とお考えください。

- 最小特権アクセスの実装
- 可能な限り、 Customer-Managed-Keys (CMKs) を使用してデータを暗号化します。の詳細についてはCMKs、「Amazon Key Management Service」を参照してください。
- データストアPIIで PHIまたは をクエリGETするときはPOST、 で検索ではなく、 で検索を使用します。
- 機密性の高い重要な監査機能へのアクセスを制限します。
- 更新または一括インポートを使用してリソースを作成する場合はAPIs、データストアやジョブの名前PIIを含む PHIまたは を、表示可能なフィールドまたは論理 FHIR ID () で使用しないでくださいLID。
- 作成、読み取り、更新、削除、または検索リクエストを送信するときは、 HTTPヘッダーPHIで を使用しないでください。
- AWS CloudTrail が AWS HealthLake 使用を監査し、予期しないアクティビティがないことを確認します。
- Amazon S3 バケットを安全に使用するためのベストプラクティスを確認します。詳細については、Amazon S3ユーザーガイド」の「セキュリティのベストプラクティス」を参照してください。

AWS HealthLake エンドポイントとクォータ

以下のセクションには、 AWS HealthLake クォータとエンドポイントに関する情報が含まれています。調整可能なクォータの場合、Service Quotas コンソールを使用してクォータの引き上げをリクエストできます。詳細については、「Service Quotas ユーザーガイド」の「クォータの引き上げのリクエスト」を参照してください。

サービスエンドポイント

この表は、特定のリージョンで利用可能な HealthLake サービスエンドポイントを示しています。

リージョン名	リージョン	エンドポイント	プロトコル	
米国東部 (オハイオ)	us-east-2	healthlake.us-east-2.amazonaws.com healthlake-fips.us-east-2.amazonaws.com	HTTPS HTTPS	
米国東部 (バージニア北部)	us-east-1	healthlake.us-east-1.amazonaws.com healthlake-fips.us-east-1.amazonaws.com	HTTPS HTTPS	
米国西部 (オレゴン)	us-west-2	healthlake.us-west-2.amazonaws.com healthlake-fips.us-west-2.amazonaws.com	HTTPS HTTPS	
アジアパシフィック (ムンバイ)	ap-south-1	healthlake.ap-south-1.amazonaws.com	HTTPS	
アジアパシフィック (シドニー)	ap-southeast-2	healthlake.ap-southeast-2.amazonaws.com	HTTPS	

リージョン名	リージョン	エンドポイント	プロトコル	
欧州 (ロンドン)	eu-west-2	healthlake.eu-west-2.amazonaws.com	HTTPS	

のサービスクォータ HealthLake

のデフォルトのクォータは次のとおりです HealthLake。

名前	デフォルト	引き上げ可能	説明
医療メモの文字数	サポートされている各リージョン: 10,000	いいえ	DocumentReference リソースタイプ (POST/PUT リクエスト) 内の個々の医療メモの最大文字数。
同時 S tartFHIRImportジョブの数	サポートされている各リージョン: 1	[いいえ]	同時 S tartFHIRImport ジョブの最大数。
concurrentStartFHIRExportJob ジョブの数	サポートされている各リージョン: 1	[いいえ]	同時 S tartFHIRExport ジョブの最大数。
アカウントごとのデータストアの数	サポートされている各リージョン: 10	可能	アカウントごとの、アクティブなデータストアのデフォルトの最大数。

名前	デフォルト	引き上げ可能	説明
S tartFHIRImportジョブ内のファイル数	サポートされている各リージョン: 10,000	いいえ	S tartFHIRImportジョブ内のファイルの最大数。
バンドルあたりのリソース数	サポートされている各リージョン: 160	いいえ	バンドルリクエストにおけるリソースの最大数。
アカウントごとの Bundle リクエストのレート	サポートされている各リージョン: 20	可能	アカウントごとに 1 秒あたりに実行できる POST Bundle リクエストの最大数。
データストアごとの Bundle リクエストのレート	サポートされている各リージョン: 10	可能	データストアごとに 1 秒あたりに実行できる POST Bundle リクエストの最大数。2023 年 8 月 21 日より前に作成されたデータストアの場合、1 秒あたり 1 リクエストに制限されます。
アカウントDELETEあたりの を使用した C ancelFHIRExportジョブリクエストのレート	サポートされている各リージョン: 1	[いいえ]	アカウントごとに 1 分あたりに実行できる DELETE C ancelFHIR Exportジョブリクエストの最大数。

名前	デフォルト	引き上げ可能	説明
アカウントあたりの C reateFHIR Datastore リクエストのレート	サポートされている各リージョン: 1	[いいえ]	アカウントごとに 1 分あたりに実行できる C reateFHIRDatastore リクエストの最大数。
アカウントごとの DELETE リクエストのレート	サポートされている各リージョン: 2,000	可能	アカウントごとに 1 秒あたりに実行できる DELETEリクエストの最大数。
データストアあたりのDELETEリクエストのレート	サポートされている各リージョン: 1,000	可能	データストアごとに 1 秒あたりに実行できる DELETEリクエストの最大数。2023 年 8 月 21 日より前に作成されたデータストアの場合、1 秒あたり 100 リクエストに制限されます。
アカウントあたりの D eleteFHIR Datastore リクエストのレート	サポートされている各リージョン: 1	[いいえ]	アカウントごとに 1 分あたりに実行できる D eleteFHIRDatastore リクエストの最大数。
アカウントあたりの D escribeFH IRDatastore リクエストのレート	サポートされている各リージョン: 10	いいえ	アカウントごとに 1 秒あたりに実行できる D escribeFHIRDatastore リクエストの最大数。

名前	デフォルト	引き上げ可能	説明
アカウントあたりの D escribeFHIRExportジョブリクエストのレート	サポートされている各リージョン: 10	いいえ	アカウントごとに 1 秒あたりに実行できる DescribeFHIRExportジョブリクエストの最大数。
アカウントGETあたりの を使用した DescribeFHIRExportジョブリクエストのレート	サポートされている各リージョン: 10	いいえ	アカウントごとに 1 秒あたりGETに実行できる DescribeFHIRExportジョブリクエストの最大数。
アカウントあたりの D escribeFHIRImportジョブリクエストのレート	サポートされている各リージョン: 10	いいえ	アカウントごとに 1 秒あたりに実行できる DescribeFHIRImportジョブリクエストの最大数。
アカウントごとの Discovery リクエストのレート	サポートされている各リージョン: 10	いいえ	アカウントごとに 1 分あたりに実行できる Discovery リクエストの最大数。
アカウントごとの GET リクエストのレート	サポートされている各リージョン: 6,000	可能	アカウントごとに 1 秒あたりに実行できるGETリクエストの最大数。

名前	デフォルト	引き上げ可能	説明
データストアあたりのGETリクエストのレート	サポートされている各リージョン: 3,000	可能	データストアごとに 1 秒あたりに実行できるGETリクエストの最大数。2023 年 8 月 21 日より前に作成されたデータストアの場合、1 秒あたり 100 リクエストに制限されます。
アカウントあたりの GetCapabilities リクエストのレート	サポートされている各リージョン: 10	いいえ	アカウントごとに 1 秒あたりに実行できる GetCapabilities リクエストの最大数。
アカウントあたりの L istFHIRDatastores リクエストのレート	サポートされている各リージョン: 10	いいえ	アカウントごとに 1 秒あたりに実行できる L istFHIRDatastores リクエストの最大数。
アカウントあたりの L istFHIRExport ジョブリクエストのレート	サポートされている各リージョン: 10	いいえ	アカウントごとに 1 秒あたりに実行できる L istFHIRExportジョブリクエストの最大数。
アカウントあたりの L istFHIRImport ジョブリクエストのレート	サポートされている各リージョン: 10	いいえ	アカウントごとに 1 秒あたりに実行できる L istFHIRImportジョブリクエストの最大数。

名前	デフォルト	引き上げ可能	説明
アカウントあたりの ListTagsforResource リクエストのレート	サポートされている各リージョン: 10	いいえ	アカウントごとに 1 秒あたりに実行できる ListTagsforResource リクエストの最大数。
アカウントごとの POST リクエストのレート	サポートされている各リージョン: 2,000	可能	アカウントごとに 1 秒あたりに実行できるPOST リクエストの最大数。
データストアあたりのPOSTリクエストのレート	サポートされている各リージョン: 1,000	可能	データストアごとに 1 秒あたりに実行できるPOSTリクエストの最大数。2023 年 8 月 21 日より前に作成されたデータストアの場合、1 秒あたり 100 リクエストに制限されます。
アカウントごとの PUT リクエストのレート	サポートされている各リージョン: 2,000	可能	アカウントごとに 1 秒あたりに実行できるPUTリクエストの最大数。
データストアあたりのPUTリクエストのレート	サポートされている各リージョン: 1,000	可能	データストアごとに 1 秒あたりに実行できるPUTリクエストの最大数。2023 年 8 月 21 日より前に作成されたデータストアの場合、1 秒あたり 100 リクエストに制限されます。

名前	デフォルト	引き上げ可能	説明
アカウントあたりの StartFHIRExport ジョブリクエストのレート	サポートされている各リージョン: 1	[いいえ]	アカウントごとに 1 分あたりに実行できる StartFHIRExportジョブリクエストの最大数。
アカウントPOSTあたりの を使用した StartFHIRExportジョブリクエストのレート	サポートされている各リージョン: 1	[いいえ]	アカウントごとに 1 分あたりに実行できる POST を使用した StartFHIRExportジョブリクエストの最大数。
アカウントあたりの StartFHIRImport ジョブリクエストのレート	サポートされている各リージョン: 1	[いいえ]	アカウントごとに 1 分あたりに実行できる StartFHIRImportジョブリクエストの最大数。
アカウントあたりの TagResource リクエストのレート	サポートされている各リージョン: 10	いいえ	1 秒あたりに実行できる TagResource リクエストの最大数。
アカウントあたりの UntagResource リクエストのレート	サポートされている各リージョン: 10	いいえ	アカウントごとに 1 秒あたりに実行できる UntagResource リクエストの最大数。
アカウントGETあたりの を使用した検索リクエストのレート	サポートされている各リージョン: 200	可能	アカウントごとに 1 秒あたりGETに実行できる を使用した検索リクエストの最大数。

名前	デフォルト	引き上げ可能	説明
データストアGETあたりの を使用した検索リクエストのレート	サポートされている各リージョン: 100	可能	データストアごとに 1 秒あたりGETに実行できるを使用した検索リクエストの最大数。
アカウントPOSTあたりの を使用した検索リクエストのレート	サポートされている各リージョン: 200	可能	を使用して 1 秒あたりPOSTに実行できる検索リクエストの最大数。
データストアPOSTあたりの を使用した検索リクエストのレート	サポートされている各リージョン: 100	可能	データストアごとに 1 秒あたりPOSTに実行できる を使用した検索リクエストの最大数。
インポートされた個々のファイルサイズ	サポートされている各リージョン: 5 ギガバイト	いいえ	S tartFHIRImportジョブに含まれる個々のファイルの最大サイズ (GB 単位) 。
インポートジョブの総サイズ	サポートされている各リージョン: 500 GB	いいえ	インポートジョブに含まれるファイルの最大サイズ (GB)。

トラブルシューティング

次のドキュメントは、 の使用で発生する可能性のある問題のトラブルシューティングに役立ちます AWS HealthLake。

トピック

HealthLake データストアを作成できないのはなぜですか？

2022 年 11 月 14 日に、 は新しいデータストアの作成に必要なIAMアクセス許可 HealthLake を更新しました。アクセスするユーザーまたはロールにアタッチされたポリシーを更新していない場合は、次のエラー HealthLake が発生します。

```
AccessDeniedException: Insufficient Lake Formation permission(s): Required Database on
 Catalog
```

データストアを作成するための更新されたIAMポリシー要件を確認するには、「 AWS 管理ポリシー：」を参照してください AmazonHealthLakeFullAccess。これらのポリシーをIAMユーザーまたはロールに追加する step-by-step方法については、「」を参照してくださいの使用を開始するためのアクセス許可の設定 AWS HealthLake。

データストアを作成するには、対称カスタマー所有または Amazon 所有のKMSキーも使用する必要があります。IAM ポリシーに正しいアクセス許可があることを確認してください。詳細については AWS KMS、「 AWS Key Management Service デベロッパーガイドAWS Key Management Service」の「」を参照してください。

アカウントごとに許可されるデータストアの数を超えました

HealthLake には、アカウントあたり 10 個のデータストアのクォータがあります。クォータの引き上げをリクエストする方法については、 AWS サポートセンターを参照してください。

の認可を作成する方法 FHIR RESTful APIs

ユーザーは、署名バージョン 4 の署名プロセスを使用して、 HTTPクライアントを介して送信されるリクエストに HealthLake API認証を追加する必要があります。詳細については、「署名バージョン 4 の署名プロセス」を参照してください。

AWS SDK for Python を使用して sigv4 認可を作成するには、次の例のようなスクリプトを作成します。

```
import boto3
import requests
import json
from requests_auth_aws_sigv4 import AWSSigV4

# Set the input arguments
data_store_endpoint = 'https://healthlake.us-east-1.amazonaws.com/datastore/<datastore
 id>/r4//'
resource_path = "Patient"
requestBody = {"resourceType": "Patient", "active": True, "name": [{"use":
 "official","family": "Dow","given": ["Jen"]},{"use": "usual","given":
 ["Jen"]}],"gender": "female","birthDate": "1966-09-01"}
region = 'us-east-1'

#Frame the resource endpoint
```

```
resource_endpoint = data_store_endpoint+resource_path
session = boto3.session.Session(region_name=region)
client = session.client("healthlake")

# Frame authorization
auth = AWSSigV4("healthlake", session=session)

# Calling data store FHIR endpoint using SigV4 auth

r = requests.post(resource_endpoint, json=requestBody, auth=auth, )
print(r.json())
```

AWS SDK for Python を使用した sigv4 認可の使用に関する追加情報は、Boto3 認証情報」トピックを参照してください。

データは FHIR R4 形式ではありません。 を引き続き使用できますHealthLakeか ？

データストアにインポートできるのは FHIR R4 形式の HealthLake データのみです。ユーザーがデータを変換できるように製品を提供するパートナーのリストについては、AWS HealthLake 「パートナー」を参照してください。

カスタマーマネージドKMSキーで暗号化されたデータストアFHIRRESTfulAPIsに を使用すると AccessDenied 、エラーが発生するのはなぜですか ？

ユーザーまたはロールがデータストアにアクセスするには、カスタマーマネージドキーとIAMポリシーの両方に対するアクセス許可が必要です。ユーザーには、カスタマーマネージドキーを使用するために必要なIAMアクセス許可が必要です。ユーザーがカスタマーマネージドKMSキーを使用するアクセス許可を付与 HealthLakeしたグラントを取り消したか、廃止した場合、 HealthLake はエラーを返します AccessDenied。

HealthLake には、顧客データにアクセスし、データストアにインポートされた新しいFHIRリソースを暗号化し、リクエストされたときにFHIRリソースを復号するためのアクセス許可が必要です。

詳細については、「キー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。

インポートが失敗したのはなぜですか？

インポートジョブが成功すると output inputFileName.ndjson ファイルを含むフォルダが生成されますが、個々のレコードはインポートに失敗する可能性があります。この場合、インポートに失敗したレコードのマニフェストを含む 2 番目のFAILUREフォルダが生成されます。マニフェストファイルにアクセスするためのジョブ出力場所は です JobPropertiesJobOutputDataConfig。S3Configuration.S3Uri.

このマニフェストファイルには、すべての成功したレスポンスの場所 (successOutput.successOutputS3Uri)、すべての失敗したレスポンスの場所 (failureOutput.failureOutputS3Uri)、追加のジョブメトリクスなど、ジョブ出力に関する詳細が含まれています。マニフェストファイルのコンテンツはプログラムで解析できます。次のサンプルマニフェストファイルには、入出力 Amazon S3 バケットと、スキャンされたリソースの数と正常にインポートされたリソースの数に関する情報が一覧表示されます。

```
            {
            "inputDataConfig": {
            "s3Uri": "s3://amzn-s3-demo-source-bucket/healthlake-input/invalidInput/"
            },
            "outputDataConfig": {
            "s3Uri": "s3://amzn-s3-demo-logging-
bucket/32839038a2f47f17c2fe0f53f0c3a0ba-FHIR_IMPORT-19dd7bb7bcc8ee12a09bf6d322744a3d/",
            "encryptionKeyID": "arn:aws:kms:us-west-2:123456789012:key/
fbbbfee3-20b3-42a5-a99d-c48c655ed545"
            },
            "successOutput": {
            "successOutputS3Uri": "s3://amzn-s3-demo-logging-
bucket/32839038a2f47f17c2fe0f53f0c3a0ba-FHIR_IMPORT-19dd7bb7bcc8ee12a09bf6d322744a3d/
SUCCESS/"
            },
            "failureOutput": {
            "failureOutputS3Uri": "s3://amzn-s3-demo-logging-
bucket/32839038a2f47f17c2fe0f53f0c3a0ba-FHIR_IMPORT-19dd7bb7bcc8ee12a09bf6d322744a3d/
FAILURE/"
            },
            "numberOfScannedFiles": 1,
            "numberOfFilesImported": 1,
            "sizeOfScannedFilesInMB": 0.023627,
            "sizeOfDataImportedSuccessfullyInMB": 0.011232,
            "numberOfResourcesScanned": 9,
```

```
            "numberOfResourcesImportedSuccessfully": 4,
            "numberOfResourcesWithCustomerError": 5,
            "numberOfResourcesWithServerError": 0
            }
```

インポートジョブが失敗した理由を分析するには、D escribeFHIRImportジョブAPIを使用して を分析します JobProperties。以下が推奨されます。

- ステータスが FAILEDで、メッセージが存在する場合、失敗は入力データサイズや HealthLake クォータを超える入力ファイル数などのジョブパラメータに関連しています。
- インポートジョブのステータスが COMPLETED_WITH_ の場合はERRORS、マニフェストファイル Manifest.json で、正常にインポートされなかったファイルに関する情報を確認します。
- インポートジョブのステータスが FAILEDで、メッセージが存在しない場合は、ジョブの出力場所に移動してマニフェストファイル Manifest.json にアクセスします。

入力ファイルごとに、インポートに失敗したリソースの入力ファイル名を持つ失敗出力ファイルがあります。レスポンスには、入力データの場所に対応する行番号 (lineId) 、FHIRレスポンスオブジェクト (UpdateResourceResponse) 、およびレスポンスのステータスコード (statusCode) が含まれます。

サンプル出力ファイルは次のようになります。

```
{"lineId":3, UpdateResourceResponse:{"jsonBlob":
{"resourceType":"OperationOutcome","issue":
[{"severity":"error","code":"processing","diagnostics":"1 validation error detected:
 Value 'Patient123' at 'resourceType' failed to satisfy constraint: Member must satisfy
 regular expression pattern: [A-Za-z]{1,256}"}]}, "statusCode":400}
{"lineId":5, UpdateResourceResponse:{"jsonBlob":
{"resourceType":"OperationOutcome","issue":
[{"severity":"error","code":"processing","diagnostics":"This property must be an
 simple value, not a com.google.gson.JsonArray","location":["/EffectEvidenceSynthesis/
name"]},{"severity":"error","code":"processing","diagnostics":"Unrecognised
 property '@telecom'","location":["/EffectEvidenceSynthesis"]},
{"severity":"error","code":"processing","diagnostics":"Unrecognised
 property '@gender'","location":["/EffectEvidenceSynthesis"]},
{"severity":"error","code":"processing","diagnostics":"Unrecognised
 property '@birthDate'","location":["/EffectEvidenceSynthesis"]},
{"severity":"error","code":"processing","diagnostics":"Unrecognised
```

```
 property '@address'","location":["/EffectEvidenceSynthesis"]},
{"severity":"error","code":"processing","diagnostics":"Unrecognised
 property '@maritalStatus'","location":["/EffectEvidenceSynthesis"]},
{"severity":"error","code":"processing","diagnostics":"Unrecognised
 property '@multipleBirthBoolean'","location":["/EffectEvidenceSynthesis"]},
{"severity":"error","code":"processing","diagnostics":"Unrecognised
 property '@communication'","location":["/EffectEvidenceSynthesis"]},
{"severity":"warning","code":"processing","diagnostics":"Name should be usable as an
 identifier for the module by machine processing applications such as code generation
 [name.matches('[A-Z]([A-Za-z0-9_]){0,254}')]","location":["EffectEvidenceSynthesis"]},
{"severity":"error","code":"processing","diagnostics":"Profile http://hl7.org/fhir/
StructureDefinition/EffectEvidenceSynthesis, Element 'EffectEvidenceSynthesis.status':
 minimum required = 1, but only found 0","location":["EffectEvidenceSynthesis"]},
{"severity":"error","code":"processing","diagnostics":"Profile
 http://hl7.org/fhir/StructureDefinition/EffectEvidenceSynthesis,
 Element 'EffectEvidenceSynthesis.population': minimum required
 = 1, but only found 0","location":["EffectEvidenceSynthesis"]},
{"severity":"error","code":"processing","diagnostics":"Profile
 http://hl7.org/fhir/StructureDefinition/EffectEvidenceSynthesis,
 Element 'EffectEvidenceSynthesis.exposure': minimum required =
 1, but only found 0","location":["EffectEvidenceSynthesis"]},
{"severity":"error","code":"processing","diagnostics":"Profile http://
hl7.org/fhir/StructureDefinition/EffectEvidenceSynthesis, Element
 'EffectEvidenceSynthesis.exposureAlternative': minimum required
 = 1, but only found 0","location":["EffectEvidenceSynthesis"]},
{"severity":"error","code":"processing","diagnostics":"Profile http://hl7.org/fhir/
StructureDefinition/EffectEvidenceSynthesis, Element 'EffectEvidenceSynthesis.outcome':
 minimum required = 1, but only found 0","location":["EffectEvidenceSynthesis"]},
{"severity":"information","code":"processing","diagnostics":"Unknown
 extension http://synthetichealth.github.io/synthea/disability-adjusted-
life-years","location":["EffectEvidenceSynthesis.extension[3]"]},
{"severity":"information","code":"processing","diagnostics":"Unknown extension
 http://synthetichealth.github.io/synthea/quality-adjusted-life-years","location":
["EffectEvidenceSynthesis.extension[4]"]}]}, "statusCode":400}
{"lineId":7, UpdateResourceResponse:{"jsonBlob":
{"resourceType":"OperationOutcome","issue":
[{"severity":"error","code":"processing","diagnostics":"2 validation errors detected:
 Value at 'resourceId' failed to satisfy constraint: Member must satisfy regular
 expression pattern: [A-Za-z0-9-.]{1,64}; Value at 'resourceId' failed to satisfy
 constraint: Member must have length greater than or equal to 1"}]}, "statusCode":400}
{"lineId":9, UpdateResourceResponse:{"jsonBlob":
{"resourceType":"OperationOutcome","issue":
[{"severity":"error","code":"processing","diagnostics":"Missing required id field in
 resource json"}]}, "statusCode":400}
```

```
{"lineId":15, UpdateResourceResponse:{"jsonBlob":
{"resourceType":"OperationOutcome","issue":
[{"severity":"error","code":"processing","diagnostics":"Invalid JSON found in input
 file"}]}, "statusCode":400}
```

この例では、入力ファイルから対応する入力行の 3、4、7、9、15 行で障害が発生したことを示しています。これらの各行の説明は次のとおりです。

- 3 行目では、入力ファイルの 3 行目で resourceType 提供された が無効であることがレスポンスで説明されています。
- 5 行目には、入力ファイルの 5 行目にFHIR検証エラーがあることがレスポンスで説明されています。
- 7 行目のレスポンスでは、入力として resourceId 提供された に検証の問題があることが説明されています。
- 9 行目のレスポンスでは、入力ファイルに有効なリソース ID が含まれている必要があることが説明されています。
- 行 15 で、入力ファイルのレスポンスは、ファイルが有効なJSON形式ではないことです。

処理できなかったリソースを見つける DocumentReferenceにはどうすればよいですか？

DocumentReference リソースが有効でない場合、 HealthLake は統合された医療NLP出力の代わりに検証エラーを示す拡張機能を提供します。NLP 処理中に検証エラーの原因となった DocumentReference リソースを見つけるために、顧客は HealthLake検索キーcm-decoration-statusと検索値 VALIDATION_ERROR で の検索関数を使用できます。この検索では、検証エラーの原因となったすべての DocumentReference リソースと、エラーの性質を説明するエラーメッセージが一覧表示されます。検証エラーがある DocumentReference リソースの拡張フィールドの構造は、次の例のようになります。

```
"extension": [
          {
              "extension": [
                  {
                      "url": "http://healthlake.amazonaws.com/aws-cm/status/",
                      "valueString": "VALIDATION_ERROR"
```

```
                },
                {
                    "url": "http://healthlake.amazonaws.com/aws-cm/message/",
                    "valueString": "Resource led to too many nested objects after NLP
operation processed the document. 10937 nested objects exceeds the limit of 10000."
                }
            ],
            "url": "http://healthlake.amazonaws.com/aws-cm/"
        }
    ]
```

デNLPコレーションによって 10,000 個を超えるネストされたオブジェクトが作成された場合にも、VALIDATION_ERROR が発生する可能性があります。この場合、処理する前にドキュメントを小さなドキュメントに分割する必要があります。

Amazon Athena を使用するための既存のデータストアの移行

2022 年 11 月 14 日より前に作成された データストアは機能しますが、 を使用して Athena でクエリすることはできませんSQL。Athena で既存のデータストアをクエリするには、まず新しいデータストアに移行する必要があります。

データを新しいデータストアに移行するには

1. 新しいデータストアを作成します。
2. 既存の から Amazon S3 バケットにデータをエクスポートします。
3. Amazon S3 バケットから新しいデータストアにデータをインポートします。

Amazon S3 バケットにデータをエクスポートすると、追加料金が発生します。追加料金は、エクスポートするデータのサイズによって異なります。

Athena の検索結果を他の AWS サービスに接続する

Athena の検索結果を他の AWS サービスと共有する際に問題が発生する可能性があります。

SQL 検索クエリ`json_extract[1]`の一部として を使用すると、問題が発生する可能性があります。

この問題を解決するには、 を に更新する必要がありますCATVAR。

保存結果、テーブル (静的) 、またはビュー (動的) を作成しようとすると、この問題が発生する可能性があります。

新しいデータストアにデータをインポートした後、Athena コンソールが動作しない

新しいデータストアにデータをインポートすると、そのデータはすぐに使用できなくなる場合があります。これは、データが Iceberg テーブルに取り込まれる時間を確保するために必要です。後でもう一度試してください。

新しいデータレイク管理者を追加するPutDataLakeSettings ときに Lake Formation 許可エラーが表示されるのはなぜですか？

IAM ユーザーまたはロールに AWSLakeFormationDataAdmin AWS 管理ポリシーが含まれている場合、新しいデータレイク管理者を追加することはできません。次のエラーが表示されます。

```
User arn:aws:sts::111122223333:assumed-role/lakeformation-admin-user is not authorized
 to perform: lakeformation:PutDataLakeSettings on resource: arn:aws:lakeformation:us-
east-2:111122223333:catalog:111122223333 with an explicit deny in an identity-based
 policy
```

AWS 管理ポリシーAdministratorAccessは、 AWS Lake Formation データレイク管理者として IAM ユーザーまたはロールを追加するために必要です。IAM ユーザーまたはロールにも が含まれているAWSLakeFormationDataAdmin場合、アクションは失敗します。AWSLakeFormationDataAdmin AWS 管理ポリシーには、 AWS Lake Formation APIオペレーション の明示的な拒否が含まれていますPutDataLakeSetting。

AdministratorAccess AWS 管理ポリシーAWSを使用するためのフルアクセス権を持つ管理者であっても、AWSLakeFormationDataAdminポリシーによって制限できます。

HealthLake統合された自然言語処理機能を有効にするにはどうすればよいですか？

2023 年 2 月 20 日現在、 HealthLake データストアのデフォルトの動作が変更されています。

現在のデータストア: 現在の HealthLake データストアはすべて、base64 でエンコードされたDocumentReferenceリソースでの自然言語処理 (NLP) の使用を停止します。つまり、新し

いDocumentReferenceリソースは を使用して分析されずNLP、リソースタイプのテキストに基づいて新しいDocumentReferenceリソースは生成されません。既存のDocumentReferenceリソースの場合、 で生成されたデータとリソースはNLP残りますが、2023 年 2 月 20 日以降は更新されません。

新しいデータストア: 2023 年 2 月 20 日以降に作成された HealthLake データストアは、base64 でエンコードされたDocumentReferenceリソースで自然言語処理 (NLP) を実行しません。

この機能を有効にするには、 を使用してケースを作成する必要がありますAWS Support Center Console。ケースを作成するには、 にログインし AWS アカウント、ケースの作成を選択します。ケースとケース管理の作成の詳細については、「 サポート ユーザーガイド」の「サポートケースとケース管理の作成」を参照してください。

データストアのステータスが「作成中」から変更されていない

新しい HealthLake データストアを作成しようとしたときに、データストアのステータスが「作成中」から変更されていない場合は、 を使用するように Athena を更新する必要があります AWS Glue Data Catalog。

詳細については、「Amazon Athena ユーザーガイド」の AWS 「 Glue データカタログへのアップグレード step-by-step」を参照してください。 Amazon Athena

を正常にアップグレードした後 AWS Glue Data Catalog、データストアを作成できるようになりました。

古いデータストアを削除するには、 を使用してケースを作成しますAWS Support Center Console。ケースを作成するには、 にログインし AWS アカウント、ケースの作成を選択します。詳細については、「 サポート ユーザーガイド」の「サポートケースとケース管理の作成」を参照してください。

SDK データストアの作成ステータスが例外または不明なステータスを返す

リストデータストアまたはデータストアAPI呼び出しが例外または不明なデータストアステータスを返す場合は、 を最新バージョンSDKに更新してください。

への 10MB ドキュメントのFHIRPOSTAPIオペレーションで、413Request Entity Too Large エラー HealthLake が発生します。

AWS HealthLake では、レイテンシーとタイムアウトの増加を避けるために、同期の作成と更新APIの制限は 5MB です。

一括インポート を使用して、バイナリ ResourceType を使用して最大 164MB の大きなドキュメントを取り込むことができますAPI。

AWS HealthLake デベロッパーガイドのドキュメント履歴

次の表は、 AWS HealthLake リリースのドキュメントの変更点をまとめたものです。

- API バージョン： 最新
- ドキュメントの最終更新日： 10/25/2024

変更	説明	日付
HealthLake が FHIRhistoryと のvreadインタラクションをサポート	HealthLake では、特定のリソースの履歴を取得するためのFHIRhistoryインタラクションと、リソースのバージョン固有の読み取りを実行するためのvreadインタラクションがサポートされるようになりました。	2024 年 10 月 25 日
HealthLake は、新しいFHIR検索パラメータ、拡張子、リソースタイプをサポートするようになりました。	HealthLake は、新しいFHIR検索パラメータ、拡張子、リソースタイプをサポートするようになりました。	2023 年 12 月 9 日
HealthLake が FHIRフレームワークSMARTで をサポートするようになりました	HealthLake は、FHIR有効なHealthLake データストアSMARTでの の作成をサポートするようになりました。	2023 年 5 月 31 日
HealthLake でプロファイル検証がサポートされるようになりました	HealthLake でFHIRプロファイル検証がサポートされるようになりました。	2023 年 5 月 31 日
HealthLake が をサポートするようになりました export	HealthLake は、 FHIRRESTAPIオペレーション を使用したファイルのエクスポートを	2023 年 5 月 31 日

	サポートするようになりましたexport。	
アジアパシフィック (ムンバイ) リージョン	AWS HealthLake がアジアパシフィック (ムンバイ) リージョンで利用可能になりました。	2023 年 4 月 4 日
統合された自然言語処理がオフになりました	HealthLake は、2023 年 2 月 20 日現在、すべてのデータストアで統合自然言語処理 (NLP) をオフにしました。	2023 年 2 月 20 日
HealthLake と Amazon Athena の統合	Athena を使用して、2022 年 11 月 14 日以降に作成されたデータストアをクエリできるようになりました。	2022 年 11 月 14 日
インポートジョブの合計サイズの増加	S tartFHIRImportジョブリクエスト内のすべてのファイルの最大合計サイズが 500 GB になりました。	2022 年 10 月 3 日
バンドルのサポート	HealthLake は、複数のリソースを取り込むための Bundle リソースタイプをサポートするようになりました。	2022 年 8 月 5 日
でのCRUDオペレーションのクォータを更新 HealthLake	HealthLake は、 CRUD リクエストに対してより高い制限をサポートするようになりました。	2022 年 7 月 14 日
サポートを含める	HealthLake は、データストアクエリ_includeで をサポートするようになりました。	2022 年 7 月 14 日
AWS HealthLake が一般公開されました	HealthLake が一般公開されました。	2020 年 7 月 30 日

AWS 用語集

最新の AWS 用語については、 AWS の用語集 リファレンスのAWS 用語集を参照してください。

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。