



デベロッパーガイド

# Amazon Managed Blockchain Query



# Amazon Managed Blockchain Query: デベロッパーガイド

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

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

# Table of Contents

Amazon Managed Blockchain (AMB) クエリとは .....	1
初めての AMB クエリユーザーですか? .....	1
主要なコンセプト .....	2
Amazon Managed Blockchain (AMB) クエリを使用する際の考慮事項と制限事項 .....	2
設定 .....	6
前提条件と考慮事項 .....	6
にサインアップする AWS .....	6
適切なアクセス許可を持つ IAM ユーザーを作成する .....	6
のインストールと設定 AWS Command Line Interface .....	7
を使用して AMB AWS マネジメントコンソール クエリを使用してブロックチェーンをクエリ する .....	7
開始方法 .....	9
IAM ポリシーを作成する .....	9
Go の使用例 .....	10
Node.js の使用例 .....	16
Python の使用例 .....	20
を使用した例 AWS マネジメントコンソール .....	22
AMB クエリのユースケース .....	24
現在および過去のトークン残高をクエリする .....	24
履歴トランザクションデータを取得する .....	24
特定のアドレスのすべてのトークン残高を取得する .....	24
トランザクションに対して出力されたイベントを一覧表示する .....	25
契約によって作成されたすべてのトークンを取得する .....	25
契約を一覧表示し、契約情報を取得する .....	25
AMB クエリ API リファレンス .....	27
セキュリティ .....	28
データ暗号化 .....	28
転送中の暗号化 .....	29
ID とアクセス管理 .....	29
オーディエンス .....	29
アイデンティティを使用した認証 .....	30
ポリシーを使用したアクセスの管理 .....	31
Amazon Managed Blockchain (AMB) クエリと IAM の連携 .....	33
アイデンティティベースのポリシーの例 .....	38

---

トラブルシューティング .....	42
API 使用状況メトリクス .....	44
Amazon CloudWatch の API 使用状況メトリクス .....	44
ドキュメント履歴 .....	46
.....	xlviii

# Amazon Managed Blockchain (AMB) クエリとは

Amazon Managed Blockchain (AMB) は、パブリックブロックチェーンとプライベートブロックチェーンの両方で回復力のある Web3 アプリケーションを構築するのに役立つように設計されたフルマネージドサービスです。AMB Access を使用すると、複数のブロックチェーンに瞬時にサーバーレスにアクセスできます。特殊なブロックチェーンインフラストラクチャをデプロイし、ブロックチェーンネットワークに接続したままにすることなく、Web3-readyアプリケーションを構築します。AMB クエリを使用すると、開発者向けの API オペレーションを使用して、複数のブロックチェーンからのリアルタイムデータと履歴データにアクセスできます。標準化されたブロックチェーンデータは、特殊なブロックチェーンインフラストラクチャや ETL (抽出、変換、ロード) を必要とせずに、AWS のサービスに統合できます。すべての AMB 機能は、機関グレードとメインストリームのコンシューマーアプリケーションビルドのために安全にスケーリングされます。

Amazon Managed Blockchain (AMB) クエリは、開発者向けの API オペレーションを使用して、標準化されたマルチブロックチェーンデータセットへのサーバーレスアクセスを提供します。AMB Query を使用すると、ブロックチェーンデータの解析、契約の追跡、特殊なインデックス作成インフラストラクチャの維持にオーバーヘッドを必要とせずに、1 つ以上のパブリックブロックチェーンのデータを必要とするアプリケーションをすばやく出荷できます。ファンジブルトークンまたは非ファンジブルトークン (NFTs) の履歴トークン残高を分析したり、特定のウォレットアドレスのトランザクション履歴を表示したり、Ether などのネイティブ暗号通貨のディストリビューションでデータ分析を実行したりする場合、AMB Query はブロックチェーンデータへのアクセスを許可します。

## 初めての AMB クエリユーザーですか？

AMB クエリを初めて使用する場合は、まず以下のセクションを読むことをお勧めします。

- [主な概念: Amazon Managed Blockchain \(AMB\) クエリ](#)
- [Amazon Managed Blockchain \(AMB\) クエリの設定](#)
- [Amazon Managed Blockchain \(AMB\) クエリの開始方法](#)
- [Amazon Managed Blockchain \(AMB\) クエリのユースケース](#)

# 主な概念: Amazon Managed Blockchain (AMB) クエリ

## Note

このガイドでは、ブロックチェーンの基本的な概念に精通していることを前提としています。これらの概念には、分散、トークン、契約、トランザクション、proof-of-work、ウォレット、パブリックキーとプライベートキー、ステーク、マイニング、半分などが含まれます。

Amazon Managed Blockchain (AMB) クエリでは、マルチブロックチェーンネットワークデータに簡単にアクセスできるため、ブロックチェーンアクティビティに関連するコンテキストデータを抽出しやすくなります。AMB クエリを使用して、Bitcoin Mainnet や Ethereum Mainnet などのパブリックブロックチェーンネットワークからデータを読み取ることができます。アドレスの現在および過去の残高などの情報を取得したり、特定の期間のブロックチェーントランザクションのリストを取得したりすることもできます。さらに、トランザクションイベントなど、特定のトランザクションの詳細を取得できます。トランザクションイベントは、アプリケーションのビジネスロジックでさらに分析または使用できます。

## Amazon Managed Blockchain (AMB) クエリを使用する際の考慮事項と制限事項

AMB クエリを使用する場合は、次の点を考慮してください。

- 使用可能なリージョン

AMB クエリは、米国東部 (バージニア北部) us-east-1リージョンでサポートされています。

- サービスエンドポイント

AMB クエリには、次のエンドポイントを使用してアクセスできます。

<https://managedblockchain-query.us-east-1.amazonaws.com>.

- サポートされているブロックチェーンネットワーク

AMB クエリは、次のパブリックブロックチェーンネットワークをサポートしています。

- Bitcoin Mainnet — proof-of-work コンセンサスによって保護され、Bitcoin (BTC) 暗号通貨が発行されて取引されるパブリック Bitcoin ブロックチェーンネットワーク。Mainnet のトランザクションには実際の値があり (つまり、実際のコストが発生します)、パブリックブロックチェーンに記録されます。
  - Bitcoin Testnet — Bitcoin Mainnet のテストネット。このネットワーク上のビットコイン (BTC) は Mainnet BTC とは別個に区別され、通常は値がありません。
  - Ethereum Mainnet — パブリック Ethereum ブロックチェーンの proof-of-stake メインネットワーク。Mainnet のトランザクションは実際の値 (つまり、実際のコストが発生します) を持ち、分散台帳に記録されます。
  - Sepolia Testnet — Ethereum Mainnet のテストネット。このネットワークの Ether (ETH) は Mainnet ETH とは独立しており、通常は値がありません。
- サポートされているブロックチェーントークンと契約

AMB クエリは、次のネイティブおよび標準の Ethereum 契約トークンをサポートしています。

- パブリックブロックチェーンネイティブトークン

- Bitcoin (BTC) — これは Bitcoin 関連のブロックチェーンのネイティブトークンです。
- Ether (ETH) — これは Ethereum 関連のブロックチェーンのネイティブトークンです。

- Ethereum 契約標準

- ERC-20 トークン標準 — ERC-20 は、ファンジブルトークンの標準です。各 ERC-20 トークンを別の ERC-20 トークンと完全に同じ (タイプと値) にするプロパティがあります。つまり、1 つのトークンはであり、常に他のすべてのトークンと等しくなります。詳細については、Ethereum.org の [ERC-20 トークン標準](#) を参照してください。
- ERC-721 非ファンジブルトークン標準 — ERC-721 は非ファンジブルトークン (NFTs)。このタイプのトークンは一意であり、おそらくその経過時間、希少性、またはその他のプロパティが原因で、同じ契約の別のトークンとは異なる値を持つことができます。詳細については、Ethereum.org の [ERC-721 トークン標準](#) を参照してください。

ERC-1155 マルチトークン標準 — ERC-1155 は、任意の数のファンジブルトークンタイプと非ファンジブルトークンタイプを表して制御できる契約インターフェイスを作成する標準です。このようにして、ERC-1155 トークンは [ERC-20](#) および [ERC-721](#) トークンと同じように機能し、両方を同時に機能させることができます。ERC-1155 トークンは、ERC-20 標準と ERC-721 標準の両方の機能を改善し、明確な実装エラーを修正しながら、より効率的にします。詳細については、Ethereum.org の [ERC-1155 トークン標準](#) を参照してください。

- 確定性

ブロックチェーンでは、確定性は有効なトランザクションが取り消される可能性が低いことを意味します。Bitcoin Mainnet の場合、AMB クエリは 6 ブロック後にトランザクションを確定と見なします。Bitcoin Testnet の場合、トランザクションは 6 ブロック後または 60 分後のいずれか早い方を最終と見なします。サポートされている Ethereum ネットワークの場合、AMB クエリは 64 ブロック後にトランザクションを確定と見なします。


AMB クエリのトークン残高と契約 API オペレーションは、確定に達したデータのみを返します。ただし、AMB Query のトランザクションおよびトランザクションイベント API オペレーションは、ブロックチェーンネットワークで確認されたトランザクションのデータを、まだ確定していない場合でも返すことができます。

- NULL アドレスはサポートされていません

AMB クエリは NULL (0x00) アドレスをサポートしていません。

- API コールの署名バージョン 4 の署名

AMB クエリ APIs を呼び出すときは、[署名バージョン 4 の署名プロセス](#)を使用して認証された HTTPS 接続を介して呼び出すことができます。つまり、アカウント内の AWS 承認された IAM プリンシパルのみが AMB クエリ API コールを実行できます。これを行うには、AWS 呼び出しで認証情報 (アクセスキー ID とシークレットアクセスキー) を指定する必要があります。

 Important

ユーザー向けアプリケーションにクライアント認証情報を埋め込まないでください。

- AMB クエリが Bitcoin トランザクション識別子とトランザクションハッシュをサポート

Bitcoin ネットワークの場合、AMB クエリ API オペレーションはトランザクション識別子 (transactionId) とトランザクションハッシュ () の両方をサポートします transactionHash。transactionId は、監視データを含まないトランザクションのダブル SHA ハッシュです。transactionHash は、監視データ (監視トランザクション ID と呼ばれます) を含むトランザクションのダブル SHA ハッシュです。

Bitcoin ネットワークの [GetTransaction](#) または [ListTransactionEvents](#) API オペレーションを呼び出すときは、`transactionId` または のいずれかを指定できます `transactionHash`。また、`transactionId` または を返す Bitcoin ネットワーク上のすべての AMB クエリオペレーション `transactionHash` には、レスポンスの一部として両方の値が含まれます。

# Amazon Managed Blockchain (AMB) クエリの設定

Amazon Managed Blockchain (AMB) クエリを初めて使用する前に、このセクションの手順に従って AWS アカウントを作成します。次のセクションでは、AMB クエリの使用を開始する方法について説明します。

## 前提条件と考慮事項

Amazon Web Services を初めて使用するには、AWS アカウントが必要です。

### にサインアップする AWS

Amazon Web Services (AWS) にサインアップすると AWS のサービス、Amazon Managed Blockchain (AMB) クエリを含むすべての AWS アカウントが自動的にサインアップされます。サービスを実際に使用した分の料金のみが請求されます。

AWS アカウント をすでにお持ちの場合は、次のステップに進みます。AWS アカウントをお持ちでない場合は、次に説明する手順に従ってアカウントを作成してください。

AWS アカウントを作成するには

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

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

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

## 適切なアクセス許可を持つ IAM ユーザーを作成する

AMB クエリを作成して操作するには、必要な Managed Blockchain アクションを許可するアクセス許可を持つ AWS Identity and Access Management (IAM) プリンシパル (ユーザーまたはグループ) を作成する必要があります。

IAM プリンシパルのみが AMB クエリ API リクエストを行うことができます。AMB クエリ APIs を呼び出すときは、[署名バージョン 4 の署名プロセス](#)を使用して認証された HTTPS 接続を介して呼び出すことができます。つまり、アカウント内の AWS 承認された IAM プリンシパルのみが AMB クエリ API コールを実行できます。これを行うには、AWS 呼び出しで認証情報 (アクセスキー ID とシークレットアクセスキー) を指定する必要があります。

IAM ユーザーを作成する方法については、「[アカウントでの IAM ユーザーの作成 AWS](#)」を参照してください。アクセス許可ポリシーをユーザーにアタッチする方法の詳細については、「[IAM ユーザーのアクセス許可の変更](#)」を参照してください。AMB クエリを操作するためのアクセス許可をユーザーに付与するために使用できるアクセス許可ポリシーの例については、「」を参照してください。[Amazon Managed Blockchain \(AMB\) クエリのアイデンティティベースのポリシーの例](#)。

## のインストールと設定 AWS Command Line Interface

まだインストールしていない場合は、最新のコマンドラインインターフェイス (CLI) AWS をインストールして、ターミナルの AWS リソースを操作します。詳細については、「[Installing or updating the latest version of the AWS CLI](#)」を参照してください。

### Note

CLI アクセスには、アクセスキー ID とシークレットアクセスキーが必要です。長期のアクセスキーの代わりに一時的な認証情報をできるだけ使用します。一時的な認証情報には、アクセスキー ID、シークレットアクセスキー、および認証情報の失効を示すセキュリティトークンが含まれています。詳細については、IAM [ユーザーガイドの「AWS リソースでの一時的な認証情報の使用」](#)を参照してください。

## AWS マネジメントコンソールを使用して Amazon Managed Blockchain (AMB) クエリを使用してブロックチェーンをクエリする

Amazon Managed Blockchain (AMB) クエリにアクセスし、を使用してサポートされているブロックチェーンネットワークに対してクエリを実行できます AWS マネジメントコンソール。次の手順は、これを行う方法を示しています。

1. <https://console.aws.amazon.com/managedblockchain/> で Amazon Managed Blockchain コンソールを開きます。

2. クエリセクションからクエリエディタを選択します。
3. サポートされているブロックチェーンネットワークのいずれかを選択します。
4. 実行するクエリタイプを選択します。
5. 選択したクエリタイプに関連するパラメータを入力し、クエリを実行します。

AMB クエリはクエリを実行し、クエリ結果ウィンドウに結果が表示されます。

# Amazon Managed Blockchain (AMB) クエリの開始方法

Amazon Managed Blockchain (AMB) クエリを使用してタスクを実行する方法については、このセクションのstep-by-stepのチュートリアルを参照してください。これらの手順には、いくつかの前提条件が必要です。AMB クエリを初めて使用する場合は、このガイドの「セットアップ」セクションを参照してください。詳細については、「[Amazon Managed Blockchain \(AMB\) クエリの設定](#)」を参照してください。

## Note

これらの例の一部の変数は意図的に難読化されています。これらの例を実行する前に、有効なものに置き換えてください。

## トピック

- [AMB クエリ API オペレーションにアクセスするための IAM ポリシーを作成する](#)
- [Go を使用して Amazon Managed Blockchain \(AMB\) クエリ API リクエストを作成する](#)
- [Node.js を使用して Amazon Managed Blockchain \(AMB\) クエリ API リクエストを作成する](#)
- [Python を使用して Amazon Managed Blockchain \(AMB\) クエリ API リクエストを作成する](#)
- [で Amazon Managed Blockchain \(AMB\) クエリ AWS マネジメントコンソール を使用して GetTokenBalance オペレーションを実行する](#)

## AMB クエリ API オペレーションにアクセスするための IAM ポリシーを作成する

AMB クエリ API リクエストを行うには、Amazon Managed Blockchain (AMB) クエリの適切な IAM アクセス許可を持つユーザー認証情報 (AWS\_ACCESS\_KEY\_ID および AWS\_SECRET\_ACCESS\_KEY) を使用する必要があります。AWS CLI がインストールされているターミナルで、次のコマンドを実行して IAM ポリシーを作成し、AMB クエリ API オペレーションにアクセスします。

```
cat <<EOT > ~/amb-query-access-policy.json
{
  "Version": "2012-10-17",
```

```
"Statement": [
  {
    "Sid" : "AMBQueryAccessPolicy",
    "Effect": "Allow",
    "Action": [
      "managedblockchain-query:*"
    ],
    "Resource": "*"
  }
]
}
EOT
aws iam create-policy --policy-name AmazonManagedBlockchainQueryAccess --policy-
document file://$HOME/amb-query-access-policy.json
```

ポリシーを作成したら、そのポリシーを IAM ユーザーのロールにアタッチして有効にします。で AWS マネジメントコンソール、IAM サービスに移動し、サービスを使用する IAM ユーザーに割り当てられた AmazonManagedBlockchainQueryAccess ロールにポリシーをアタッチします。詳細については、[「ロールの作成」](#) および [「IAM ユーザーへの割り当て」](#) を参照してください。

#### Note

AWS では、ワイルドカードを使用するのではなく、特定の API オペレーションへのアクセスを許可することをお勧めします\*。詳細については、[「特定の Amazon Managed Blockchain \(AMB\) クエリ API アクションへのアクセス」](#) を参照してください。

## Go を使用して Amazon Managed Blockchain (AMB) クエリ API リクエストを作成する

Amazon Managed Blockchain (AMB) クエリを使用すると、ブロックチェーンで確認されると、ブロックチェーンデータへの即時アクセスに依存するアプリケーションを構築できます。AMB クエリを使用すると、ウォレットのトランザクション履歴の入力、トランザクションハッシュに基づくトランザクションに関するコンテキスト情報の提供、ネイティブトークンと ERC-721, ERC-1155, ERC-20 トークンのバランスの取得など、いくつかのユースケースが可能になります。

次の例は、Go 言語で作成され、AMB クエリ API オペレーションを使用します。Go の詳細については、[「Go ドキュメント」](#) を参照してください。AMB クエリ API の詳細については、[「Amazon Managed Blockchain \(AMB\) クエリ API リファレンスドキュメント」](#) を参照してください。

次の例では、ListTransactionsおよび GetTransaction API アクションを使用して、Ethereum Mainnet 上の特定の外部所有アドレス (EOA) のすべてのトランザクションのリストを取得し、次の例では、リストから 1 つのトランザクションのトランザクション詳細を取得します。

Example— Go を使用して ListTransactions API アクションを実行する

次のコードを ListTransactions ディレクトリ listTransactions.go の という名前のファイルにコピーします。

```
package main

import (
    "fmt"
    "github.com/aws/aws-sdk-go/aws"
    "github.com/aws/aws-sdk-go/aws/session"
    "github.com/aws/aws-sdk-go/service/managedblockchainquery"
    "time"
)

func main() {

    // Set up a session
    ambQuerySession := session.Must(session.NewSessionWithOptions(session.Options{
        Config: aws.Config{
            Region: aws.String("us-east-1"),
        },
    }))
    client := managedblockchainquery.New(ambQuerySession)

    // Inputs for ListTransactions API
    ownerAddress := "0x00000bf26964af9d7eed9e03e53415d*****"
    network := managedblockchainquery.QueryNetworkEthereumMainnet
    sortOrder := managedblockchainquery.SortOrderAscending
    fromTime := time.Date(1971, 1, 1, 1, 1, 1, 1, time.UTC)
    toTime := time.Now()
    nonFinal := "NONFINAL"
    // Call ListTransactions API. Transactions that have reached finality are always
    returned
    listTransactionRequest, listTransactionResponse :=
    client.ListTransactionsRequest(&managedblockchainquery.ListTransactionsInput{
        Address: &ownerAddress,
        Network: &network,
        Sort: &managedblockchainquery.ListTransactionsSort{
```

```

        SortOrder: &sortOrder,
    },
    FromBlockchainInstant: &managedblockchainquery.BlockchainInstant{
        Time: &fromTime,
    },
    ToBlockchainInstant: &managedblockchainquery.BlockchainInstant{
        Time: &toTime,
    },

    ConfirmationStatusFilter: &managedblockchainquery.ConfirmationStatusFilter{
        Include: []*string{&nonFinal},
    },
})
errors := listTransactionRequest.Send()

if errors == nil {
    // handle API response
    fmt.Println(listTransactionResponse)
} else {
    // handle API errors
    fmt.Println(errors)
}
}

```

ファイルを保存したら、ListTransactions ディレクトリ内の コマンド を使用してコードを実行します。go run listTransactions.go。

次の出力は次のようになります。

```

{
  Transactions: [
    {
      ConfirmationStatus: "FINAL",
      Network: "ETHEREUM_MAINNET",
      TransactionHash:
"0x12345ea404b45323c0cf458ac755ecc45985fbf2b18e2996af3c8e8693354321",
      TransactionTimestamp: 2020-06-01 01:59:11 +0000 UTC
    },
    {
      ConfirmationStatus: "FINAL",
      Network: "ETHEREUM_MAINNET",
      TransactionHash:
"0x1234547c65675d867ebd2935bb7ebe0996e9ec8e432a579a4516c7113bf54321",

```

```

    TransactionTimestamp: 2021-09-01 20:06:59 +0000 UTC
  },
  {
    ConfirmationStatus: "NONFINAL",
    Network: "ETHEREUM_MAINNET",
    TransactionHash:
"0x123459df7c1cd42336cd1c444cae0eb660ccf13ef3a159f05061232a24954321",
    TransactionTimestamp: 2024-01-23 17:10:11 +0000 UTC
  }
]
}

```

### Example— Go を使用して GetTransaction API アクションを実行する

この例では、前の出力のトランザクションハッシュを使用します。次のコードを `GetTransaction` ディレクトリ `GetTransaction.go` の という名前のファイルにコピーします。

```

package main

import (
    "fmt"
    "github.com/aws/aws-sdk-go/aws"
    "github.com/aws/aws-sdk-go/aws/session"
    "github.com/aws/aws-sdk-go/service/managedblockchainquery"
)

func main() {

    // Set up a session
    ambQuerySession := session.Must(session.NewSessionWithOptions(session.Options{
        Config: aws.Config{
            Region: aws.String("us-east-1"),
        },
    }))
    client := managedblockchainquery.New(ambQuerySession)

    // inputs for GetTransaction API
    transactionHash :=
"0x123452695a82868950d9db8f64dfb2f6f0ad79284a6c461d115ede8930754321"
    network := managedblockchainquery.QueryNetworkEthereumMainnet

    // Call GetTransaction API. This operation will return transaction details for all
    // transactions that are confirmed on the blockchain, even if they have not

```

```
// reached #nality.
getTransactionRequest, getTransactionResponse :=
client.GetTransactionRequest(&managedblockchainquery.GetTransactionInput{
    Network:          &network,
    TransactionHash: &transactionHash,
})

errors := getTransactionRequest.Send()
if errors == nil {
    // handle API response
    fmt.Println(getTransactionResponse)
} else {
    // handle API errors
    fmt.Println(errors)
}
}
```

ファイルを保存したら、GetTransaction ディレクトリ内の次のコマンドを使用してコードを実行しますgo run GetTransaction.go。

次の出力は次のようになります。

```
{
  Transaction: {
    BlockHash: "0x000005c6a71d1afbc005a652b6ceca71cd516d97b0fc514c2a1d0f2ca3912345",
    BlockNumber: "11111111",
    CumulativeGasUsed: "5555555",
    EffectiveGasPrice: "444444444444",
    From: "0x9157f4de39ab4c657ad22b9f19997536*****",
    GasUsed: "22222",
    Network: "ETHEREUM_MAINNET",
    NumberOfTransactions: 111,
    SignatureR: "0x99999894fd2df2d039b3555dab80df66753f84be475069dfaf6c6103*****",
    SignatureS: "0x77777a101e7f37dd2dd0bf878b39080d5ecf3bf082c9bd4f40de783e*****",
    SignatureV: 0,
    ConfirmationStatus: "FINAL",
    ExecutionStatus: "SUCCEEDED",
    To: "0x5555564f282bf135d62168c1e513280d*****",
    TransactionHash:
"0x123452695a82868950d9db8f64dfb2f6f0ad79284a6c461d115ede8930754321",
    TransactionIndex: 11,
    TransactionTimestamp: 2022-02-02 01:01:59 +0000 UTC
  }
}
```

```
}
```

GetTokenBalance API は、ある時点で外部所有アカウント (EOA) の現在の残高を取得するために使用できるネイティブトークン (ETH と BTC) の残高を取得する方法を提供します。

Example— GetTokenBalance API アクションを使用して Go でネイティブトークンのバランスを取得します。

次の例では、GetTokenBalance API を使用して Ethereum Mainnet のアドレス Ether (ETH) 残高を取得します。次のコードを GetTokenBalance ディレクトリGetTokenBalanceEth.goの という名前のファイルにコピーします。

```
package main

import (
    "fmt"
    "github.com/aws/aws-sdk-go/aws"
    "github.com/aws/aws-sdk-go/aws/session"
    "github.com/aws/aws-sdk-go/service/managedblockchainquery"
)

func main() {
    // Set up a session
    ambQuerySession := session.Must(session.NewSessionWithOptions(session.Options{
        Config: aws.Config{
            Region: aws.String("us-east-1"),
        },
    }))
    client := managedblockchainquery.New(ambQuerySession)

    // inputs for GetTokenBalance API
    ownerAddress := "0xBeE510AF9804F3B459C0419826b6f225*****"
    network := managedblockchainquery.QueryNetworkEthereumMainnet
    nativeTokenId := "eth" //Ether on Ethereum mainnet

    // call GetTokenBalance API
    getTokenBalanceRequest, getTokenBalanceResponse :=
    client.GetTokenBalanceRequest(&managedblockchainquery.GetTokenBalanceInput{
        TokenIdentifier: &managedblockchainquery.TokenIdentifier{
            Network:      &network,
            TokenId: &nativeTokenId,
        },
        OwnerIdentifier: &managedblockchainquery.OwnerIdentifier{
```

```
        Address: &ownerAddress,
    },
})
errors := getTokenBalanceRequest.Send()

if errors == nil {
    // process API response
    fmt.Println(getTokenBalanceResponse)
} else {
    // process API errors
    fmt.Println(errors)
}
}
```

ファイルを保存したら、GetTokenBalance ディレクトリ内の次のコマンドを使用してコードを実行します。go run GetTokenBalanceEth.go。

次の出力は次のようになります。

```
{
  AtBlockchainInstant: {
    Time: 2020-12-05 11:51:01 +0000 UTC
  },
  Balance: "4343260710",
  LastTransactionHash:
  "0x00000ce94398e56641888f94a7d586d51664eb9271bf2b3c48297a50a0711111",
  LastTransactionTime: 2023-03-14 18:33:59 +0000 UTC,
  OwnerIdentifier: {
    Address: "0x12345d31750D727E6A3a7B534255BADd*****"
  },
  TokenIdentifier: {
    Network: "ETHEREUM_MAINNET",
    TokenId: "eth"
  }
}
```

## Node.js を使用して Amazon Managed Blockchain (AMB) クエリ API リクエストを作成する

これらのノード例を実行するには、次の前提条件が適用されます。

1. マシンにはノードバージョンマネージャー (nvm) と Node.js がインストールされている必要があります。OS のインストール手順については、[こちらを参照してください](#)。
2. `node --version` コマンドを使用して、Node バージョン 14 以降を使用していることを確認します。必要に応じて、`nvm install 14` コマンドを使用し、続いて `nvm use 14` コマンドを使用してバージョン 14 をインストールできます。
3. 環境変数 `AWS_ACCESS_KEY_ID` と `AWS_SECRET_ACCESS_KEY` には、アカウントに関連付けられている認証情報が含まれている必要があります。

次のコマンドを使用して、これらの変数をクライアントの文字列としてエクスポートします。以下の強調表示された値を IAM ユーザーアカウントの適切な値に置き換えます。

```
export AWS_ACCESS_KEY_ID="AKIAIOSFODNN7EXAMPLE"
export AWS_SECRET_ACCESS_KEY="wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY"
```

#### Note

- すべての前提条件を完了したら、HTTPS 経由で署名付きリクエストを送信して Amazon Managed Blockchain (AMB) クエリ API オペレーションにアクセスし、[Node.js のネイティブ https モジュール](#)を使用してリクエストを行うか、[AXIOS](#) などのサードパーティーライブラリを使用して AMB クエリからデータを取得できます。
- これらの例では Node.js にサードパーティーの HTTP クライアントを使用していますが、AWS JavaScript SDK を使用して AMB クエリにリクエストを行うこともできます。
- 次の例は、Axios と AWS SigV4 用の SDK モジュールを使用して AMB クエリ API リクエストを行う方法を示しています。

次の `package.json` ファイルをローカル環境の作業ディレクトリにコピーします。

```
{
  "name": "amb-query-examples",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
```

```
"license": "ISC",
"dependencies": {
  "@aws-crypto/sha256-js": "^4.0.0",
  "@aws-sdk/credential-provider-node": "^3.360.0",
  "@aws-sdk/protocol-http": "^3.357.0",
  "@aws-sdk/signature-v4": "^3.357.0",
  "axios": "^1.4.0"
}
}
```

Example— AMB クエリ GetTokenBalance API を使用して、特定の外部所有アドレス (EOA) から過去のトークン残高を取得する

GetTokenBalance API を使用して、さまざまなトークン (ERC20, ERC721, ERC1155 など) とネイティブコイン (ETH、BTC など) のバランスを取得できます。これを使用して、履歴 timestamp (Unix タイムスタンプ - 秒) に基づいて外部所有アカウント (EOA) の現在のバランスを取得できます。この例では、[GetTokenBalance](#) API を使用して、Ethereum Mainnet の ERC20 トークン USDC のアドレスバランスを取得します。

GetTokenBalance API をテストするには、次のコードを という名前のファイルにコピーし token-balance.js、そのファイルを同じ作業ディレクトリに保存します。

```
const axios = require('axios').default;
const SHA256 = require('@aws-crypto/sha256-js').Sha256
const defaultProvider = require('@aws-sdk/credential-provider-node').defaultProvider
const HttpRequest = require('@aws-sdk/protocol-http').HttpRequest
const SignatureV4 = require('@aws-sdk/signature-v4').SignatureV4

// define a signer object with AWS service name, credentials, and region
const signer = new SignatureV4({
  credentials: defaultProvider(),
  service: 'managedblockchain-query',
  region: 'us-east-1',
  sha256: SHA256,
});

const queryRequest = async (path, data) => {
  //query endpoint
  let queryEndpoint = `https://managedblockchain-query.us-east-1.amazonaws.com/
  ${path}`;

  // parse the URL into its component parts (e.g. host, path)
```

```
const url = new URL(queryEndpoint);

// create an HTTP Request object
const req = new HttpRequest({
  hostname: url.hostname.toString(),
  path: url.pathname.toString(),
  body: JSON.stringify(data),
  method: 'POST',
  headers: {
    'Content-Type': 'application/json',
    'Accept-Encoding': 'gzip',
    host: url.hostname,
  }
});

// use AWS SignatureV4 utility to sign the request, extract headers and body
const signedRequest = await signer.sign(req, { signingDate: new Date() });

try {
  //make the request using axios
  const response = await axios({...signedRequest, url: queryEndpoint, data: data})

  console.log(response.data)
} catch (error) {
  console.error('Something went wrong: ', error)
  throw error
}

}

let methodArg = 'get-token-balance';

let dataArg = {
  " atBlockchainInstant": {
    "time": 1688071493
  },
  "ownerIdentifier": {
    "address": "0xf3B0073E3a7F747C7A38B36B805247B2*****" // externally owned
address
  },
  "tokenIdentifier": {
```

```
    "contractAddress": "0xA0b86991c6218b36c1d19D4a2e9Eb0cE*****", //USDC contract
    address
      "network": "ETHEREUM_MAINNET"
    }
  }

//Run the query request.
queryRequest(methodArg, dataArg);
```

コードを実行するには、ファイルと同じディレクトリでターミナルを開き、次のコマンドを実行します。

```
npm i
node token-balance.js
```

このコマンドはスクリプトを実行し、コードで定義された引数を渡して、Ethereum Mainnet にリストされている EOA の ERC20 USDC 残高をリクエストします。応答は次の例のようになります。

```
{
  atBlockchainInstant: { time: 1688076218 },
  balance: '140386693440144',
  lastUpdatedTime: { time: 1688074727 },
  ownerIdentifier: { address: '0xf3b0073e3a7f747c7a38b36b805247b2*****' },
  tokenIdentifier: {
    contractAddress: '0xa0b86991c6218b36c1d19d4a2e9eb0ce*****',
    network: 'ETHEREUM_MAINNET'
  }
}
```

## Python を使用して Amazon Managed Blockchain (AMB) クエリ API リクエストを作成する

これらの Python の例を実行するには、次の前提条件が適用されます。

1. マシンに Python がインストールされている必要があります。OS のインストール手順については、[こちらを参照してください](#)。
2. [AWS SDK for Python \(Boto3\)](#) をインストールします。
3. [AWS コマンドラインインターフェイス](#) をインストールし、コマンドを実行して、Access Key ID、Secret Access Key およびの変数 `aws configure` を設定します。Region。

すべての前提条件を完了したら、SDK for Python over HTTPS を使用して AWS Amazon Managed Blockchain (AMB) クエリ API リクエストを行うことができます。

次の Python の例では、boto3 のモジュールを使用して、必要な SigV4 ヘッダーがアタッチされたリクエストを AMB クエリ ListTransactionEvents API オペレーションに送信します。この例では、Ethereum Mainnet 上の特定のトランザクションによって出力されるイベントのリストを取得します。

次の list-transaction-events.py ファイルをローカル環境の作業ディレクトリにコピーします。

```
import json
from botocore.auth import SigV4Auth
from botocore.awsrequest import AWSRequest
from botocore.session import Session
from botocore.httpsession import URLLib3Session

def signed_request(url, method, params, service, region):

    session = Session()
    sigv4 = SigV4Auth(session.get_credentials(), service, region)
    data = json.dumps(params)
    request = AWSRequest(method, url, data=data)
    sigv4.add_auth(request)
    http_session = URLLib3Session()
    response = http_session.send(request.prepare())

    return(response)

url = 'https://managedblockchain-query.us-east-1.amazonaws.com/list-transaction-events'
method = 'POST'
params = {
    'network': 'ETHEREUM_MAINNET',
    'transactionHash': '0x125714bb4db48757007fff2671b37637bbfd6d47b3a4757ebbd0c5222984f905'
}
service = 'managedblockchain-query'
region = 'us-east-1'

# Call the listTransactionEvents operation. This operation will return transaction
# details for
# all transactions that are confirmed on the blockchain, even if they have not reached
# finality.
listTransactionEvents = signed_request(url, method, params, service, region)
```

```
print(json.loads(ListTransactionEvents.content.decode('utf-8')))
```

サンプルコードを実行するには `ListTransactionEvents`、ファイルを作業ディレクトリに保存し、コマンドを実行します `python3 list-transaction-events.py`。このコマンドはスクリプトを実行し、コードで定義された引数を渡して、Ethereum Mainnet で指定されたトランザクションハッシュに関連付けられたイベントをリクエストします。応答は次の例のようになります。

```
{
  'events':
  [
    {
      'contractAddress': '0x95ad61b0a150d79219dcf64e1e6cc01f*****',
      'eventType': 'ERC20_TRANSFER',
      'from': '0xab5801a7d398351b8be11c439e05c5b3*****',
      'network': 'ETHEREUM_MAINNET',
      'to': '0xdead0000000000000000000000000000420694206942*****',
      'transactionHash':
      '0x125714bb4db48757007fff2671b37637bbfd6d47b3a4757ebbd0c522*****',
      'value': '410241996771871894771826174755464'
    }
  ]
}
```

## で Amazon Managed Blockchain (AMB) クエリ AWS マネジメント コンソール を使用して GetTokenBalance オペレーションを実行する

次の例は、の Amazon Managed Blockchain (AMB) クエリを使用して Ethereum Mainnet でトークンの残高を取得する方法を示しています。AWS マネジメントコンソール

### Example

1. <https://console.aws.amazon.com/managedblockchain/> で Amazon Managed Blockchain コンソールを開きます。
2. クエリセクションからクエリエディタを選択します。
3. ブロックチェーンネットワークとして ETHEREUM\_MAINNET を選択します。
4. クエリタイプとして GetTokenBalance を選択します。

5. トークンのブロックチェーンアドレスを入力します。
6. トークンの契約アドレスを入力します。
7. トークンのオプションのトークン ID を入力します。
8. トークン残高の日付を選択します。
9. トークン残高のオプションの At time を入力します。
10. [Run query] (クエリの実行) を選択します。

AMB クエリはクエリを実行し、クエリ結果ウィンドウに結果が表示されます。

# Amazon Managed Blockchain (AMB) クエリのユースケース

このトピックでは、AMB クエリのユースケースを一覧表示します。

## トピック

- [現在および過去のトークン残高をクエリする](#)
- [履歴トランザクションデータを取得する](#)
- [特定のアドレスのすべてのトークン残高を取得する](#)
- [トランザクションに対して出力されたイベントを一覧表示する](#)
- [契約によって作成されたすべてのトークンを取得する](#)
- [契約を一覧表示し、契約情報を取得する](#)

## 現在および過去のトークン残高をクエリする

[GetTokenBalance](#) API は、サポートされているトークン (ERC20、ERC721、ERC1155 とネイティブコイン (ETH、BTC) のバランスを取得し、外部所有アカウント (EOAs) のユニバーサルタイムスタンプ (Unix タイムスタンプ、秒単位) を使用して、現在または過去のバランスを取得します。たとえば、[GetTokenBalance](#) API オペレーションを使用して、Ethereum Mainnet の ERC20 トークン USDC のアドレスバランスを取得できます。[BatchGetTokenBalance](#) API オペレーションを使用して、トークンとネイティブコインのバランスをバッチ取得することもできます。

詳細については、「[Amazon Managed Blockchain \(AMB\) クエリリファレンスガイド](#)」を参照してください。

## 履歴トランザクションデータを取得する

Amazon Managed Blockchain (AMB) クエリを使用すると、Ethereum や Bitcoin などのパブリックブロックチェーンから履歴データを取得できます。この機能により、ブロックチェーンウォレットのトランザクション履歴の取得や、トランザクションハッシュに基づくトランザクションに関するコンテキスト情報の提供など、いくつかのユースケースが可能になります。[ListTransactions](#) API オペレーションを使用して Ethereum Mainnet 上の特定の外部所有アドレス (EOA) のトランザクションのリストを取得し、[GetTransaction](#) API オペレーションを使用してリストから 1 つのトランザクションの詳細を取得できます。

詳細については、「[Amazon Managed Blockchain \(AMB\) クエリリファレンスガイド](#)」を参照してください。

## 特定のアドレスのすべてのトークン残高を取得する

[ListTokenBalances](#) API オペレーションを使用して、ウォレット、ユーザーインターフェイス、web3 ユーティリティなどの残高を取得できます。この API オペレーションは、単一の API オペレーションを使用して、特定のパブリックブロックチェーン上のトークン (ERC20、ERC721、ERC1155) とネイティブコイン (ETH、BTC) にわたるアドレスのすべての残高のリストを返します。たとえば、外部所有のアドレス (EOA) とネットワーク (Ethereum Mainnet) を指定し、レスポンスでトークンとネイティブコインバランスのリストを受け取ることができます。

詳細については、「[Amazon Managed Blockchain \(AMB\) クエリリファレンスガイド](#)」を参照してください。

## トランザクションに対して出力されたイベントを一覧表示する

[ListTransactionEvents](#) API オペレーションを使用して、ハッシュ (トランザクション識別子) によって識別される、特定のトランザクションの結果として出力される契約イベントのリストを取得できます。たとえば、[ListTransactionEvents](#) を使用して、ERC20 契約からの転送イベントや取り消しイベントなど、Ethereum ブロックチェーンで ERC20 トークン契約の関数を呼び出すトランザクションの結果イベントを取得できます。

詳細については、「[Amazon Managed Blockchain \(AMB\) クエリリファレンスガイド](#)」を参照してください。

## 契約によって作成されたすべてのトークンを取得する

[ListTokenBalances](#) API オペレーションを使用して、契約アドレスを入力として渡したときに、契約によって作成されたすべてのサポートされているトークン (ERC20、ERC721、ERC1155) のリストを返すことができます。例えば、[ListTokenBalances](#) API オペレーションを使用して、Ethereum ブロックチェーンの ERC721 契約標準によって作成された非ファンジブルトークン (NFTs) に関する情報を取得できます。

詳細については、「[Amazon Managed Blockchain \(AMB\) クエリリファレンスガイド](#)」を参照してください。

## 契約を一覧表示し、契約情報を取得する

[ListAssetContracts](#) API オペレーションを使用して、特定のアドレスによってデプロイされた ERC-721、ERC-1155、または ERC-20 契約を一覧表示できます。さらに、契約アドレスがある場合

は、[GetAssetContract](#) API オペレーションを使用して、契約タイプのデプロイ先のアドレスや関連するトークンメタデータなど、契約のプロパティを取得できます。

詳細については、「[Amazon Managed Blockchain \(AMB\) クエリリファレンスガイド](#)」を参照してください。

# Amazon Managed Blockchain (AMB) クエリ API リファレンス

Amazon Managed Blockchain (AMB) クエリは、サポートされているブロックチェーンをクエリするための API オペレーションを提供します。これには、トークン、トランザクション、および契約をクエリするための APIs が含まれます。詳細については、[「AMB クエリ API リファレンス」](#)を参照してください。

# Amazon Managed Blockchain (AMB) クエリのセキュリティ

のクラウドセキュリティ AWS が最優先事項です。お客様は AWS、セキュリティを最も重視する組織の要件を満たすように構築されたデータセンターとネットワークアーキテクチャからメリットを得られます。

セキュリティは、AWS お客様とお客様の間の責任共有です。[責任共有モデル](#)では、これをクラウドのセキュリティとクラウドのセキュリティの両方として説明しています。

- クラウドのセキュリティ – AWS は、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があります AWS クラウド。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。サードパーティーの監査人は、[AWS コンプライアンスプログラム](#) の一環として、セキュリティの有効性を定期的にテストおよび検証します。Amazon Managed Blockchain (AMB) クエリに適用されるコンプライアンスプログラムの詳細については、[AWS 「コンプライアンスプログラムによる対象範囲内のサービス」](#) を参照してください。
- クラウド内のセキュリティ – お客様の責任は、使用する AWS サービスによって決まります。また、お客様は、お客様のデータの機密性、企業の要件、および適用可能な法律や規制といった他の要因 についても責任を担います。

データ保護、認証、アクセスコントロールを提供するために、Amazon Managed Blockchain は AWS Managed Blockchain で実行されているオープンソースフレームワークの機能を使用します。

このドキュメントは、AMB クエリを使用する際の責任共有モデルの適用方法を理解するのに役立ちます。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ように AMB クエリを設定する方法について説明します。また、AMB クエリ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

- [データ暗号化](#)
- [Amazon Managed Blockchain \(AMB\) クエリの Identity and Access Management](#)

## データ暗号化

データ暗号化は、権限のないユーザーがブロックチェーンネットワークおよび関連するデータストレージシステムからデータを読み取るのを防ぐのに役立ちます。これには、転送中のデータと呼ばれる、ネットワークを通過するときに傍受される可能性のあるデータが含まれます。

## 転送中の暗号化

デフォルトでは、Managed Blockchain は HTTPS/TLS 接続を使用して、AWS CLI クライアントから AWS サービスエンドポイントに送信されるすべてのデータを暗号化します。

## Amazon Managed Blockchain (AMB) クエリの Identity and Access Management

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 AMB クエリリソースの使用を許可する (アクセス許可を付与する) かを制御します。IAM は、追加料金なしで使用できる AWS のサービス です。

### トピック

- [オーディエンス](#)
- [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](#)
- [Amazon Managed Blockchain \(AMB\) クエリと IAM の連携](#)
- [Amazon Managed Blockchain \(AMB\) クエリのアイデンティティベースのポリシーの例](#)
- [Amazon Managed Blockchain \(AMB\) クエリのアイデンティティとアクセスのトラブルシューティング](#)

## オーディエンス

AWS Identity and Access Management (IAM) の使用方法は、ロールによって異なります。

- サービスユーザー - 機能にアクセスできない場合は、管理者にアクセス許可をリクエストします (「[Amazon Managed Blockchain \(AMB\) クエリのアイデンティティとアクセスのトラブルシューティング](#)」を参照)。
- サービス管理者 - ユーザーアクセスを決定し、アクセス許可リクエストを送信します (「[Amazon Managed Blockchain \(AMB\) クエリと IAM の連携](#)」を参照)
- IAM 管理者 - アクセスを管理するためのポリシーを作成します (「[Amazon Managed Blockchain \(AMB\) クエリのアイデンティティベースのポリシーの例](#)」を参照)

## 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。、IAM ユーザー 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で認証される必要があります。

AWS IAM アイデンティティセンター (IAM Identity Center)、シングルサインオン認証、Google/Facebook 認証情報などの ID ソースからの認証情報を使用して、フェデレーテッド ID としてサインインできます。サインインの詳細については、「AWS サインイン ユーザーガイド」の「[AWS アカウントにサインインする方法](#)」を参照してください。

プログラムによるアクセスの場合、は SDK と CLI AWS を提供してリクエストを暗号化して署名します。詳細については、「IAM ユーザーガイド」の「[API リクエストに対するAWS 署名バージョン 4](#)」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、すべての AWS のサービスおよび リソースへの完全なアクセス権を持つ AWS アカウント ルートユーザーと呼ばれる 1 つのサインインアイデンティティから始めます。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。ルートユーザー認証情報を必要とするタスクについては、「IAM ユーザーガイド」の「[ルートユーザー認証情報が必要なタスク](#)」を参照してください。

### 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、人間のユーザーが一時的な認証情報 AWS のサービス を使用してにアクセスするには、ID プロバイダーとのフェデレーションを使用する必要があります。

フェデレーション ID は、エンタープライズディレクトリ、ウェブ ID プロバイダー、または ID Directory Service ソースの認証情報 AWS のサービス を使用してにアクセスするユーザーです。フェデレーテッドアイデンティティは、一時的な認証情報を提供するロールを引き受けます。

アクセスを一元管理する場合は、AWS IAM アイデンティティセンターをお勧めします。詳細については、「AWS IAM アイデンティティセンター ユーザーガイド」の「[IAM アイデンティティセンターとは](#)」を参照してください。

### IAM ユーザーとグループ

[IAM ユーザー](#)は、特定の個人やアプリケーションに対する特定のアクセス許可を持つアイデンティティです。長期認証情報を持つ IAM ユーザーの代わりに一時的な認証情報を使用することをお勧め

します。詳細については、IAM ユーザーガイドの「[ID プロバイダーとのフェデレーションを使用し  
てにアクセスする必要がある AWS](#)」を参照してください。

[IAM グループ](#)は、IAM ユーザーの集合を指定し、大量のユーザーに対するアクセス許可の管理を容易にします。詳細については、「IAM ユーザーガイド」の「[IAM ユーザーに関するユースケース](#)」を参照してください。

## IAM ロール

[IAM ロール](#)は、特定のアクセス許可を持つアイデンティティであり、一時的な認証情報を提供します。ユーザーから [IAM ロール \(コンソール\)](#) に切り替えるか、または [API オペレーション](#) を呼び出すことで、[ロール](#) を引き受けることができます。AWS CLI AWS 詳細については、「IAM ユーザーガイド」の「[ロールを引き受けるための各種方法](#)」を参照してください。

IAM ロールは、フェデレーションユーザーアクセス、一時的な IAM ユーザーのアクセス許可、クロスアカウントアクセス、クロスサービスアクセス、および Amazon EC2 で実行するアプリケーションに役立ちます。詳細については、IAM ユーザーガイドの [IAM でのクロスアカウントリソースアクセス](#) を参照してください。

## ポリシーを使用したアクセスの管理

でアクセスを制御する AWS には、ポリシーを作成し、ID AWS またはリソースにアタッチします。ポリシーは、アイデンティティまたはリソースに関連付けられたときにアクセス許可を定義します。は、プリンシパル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。ほとんどのポリシーは JSON ドキュメント AWS として保存されます。JSON ポリシードキュメントの詳細については、「IAM ユーザーガイド」の「[JSON ポリシー概要](#)」を参照してください。

管理者は、ポリシーを使用して、どのプリンシパルがどのリソースに対して、どのような条件でアクションを実行できるかを定義することで、誰が何にアクセスできるかを指定します。

デフォルトでは、ユーザーやロールにアクセス許可はありません。IAM 管理者は IAM ポリシーを作成してロールに追加し、このロールをユーザーが引き受けられるようにします。IAM ポリシーは、オペレーションの実行方法を問わず、アクセス許可を定義します。

## アイデンティティベースのポリシー

アイデンティティベースのポリシーは、アイデンティティ (ユーザー、グループ、またはロール) 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、アイデンティティがどのリソースに対してどのような条件下でどのようなアクションを実行できるかを制御し

ます。アイデンティティベースポリシーの作成方法については、IAM ユーザーガイドの [カスタマー管理ポリシーでカスタム IAM アクセス許可を定義する](#) を参照してください。

アイデンティティベースのポリシーは、インラインポリシー (単一の ID に直接埋め込む) または管理ポリシー (複数の ID にアタッチされたスタンドアロンポリシー) にすることができます。管理ポリシーとインラインポリシーのいずれかを選択する方法については、「IAM ユーザーガイド」の [管理ポリシーとインラインポリシーのいずれかを選択する](#) を参照してください。

## 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、リソースに添付する JSON ポリシードキュメントです。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制御できます。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。

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では、IAM の AWS マネージドポリシーを使用できません。

## その他のポリシータイプ

AWS は、より一般的なポリシータイプによって付与されるアクセス許可の最大数を設定できる追加のポリシータイプをサポートしています。

- アクセス許可の境界 – アイデンティティベースのポリシーで IAM エンティティに付与することのできるアクセス許可の数の上限を設定します。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 (SCP) - AWS Organizations内の組織または組織単位の最大のアクセス許可を指定します。詳細については、「AWS Organizations ユーザーガイド」の [サービスコントロールポリシー](#) を参照してください。
- リソースコントロールポリシー (RCP) – は、アカウント内のリソースで利用できる最大数のアクセス許可を定義します。詳細については、「AWS Organizations ユーザーガイド」の [リソースコントロールポリシー \(RCP\)](#) を参照してください。
- セッションポリシー – ロールまたはフェデレーションユーザーの一時セッションを作成する際にパラメータとして渡される高度なポリシーです。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アクセス許可を理解するのがさらに難しくなります。が複数のポリシータイプが関与する場合にリクエストを許可するかどうか AWS を決定する方法については、「IAM ユーザーガイド」の「[ポリシー評価ロジック](#)」を参照してください。

## Amazon Managed Blockchain (AMB) クエリと IAM の連携

IAM を使用して AMB クエリへのアクセスを管理する前に、AMB クエリで使用できる IAM 機能を確認してください。

### Amazon Managed Blockchain (AMB) クエリで使用できる IAM 機能

IAM 機能	AMB クエリのサポート
<a href="#">アイデンティティベースのポリシー</a>	あり
<a href="#">リソースベースのポリシー</a>	なし
<a href="#">ポリシーアクション</a>	あり
<a href="#">ポリシーリソース</a>	いいえ
<a href="#">ポリシー条件キー</a>	いいえ
<a href="#">ACL</a>	なし
<a href="#">ABAC (ポリシー内のタグ)</a>	いいえ
<a href="#">一時的な認証情報</a>	あり
<a href="#">プリンシパルアクセス権限</a>	あり
<a href="#">サービスロール</a>	いいえ
<a href="#">サービスリンクロール</a>	いいえ

AMB クエリおよびその他の AWS のサービスがほとんどの IAM 機能と連携する方法の概要については、IAM ユーザーガイドの[AWS 「IAM と連携する のサービス」](#)を参照してください。

## AMB クエリの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のサポート: あり

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

IAM アイデンティティベースのポリシーでは、許可または拒否するアクションとリソース、およびアクションを許可または拒否する条件を指定できます。JSON ポリシーで使用できるすべての要素について学ぶには、「IAM ユーザーガイド」の「[IAM JSON ポリシーの要素のリファレンス](#)」を参照してください。

AMB クエリのアイデンティティベースのポリシーの例

AMB クエリのアイデンティティベースのポリシーの例を表示するには、「」を参照してください [Amazon Managed Blockchain \(AMB\) クエリのアイデンティティベースのポリシーの例](#)。

## AMB クエリ内のリソースベースのポリシー

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、リソースに添付する JSON ポリシードキュメントです。リソースベースのポリシーには例として、IAM ロールの信頼ポリシーや Amazon S3 バケットポリシーがあげられます。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、[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。詳細については、IAM ユーザーガイドの [IAM でのクロスアカウントリソースアクセス](#) を参照してください。

## AMB クエリのポリシーアクション

ポリシーアクションのサポート: あり

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

JSON ポリシーの Action 要素にはポリシー内の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が記述されます。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アクセス許可を付与するポリシーで使用されます。

AMB クエリ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「[Amazon Managed Blockchain \(AMB\) クエリで定義されるアクション](#)」を参照してください。

AMB クエリ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を使用します。

```
managedblockchain-query:
```

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、アクションをカンマで区切ります。

```
"Action": [  
  "managedblockchain-query::ListTransaction",  
  "managedblockchain-query::GetTransaction"  
]
```

AMB クエリのアイデンティティベースのポリシーの例を表示するには、「」を参照してください [Amazon Managed Blockchain \(AMB\) クエリのアイデンティティベースのポリシーの例](#)。

## AMB クエリのポリシーリソース

ポリシーリソースのサポート: なし

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

Resource JSON ポリシー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。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。リソースレベルのアクセス許可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": "*"
```

AMB クエリリソースタイプとその ARNs [「Amazon Managed Blockchain \(AMB\) クエリで定義されるリソース」](#)を参照してください。各リソースの ARN を指定できるアクションについては、[「Amazon Managed Blockchain \(AMB\) クエリで定義されるアクション」](#)を参照してください。

AMB クエリのアイデンティティベースのポリシーの例を表示するには、「」を参照してください [Amazon Managed Blockchain \(AMB\) クエリのアイデンティティベースのポリシーの例](#)。

## AMB クエリのポリシー条件キー

サービス固有のポリシー条件キーへのサポート: なし

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

Condition 要素は、定義された基準に基づいてステートメントが実行される時期を指定します。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して、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。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。

AMB クエリ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 [「Amazon Managed Blockchain \(AMB\) クエリの条件キー」](#) 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、[「Amazon Managed Blockchain \(AMB\) Query で定義されるアクション」](#) を参照してください。

AMB クエリのアイデンティティベースのポリシーの例を表示するには、「」を参照してください [Amazon Managed Blockchain \(AMB\) クエリのアイデンティティベースのポリシーの例](#)。

## AMB ACLs

ACL のサポート: なし

アクセスコントロールリスト (ACL) は、どのプリンシパル (アカウントメンバー、ユーザー、またはロール) がリソースにアクセスするためのアクセス許可を持つかを制御します。ACL はリソースベースのポリシーに似ていますが、JSON ポリシードキュメント形式は使用しません。

## AMB クエリを使用した ABAC

ABAC (ポリシー内のタグ) のサポート: なし

属性ベースのアクセス制御 (ABAC) は、タグと呼ばれる属性に基づいてアクセス許可を定義する認可戦略です。IAM エンティティと AWS リソースにタグをアタッチし、プリンシパルのタグがリソースのタグと一致するときにオペレーションを許可するように ABAC ポリシーを設計できます。

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。

ABAC の詳細については、「IAM ユーザーガイド」の「[ABAC 認可でアクセス許可を定義する](#)」を参照してください。ABAC をセットアップする手順を説明するチュートリアルについては、「IAM 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。

## AMB クエリでの一時的な認証情報の使用

一時的な認証情報のサポート: あり

一時的な認証情報は、AWS リソースへの短期的なアクセスを提供し、フェデレーションまたはスイッチロールの使用時に自動的に作成されます。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 AWS 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「[IAM の一時的な認証情報](#)」および「[AWS のサービスと IAM との連携](#)」を参照してください。

## AMB クエリのクロスサービスプリンシパルアクセス許可

転送アクセスセッション (FAS) のサポート: あり

転送アクセスセッション (FAS) は、`sts:AssumeRole` 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 へのリクエストをリクエストする `sts:AssumeRoleWithSAML` を使用します。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。

## AMB クエリのサービスロール

サービスロールのサポート: なし

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

では、IAM ユーザーガイドの [AWS のサービスに許可を委任するロールを作成する](#) を参照してください。

#### Warning

サービスロールのアクセス許可を変更すると、AMB クエリ機能が破損する可能性があります。AMB クエリが指示する場合にのみ、サービスロールを編集します。

## AMB クエリの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 と提携するAWS のサービス](#)」を参照してください。表の「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。

## Amazon Managed Blockchain (AMB) クエリのアイデンティティベースのポリシーの例

デフォルトでは、ユーザーとロールには AMB クエリリソースを作成または変更するアクセス許可はありません。IAM 管理者は、リソースで必要なアクションを実行するための権限をユーザーに付与する IAM ポリシーを作成できます。

これらのサンプルの JSON ポリシードキュメントを使用して 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 ポリシーを作成する \(コンソール\)](#)」を参照してください。

各リソースタイプの ARNs 「サービス認可リファレンス」の「[Amazon Managed Blockchain \(AMB\) クエリのアクション、リソース、および条件キー](#)」を参照してください。

### トピック

- [ポリシーに関するベストプラクティス](#)
- [ユーザーが自分の権限を表示できるようにする](#)
- [特定の Amazon Managed Blockchain \(AMB\) クエリ API アクションへのアクセス](#)

## ポリシーに関するベストプラクティス

ID ベースのポリシーは、アカウント内で誰かが AMB クエリリソースを作成、アクセス、または削除できるかどうかを決定します。これらのアクションでは、AWS アカウントに費用が発生する場合があります。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 – ユーザーとワークロードにアクセス許可の付与を開始するには、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することで、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 [ジョブ機能のAWS マネージドポリシー](#) 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 で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 SSL を使用して送信するように指定できます。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ている場合に 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 CloudFormation。詳細については、IAM ユーザーガイドの [IAM JSON ポリシー要素:条件](#) を参照してください。
- IAM アクセスアナライザーを使用して IAM ポリシーを検証し、安全で機能的な権限を確保する - IAM アクセスアナライザーは、新規および既存のポリシーを検証して、ポリシーが IAM ポリシー言語 (JSON) および IAM のベストプラクティスに準拠するようにします。IAM アクセスアナライザーは 100 を超えるポリシーチェックと実用的な推奨事項を提供し、安全で機能的なポリシーの作成をサポートします。詳細については、IAM ユーザーガイドの [IAM Access Analyzer でポリシーを検証する](#) を参照してください。
- 多要素認証 (MFA) を要求する – 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、MFA をオンにしてセキュリティを強化します。API オペレーション

が呼び出されるときに MFA を必須にするには、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 [MFA を使用した安全な API アクセス](#) 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 でのセキュリティベストプラクティス](#)」を参照してください。

## ユーザーが自分の権限を表示できるようにする

この例では、ユーザーアイデンティティにアタッチされたインラインおよびマネージドポリシーの表示を IAM ユーザーに許可するポリシーの作成方法を示します。このポリシーには、コンソールで、または AWS CLI または AWS API を使用してプログラムでこのアクションを実行するアクセス許可が含まれています。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
        "iam:ListGroupsWithUser",
        "iam:ListAttachedUserPolicies",
        "iam:ListUserPolicies",
        "iam:GetUser"
      ],
      "Resource": ["arn:aws:iam::*:user/${aws:username}"]
    },
    {
      "Sid": "NavigateInConsole",
      "Effect": "Allow",
      "Action": [
        "iam:GetGroupPolicy",
        "iam:GetPolicyVersion",
        "iam:GetPolicy",
        "iam:ListAttachedGroupPolicies",
        "iam:ListGroupPolicies",
        "iam:ListPolicyVersions",
        "iam:ListPolicies",
        "iam:ListUsers"
      ],
      "Resource": "*"
    }
  ]
}
```

```
    }  
  ]  
}
```

## 特定の Amazon Managed Blockchain (AMB) クエリ API アクションへのアクセス

### Note

AMB クエリにアクセスして API コールを行うには、AMB クエリに適切な IAM アクセス許可を持つユーザー認証情報 (AWS\_ACCESS\_KEY\_ID および AWS\_SECRET\_ACCESS\_KEY) が必要です。

Example すべての Amazon Managed Blockchain (AMB) クエリ APIs にアクセスするための IAM ポリシー

この例では、すべての AMB クエリ API AWS アカウント へのアクセス権を IAM ユーザーに付与します。 APIs

JSON

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "AccessAllAMBQueryAPIs",  
      "Effect": "Allow",  
      "Action": [  
        "managedblockchain-query:*"  
      ],  
      "Resource": "*"   
    }  
  ]  
}
```

## Example Amazon Managed Blockchain (AMB) クエリ `ListTransactions` と `GetTransaction` APIs にアクセスするための IAM ポリシー

この例では、AMB クエリ `ListTransaction` と `GetTransaction` API AWS アカウント へのアクセス権を IAM ユーザーに付与します。 APIs

### Note

この例 APIs を他の API に置き換えたり追加したりして、他の APIs または複数の APIs へのアクセスを許可したりできます。AMB クエリ APIs 「Amazon Managed Blockchain (AMB) クエリ API リファレンスガイド」を参照してください。

## JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccessAMBQueryAPIs",
      "Effect": "Allow",
      "Action": [
        "managedblockchain-query:ListTransactions",
        "managedblockchain-query:GetTransaction"
      ],
      "Resource": "*"
    }
  ]
}
```

## Amazon Managed Blockchain (AMB) クエリのアイデンティティとアクセスのトラブルシューティング

以下の情報は、AMB クエリと IAM の使用時に発生する可能性がある一般的な問題の診断と修正に役立ちます。

### トピック

- [AMB クエリでアクションを実行する権限がありません](#)

## AMB クエリでアクションを実行する権限がありません

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、そのアクションを実行できるようにポリシーを更新する必要があります。

次のエラー例は、mateojackson IAM ユーザーがコンソールを使用して、ある *my-example-widget* リソースに関する詳細情報を表示しようとしたことを想定して、その際に必要な `managedblockchain-query::GetWidget` アクセス許可を持っていない場合に発生するものです。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
managedblockchain-query::GetWidget on resource: my-example-widget
```

この場合、`managedblockchain-query::GetWidget` アクションを使用して *my-example-widget* リソースへのアクセスを許可するように、mateojackson ユーザーのポリシーを更新する必要があります。

サポートが必要な場合は、AWS 管理者にお問い合わせください。サインイン資格情報を提供した担当者が管理者です。

# Amazon CloudWatch での Amazon Managed Blockchain (AMB) クエリ API 使用状況メトリクス

## Amazon CloudWatch の API 使用状況メトリクス

CloudWatch に発行される API 使用状況メトリクスは、Amazon Managed Blockchain (AMB) クエリサービスクォータに対応しています。使用量がサービスクォータに近づいたときに警告するようにアラームを設定できます。CloudWatch とサービスクォータの統合の詳細については、「Amazon CloudWatch ユーザーガイド」の「[AWS 使用状況メトリクス](#)」を参照してください。

AMB クエリは、AWS/Usage 名前空間に次の API メトリクスを Amazon Managed Blockchain Query サービス名で発行します。

メトリクス	説明
CallCount	AMB クエリで API に対して行われた呼び出しの合計数。SUM は、指定した期間に API に対して行われた呼び出しの総数を表します。

Amazon Managed Blockchain (AMB) クエリは、次のディメンションを使用して使用状況メトリクスを AWS/Usage 名前空間に発行します。

ディメンション	説明
Service	リソースを含む AWS サービスの名前。Amazon Managed Blockchain Query は常にこのディメンションの値です。
タイプ	レポートされるエンティティのタイプ。API は常にこのディメンションの値です。
[リソース]	レポートされるリソースのタイプ。使用される <a href="#">AMB クエリ API オペレーション</a> の名前は、このディメンションの値になります。

ディメンション	説明
Class	レポートされるリソースのクラス。Noneは常にこのディメンションの値です。

# AMB クエリユーザーガイドのドキュメント履歴

次の表に、AMB クエリのドキュメントリリースを示します。

変更	説明	日付
<a href="#">AMB クエリが Bitcoin トランザクション識別子とトランザクションハッシュをサポート</a>	Bitcoin ネットワークの場合、AMB クエリ API オペレーションはトランザクション識別子 (transactionId ) とトランザクションハッシュ () の両方をサポートし、transactionHash をサポートしません。	2024 年 3 月 21 日
<a href="#">Amazon CloudWatch での API 使用状況メトリクスのサポート</a>	AMB Query に、CloudWatch での API 使用状況メトリクスのサポートが追加されました。これらの使用状況メトリクスは、AMB クエリサービスクォータに対応しています。	2024 年 2 月 8 日
<a href="#">確定に達していないトランザクションのサポート</a>	AMB Query は、 <a href="#">確定に達していないトランザクション</a> のサポートを追加しました。また、GetTransaction オペレーションのレスポンスから status プロパティのサポートも削除されます。代わりに、プロパティ confirmationStatus と executionStatus プロパティを使用してトランザクションのステータスを決定します。	2024 年 2 月 1 日
<a href="#">トランザクションデータ型の status プロパティの廃止</a>	Amazon Managed Blockchain (AMB) クエリは、トランザクションデータ型の	2023 年 12 月 20 日

statusプロパティを廃止しました。confirmationStatus およびexecutionStatus フィールドを使用して、トランザクションstatusのが FINALかかを判断する必要があります。FAILED。

### [Sepolia Testnet のサポート](#)

Amazon Managed Blockchain (AMB) Query は、Ethereum Sepolia Testnet でのクエリをサポートするようになりました。

2023 年 10 月 19 日

### [アセット契約のサポート](#)

[ListAssetContracts](#) API オペレーションを使用して、特定のアドレスによってデプロイされたを一覧表示できます。さらに、契約住所がある場合は、[GetAssetContract](#) API オペレーションを使用して契約の詳細を取得できます。

2023 年 10 月 16 日

### [Bitcoin Testnet のサポート](#)

Amazon Managed Blockchain (AMB) クエリが Bitcoin Testnet のクエリをサポートするようになりました。

2023 年 10 月 16 日

### [初回リリース](#)

AMB クエリサービスの初回リリース。

2023 年 7 月 27 日

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。