# Azure Database for PostgreSQLのハイパースケール (Citus) の概要

Azure Database for PostgreSQLは、クラウドで可用性の高いPostgreSQLデータベースを実行、管理、および拡張するために使用するマネージドサービスです。この手順では、Azureポータルを使用してAzure Database for PostgreSQLのハイパースケール (Citus) のサーバーグループを作成する方法について説明します。分散データ（ノード間でのシャーディングテーブル、サンプルデータの読み込み、複数のノードで実行されるクエリの実行）について説明します。

# ハイパースケール (Citus) の概要

Azure Database for PostgreSQLのハイパースケール (Citus) は、スケールアウトするために構築された安心なPostgresです。複数のマシンのクラスタに、データ、およびクエリを配布（シャード）します。ハイパースケール (Citus) は、（フォークではなく）拡張機能として、新しいPostgreSQLリリースをサポートしており、既存のPostgreSQLツールとの互換性を維持しながら、ユーザーが新機能の恩恵を受けられるようにします。既存のワークロードにおけるパフォーマンスとスケーラビリティの2つの主要な問題を解決します。通常のPostgreSQLと同様に、ハイパースケール (Citus) もデータベースを管理します。高可用性、バックアップ、監視、アラート、その他の追加機能を提供し、これらはPaaS（サービスとしてのプラットフォーム）の一部として提供されます。

# ハイパースケール (Citus) のアーキテクチャ:

クラスタには、コーディネーターと呼ばれる1つの特殊ノードが必ずあります（他のノードはワーカーと呼ばれます）。アプリケーションは、コーディネーターにクエリを送信し、コーディネーターは関連付けられたワーカーにクエリを中継し、結果を蓄積します。

各クエリについて、**コーディネーター**は、それを単一のワーカーにルーティングするか、必要なデータが単一あるいは複数のワーカーに存在するかに応じて、クエリをいくつかに並列化します。ハイパースケール (Citus) が複数のワーカーにクエリを分散する方法に関するシナリオを次に示します。

スクリーンショット が含まれている画像

自動的に生成された説明

1. このウインドウの右下にある**Next**をクリックします。

# Azure Database for PostgreSQLのハイパースケール (Citus) の概要

まず、提供された資格情報を使用してAzureポータルにログインする必要があります。

# Azureポータルへのサイン・イン

□1. Azure Portalに既にログインしている場合は、次のページにスキップします。このウィンドウの右下にある**Next**をクリックします。

□2. ブラウザで <https://portal.azure.com> を開き、ブラウザのウインドウを最大化します。

□3. Pick an accountというダイアログが表示されたら、+ Use another accountを選択します。

□4. Sign inダイアログの、Email, phone or Skypeフィールドに  
userxxxxxx@cloudplatimmersionlabs.onmicrosoft.com を入力しNextをクリックします。

□5. **Password**フィールドに**xxxxxxxx**を入力します。

□6. **Sign in**をクリックします。

□7. **Stay signed in?** とタイトルがついた、**No**と**Yes**ボタンがあるポップアップが表示されるかもしれません。**No**を選択します。

□8. Welcome to Microsoft AzureとタイトルがついたStart TourとMaybe Laterボタンがあるポップアップが表示されるかもしれません。Maybe Laterを選択します。

□9. このウインドウの右下にある**Next**をクリックします。

# PostgreSQLのハイパースケール (Citus) 拡張を使い始める

これらの手順では、Azureポータルを使用してハイパースケール (Citus) サーバーグループを作成する方法について説明します。通常、これは約10分かかりますが、時間の節約のために事前に作成してあります。以下の手順では、プロセスを通じてこれがいかにシンプルで簡単であるかが分かりますが、最終的には事前に作成されたものを使用します。

# Azure Database for PostgreSQLのハイパースケール (Citus) を作成する

次の手順に従って、Azureポータルを使用してPostgreSQLハイパースケール (Citus) サーバーグループを作成するプロセスを理解します。

□1. Azureポータルの左上にある**+ Create a resource**をクリックします。

□2. Azure MarketplaceのNewページで**Databases**を選択し、Databasesページで**Azure Database for PostgreSQL**を選択します。

□3. デプロイメントオプションページで、Hyperscale (Citus) on Azure Database for PostgreSQLの下の**Create**ボタンをクリックします。

□4. 以下の情報を新しいサーバーの詳細に入力します。

* Subscription: あなたのセッションのサブスクリプションがデフォルトになっています。
* Resource Group: **select existing...**ドロップダウンをクリックし、**rg000000**を選択します。
* Server group name: **sg000000**を入力します。
* Admin username: 現時点では**citus**という値が必須です。
* Password: **xxxxxxxx**を入力し、**Confirm Password**にも同じパスワードを入力します。
* Location: **eastus2**を選択します。
* Compute + Storage: **Configure server group**をクリックします。このセクションの設定はそのままにして**Save**をクリックします。

|  |
| --- |
| 注: ハイパースケール (Citus) デプロイを作成する場合、最大20のワーカーノードを水平方向にスケーリングできます。20以上のノードが必要な場合は、サポートチケットを作成するだけで、有効になります。コーディネーターと同様に、すべてのワーカー（コア、ストレージ）をセットアップ/ダウンできます。RAMは、コア数とサーバーの種類（コーディネーターまたはワーカー）で決まります。 |

□5. Review + createをクリックするとサマリーが表示されます。ここでは**絶対にCreateをクリックしない**でください。時間を節約するために事前に作成してあります。

|  |
| --- |
| 注: Createをクリックすると、デプロイに最大10分かかります。待機中にデプロイメントを監視するページにリダイレクトされます。 |

□6. Azureポータルの左上にある**Home**をクリックします。

□7. Azure servicesの下にあるAzure Database for PostgreSQL serversをクリックします。

□8. **sg000000**をクリックします。  
これは、ハイパースケール (Citus) サーバーグループを管理できるAzure Portalの概要ブレードです。この概要タブには、サーバーグループへの接続に使用するコーディネーター名が右上に表示されます。

□9. 左にある**Connection Strings**をクリックし、接続文字列の書式の数字を表示します。

□10. 左にある**Configure**をクリックし、デプロイメントの設定を表示します。

□11. このウインドウの右下にある**Next**をクリックします。

# ハイパースケール (Citus) の使用を開始する

Azureポータルのクラウドシェルを使用してハイパースケール (Citus) サーバーグループに接続するには、ストレージアカウントを作成する必要があります。ストレージアカウントを使用すると、クラウドシェルに関連付けられたファイルを保存できるため、スクリプトの実行、データファイルのダウンロード、Azureリソースの管理など、さまざまなAzureポータルアクティビティで使用できます。

# クラウドシェルを作成する

□1. ポータルのバナーでクラウドシェルのアイコンをクリックします。

□2. Welcome to Azure Cloud Shellで**Bash**をクリックします。

□3. You have no storage mountedの画面で、Show advanced settingsをクリックします。

□4. サブスクリプションとリージョンのデフォルト値を使います。

□5. リソースグループは既存の**rg000000**を使うようにしてください。

□6. ストレージアカウントには、Create newを選択し、**sg000000shell**をペーストします。

□7. ファイルシェアには、Create newを選択し、**sg000000shell**を入力してください。

□8. Create Storageをクリックします。

|  |
| --- |
| 注: クラウドシェルを作成・開始するのに1分程度を要します。 |

□9. 次の手順でファイアウォールを構成するには、クラウドシェルのクライアントIPアドレスが必要です。コマンドプロンプトで次のコマンドを入力し、returnキーを押してから、クラウドシェルのIPアドレスをコピーまたはメモします。

|  |
| --- |
| curl -s <https://ifconfig.co> |

|  |
| --- |
| 注: bashコンソールでペーストするには右クリック後にpasteを選択します。 |

□11. このウインドウの右下にある**Next**をクリックします。

# ハイパースケール (Citus) の利用を開始する

Azure Database for PostgreSQLのハイパースケール (Citus) は、サーバーレベルでファイアウォールを使用します。既定では、ファイアウォールはすべての外部アプリケーションとツールがコーディネーターノードおよび内部のデータベースに接続するのを防ぎます。特定のIPアドレス範囲のファイアウォールを開くルールを追加する必要があります。

右上のOverviewペインには、接続先のクラスターのコーディネーターホスト名のアドレスが表示されます。

# サーバーレベルのファイアウォールのルールを設定する

□1. Securityの下のOverviewペインの左のナビゲーションで**Firewall**をクリックします。

□2. クラウドシェルで確認した**IPアドレス**を**START IP**と**END IP**に入力します。

□3. FIREWALL RULE NAMEにCloudShellと入力します。

□4. ペインの左上にある**Save**をクリックします。

|  |
| --- |
| 注: ハイパースケール(Citus) サーバーはポート5432を介して通信します。企業ネットワーク内から接続しようとしている場合、ポート5432を超える送信トラフィックは、ネットワークのファイアウォールで許可されない場合があります。その場合は、IT部門がポート5432を開かない限り、ハイパースケール (Citus) サーバーに接続できません。 |

# Azure Database for PostgreSQLのハイパースケール (Citus)に接続する

ハイパースケール(Citus)を作成すると、**citus**という名前の既定のデータベースが作成されます。データベースサーバーに接続するには、接続文字列と管理者パスワードが必要です。最初の接続には最大2分かかる場合があります。何らかの理由でシェルがタイムアウトして再起動した場合は、curl -s https://ifconfig.coコマンドをもう一度実行し、ファイアウォールが新しいIPアドレスで更新されていることを確認する必要があります。

# Psqlでデータベースに接続する

□1. クラウドシェルの右上にある**最大化**ボックスをクリックして全画面にします。

□2. Bashプロンプトで、Psqlユーティリティを用いてAzure Database for PostgreSQLに接続します。最初の接続には最大2分かかる場合があります。以下のコマンドをコピー＆ペーストして[enter]を押します。

|  |
| --- |
| psql "host=sg000000-c.postgres.database.azure.com port=5432 dbname=citus user=citus password='xxxxxxxx' sslmode=require" |

# テーブルを作成しスケールアウトする

Psqlを使用してハイパースケール (Citus) コーディネーターノードに接続すると、いくつかの基本的なタスクを完了できます。

この経験では、主に分散テーブルとそれらに慣れることに焦点を当てます。これから作業するデータモデルは単純です：GitHubのユーザーデータとイベントデータ。イベントには、フォークの作成、組織に関連するgitコミットなどが含まれます。Psql経由で接続したら、テーブルを作成してみましょう。

□3. Psqlコンソールで以下をコピー＆ペーストしてテーブルを作成します。

|  |
| --- |
| CREATE TABLE github\_events  (  event\_id bigint,  event\_type text,  event\_public boolean,  repo\_id bigint,  payload jsonb,  repo jsonb,  user\_id bigint,  org jsonb,  created\_at timestamp  );  CREATE TABLE github\_users  (  user\_id bigint,  url text,  login text,  avatar\_url text,  gravatar\_id text,  display\_login text  ); |

github\_eventsのペイロードフィールドには、JSONBデータ型があります。JSONBはPostgresのバイナリ形式のJSONデータ型です。データ型を使用すると、柔軟なスキーマを1つの列に簡単に格納できます。Postgresは、この型にGINインデックス（Generalized Inverted Index、汎用転置インデックス）を作成し、その中のすべてのキーと値にインデックスを付けることができます。インデックスを使用すると、さまざまな条件でペイロードを高速かつ簡単に照会できます。データを読み込む前に、いくつかのインデックスを作成してみましょう。

□4. Psqlコンソールで以下をコピー＆ペーストしてインデックスを作成します。

|  |
| --- |
| CREATE INDEX event\_type\_index ON github\_events (event\_type);  CREATE INDEX payload\_index ON github\_events USING GIN (payload jsonb\_path\_ops); |

次に、コーディネーターノード上のPostgresテーブルを指定し、ハイパースケール (Citus) にワーカー全体でシャードするように伝えます。そのために、それをシャードするキーを指定する各テーブルに対してクエリを実行します。この例では、user\_idのイベントテーブルとユーザーテーブルの両方をシャードします。

□5. Psqlコンソールで以下をコピー＆ペーストします。

|  |
| --- |
| SELECT create\_distributed\_table('github\_events', 'user\_id');  SELECT create\_distributed\_table('github\_users', 'user\_id'); |

テーブルごとに、このコマンドはワーカーノードにシャードを作成します。各シャードは、一連のユーザーを保持する単純なpostgresqlテーブルです（user\_idでシャード化したので）。また、コーディネーターノードにメタデータを作成して、分散テーブルのセットとワーカーノードのシャードの局所性を追跡します。user\_idで両方のテーブルをシャードしたので、テーブルは自動的にコロケーションされます。つまり、両方のテーブルの1つのuser\_idに関連するすべてのデータが同じワーカーノード上にあります。これは、コロケーションされたシャード全体で、ワーカーノード上での２つのテーブル間の結合をローカルに実行する場合に役立ちます。

|  |
| --- |
| 注: ハイパースケール (Citus) サーバー内には、3種類のテーブルがあります。   * **分散テーブル** – ワーカーノードを跨いで分散（スケールアウト）。一般的に大きなテーブルはパフォーマンスを改善するために分散したテーブルであるべきです。 * **参照テーブル** – 全てのノードに複製されます。分散テーブルとの結合を可能にします。典型的には国や製品カテゴリのような小さなテーブルに用いられます。 * **ローカルテーブル** – コーディネーターノードに置かれるテーブルで、管理テーブルがローカルテーブルの典型例です。 |

データをロードする準備が整いました。以下のコマンドでBashのクラウドシェルを「シェル実行」し、ファイルをダウンロードします。

□6. Psqlコンソールでデータファイルをダウンロードするために以下をコピー＆ペーストとします。

|  |
| --- |
| \! curl -O https://examples.citusdata.com/users.csv  \! curl -O https://examples.citusdata.com/events.csv |

□7. Psqlコンソールでデータファイルをロードするために以下をコピー＆ペーストします。

|  |
| --- |
| \copy github\_events from 'events.csv' WITH CSV  \copy github\_users from 'users.csv' WITH CSV |

重い本番ワークロードの場合、COPYコマンドが単一ノードのPostgresよりもハイパースケール (Citus) で高速な理由は、COPYがファンアウトされワーカーノード間で並行して実行されることによります。

# クエリの実行

ここから、実際にいくつかのクエリを実行するので、楽しい時間です。簡単なカウント(\*)から始めて、読み込んだデータの量を確認します。

□8. Psqlコンソールでgithub\_eventsのレコードカウントを取得するために以下をコピー＆ペーストします。

|  |
| --- |
| SELECT count(\*) from github\_events; |

この単純なクエリは、先ほど作成されたシャードキー**user\_id**に基づいてコントローラーがすべてのワーカーに対してリファクタリングし、集計したレコード数が返されました。

JSONBペイロード列には、多くのデータがありますが、イベントの種類によって異なります。PushEventイベントには、プッシュの個別のコミットの数を含むサイズが含まれています。これを使用して、1時間あたりのコミットの合計数を検索できます。

□9. Psqlコンソールで時間あたりのコミット数を見るために以下をコピー＆ペーストします。

|  |
| --- |
| SELECT date\_trunc('hour', created\_at) AS hour,  sum((payload->>'distinct\_size')::int) AS num\_commits  FROM github\_events  WHERE event\_type = 'PushEvent'  GROUP BY hour  ORDER BY hour; |

|  |
| --- |
| 注: 結果ビューでスタックした場合は、[q]と入力し、[Enter]を押してビューモードを終了します。 |

これまでのところ、クエリにはgithub\_eventsだけが関係していましたが、この情報をgithub\_usersと組み合わせることができます。ユーザーとイベントの両方を同じ識別子（**user\_id**)でシャードしたので、一致するユーザーIDを持つ両方のテーブルの行は同じデータベースノードと同じ位置に配置され、簡単に結合できます。user\_idでクエリを結合すると、ハイパースケール (Citus) コントローラーは、ワーカーノードで並行して実行するために結合実行をシャードに押し下げるでしょう。

□10. Psqlコンソールでレポジトリ数が最大のユーザを見つけるために以下をコピー＆ペーストします。

|  |
| --- |
| SELECT login, count(\*)  FROM github\_events ge  JOIN github\_users gu  ON ge.user\_id = gu.user\_id  WHERE event\_type = 'CreateEvent' AND  payload @> '{"ref\_type": "repository"}'  GROUP BY login  ORDER BY count(\*) DESC  LIMIT 20; |

本番ワークロードでは、次の理由により、上記のクエリはハイパースケール (Citus) 上では高速です。

* シャードが小さく、インデックスも小さい。これはリソースの利用効率の向上とインデックス/キャッシュのヒット率の向上に寄与します。
* 複数のワーカーノードによる並列実行

□11. このウインドウの右下にある**Next**をクリックします。

# 結論

このラボではデータベースクラスターを展開しシャーディングキーを設定することで、Microsoft Azure上でPostgresを水平にスケールする方法を学びました。

他にも特にここで学んだこととして以下がありました。

* Azure Database for PostgreSQLにハイパースケール (Citus) を展開する方法
* Azureクラウドシェルの作り方
* Psqlを用いたハイパースケール (Citus) への接続方法
* スキーマの作成方法、シャーディングキーの設定方法、サーバーグループへのデータのロード方法

Azure Database for PostgreSQLのハイパースケール (Citus) を使用すると、Postgresデータベースクラスター（「サーバーグループ」と呼ばれる）にデータとクエリを分散できるため、サーバーグループ内のすべてのノードで、すべてのメモリ、コンピューティング、およびディスクが利用できるというパフォーマンス上の利点をアプリケーションに提供できます。

自分のサブスクリプションでハイパースケール (Citus) を試してみたい場合は、以下のリンクを参照してください。

* Quickstart create Hyperscale (Citus) (https://docs.microsoft.com/en-us/azure/postgresql/quickstart-create- hyperscale-portal)

今日は時間をいただきありがとうございます。またこの経験を完了されおめでとうございます。