

Azure Database for PostgreSQL のハイパースケール (Citius) 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 (Citius) 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。

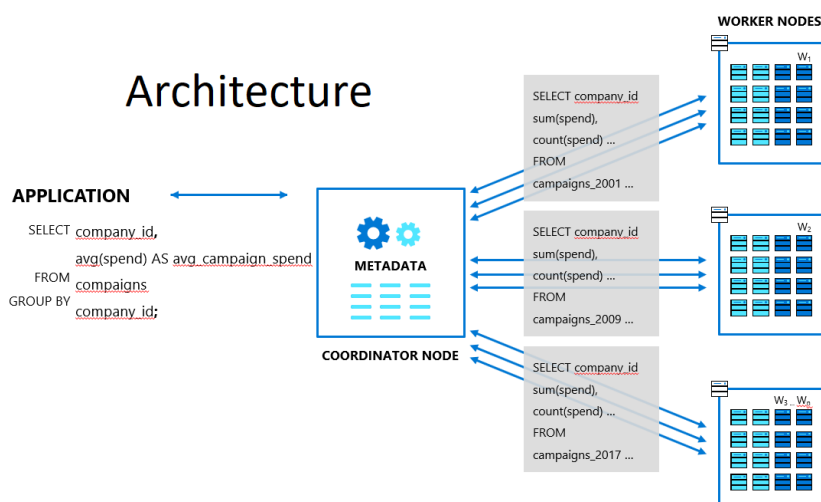
ハイパースケール (Citius) の概要

Azure Database for PostgreSQL のハイパースケール (Citius) 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ーに、データ、およびクエリを配布（シャーディング）します。ハイパースケール (Citius) 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 (Citius) 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 (Citius) のアーキテクチャ:

クラスターには、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 (Citius) 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。



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 (Citrus) 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、ブラウザのウィ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 userxxxxx@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. このウィ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 (Citius) 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 (Citius) サーバークラスタ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 (Citius) 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 (Citius) サーバークラスタ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 (Citius) 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 **citius** という値が必須です。
 - Password: **xxxxxxx** 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 (Citius) 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ルアップできます。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 (Citrus) 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. このウィ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 (Citrus) 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 (Citrus) サーバ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. このウィ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 (Citrus) の利用を開始する

Azure Database for PostgreSQL のハイパースケール (Citrus) 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(Citrus) 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 (Citrus) 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. PsqI 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（Citius）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. PsqI 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＆ペーストとします。

```
¥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＆ペーストとします。

```
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 (Citius) 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 (Citius) 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。