

▶ PostgreSQL 用 Amazon Aurora

▶ すべての研究室の前提条件

ラボ1:新しいAuroraクラスタを 手動で作成する

▶ ラボ1.5: Cloud9の設定とデータ ベースの初期化

ラボ2: 高速クローニング

ラボ3: クエリ・プランの管理

▶ ラボ4: クラスターキャッシュの 管理

▶ ラボ5: データベース・アクティ ビティ・ストリーミング

ラボ6: RDSパフォーマンスイン サイト

#### ラボ7: データセットの作成とオ ートスケール

- ▶ ラボ8: フォールトトレランスの テスト
- ▶ ラボ9: オーロラ・グローバル・ データベース
- ▶ ラボ10: Auroraサーバーレスv1
- ▶ ラボ11:オーロラ機械学習
- ▶ ラボ12:グラビトン2とx86の比較
- ▶ ラボ13: Auroraサーバーレスv2
- ▶ 研究室14: Aurora PostgreSQLのための信頼できる言語拡張
- ▶ PostgreSQLアドバンスラボ

クリーンアップ

貢献者と改訂履歴

その他のリソース

PostgreSQL 用 Amazon Aurora Labs 〉 ラボ7:データセットの作成とオートスケール

## ラボ7:データセットの作成とオートスケー ル

Auroraオートスケーリングは、プロビジョニングされたAurora DBクラスタのAuroraレプリカの数を動的に調整することで、接続性やワークロードの突然の増加に対応できるようにします。接続性やワークロードが減少すると、Aurora Auto Scalingは不要なAurora Replicasを削除します。

このラボでは、ロードジェネレータスクリプトを使用して、Aurora リードレプリカ自動スケーリングが実際にどのように機能するかを説明します。

このラボには以下のタスクが含まれています:

- 1. オーロラレプリカの自動スケーリングを設定する
- 2. pgbenchの初期化とデータセットの作成
- 3. 読み取り専用のワークロードを実行する

## 前提条件

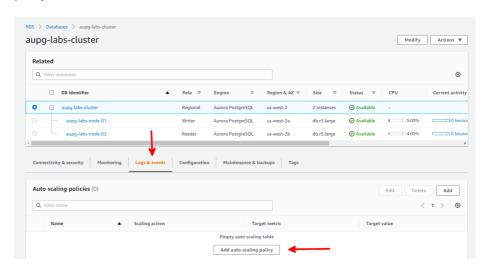
このラボでは、以下のラボモジュールを最初に完了する必要があります。

- すべての研究室の前提条件
- ラボ1:新しいAuroraクラスタの作成(「**すべてのラボの前提条件**」セクションで Aurora PostgreSQLクラスタを使用しない素の最小ラボ環境に従った場合は必須。それ 以外は任意)
- Cloud9を設定し、データベースを初期化する(「**すべてのラボの前提条件**」セクションでAurora PostgreSQLクラスタを使用しないベア最小ラボ環境に従った場合は必須。そうでない場合はスキップしてください)。

## 1.レプリカ自動スケーリングポリシーを作成する

DBクラスタにリードレプリカ自動スケーリング構成を追加します。これにより、DBクラスターは任意の時点でDBクラスターで動作するリーダーDBインスタンスの数を負荷に基づいてスケーリングできるようになります。

Auroraクラスタ名をクリックし、[Logs & events]タブに移動します。Add auto scaling policyボタンをクリックします。



© 2008 - 2024, Amazon Web Services, Inc.またはその関連会社。無断複写・転載を禁じます。 プライバシーポリシー 利用規約 クッキーの設定

▶ PostgreSQL 用 Amazon Aurora Labs

▶ すべての研究室の前提条件

ラボ1:新しいAuroraクラスタを 手動で作成する

▶ ラボ1.5: Cloud9の設定とデータ ベースの初期化

ラボ2: 高速クローニング

ラボ3: クエリ・プランの管理

▶ ラボ4: クラスターキャッシュの 管理

▶ ラボ5: データベース・アクティ ビティ・ストリーミング

ラボ6: RDSパフォーマンスイン サイト

#### ラボ7: データセットの作成とオ ートスケール

▶ ラボ8: フォールトトレランスの テスト

▶ ラボ9: オーロラ・グローバル・ データベース

▶ ラボ10: Auroraサーバーレスv1

▶ ラボ11: オーロラ機械学習

▶ ラボ12:グラビトン2とx86の比較

▶ ラボ13: Auroraサーバーレスv2

▶ 研究室14: Aurora PostgreSQLのための信頼できる言語拡張

▶ PostgreSQLアドバンスラボ

クリーンアップ

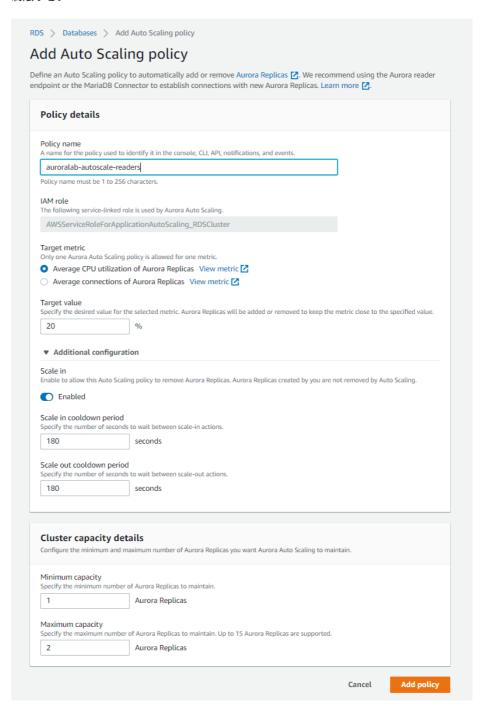
貢献者と改訂履歴

その他のリソース

**Policy Name** にauroralab-autoscale-readers を入力します。**Target メトリックには、Aurora レプリカの平均 CPU 使用**率を選択します。**Target 値に** 20% を入力します。本番環境で使用する場合、この値はもっと高く設定する必要があるかもしれませんが、ここではデモのために低い値を使用しています。

次に、[追加設定]セクションを展開し、[スケールインのクールダウン期間]と[スケールアウトのクールダウン期間]の両方を180秒に変更します。これにより、後続のラボでのスケーリング操作間の待機時間が短縮されます。

**クラスタ容量の詳細**」セクションで、「最小容量」を「1」に、「最大容量」を「2」に設定する。本番使用ケースでは異なる値を使用する必要があるかもしれないが、デモンストレーションを目的とし、ラボに関連するコストを制限するため、リーダーの数を2つに制限する。



次にポリシーの追加をクリックします。

## 2.pgbenchの初期化とデータセットの作成

Cloud9ターミナルウィンドウを開くセクションを参照してCloud9ターミナルウィンドウを開き、ターミナルウィンドウに以下のコマンドを貼り付けてpgbenchを初期化し、デー

- ▶ PostgreSQL 用 Amazon Aurora
- ▶ すべての研究室の前提条件

ラボ1:新しいAuroraクラスタを 手動で作成する

▶ ラボ1.5: Cloud9の設定とデータ ベースの初期化

ラボ2:高速クローニング

ラボ3: クエリ・プランの管理

- ▶ ラボ4: クラスターキャッシュの 管理
- ▶ ラボ5: データベース・アクティ ビティ・ストリーミング

ラボ6: RDSパフォーマンスイン サイト

## ラボ7: データセットの作成とオ ートスケール

- ▶ ラボ8: フォールトトレランスの テスト
- ▶ ラボ9: オーロラ・グローバル・ データベース
- ▶ ラボ10: Auroraサーバーレスv1
- ▶ ラボ11:オーロラ機械学習
- ▶ ラボ12: グラビトン2とx86の比較
- ▶ ラボ13: Auroraサーバーレスv2
- ▶ 研究室14: Aurora PostgreSQLのための信頼できる言語拡張
- ▶ PostgreSQLアドバンスラボ

クリーンアップ

貢献者と改訂履歴

その他のリソース

タセットの作成を開始します。

```
pgbench -i--scale=1000
```

データの読み込みには数分かかることがありますが、完了すると同様の出力が表示されます:

```
古いテーブルの削除...NOTICE: テーブル "pgbench_accounts "が 存在しない、スキップNOTICE:
200000 of 100000000 tuples (0%) done (経過 0.16 s, 残り 77.78 s)
300000 of 100000000 tuples (0%) done (経過 0.23 s, 残り 76.16 s)
400000 of 100000000 tuples (0%) done (経過 0.33 s, 残り 82.14 s)
500000 of 100000000 tuples (0%) done (経過 0.46 s, 残り 91.65 s)
600000 of 100000000 tuples (0%) done (elapsed 0.56 s, remaining 92.33 s)
700000 of 100000000 tuples (0%) done (elapsed 0.66 s, remaining 94.29 s)
800000 of 100000000 tuples (0%) done (elapsed 0.77 s, remaining 94.92 s).77 s, 残り 94.
100000000個の タプルの うち 900000 個 (0%) don e (elapse d 0.87 s, 残り 95.93 s)
100000000個の タプルの うち 1000000 個(1%) done (elapsed 0.95 s,残り 94.27s) 100000000€
100000000個の タプルの うちの 1100000個(1%) done (経過 1.04 s, 残り 93.48 s)
100000000個の タプルの うちの 1200000個(1%) done (経過 1.14 s, 残り 93.96 s)
1000000000 個の タプルの うち1300000個(1%) done (elapsed 1.24 s, remaining 94.43 s)
1000000000 個の タプルの うち 1400000個(1%) done (elapsed 1.35 s, remaining 95.05 s)
100000000個の タプルの うち 1500000個(1%) done (elapsed 1.43 s, remaining 94.22 s).43 s
... 98400000 of 100000000 tuples (98%) done (elapsed 119.10 s, 残り 1.94 s)
98500000 of 100000000 tuples (98%) done (elapsed 119.23 s, 残り 1.82 s)
98600000 of 100000000 tuples (98%) done (経過 119.34 s, 残り 1.69 s)
98700000 of 100000000 tuples (98%) done (経過 119.49 s, 残り 1.57 s)
98800000 of 1000000 00 tupl e s (98%) done (elapsed 119.60 s, remaining 1.45 s)
98900000 of 100000000 tuples (98%) done (elapsed 119.72 s, remaining 1. 33 s) 987000
100000000個の タプルの うちの 99000000個(99%) done (elapsed 119.82 s, remaining 1.21 s)
100000000個の タプルの うちの 99100000 個(99%) done (elapsed 119.94 s, remaining 1.09 s
1000000000個の タプルの うちの 99200000 個(99%) done (elapsed 120.05 s, 残り 0.97 s)
100000000個の タプルの うち 99300000 個 (99%) done (経過 120.12 s, 残り 0.85 s)
100000000個の タプルの うち 99400000 個 (99%) done (経過 120.21 s, 残り 0.73 s)
99500000 個の 100000000 タプル (99%) done (経過 120.32 s, 残り 0.60 s)
99600000 個の 100000000 タプル (99%) done (経過 120.41 s, 残り 0.48 s)
100000000 個の タプルの 99700000 個(99%) done (elapsed 120.55 s, remain i ng 0.36 s)
 100000000 個の タプルの 99800000 個(99%) done (elapsed 120.68 s, remaining 0.24 s) 3
100000000個の タプルの うち 99900000個(99%) done (elapsed 120.80 s, remaining 0.12 s)
100000000個の タプルの うち 10000000個(100%) done (elapsed 120.89 s, remaining 0.00 s)
vacuuming...creating primary keys..done.
```

## 3.読み取り専用ワークロードの実行

データロードが正常に完了したら、クラスタ上で読み取り専用のワークロードを実行できます(自動スケーリングポリシーをトリガできるようにします)。DBクラスタ・トポロジーへの影響も観察します。

このステップでは、クラスタのリーダーエンドポイントを使用します。DBクラスタが自動的に作成された場合、リーダエンドポイントはCloudFormationスタックのOutputs に"Amazon Aurora PostgreSQL Labs Stackset "と記述されています。 readerEndpointという Outputキーを参照してください。手動でクラスタを作成した場合は、RDS Console - Databases セクションでに行き、Aurora クラスタの名前をクリックし、Connectivity & securityタブに行くことでリーダーエンドポイントを見つけることができます。

Cloud9ターミナルウィンドウからロード生成スクリプトを実行し、[readerEndpoint]プレースホルダを実際のAuroraクラスタリーダエンドポイントに置き換えます:

```
pgbench -h [readerEndpoint] -c 100 --select-only -T 600 -C
```

- ▶ PostgreSQL 用 Amazon Aurora Lahs
- ▶ すべての研究室の前提条件

ラボ1:新しいAuroraクラスタを 手動で作成する

▶ ラボ1.5: Cloud9の設定とデータ ベースの初期化

ラボ2: 高速クローニング

ラボ3: クエリ・プランの管理

- ▶ ラボ4: クラスターキャッシュの 管理
- ▶ ラボ5: データベース・アクティ ビティ・ストリーミング

ラボ6: RDSパフォーマンスイン サイト

## ラボ7:データセットの作成とオ ートスケール

- ▶ ラボ8: フォールトトレランスの テスト
- ▶ ラボ9: オーロラ・グローバル・ データベース
- ▶ ラボ10: Auroraサーバーレスv1
- ▶ ラボ11:オーロラ機械学習
- ▶ ラボ12:グラビトン2とx86の比較
- ▶ ラボ13: Auroraサーバーレスv2
- ▶ 研究室14: Aurora PostgreSQLのための信頼できる言語拡張
- ▶ PostgreSQLアドバンスラボ

クリーンアップ

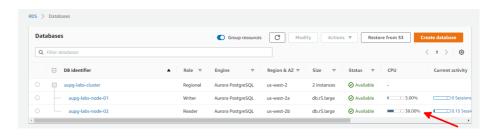
貢献者と改訂履歴

その他のリソース

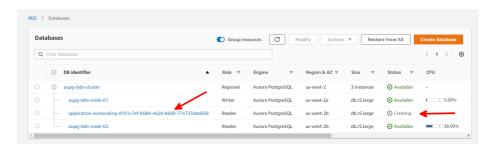
# TeamRole: -/environment \$ ggbench -h aupg-labs-cluster.cluster-ro-ctuoyfdmedaj.us-west-2.rds.amazonaws.com -c 100 --select-only -T 600 -C starting vacuum...ERROR: cannot execute VACUUM during recovery (ignoring this error and continuing anyway) ERROR: cannot execute VACUUM during recovery (ignoring this error and continuing anyway) ERROR: cannot execute VACUUM during recovery (ignoring this error and continuing anyway) ERROR: cannot execute RUNLATE TABLE in a read-only transaction (ignoring this error and continuing anyway) end.

次に、別のブラウザタブでAmazon RDS管理 コンソールと
と
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
<l>に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に</l>
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に</l>
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に
に</l>
に
に
に
に
に
に</l

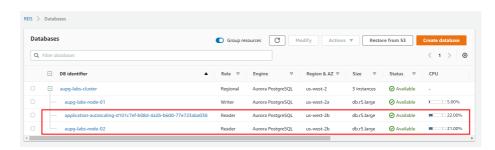
リーダー・ノードは現在負荷を受けていることに注意してください。メトリクスが受信負荷を完全に反映するまで、1分以上かかる場合があります。



数分後、インスタンスのリストに戻り、クラスタに新しいリーダーがプロビジョニングされていることに気づく。



新しいレプリカを追加するには5~7分かかります。新しいレプリカが利用可能になると、 負荷が分散して安定することに注意してください(安定するまで数分かかる場合があります)。



これでCloud9ターミナル・ウィンドウに戻り、CTRL+Cを押して実行中のpgbenchジョブを終了できます。しばらくすると、追加のリーダーは自動的に削除されます。

前へ

次のページ