CCM と QPM を使用して Amazon Aurora PostgreSQL のリカバリのパフォーマンスと実行プランを最適化してください

AWS の規範的ガイダンス

AWS の規範的ガイダンス: CCM と QPM を使用して Amazon Aurora PostgreSQL のリカバリのパフォーマンスと実行プランを最適化してください

Table of Contents

CCM と QPM を使用して Amazon Aurora PostgreSQL のリカバリのパフォーマンスと実行プランを最適化してください

Amazon Web Services ラウナック・リシャブ、ロヒト・カプール、スジタ・サシクマラン (AWS)

2023 年 1 月 (ドキュメント履歴)

ビジネスが拡大するにつれて、重要な意思決定を行うためにますます多くのデータを使用します。データ量が増えるにつれ、データベースのパフォーマンスを最適化し、システム変更中も安定させることが重要です。パフォーマンスが低いと顧客満足度やビジネス収益に影響する可能性があるため、金融取引や顧客注文を伴うワークロードなど、トランザクションの多いワークロードには、安定した、一貫性のある高速なパフォーマンスが必要です。Amazon Aurora PostgreSQL 互換のデータベースインスタンスなど、トランザクションの多いワークロードを処理するデータベースでは、利用可能なパフォーマンス最適化機能を理解して実装することが重要です。

Amazon Aurora PostgreSQL と互換性があるのは、PostgreSQL デプロイメントのセットアップ、運用、スケーリングに役立つフルマネージド型のリレーショナルデータベースエンジンです。自立型のストレージアーキテクチャとその機能により、メンテナンスオーバーヘッドを最小限に抑えながら実際のワークロードシナリオでパフォーマンスを最適化できるため、広く使用されているデータベースエンジンです。

これらの機能のうち 2 つは、クラスターキャッシュ管理 (CCM) とクエリプラン管理 (QPM) です。CCM により、フェイルオーバー時にアプリケーションとデータベースのパフォーマンスを回復できます。この機能により、オプティマイザによって生成されたクエリ実行計画を SQL アプリケーションで管理できます。これらの機能はどちらも、データベースをより細かく制御できるため、SQL クエリのパフォーマンスを最適化するのに役立ちます。このガイドは、管理者、プロダクトオーナー、データベースアーキテクト (DBA) が CCM と QPM を導入することの利点と潜在的なビジネス成果を理解するのに役立つことを目的としています。

対象者

このガイドの対象者は、Amazon Aurora PostgreSQL 互換データベースインスタンスのパフォーマンスを最適化するために利用できる機能を理解し、それらの機能の使用例を理解したいビジネス関係者を対象としています。

ターゲットを絞ったビジネス成果

このガイドを参考に、クラスターキャッシュ管理 (CCM) で次のようなビジネス上の成果を達成する。

- フェイルオーバーが発生した場合は、迅速に回復して、安定した最適なワークロードパフォーマンスを維持できます。
- フェイルオーバー後のワークロードパフォーマンスの低下によるビジネス損失を減らします。
- フェイルオーバー後の不要な I/O コストを防ぐのに役立ちます。

このガイドを使用すると、クエリプラン管理 (QPM) で次のようなビジネス成果を達成できます。

- オプティマイザに、承認された少数の計画から強制的に的に的に的に選択させることで、計画の安定性を改善する。これにより、システムまたはデータベースが変更された後に、オプティマイザが特定の SQL ステートメントに対して最適ではない計画を選択したときに発生します。
- 計画を一元元元元元元元的に元的に元的に元的に元的に元的に元
- 使用されていないインデックスを特定し、インデックスの追加または削除の影響を評価します。
- オプティマイザーが特定した新しい最小コストプランを自動的に認識します。
- パフォーマンスを改善する計画変更のみが承認されるように決定し、新しいオプティマイザの機能を少ないリスクでテストする。

クラスターキャッシュ管理

キャッシュは、ディスク I/O の削減に役立つため、データベース (DB) の最も重要な機能の 1 つです。最も頻繁にアクセスされるデータは、バッファキャッシュと呼ばれるメモリ領域に格納されます。クエリが頻繁に実行されると、ディスクではなくキャッシュから直接データが取得されます。これはより高速で、スケーラビリティとアプリケーションパフォーマンスが向上します。PostgreSQL のキャッシュサイズは、`shared_buffers`パラメーターを使用して設定します。詳細については、「メモリ (PostgreSQL ドキュメント)」を参照してください。

フェイルオーバー後、Amazon Aurora PostgreSQL 互換エディションのクラスターキャッシュ管理 (CCM) は、アプリケーションとデータベースの復旧パフォーマンスを向上させるように設計されています。CCM を使用しない一般的なフェイルオーバーが発生した場合、一時的だがパフォーマンスが大幅に低下することがあります。これは、フェイルオーバー DB インスタンスの起動時にバッファキャッシュが空になることが原因で発生します。空のキャッシュは、コールドキャッシュとも呼ばれます。DB インスタンスはディスクから読み取る必要がありますが、これはキャッシュからの読み取りよりも時間がかかります。

CCM を実装する場合、優先するリーダー DB インスタンスを選択すると、CCM はそのキャッシュメモリをプライマリ (ライター) DB インスタンスのキャッシュメモリと継続的に同期します。フェイルオーバーが発生すると、優先リーダー DB インスタンスが新しいライター DB インスタンスに昇格します。ウォームキャッシュと呼ばれるキャッシュメモリが既に搭載されているため、フェイルオーバーがアプリケーションのパフォーマンスに与える影響を最小限に抑えることができます。

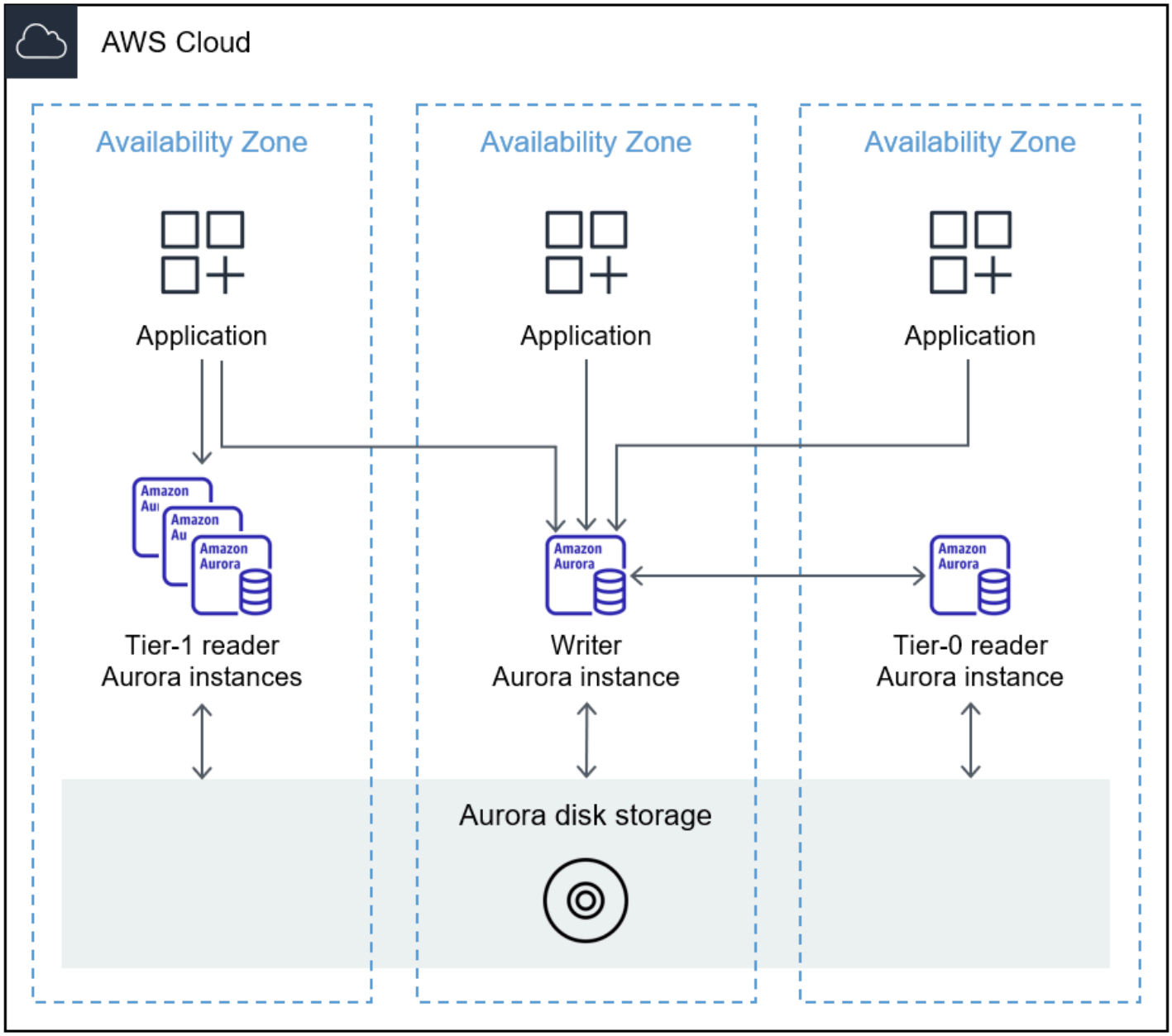
クラスターキャッシュ管理はどのように機能しますか?

フェイルオーバー DB インスタンスは、プライマリのライター DB インスタンスとは異なるアベイラビリティーゾーンにあります。優先リーダー DB インスタンスは優先フェイルオーバーターゲットで、Tier-0 の優先度レベルを割り当てることで指定されます。

> **Note**
>
> 昇格階層の優先度は、Aurora リーダーが失敗後に書き込み DB インスタンスに昇格する順序を指定する値です。有効な値は 0 ~ 15 です。ここで、0 は優先度が最も高く、15 が最も低いことを表します。昇格階層の詳細については、「Aurora DB クラスターの耐障害性」を参照してください。昇格階層を変更しても、停止は発生しません。

CCM は、ライター DB インスタンスのキャッシュを優先リーダー DB インスタンスと同期します。リーダー DB インスタンスは、現在キャッシュされているバッファアドレスのセットをブルームフィルターとしてライター DB インスタンスに送信します。ブルームフィルターは、要素がセットのメンバーかどうかをテストするために使用される、確率的でメモリ効率の高いデータ構造です。ブルームフィルターを使用すると、リーダー DB インスタンスがライター DB インスタンスに同じバッファアドレスを繰り返し送信することを防ぎます。ライター DB インスタンスはブルームフィルターを受け取ると、バッファーキャッシュ内のブロックを比較し、頻繁に使用されるバッファーをリーダー DB インスタンスに送信します。デフォルトでは、バッファの使用回数が3を超えると、そのバッファは頻繁に使用されると見なされます。

次の図は、CCM がライター DB インスタンスのバッファキャッシュを優先リーダー DB インスタンスと同期する方法を示しています。



CCM の詳細については、「Aurora PostgreSQL のクラスターキャッシュ管理によるフェイルオーバー後の高速リカバリ」(Aurora ドキュメント) と「Aurora PostgreSQL クラスターキャッシュ管理の概要」(AWSブログ記事) を参照してください。CCM の設定方法については、「クラスターキャッシュ管理の設定」(Aurora ドキュメント) を参照してください。

制約事項

CCM 機能には次の制約があります。

- リーダー DB インスタンスは、ライター DB インスタンスと同じ DB インスタンスクラスタイプとサイズ (`r5.2xlarge`またはなど) でなければなりません`db.r5.xlarge`。
- CCM は、Aurora Global Database の一部である Aurora PostgreSQL DB クラスターではサポートされません。

クラスターキャッシュ管理のユースケース

小売、銀行、金融などの一部の業界では、わずか数ミリ秒の遅延がアプリケーションのパフォーマンスの問題を引き起こし、ビジネスに大きな損失をもたらす可能性があります。CCM は、プライマリデータベースインスタンスのバッファキャッシュを優先バックアップインスタンスに継続的に同期させることで、アプリケーションとデータベースのパフォーマンスを回復させるのに役立つため、フェイルオーバーによるビジネス損失を防ぐことができます。

クエリプラン管理

統計情報、制限事項、環境設定、クエリパラメータのバインディングの変更、PostgreSQL データベースエンジンのアップグレードは、すべてクエリ計画のリグレッションの原因になる可能性があります。クエリ計画のリグレッションとは、オプティマイザが、データベース環境への特定の変更前よりも最適でない計画を選択したことです。

Amazon Aurora PostgreSQL Compatible Edition では、クエリプラン管理 (QPM) 機能は、クエリプランのリグレッションを引き起こす可能性のあるデータベース環境の変化に関係なく、プランの適応性と安定性を確保するように設計されています。QPM はオプティマイザーをある程度制御します。QPM を使用すると、オプティマイザによって生成されたクエリ計画を SQL クエリ管理できます。クエリ実行プランでは、オプティマイザーは重要なクエリについて承認済みのプランから選択して、パフォーマンスを最適化する必要があります。

企業は通常、アプリケーションやデータベースをグローバルに展開したり、開発、QA、ステージング、試作、テスト、本番環境など、アプリケーションデータベースごとに複数の環境を維持したりします。各データベース、各環境、AWS リージョンおよびすべてのデータベースのクエリ実行計画を維持することは、複雑で時間がかかる場合があります。QPM では、Amazon Aurora PostgreSQL と互換性のあるマネージドプランをあるデータベースから別のデータベースにエクスポートおよびインポートできます。これにより、クエリ実行計画を一元的に管理し、データベースをグローバルに展開できます。この機能を使用して、試作データベースにある一連の計画を調査し、それらが適切に機能することを確認してから、本番環境に読み込むことができます。

QPM には他にもいくつかの利点があります。たとえば、アプリケーションでは変更できない実行プランや、ステートメントにヒントを追加できない場合に、QPM を使用して改善できます。また、QPM は、オプティマイザーが検出した新しい最小コストプランを自動的に検出するため、パフォーマンスだけでなくコストも引き続き最適化できます。

QPM を有効化することを推奨します。QPM を有効にすると、オプティマイザーは承認された最小コストプランを使用します。これにより、リグレッションを防ぎ、最適ではない計画の管理と修正に必要な時間を短縮できます。

QPM 機能の使用方法には、事前対応型と事後対応型の 2 種類があります。プロアクティブなアプローチは、パフォーマンスの低下を防ぐのに役立つように設計されており、事後対応的なアプローチは、パフォーマンスの低下が発生した後にそれを検出して修復するように設計されています。アプローチはクエリごとに選択できます。リグレッションが発生しやすい複雑なクエリやビジネスクリティカルなクエリには、プロアクティブなアプローチを採用して、それらのクエリに最適なプランを承認できます。実行中に他のクエリでクエリプランのリグレッションが発生した場合は、事後対

応型のアプローチを使用できます。リグレッションが検出されたら、rejectedオプティマイザーが別の承認済みプランを選択するように、そのプランのステータスをに変更します。詳細については、「Aurora PostgreSQL クエリプラン管理のベストプラクティス」(Aurora ドキュメント) を参照してください。

クエリプラン管理はどのように機能しますか?

プランには、、approvedunapprovedpreferred、またはのいずれかのステータスが割り当てられますrejected。オプティマイザーは、approved各管理ステートメントの最初に生成されたプランをに設定し、次にその他のプランのステータスをに設定しますunapproved。unapproved後でプランを評価し、ステータスをapprovedpreferred、またはに変更できますrejected。詳細については、「Aurora PostgreSQL クエリプラン管理について」(Aurora ドキュメント) を参照してください。

管理計画は、マニュアルまたは自動でキャプチャできます。最も一般的な方法は、2 回以上実行されるすべてのステートメントのプランを自動的にキャプチャすることです。ただし、特定のステートメントセットの計画を手動でキャプチャすることもできます。詳細については、「Aurora PostgreSQL 実行プランのキャプチャ」(Aurora ドキュメント) を参照してください。

管理計画を設定すると、オプティマイザでは、preferredapproved管理ステートメントごとに最小コストまたは計画が使用されるようになります。詳細については、オプティマイザが実行する計画を選択する方法 (Aurora ドキュメント) を参照してください。

Amazon Aurora PostgreSQL 互換で QPM 機能を設定する手順については、「Aurora PostgreSQL のクエリ実行プランの管理」(Aurora ドキュメント) を参照してください。

制約事項

QPM を使用するには、サポートされている SQL ステートメントの要件を満たしていること、ステートメントがシステムリレーションを参照していないこと、DB インスタンスクラスに十分な vCPUs があることを確認する必要があります。詳細については、「サポートされている SQL ステートメント」と「クエリプラン管理の制限」(Aurora ドキュメント) を参照してください。

クエリ計画管理のユースケース

- クエリプランのリグレッションの防止 — データベースのバージョンを最新の状態に保つことには、パフォーマンスとセキュリティの向上、新機能へのアクセス、既知の問題の修正、規制要件

への準拠など、多くのメリットがあります。ただし、データベースを更新すると、一部のクエリでパフォーマンスが低下するリスクがあります。メジャーバージョンのアップグレードでは、既存のアプリケーションクエリとの後方互換性のない変更が含まれる可能性があるため、このリスクは高くなります。QPM を実装すると、システム変更時にリグレッションを防ぎ、パフォーマンスを安定させるのに役立ちます。統計を更新したり、インデックスを追加したり、パラメータを変更したり、Amazon Aurora PostgreSQL と互換性のある新しいバージョンにアップグレードしたりすると、QPM は新しいプランを検出しますが、承認されたプランを引き続き使用するため、プランの安定性が維持されます。

- テスト機能 — すべてのマネージド SQL ステートメントのプラン履歴を表示し、PostgreSQL の新機能やプランの変更によってパフォーマンスが向上しているかどうかを評価できます。その後、それらの機能を実装するか、新しい計画を実装するかを決定できます。詳細については、dba_plans ビューで Aurora PostgreSQL クエリ計画を検証する (Aurora ドキュメント) を参照してください。
- 計画のリグレッション化、削除せずに、修正した方が良いこともあります。詳細については、「pg_hint_plan を使ったプランの修正」(Aurora ドキュメント) を参照してください。

リソース

AWSドキュメント

- クラスターキャッシュ管理 (CCM)
- クエリ計画管理 (QPM)

AWSブログ記事

- Aurora PostgreSQL CCM 入門
- Aurora PostgreSQL QPM 入門

AWSワークショップ

- PostgreSQL 向けAmazon Aurora ラボ:CCM
- PostgreSQL 向けAmazon Aurora ラボ:QPM

ドキュメント履歴

このガイドは、このドキュメントの大きな変更点をまとめたものです。今後の更新に関する通知を受け取る場合は、RSS フィードをサブスクライブできます。

変更	説明	日付
初回刊行物	—	2023 年 1 月 20 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。