[AWS Black Belt Online Seminar] Amazon Aurora with PostgreSQL Compatibility

Solutions Architect 江川 大地
2019/8/28

AWS 公式 Webinar
https://amzn.to/JPWebinar

過去資料
https://amzn.to/JPArchive

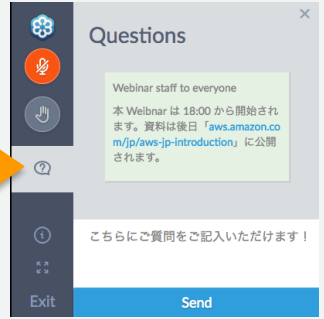
AWS Black Belt Online Seminar とは

「サービス別」「ソリューション別」「業種別」のそれぞれのテーマに分かれて、アマゾン ウェブ サービス ジャパン株式会社が主催するオンラインセミナーシリーズです。

質問を行うことができます！

- 書き込んだ質問は、主催者にしか見えません
- 今後のロードマップに関するご質問は
  お答えできませんのでご了承下さい

① 吹き出しをクリック
② 質問を入力
③ Sendをクリック



Twitter ハッシュタグは以下をご利用ください
#awsblackbelt

内容についての注意点

- 本資料では2019年8月28日時点のサービス内容および価格についてご説明しています。最新の情報はAWS公式ウェブサイト(http://aws.amazon.com)にてご確認ください。

- 資料作成には十分注意しておりますが、資料内の価格とAWS公式ウェブサイト記載の価格に相違があった場合、AWS公式ウェブサイトの価格を優先とさせていただきます。

- 価格は税抜表記となっています。日本居住者のお客様が東京リージョンを使用する場合、別途消費税をご請求させていただきます。

- AWS does not offer binding price quotes. AWS pricing is publicly available and is subject to change in accordance with the AWS Customer Agreement available at http://aws.amazon.com/agreement/. Any pricing information included in this document is provided only as an estimate of usage charges for AWS services based on certain information that you have provided. Monthly charges will be based on your actual use of AWS services, and may vary from the estimates provided.

本日のアジェンダ

- Amazon Aurora とは
- 可用性と耐久性
- 性能と拡張性
- 運用管理
- 新機能
- まとめ

Amazon Relational Database Service(Amazon RDS)

オープンソースDBとコマーシャルDBの多様な選択肢

クラウドネイティブ	オープンソース	コマーシャル
Amazon Aurora	PostgreSQL MySQL MariaDB	Microsoft SQL Server ORACLE

RDS プラットフォーム

自動フェイルオーバー バックアップ・リカバリ クロスリージョン・レプリケーション	セキュリティ 業界標準の認証・認可 自動的なパッチ適用	拡張モニタリング 定期的なメンテナンス ボタン一つでスケール

aws

Amazon Aurora

クラウド向けに再設計された MySQL, PostgreSQL と互換性のあるRDBMS
コマーシャルデータベースの性能と可用性を1/10のコストで

優れた性能と拡張性

標準的な MySQL と比べて 5 倍、標準的な PostgreSQL と比べて 3 倍のスループットを実現; リードレプリカを最大 15 個追加してスケールアウト可能

高可用性と耐久性

耐障害性、自己修復機能を兼ね備えたストレージ; 3 つの AZにわたり、6 個のコピーを保持; Amazon S3への継続的なバックアップ

高い安全性

ネットワーク分離、保管時/通信の暗号化

フルマネージド

RDSを使ったマネージドサービス ハードウェアのプロビジョニング、ソフトウェアのパッチ適用、セットアップ、構成、バックアップといった管理タスクからの解放

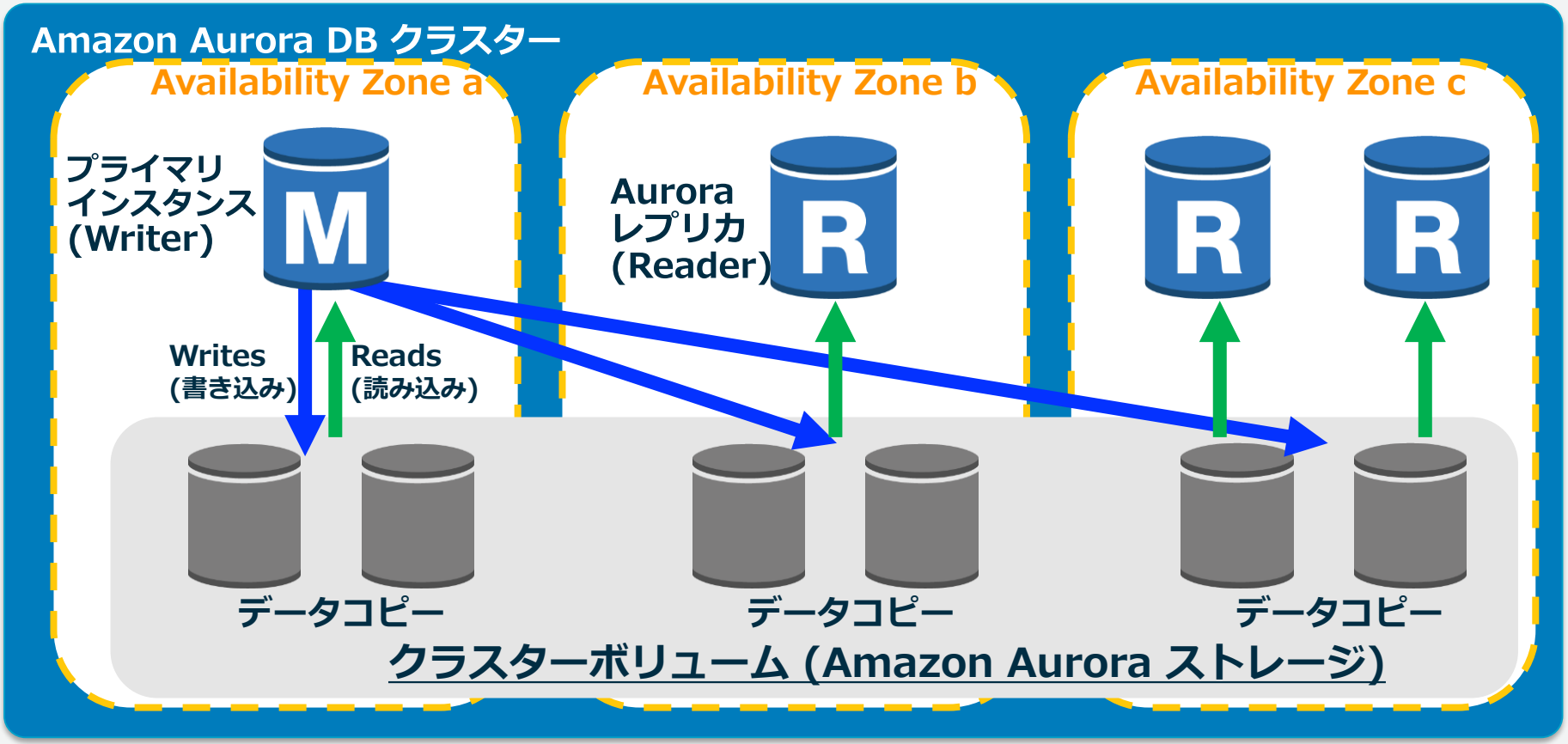
Amazon Aurora innovations
Re-imagining databases for the cloud

1. スケールアウト, 分散, マルチテナントデザイン

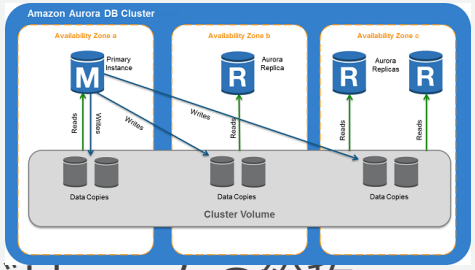
2. AWSサービスを活用したサービスオリエンテッドアーキテクチャ

3. 自動化されたタスク – 完全マネージド・サービス

Aurora を構成するコンポーネント



Aurora を構成するコンポーネント



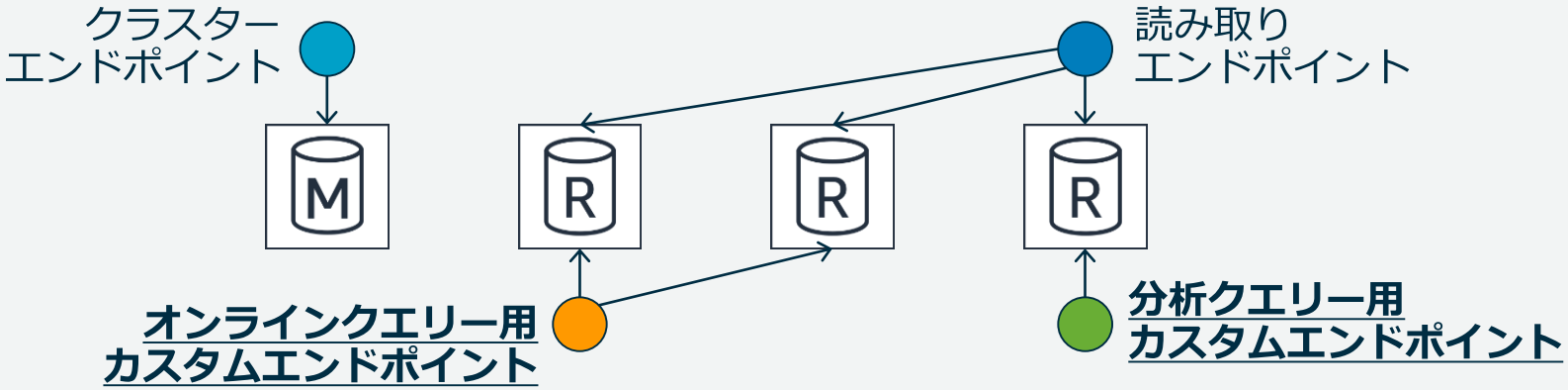
- **Amazon Aurora DB クラスター**
  - Amazon Aurora の管理単位
  - プライマリインスタンス、レプリカ、クラスターボリュームの総称
- **プライマリ DB インスタンス(Writer)** 
  - 読み込み、書き込みを行うマスターインスタンス
- **Aurora レプリカ(Reader)** 
  - 読み込みをスケールアウトさせるレプリカ(15台まで作成可能)
- **クラスターボリューム(Aurora ストレージ)**
  - 3つの AZ 間でレプリケートされる仮想ボリューム
  - プライマリインスタンスもレプリカも同じクラスターボリュームを利用

- **Aurora エンドポイント**
  - Aurora の接続先を示す URL

詳細：http://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/UserGuide/Aurora.Overview.html

Amazon Aurora エンドポイント

- Auroraクラスター内のインスタンスへの接続先
- 用途に応じて、エンドポイントを使い分ける
    - クラスターエンドポイント：常にプライマリインスタンスを指す
    - 読み取りエンドポイント：どれか１つのレプリカに接続
    - カスタムエンドポイント：ユーザーが定義可能なエンドポイント



Aurora PostgreSQL のバージョン

- Aurora PostgreSQL では以下の形式でバージョンを表す

  <major version>.<minor version>.<patch version>

  (例: 2.3.3, 1.5.2 など)

  - **major version**: PostgreSQL との対応関係あり
    - 1.y.z: 9.6系との互換性
    - 2.y.z: 10系との互換性
  - **minor version**: PostgreSQL との対応関係あり
    - 例えば、2.3.z は 10.7との互換性があることを示す
  - **patch version**: Aurora 独自のパッチバージョン

- 以下の SQL でバージョン確認が可能

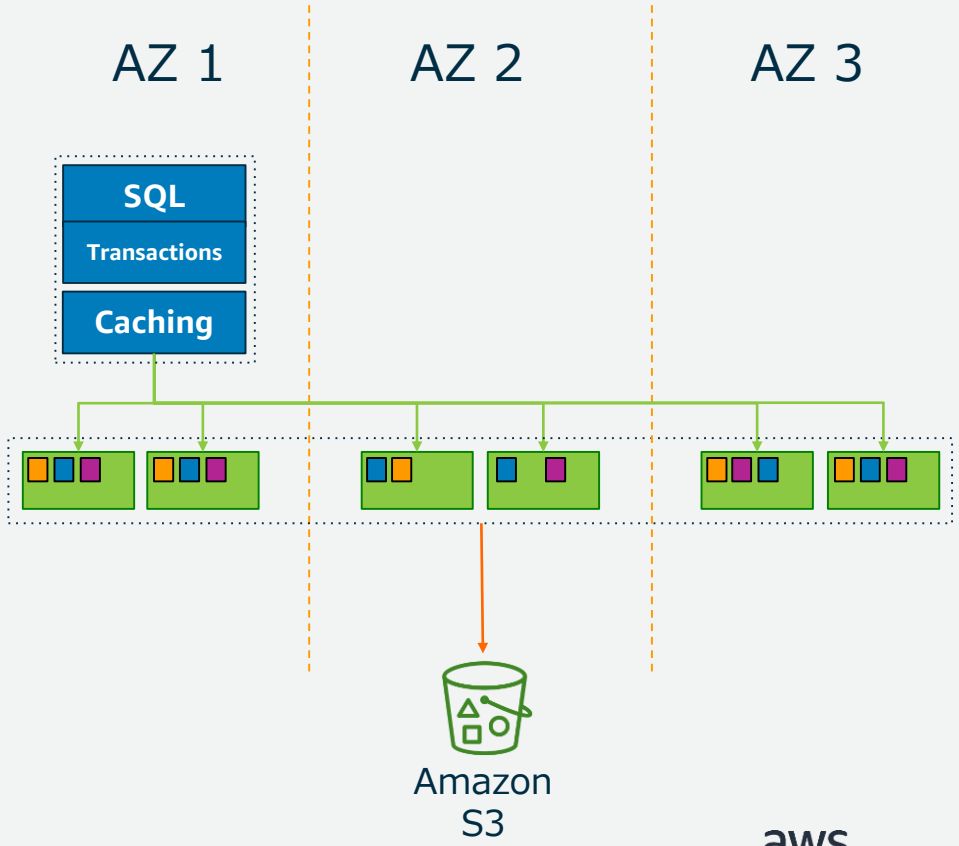
```
=>SELECT AURORA_VERSION();
```


詳細：https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/AuroraUserGuide/AuroraPostgreSQL.Updates.20180305.html

可用性と耐久性

Aurora ストレージ

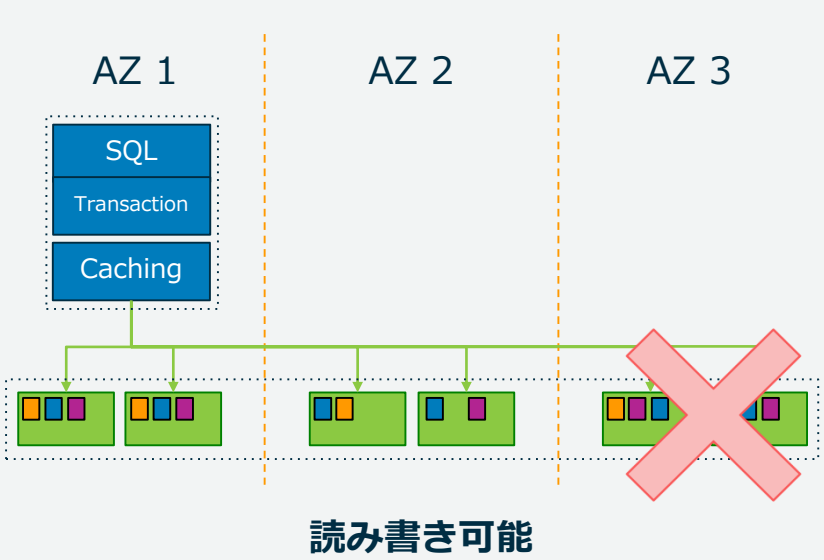
- **標準で高可用性を実現**
  - 3AZに6つのデータのコピーを作成
  - クォーラムシステムの採用
  - 継続的に S3 へ増分バックアップ

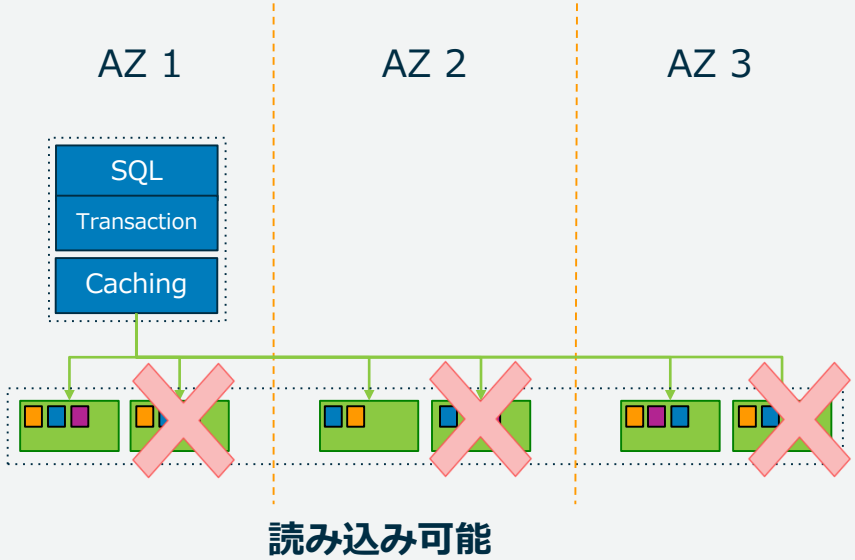


詳細：https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/AuroraUserGuide/Aurora.Overview.StorageReliability.html

ディスク障害検知と修復

- 2つのコピーに障害が起こっても、読み書きに影響は無い
- 3つのコピーに障害が発生しても読み込み可能
- 自動検知、修復(ノードの修復にはPeer to peer “gossip protocol”を利用)

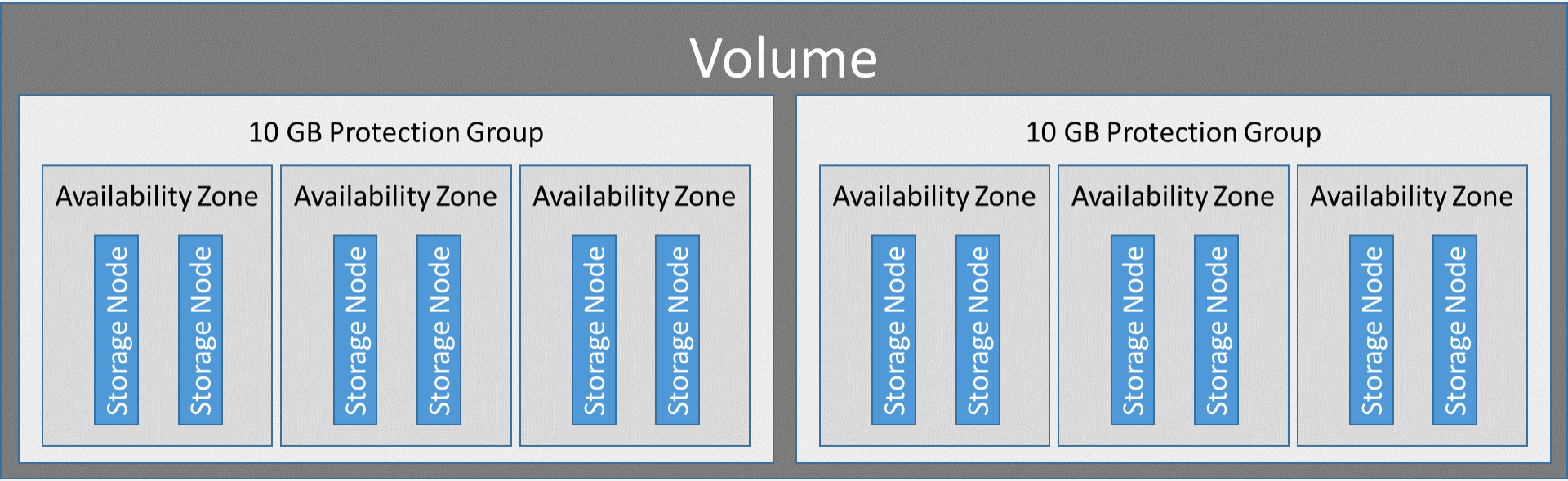




詳細： https://aws.amazon.com/jp/blogs/news/amazon-aurora-under-the-hood-quorum-and-correlated-failure/

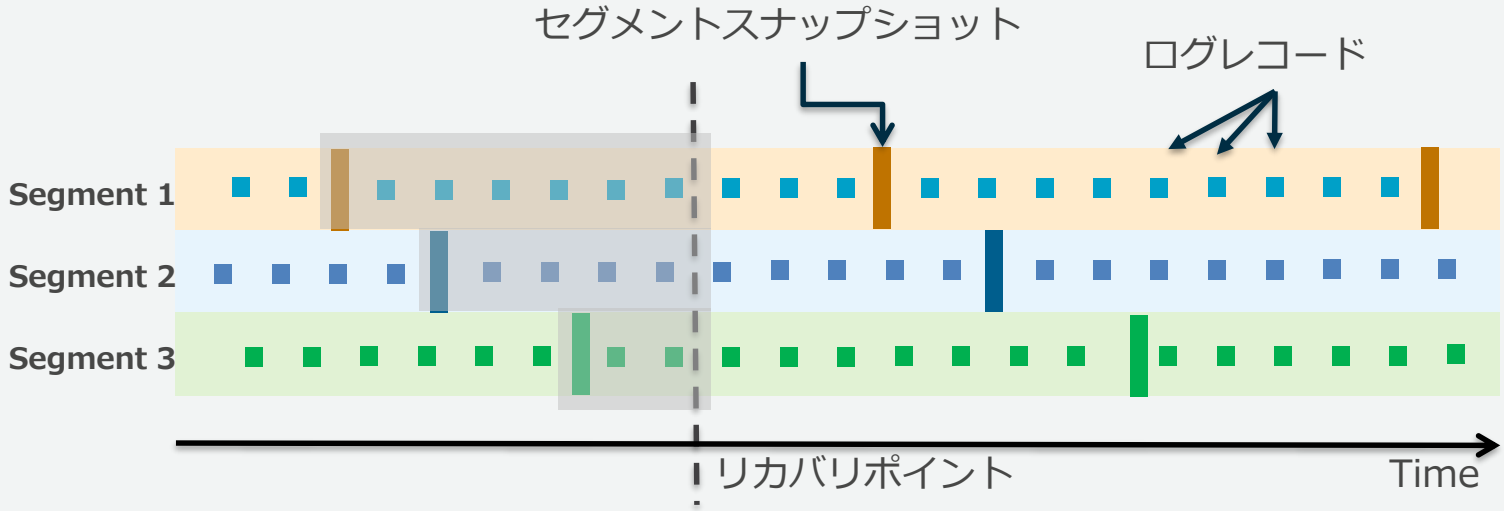
ストレージノードクラスタ

- Protection Group毎に6つのストレージノードを使用
- 各ログレコードはLog Sequence Number(LSN)を持っており不足・重複しているレコードを判別可能
  - ➢ 不足している場合はストレージノード間でゴシッププロトコルを利用し補完



詳細： http://aws.typepad.com/sajp/2017/02/introducing-the-aurora-storage-engine.html

Amazon Aurora Continuous Backup



- 各セグメントごとに Amazon S3 へ継続的なスナップショットを並列に取得
- Amazon Auroraが使用しているディスクの仕組みによりパフォーマンスへ影響を与えない
- リストア時、並列非同期に適切なセグメントのスナップショットとログを取得し、ストレージへ適用

高速なデータ修復

既存のデータベース

最後のチェックポイントからログを適用していく

PostgreSQL ではシングルプロセスなため適用完了までの時間が増加

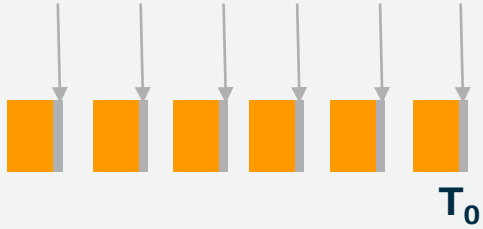
T0 でクラッシュが発生すると
最後のチェックポイントからの
ログを適用する必要がある

Amazon Aurora

リカバリ中かどうかに関わらず、ストレージノードでは継続的にログレコードを再生

並列、分散、非同期で行われる

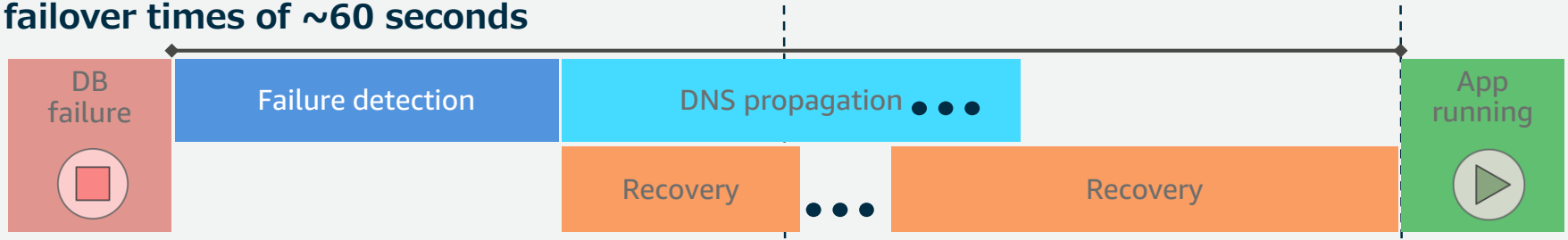
T0 でクラッシュが発生するとredo
を並列で分散して非同期でログの適用を行う



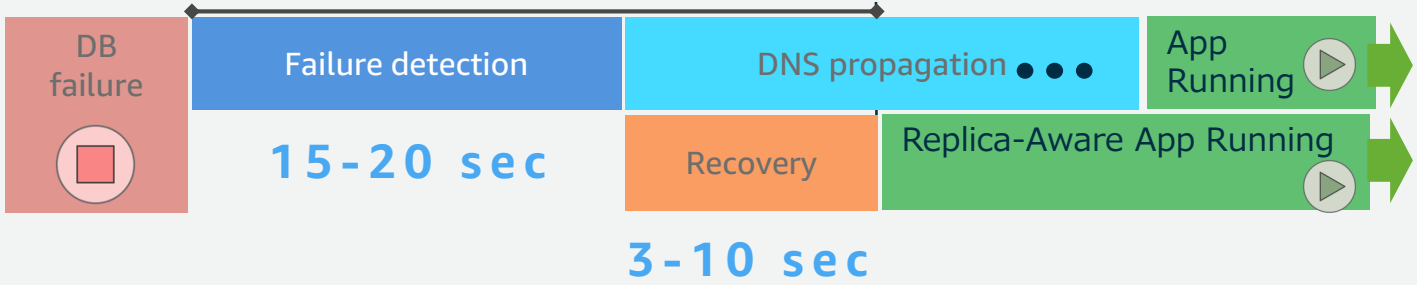
aws

高速でより予測可能なフェイルオーバー時間

Amazon RDS for PostgreSQL is good:
failover times of ~60 seconds



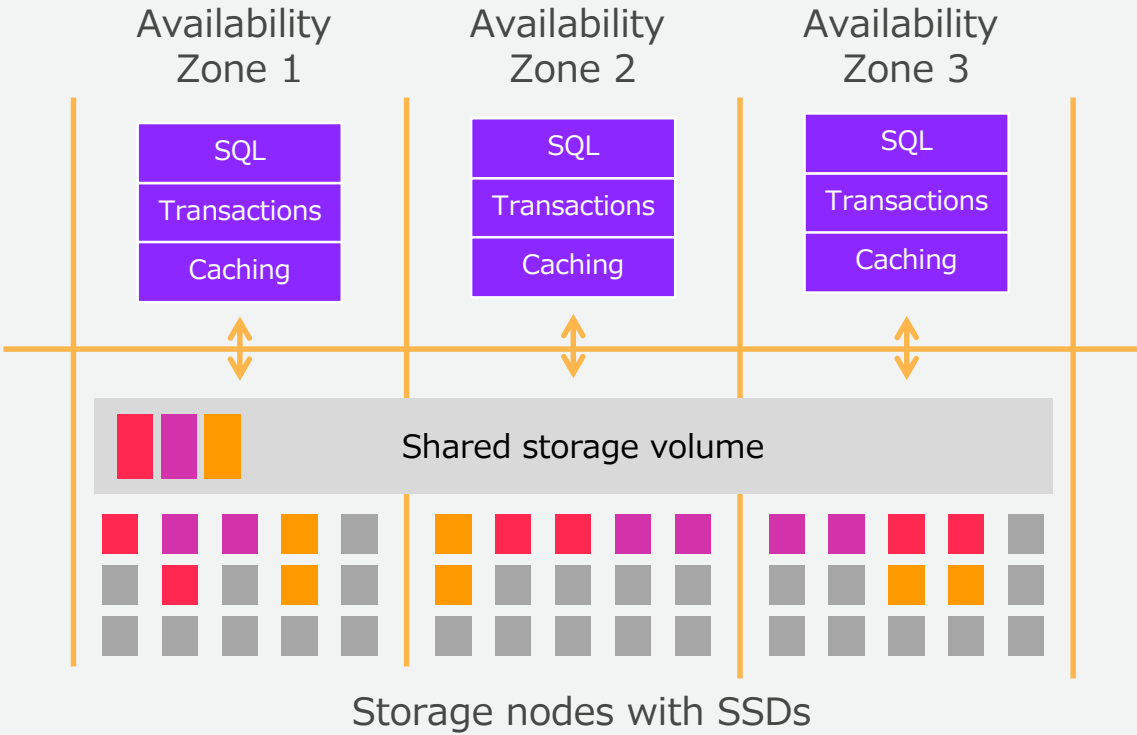
Amazon Aurora is better:
failover times < 30 seconds



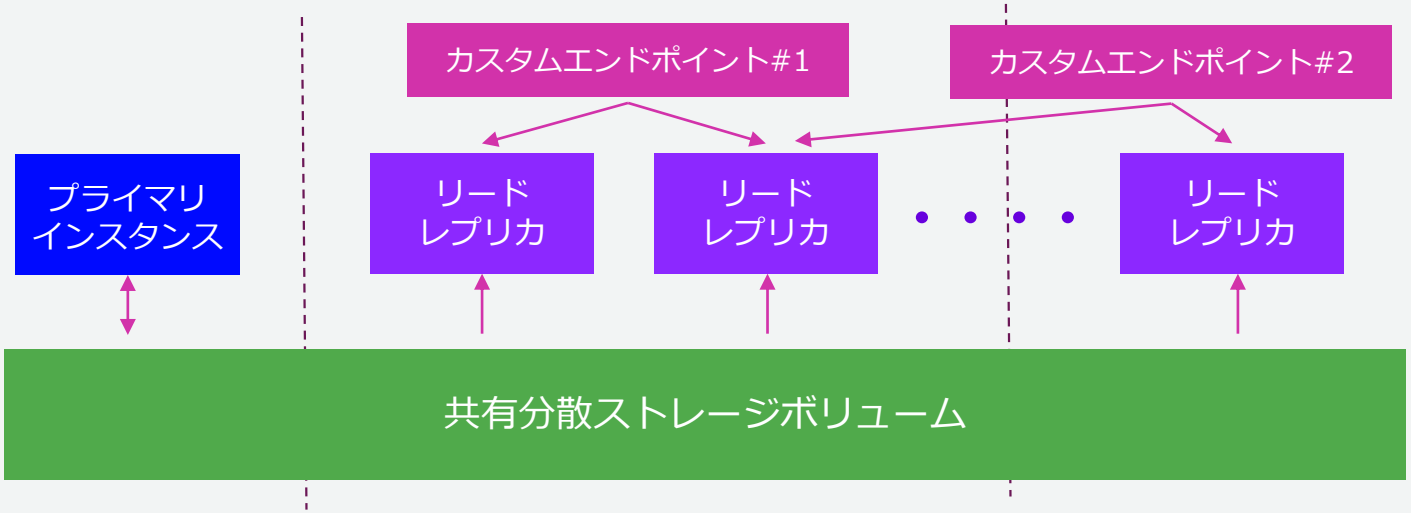
性能と拡張性

Aurora scale-out, distributed architecture

- Amazon Aurora向けに作られたlog-structured 分散ストレージシステム
- データは10GBずつ“protection groups”に保存され、**64TBまで自動的にスケールアップ**
  - 10GBから64TBまで自動でスケールアップ
  - 実際に使った分だけ課金
- ストレージボリュームは3AZに数百～数千インスタンスのストレージノードを配置



Aurora リードレプリカ



- 複数のアベイラビリティーゾーンに最大15の昇格可能なリードレプリカ
- レプリカの遅延が低い（通常10ms未満）
- フェイルオーバーの順序を設定可能
- カスタムリーダーエンドポイントによるレプリカの使い分け

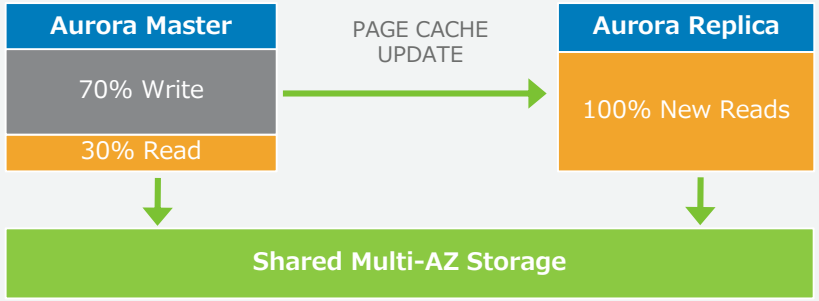
IO traffic in Aurora Replicas

PostgreSQL READ SCALING

Physical: WAL をレプリカに送信

- 書き込みは各インスタンス同様に実施
- ストレージはそれぞれ独立
(Shared Nothing)

Amazon Aurora READ SCALING

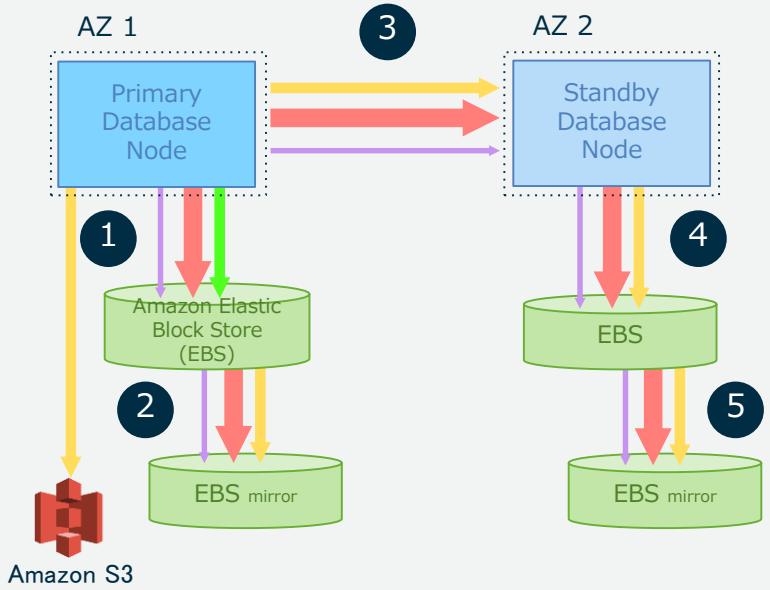


Physical: マスタからレプリカにWALを送信

- ストレージは共有。レプリカ側でストレージへの書き込みは起こらない
- ページキャッシュが適用

IO traffic in Amazon RDS for PostgreSQL

RDS FOR POSTGRESQL WITH MULTI-AZ



IO FLOW

EBSに書き込み – EBSがミラーへ複製し、両方終了後ack
スタンバイインスタンス側のEBSに書き込み

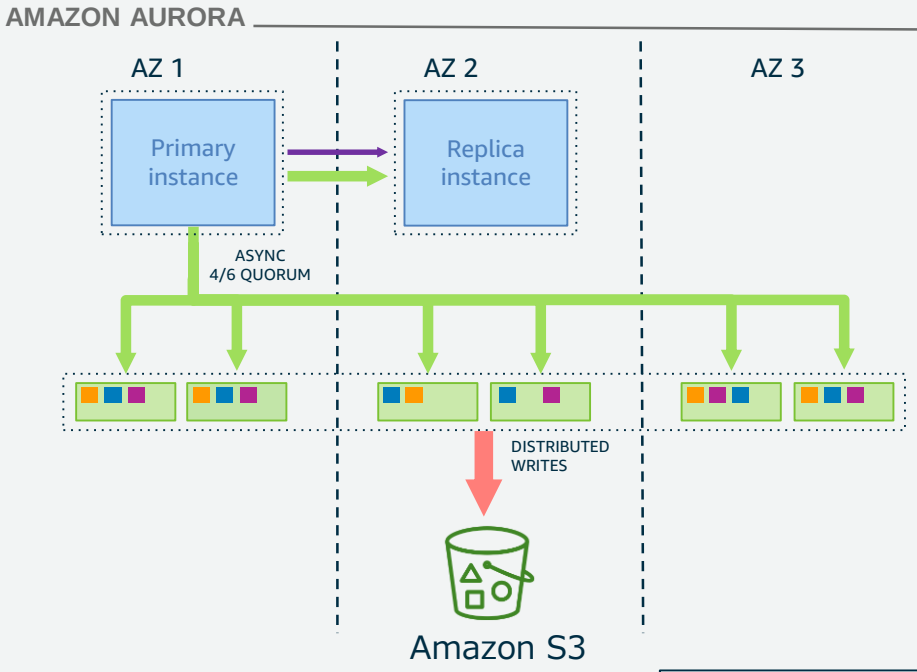
OBSERVATIONS

ステップ1, 3, 5はシーケンシャルかつ同期
それによりレイテンシーもパフォーマンスのゆらぎも増加
各ユーザー操作には様々な書き込みタイプがある
書き込み破損を避けるためにデータブロックを2回書く必要性

TYPE OF WRITE

WAL　DATA　COMMIT LOG & FILES

IO traffic in Aurora (データベース)



IO FLOW

REDOログレコードをまとめる – 完全にLSN順に並ぶ
適切なセグメントに分割する – 部分ごとに並ぶ
ストレージノードへまとめて書き込む

OBSERVATIONS

REDOログレコードのみ書き込む; 全てのステップは非同期
データブロックは書かない(チェックポイント, キャッシュ置換時)
6倍のログ書き込みだが, **1/9の**ネットワークトラフィック
ネットワークとストレージのレイテンシー異常時の耐性

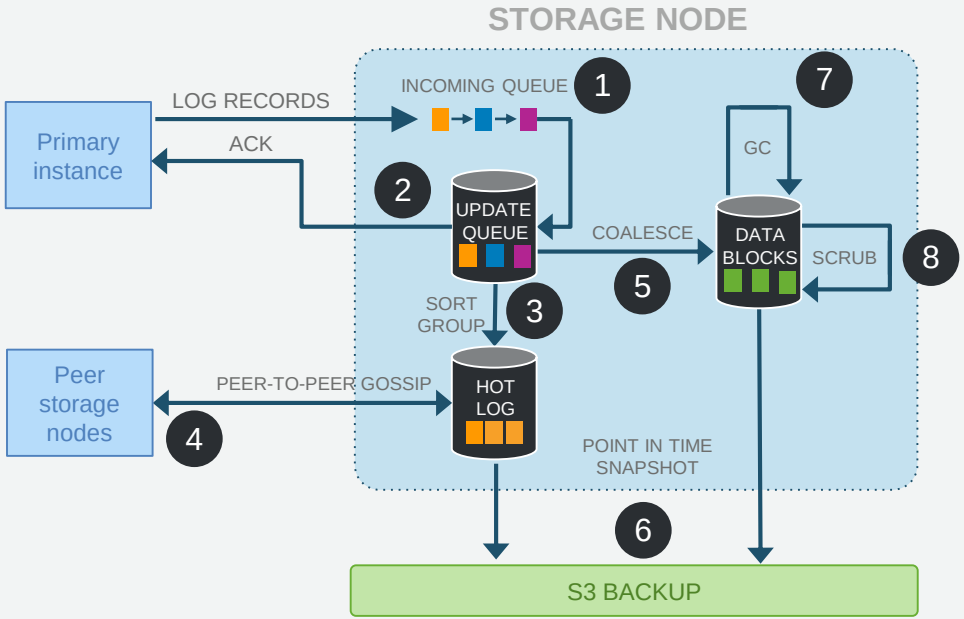
PERFORMANCE

write-only もしくは、read/write が混在するワークロードにて、PostgreSQL のコミュニティエディションに比べて、2倍以上の性能を発揮

TYPE OF WRITE

AMAZON AURORA + WAL LOG | WAL | DATA | COMMIT LOG & FILES

IO traffic in Aurora (ストレージノード)



IO FLOW

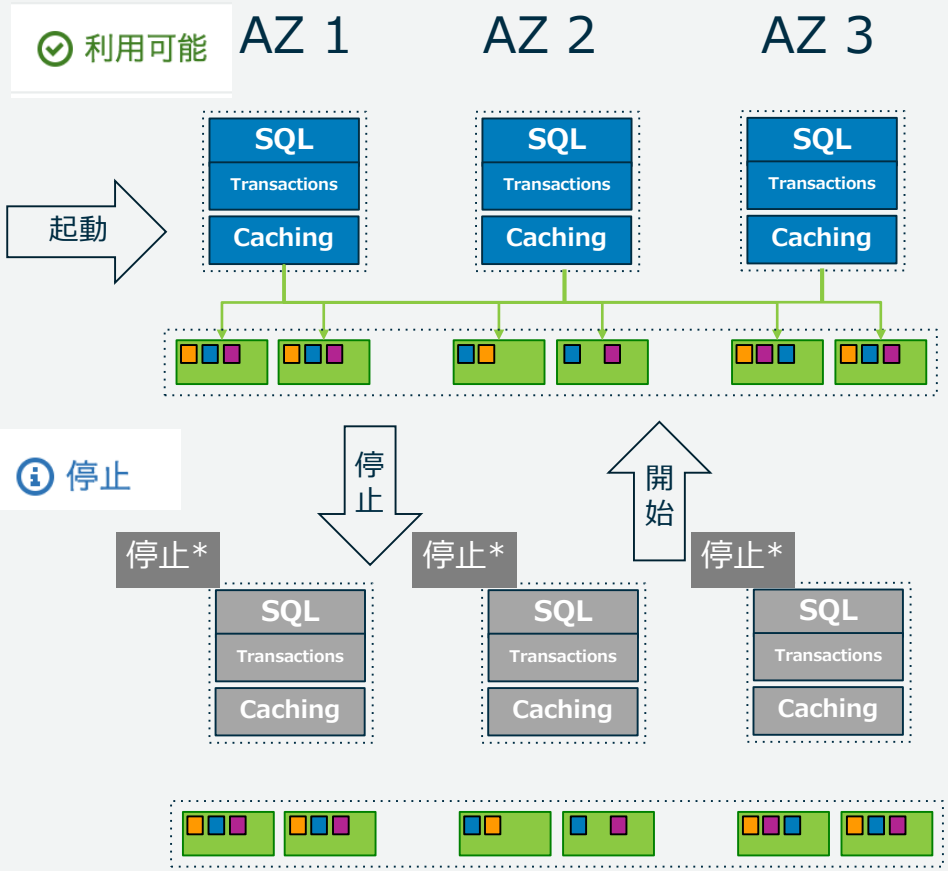
① レコードを受信しインメモリのキューに追加
② レコードをSSDに永続化してACK
③ レコードを整理してギャップを把握
④ ピアと通信して穴埋め
⑤ ログレコードを新しいバージョンのデータブロックに合体
⑥ 定期的にログと新しいバージョンのブロックをS3に転送
⑦ 定期的に古いバージョンのガベージコレクションを実施
⑧ 定期的にブロックのCRCを検証

OBSERVATIONS

全てのステップは非同期
ステップ1と 2だけがフォアグラウンドのレイテンシーに影響
インプットキューはPostgreSQLに比べて**極めて小さい**
レイテンシーにセンシティブな操作に向く
ディスク領域をバッファーに使ってスパイクに対処

運用管理

Amazon Aurora クラスター全体のライフサイクル



※ 削除直前のスナップショットを取得可能

停止に関するTips

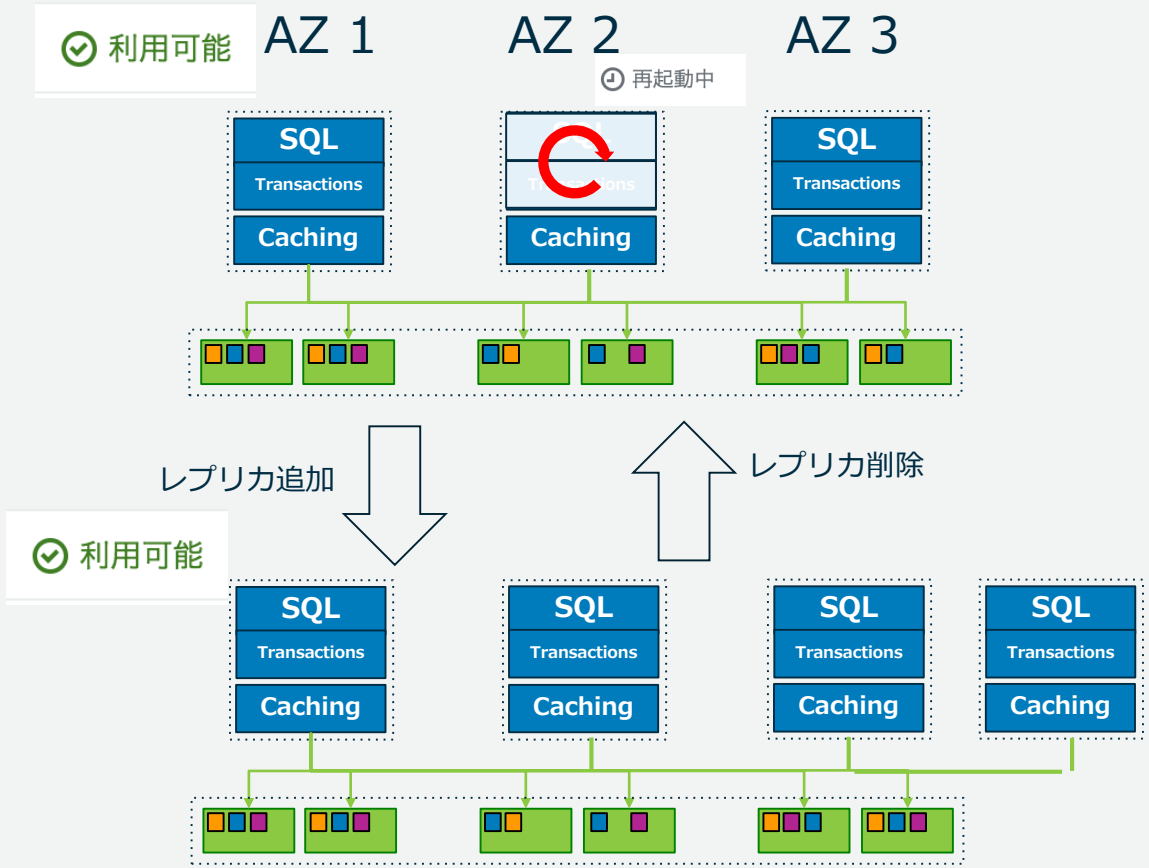
- 全インスタンス(プライマリ、レプリカ)が停止
- 停止可能な期間は最大 7 日間
- 停止に伴い、ログは削除されるため、必要に応じてエクスポート
- 停止中もストレージ料金は発生

削除に関するTips

- 削除時に最終スナップショットが取得可能
- 削除したクラスターの復元にはスナップショットが必要

＊ 実態としては削除

クラスター内のインスタンスライフサイクル



再起動に関するTips

- データベースプロセスのリスタートが発生してもキャッシュが残った状態を維持可能
- プライマリインスタンスを再起動すると、Aurora レプリカもすべて自動的に再起動

レプリカ追加に関するTips

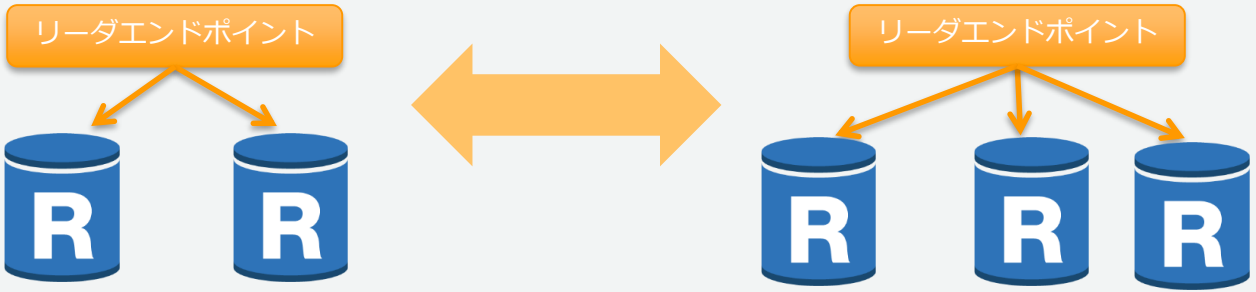
- クラスターのダウンタイムなく追加可能
- DNSの伝播時間についての考慮
- 追加したインスタンスのキャッシュについての考慮
- 最大15台まで

レプリカ削除に関するTips

- クラスターのダウンタイムなく削除可能
- DNSの伝播時間についての考慮

Auto Scaling によるレプリカの自動増減

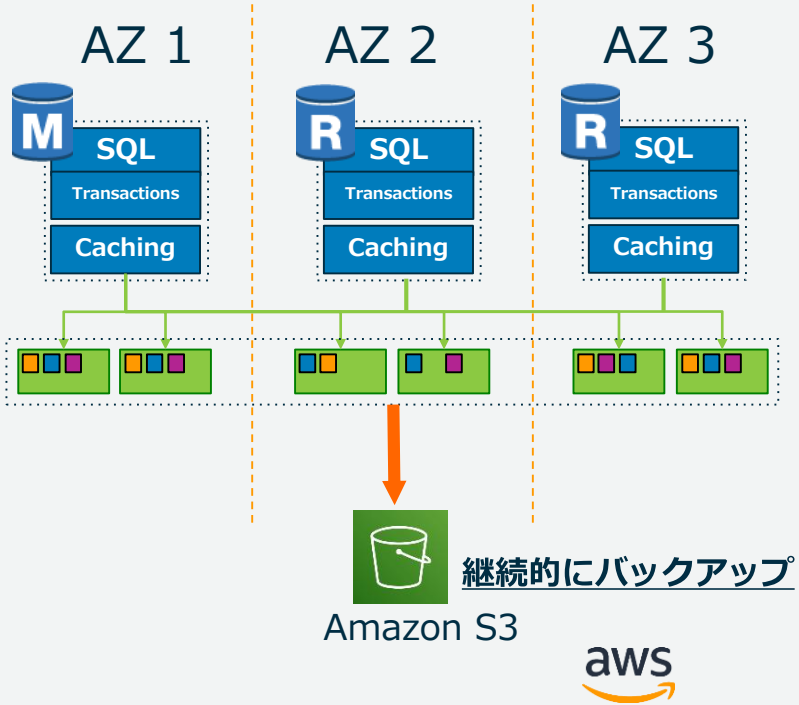
- Aurora レプリカをメトリクスに応じて動的に増減
  - 読み取りクエリの分散や余分なコストを支払うリスクを軽減
  - リーダーエンドポイント/カスタムエンドポイントを利用することで自動的なレプリカの追加削除に対応可能
  - Cooldown PeriodやMinimum / Maximum Capacityを設定可能
  - 注意点
    - 追加されるのはPrimary Instanceと同じDBインスタンスクラス
    - 監視間隔、起動時間を考慮すると急激なスパイクへの対応は困難
      ※ 予測可能なイベントでは事前にレプリカを追加
    - 増えたレプリカのキャッシュ管理



Aurora のバックアップ機能

- 自動バックアップが常に有効
  - Amazon S3 へ継続かつ自動的に増分バックアップ(バックアップウィンドウも指定不要)
  - Aurora ストレージの仕組みにより パフォーマンスへ影響は出ない
  - バックアップの保持期間(1～35日)のみ指定

- データの復元
  - 取得したスナップショットから復元可能
  - ポイントインタイムリカバリとして、5分前からバックアップの保持期間までの任意の位置に秒単位で復元可能
    - 最新の復元可能時刻はマネジメントコンソールで確認可能(Latest Restorable Time 値または Earliest Restorable Time 値)



パラメーターグループによる設定

- パラメータ設定(主にpostgresql.conf の内容)作業を省力化
  - 複数インスタンスに同じパラメーターを簡単にアタッチ可能
  - デフォルトのパラメータはチューニング済み

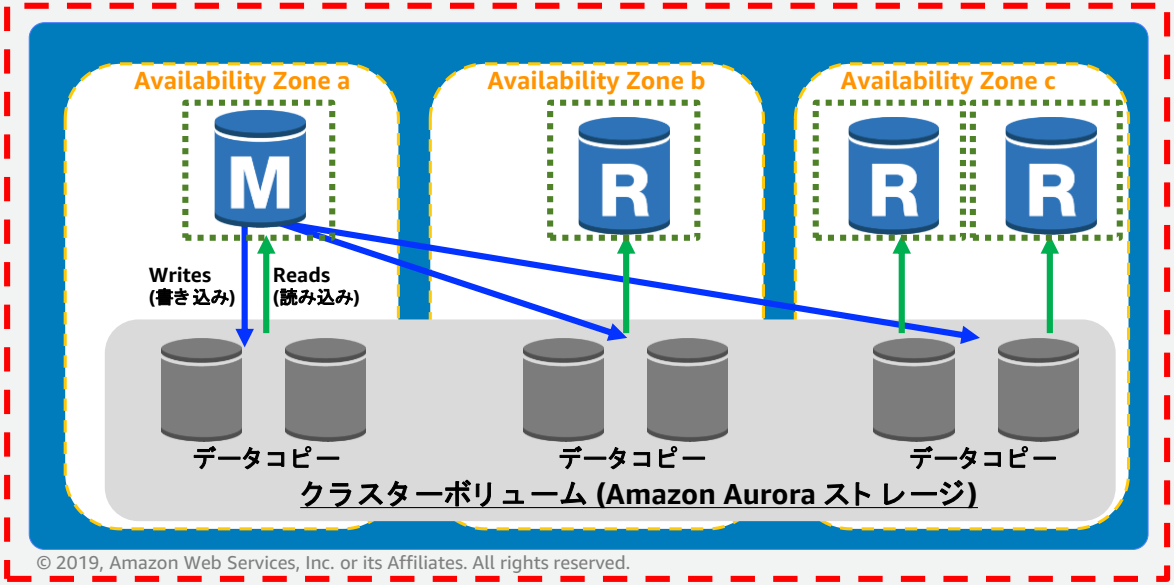
- 考慮事項
  - パラメータ変更が適用されるタイミング
    - 静的パラメータ：適用にインスタンスの再起動が必要
    - 動的パラメータ：即座に適用
  - DB クラスターパラメータとDB パラメータ(次のページで解説)
  - パラメータ変更に伴うテスト

詳細：https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/UserGuide/Appendix.PostgreSQL.CommonDBATasks.html#Appendix.PostgreSQL.CommonDBATasks.Parameters

DB クラスターパラメータとDB パラメータ

- クラスター、インスタンスそれぞれにパラメータグループを設定
  - DB クラスターパラメータグループ: クラスター内の全インスタンスに適用
  - DB パラメータグループ: 単一のインスタンスに適用

  ※ 重複するパラメータがある場合、 DB パラメータグループの設定値が優先される



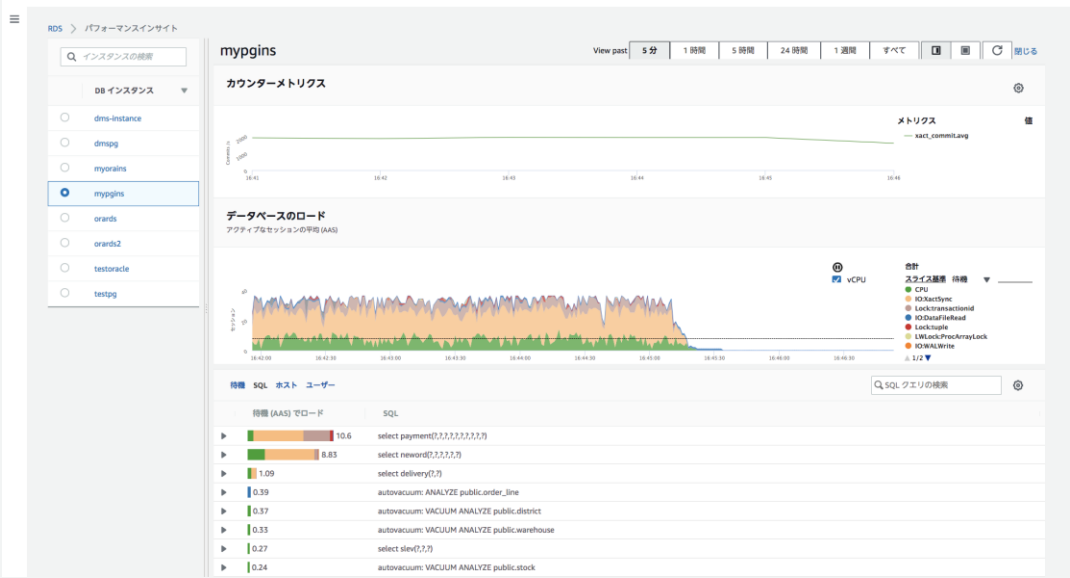
クラスターパラメータグループ

インスタンスパラメータグループ

Performance Insights によるワークロード監視

- データベースへの負荷をリアルタイムに表示
- データベースのキャパシティ、統計情報を表示

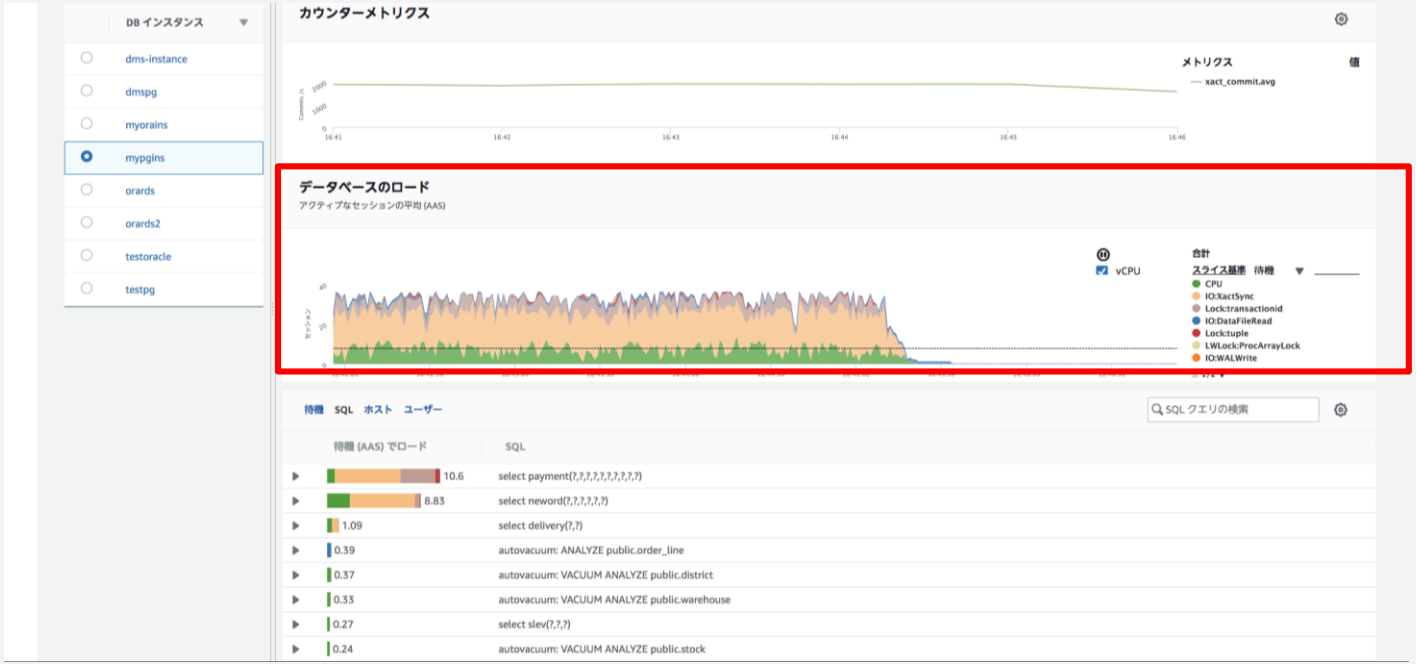
- 主要な機能
  - “データベースロード” チャート
  - “カウンターメトリクス” チャート
  - “Top N ディメンション” テーブル
    - ディメンション: 待機、SQL、ホスト、ユーザー



詳細：https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/UserGuide/USER_PerfInsights.html

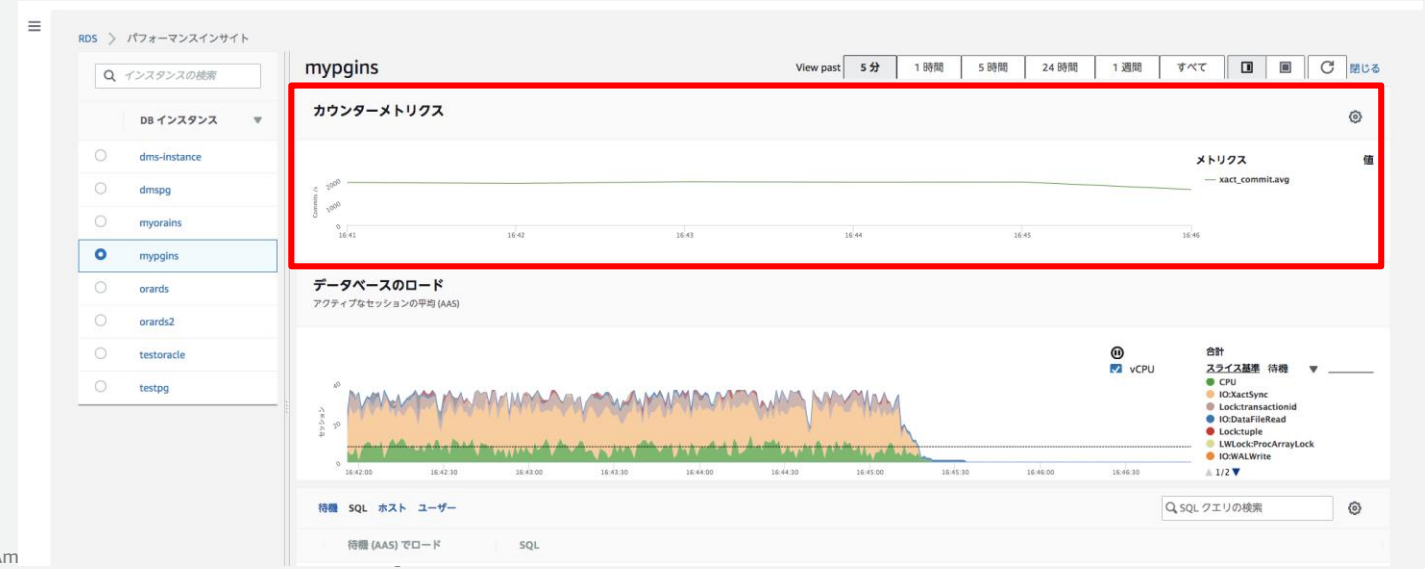
データベースロードチャート

- データベースロードとは
  - 全てのデータベースエンジンは“Active” / “Idle”属性の接続がある
  - 1秒おきに”Active”な接続の詳細な情報をサンプリング
    - SQL文、状態(CPU処理中、待機中) 、接続元ホスト、接続ユーザー



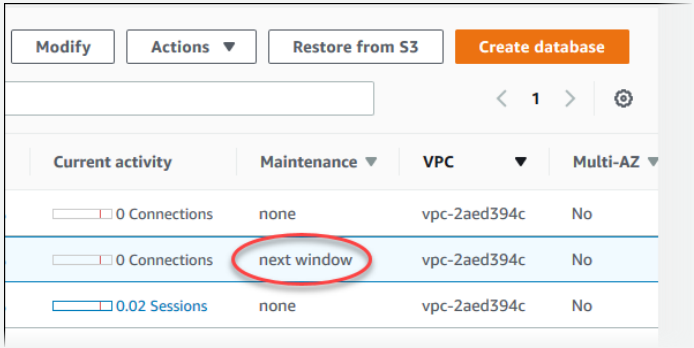
カウンターメトリクスチャート

- Performance Insights Dashboard にOS, データベースのパフォーマンスメトリクスを追加可能
  - 統計情報などの表示が可能



Amazon AuroraDB クラスターのメンテナンス

- **サービス側でメンテナンスを定期的に実行**
  - **メンテナンスウィンドウで指定した曜日・時間帯**に自動実施
  - 安全性・堅牢性に関わるソフトウェアパッチを自動適用
  - リブートやダウンタイムを伴うケースあり
  - 通常、数ヶ月に一度の頻度で発生（毎週必ずではない）
  - 指定した時間帯の数分間で実施（メンテナンス内容に依存）
- 考慮事項
  - メンテナンス有無の定期的な確認
    - マネジメントコンソール、CLI, API
  - イベント通知の設定
  - 適切なメンテナンスウィンドウ設定
    - トラフィックが少ない曜日・時間帯

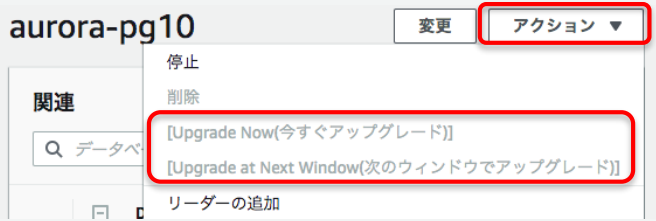


詳細：https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/AuroraUserGuide/USER_UpgradeDBInstance.Maintenance.html

マネジメントコンソールでのメンテナンス確認と適用

- メンテナンス(Maintenance)列の表示を確認
  - 必須(required): メンテナンスアクションがリソースに適用(期限あり)
  - 利用可能(available) – メンテナンスアクションが利用可能
    (自動的にはリソースに適用されないので、必要に応じて手動で適用)
  - 次のウィンドウ(next window) –次回のメンテナンスウィンドウ中に適用
  - 進行中(in progress) – メンテナンスアクション適用中

- 更新の適用
  - 対象のクラスターを選択し、以下のいずれかの操作
    - [アクション]-[Upgrade Now]
    - [アクション]-
      [Upgrade at Next Window]



AWS CLI でのメンテナンス確認と適用

- AWS CLI によりメンテナンスがスケジュールされた場合の通知、適用を自動化可能
  - メンテナンスの確認: describe-pending-maintenance-actions
    - 詳細: https://docs.aws.amazon.com/cli/latest/reference/rds/describe-pending-maintenance-actions.html
  - メンテナンスの適用：apply-pending-maintenance-action
    - 詳細: https://docs.aws.amazon.com/cli/latest/reference/rds/apply-pending-maintenance-action.html

- メンテナンス適用の例

```
$ aws rds apply-pending-maintenance-action --resource-identifier
<arn> --apply-action system-update --opt-in-type immediate
```

Aurora の更新情報の確認

以下のリファレンスをAurora の更新情報やサービス全体に関わるメンテナンス情報の確認として利用可能

- Amazon Aurora PostgreSQL のデータベースエンジンのバージョン: https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/AuroraUserGuide/AuroraPostgreSQL.Updates.20180305.html

- Amazon Aurora Forum: https://forums.aws.amazon.com/forum.jspa?forumID=227

新機能

Aurora with PostgreSQL Compatibility：新しいメジャーバージョン 10系をリリース

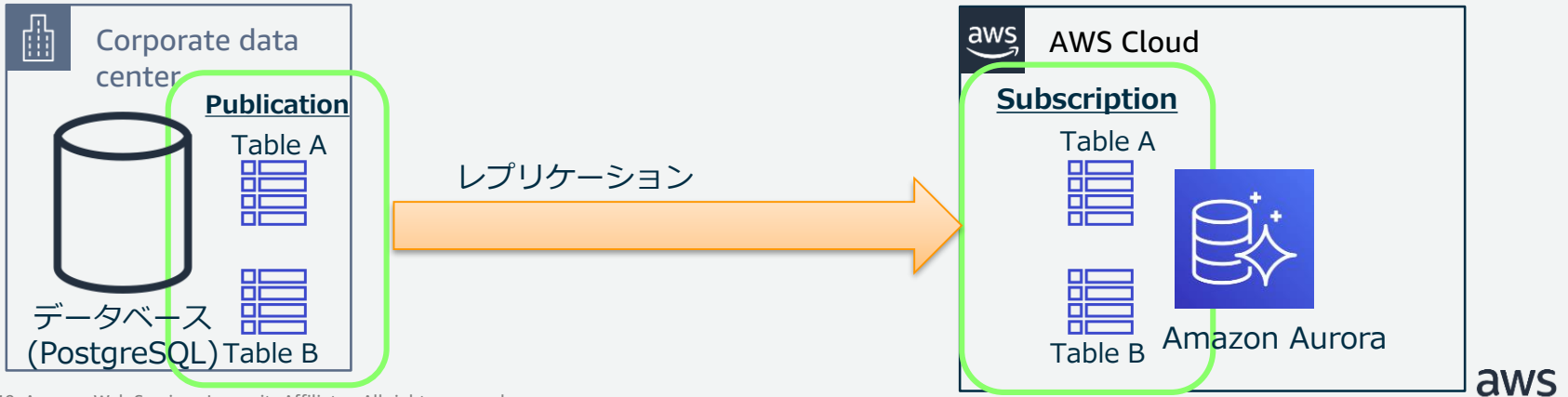
- PostgreSQL 10.y に対応(Aurora PostgreSQL としては v2.y)
  - 主な新機能：
    - ネイティブ・パーティショニング(宣言的パーティショニング)
    - パラレルクエリ強化(パラレルクエリに対応するScan方式、Join方式の追加)
    - postgres_fdw強化(リモートサーバでの集約に対応)

→その他： https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/AuroraPostgreSQL.Updates.20180305.html

詳細： https://aws.amazon.com/jp/about-aws/whats-new/2018/09/amazon-aurora-postgresql-compatibility-supports-postgresql104/

PostgreSQL 論理レプリケーションのサポート

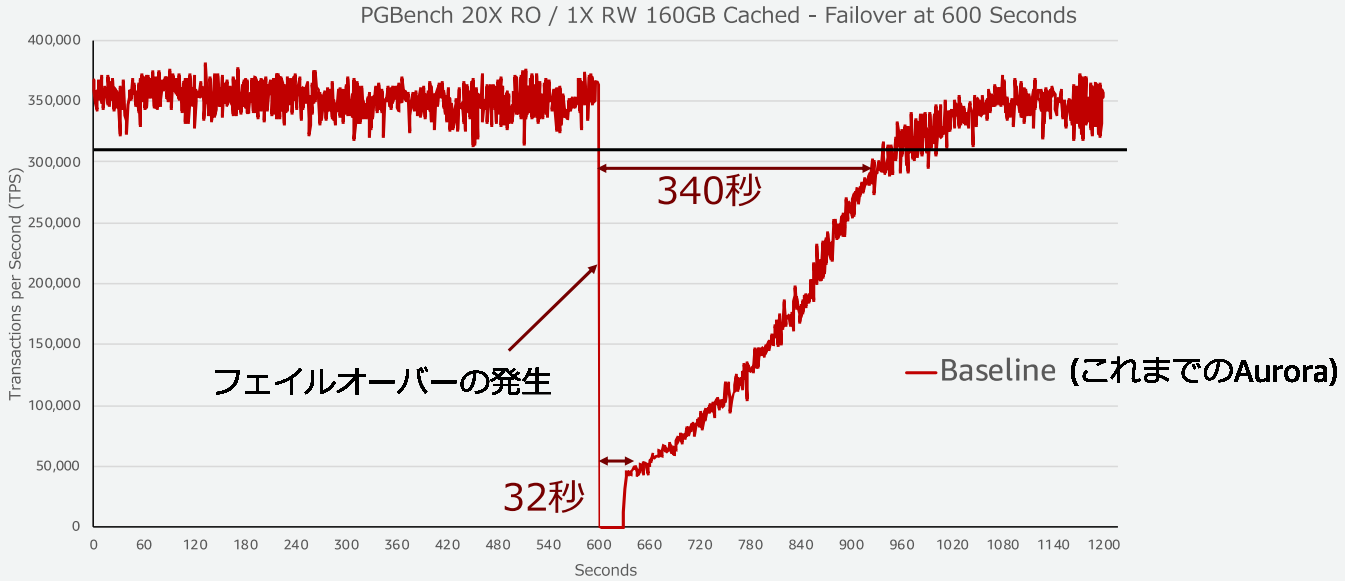
- ロジカルレプリケーションが利用可能
  - Aurora PostgreSQL と外部のPostgreSQL でのレプリケーションが可能
    - 別リージョンの RDS/Aurora PostgreSQL へレプリケーションする災害対策
    - 移行元の PostgreSQL から同期することで移行時のダウンタイムを短縮
  - 基本的な手順/注意点(制限)は PostgreSQL マニュアルを参照
    - 例)DDLなどのレプリケーションは出来ない
  - 利用可能なバージョン
    - Aurora: Aurora with PostgreSQL Compatibility 2.2.0(PostgreSQL 10.6互換)以降
    - RDS: RDS for PostgreSQL 10.4以降



詳細：https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/UserGuide/CHAP_PostgreSQL.html#PostgreSQL.Concepts.General.FeatureSupport.LogicalReplication

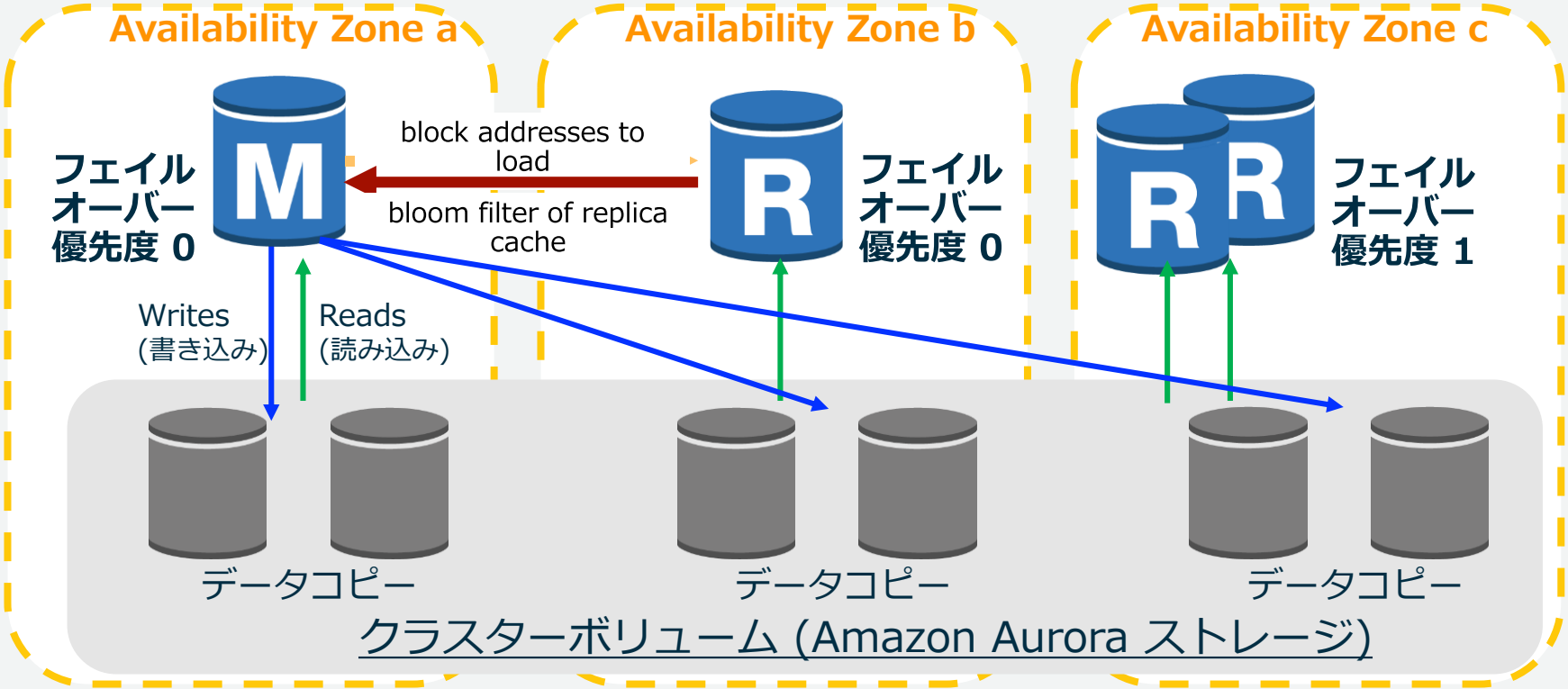
Cluster Cache Management による高速リカバリ

フェイルオーバー後のコールドキャッシュを回避



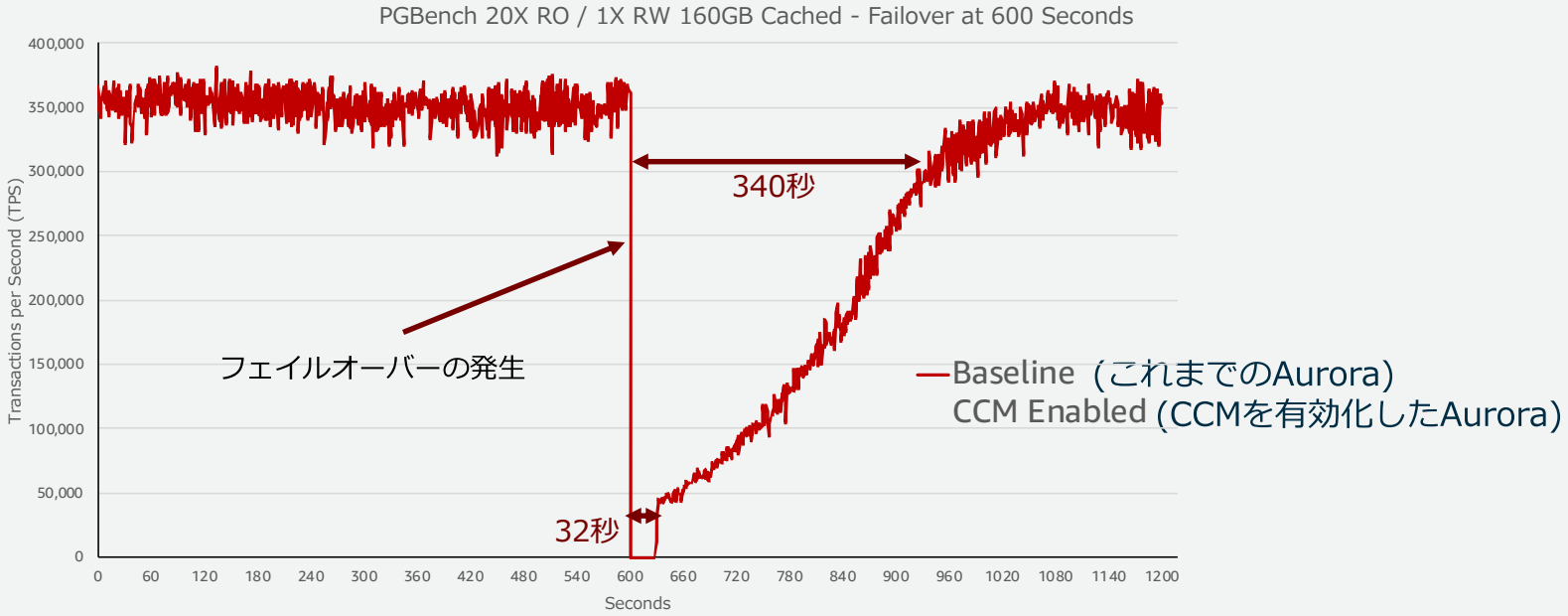
通常、データベースをバックアップしてキャッシュをウォームアップするにはしばらく時間がかかります。このフェールオーバーの例では、CCMがないと、DBの起動に32秒かかりましたが、パフォーマンスの90%を回復するには340秒でした

Cluster Cache Management(CCM) の動作



Cluster Cache Managementによる高速リカバリ

Writerとほぼ同期されたウォームキャッシュを持つレプリカへフェイルオーバー



CCMが有効になっていると、データベースはウォームアップされたキャッシュにフェイルオーバ。フェイルオーバーから32秒後には、90%のパフォーマンスを回復

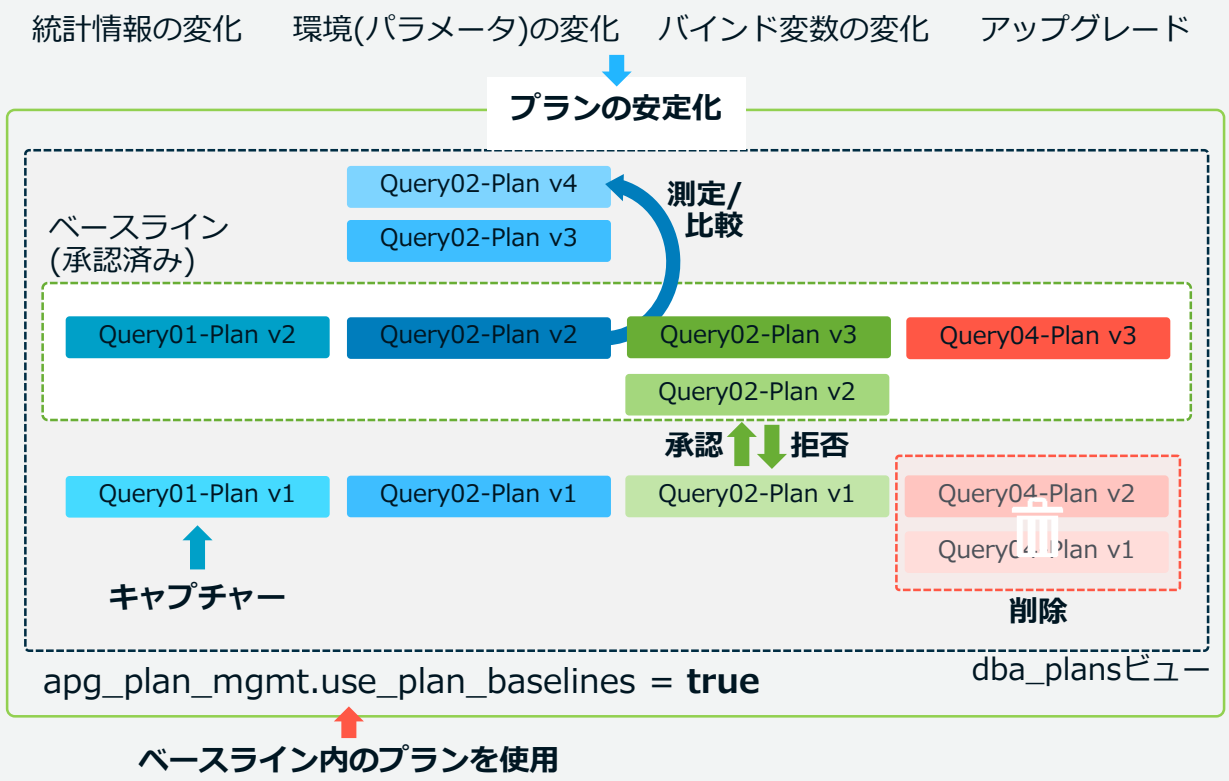
Query Plan Managementの概要

機能概要

- ✓ 手動/自動でプランのキャプチャー
- ✓ プランの測定/比較
- ✓ プランの承認/拒否
- ✓ ベースライン内のプランを使用
- ✓ pg_hint_planを使ったプランの修正
- ✓ プランの削除
- ✓ プランのエクスポート/インポート

サポートバージョン/制限

- ✓ Aurora PostgreSQL 2.1.0以上 (PostgreSQL 10.5互換)
- ✓ PL/pgSQLは未サポート

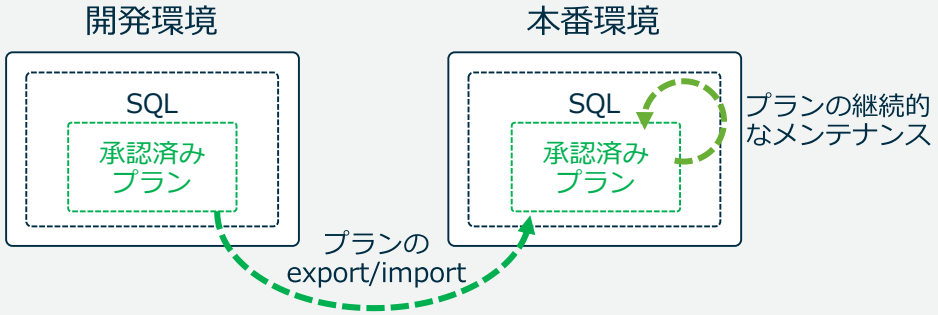


詳細：https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/AuroraUserGuide/AuroraPostgreSQL.Optimize.html

Query Plan Managementのユースケース

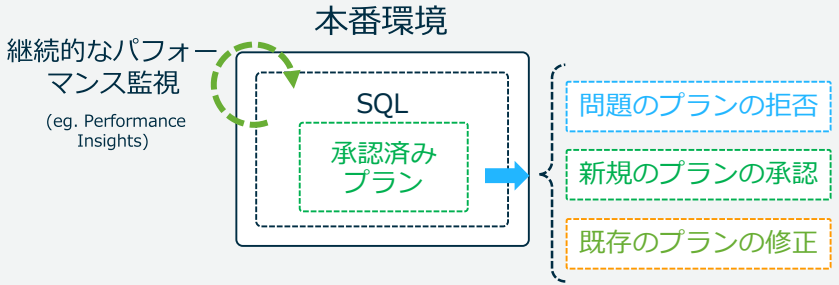
パフォーマンス低下を防止する事前予防

- 開発環境でパフォーマンスに影響を与えるSQL文を特定し**手動/自動のキャプチャーでプラン**を取得
- 開発環境から承認済みプランを**エクスポート**し、本番環境に**インポート**
- 本番環境では承認された**ベースラインのプランを強制**
- 新しくキャプチャーされたプランの効率性を**分析**し、必要な場合は承認する



パフォーマンス低下を検出し修復する事後対応

- アプリケーションのプランを**ベースラインで固定**しつつ、新しいプランのキャプチャーを継続
- 実行中のアプリケーションの**パフォーマンス低下を監視**(ex. Performance Insights)
- 既存のベースラインのプランを**拒否**し、適切な別の承認済みプランを使用させる
- **pg_hint_plan拡張**でプランを**修正**することも可能

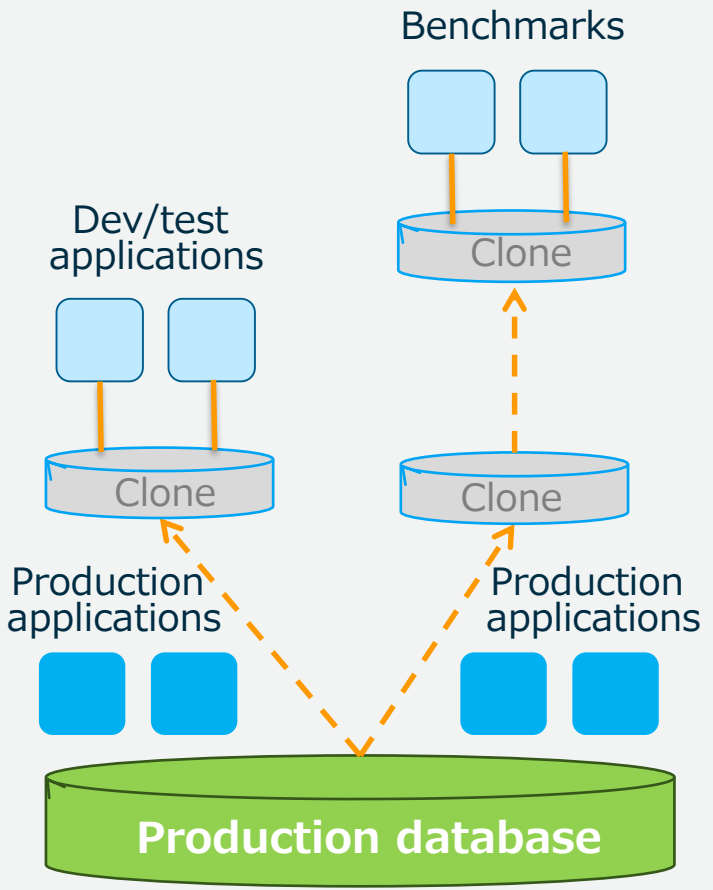


Database cloning

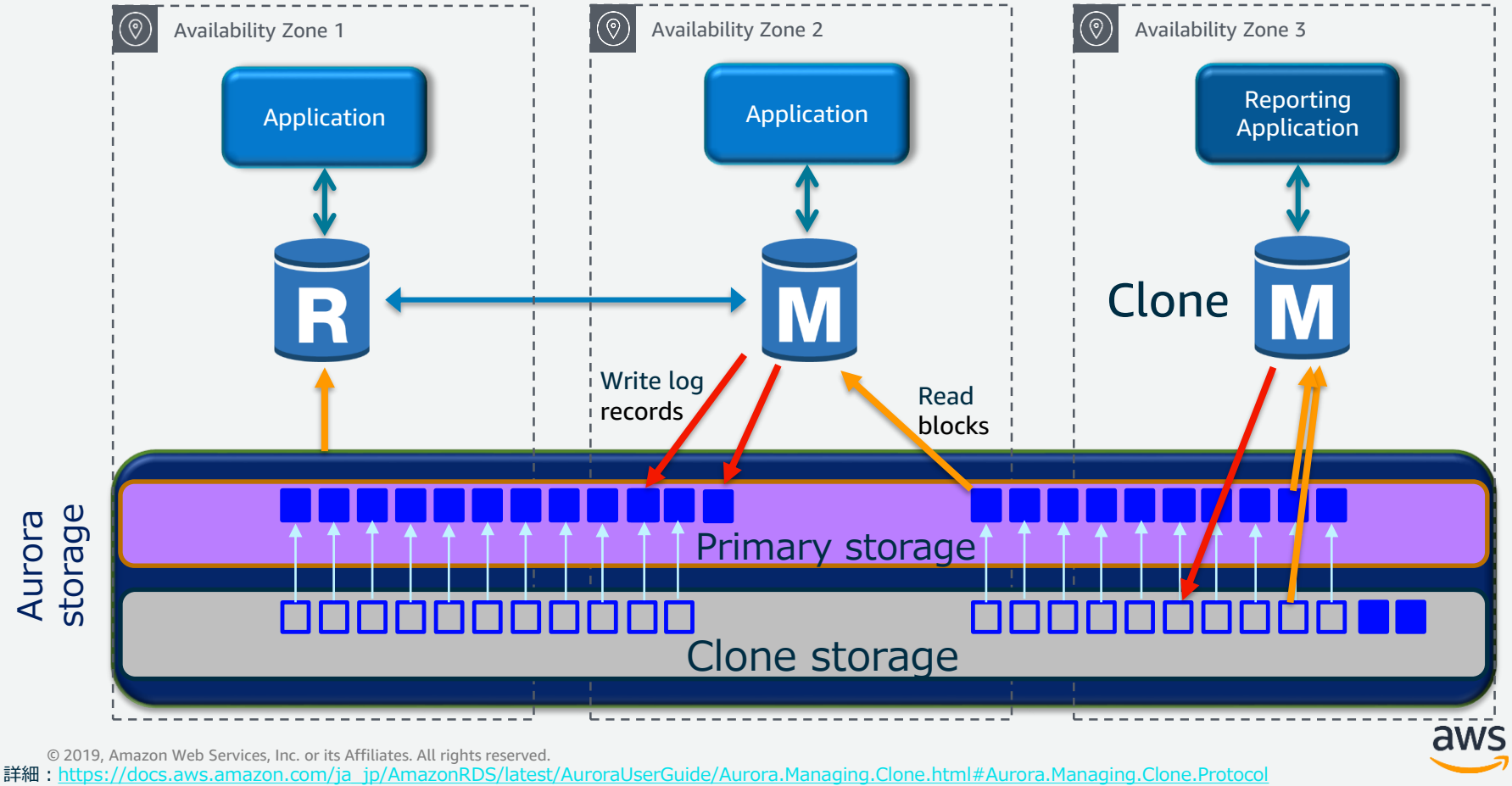
- **ストレージコストを増やすことなく データベースのコピーを作成**
  - データをコピーするわけではないため、クローンの作成はほぼ即座に完了
  - データのコピーはオリジナルボリュームとコピー先のボリュームのデータが異なる場合の書き込み時のみ発生
  - アカウント間のクローン共有も可能

ユースケース

- プロダクションデータを使用したテスト
- データベースの再構成
- プロダクションシステムに影響を及ばさずに分析目的で特定の時点でのスナップショットを保存



Database cloning のしくみ



詳細：https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/AuroraUserGuide/Aurora.Managing.Clone.html#Aurora.Managing.Clone.Protocol

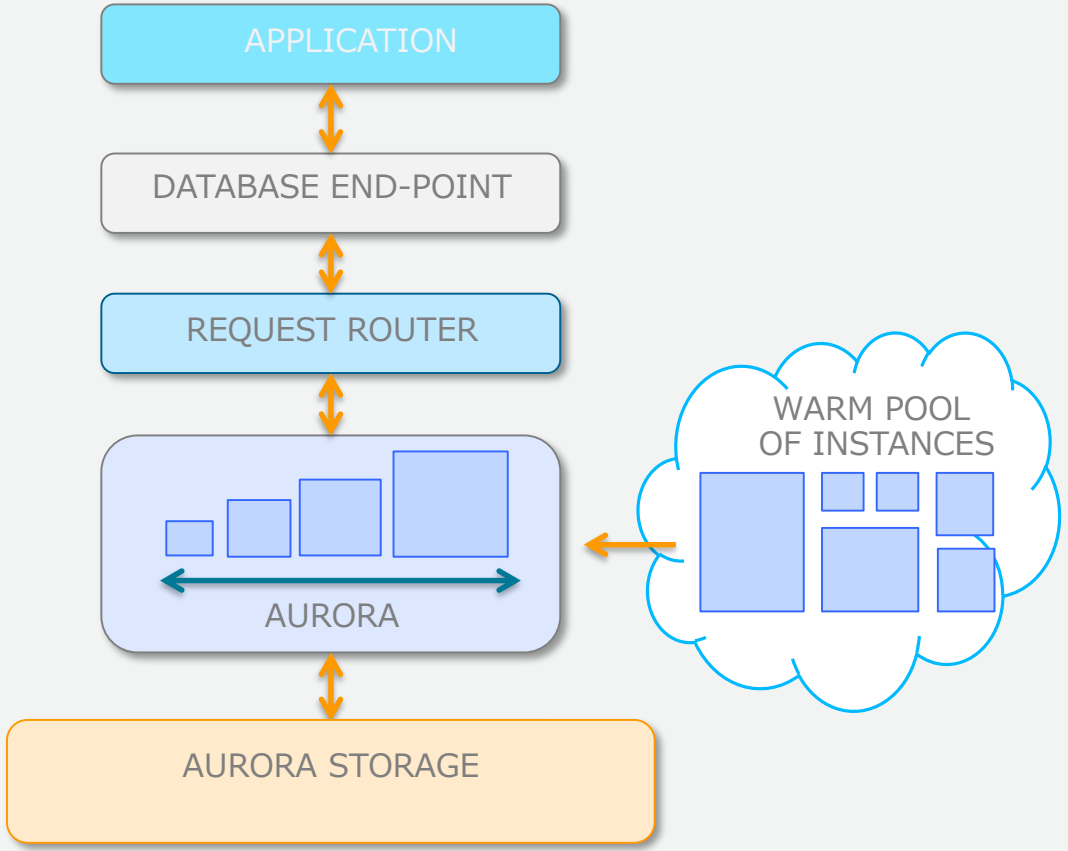
Aurora Serverless

特徴

- コンピューティングキャパシティーが自動管理される Amazon Aurora
  - オンデマンドで起動
  - 利用状況に応じたにスケーリング
  - 利用されていない場合は自動停止
- スケール時のクライアント接続の中断なし
- 秒課金(ただし、1分が最低利用料金)

ユースケース

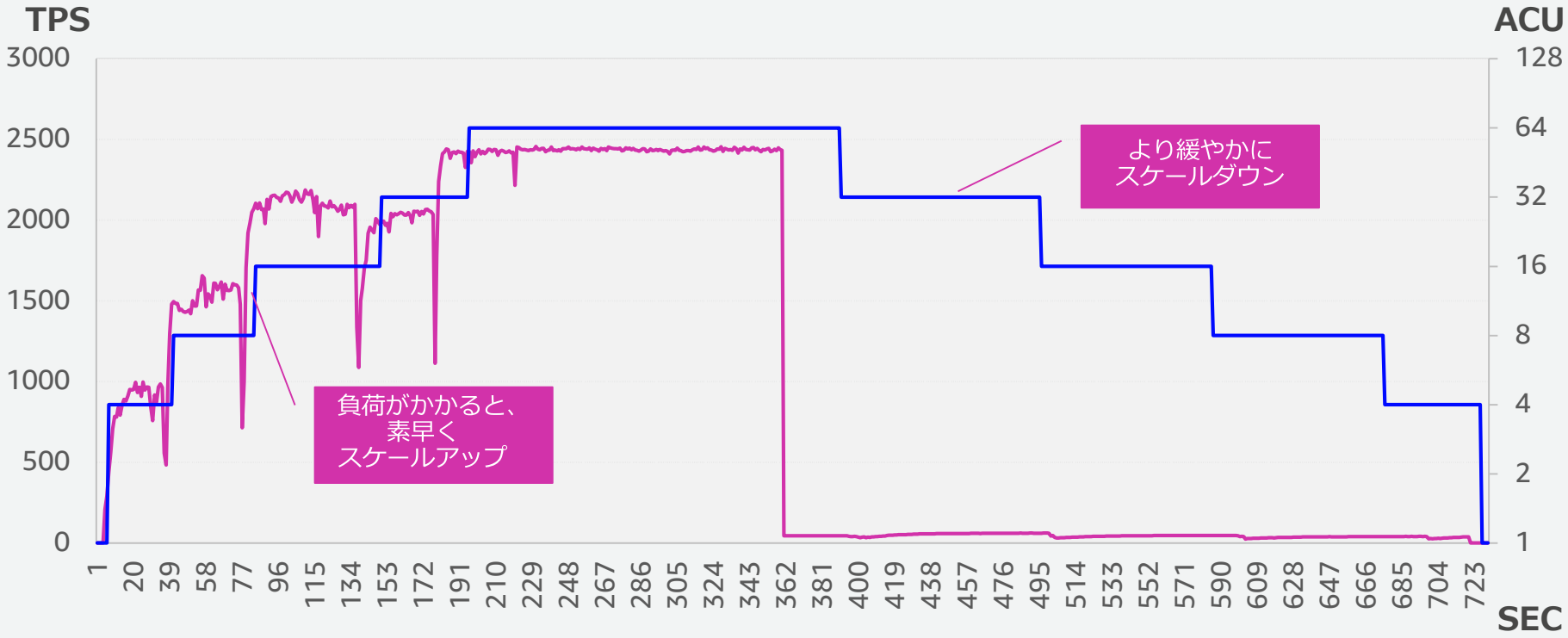
- 不定期利用のアプリケーション
- 開発・テスト用途
- リクエスト予測が難しい場合
  ※定常的にリクエストが予測できる場合は通常のAurora(*)の利用がおすすめ



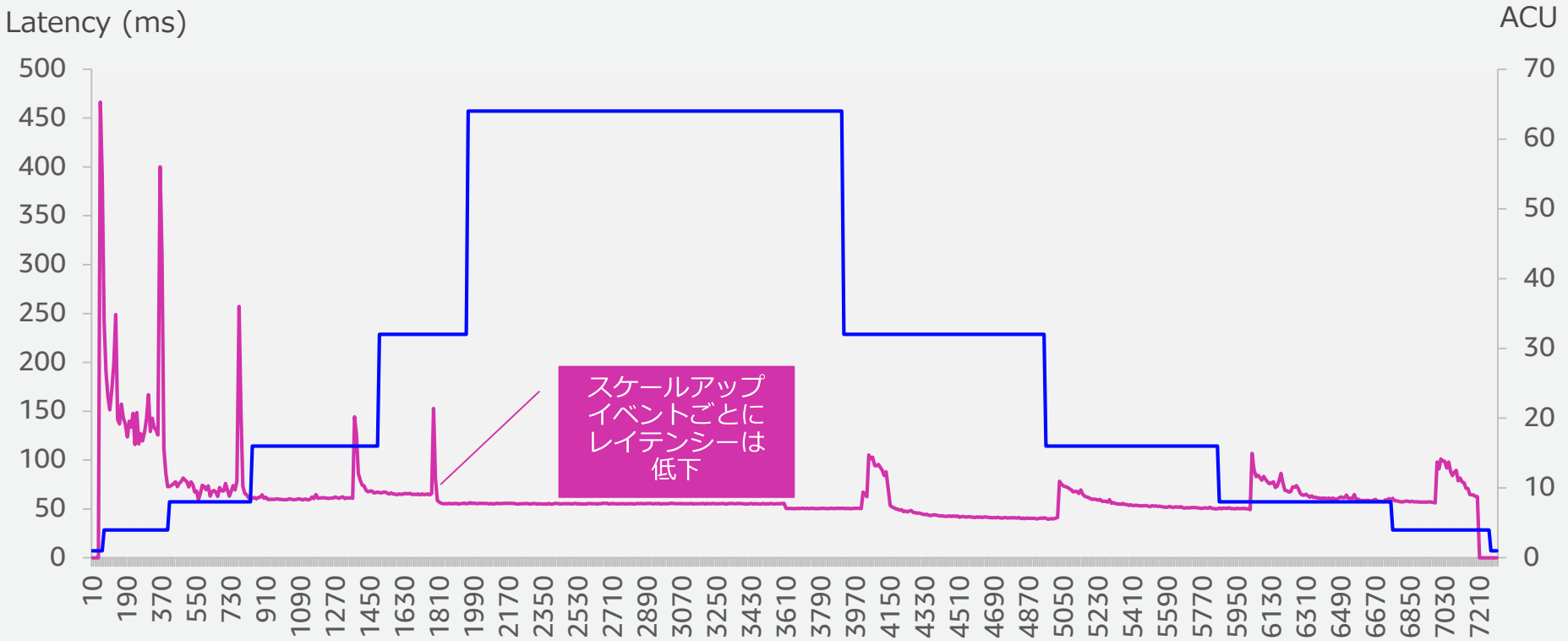
(*)通常のAurora(サーバーレスではないAurora)をprovisioned DB cluster と呼びます

詳細：https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/AuroraUserGuide/aurora-serverless.html

Aurora Serverless: 負荷に応じたスケールアップ・ダウン



Aurora Serverless: レイテンシーを即座に安定化



Amazon Aurora / RDS for PostgreSQL: IAM認証のサポート

- 通常のパスワードではなく、接続前にIAMに権限に基づいて認証トークン作成を依頼し、その認証トークンをパスワードの代わりに使用
- 認証トークンの有効期限は15分
- データベースクラスターの１秒あたりの最大接続数は、インスタンスタイプとワークロードに応じて制限

```
GRANT rds_iam TO foo;
$ export PGPASSWORD="$(aws rds generate-db-auth-token ¥
> --hostname $HOST --port 5432 --region $REGION --username foo)"
$ psql "host=$HOST port=5432  dbname=$DBNAME user=foo ¥
> sslmode=verify-full sslrootcert=certificateFile"
```

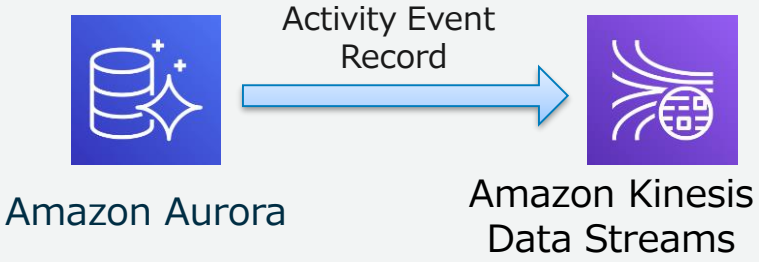
厳密なパスワード管理

- 機能概要
  - ロール/ユーザに対するパスワードを誰が変更可能かを制御する機能
  - rds_password ロールを持つメンバのみをパスワード変更とする
  - パスワード要件(有効期限など)の設定を可能に
- 利用の仕方
  - rds.restrict_password_commands = 1 に設定(パラメータグループ)
  - rds_password ロール
- 注意点
  - RDS/Aurora PostgreSQL 10.6以降で利用可能
  - rds_superuser は rds_passwordロールを常に持つ

詳細：https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/UserGuide/Appendix.PostgreSQL.CommonDBATasks.html#Appendix.PostgreSQL.CommonDBATasks.RestrictPasswordMgmt

Database Activity Stream による リアルタイム監視

- データベースアクティビティをニアリアルタイムで Amazon Kinesis Data Stream へプッシュ
  - SQL コマンド、接続情報などがプッシュされる
  - DBアクティビティ情報は暗号化されている
  - 以下のパートナー製品が本機能を利用した監査に対応
    - SecureSphere Database Audit and Protection(Imperva)
    - Data Center Security Suite(McAfee)
    - Infosphere Guardium(IBM)

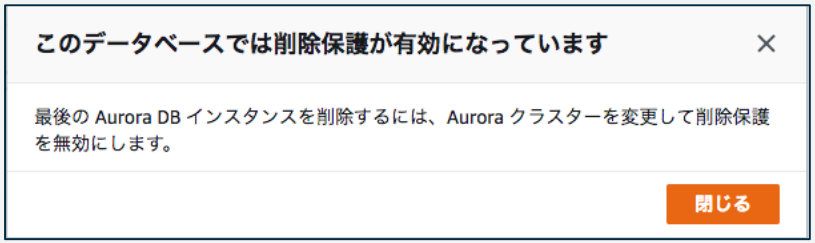


詳細： https://aws.amazon.com/jp/about-aws/whats-new/2019/05/amazon-aurora-with-postgresql-compatibility-supports-database-activity-streams/

Deletion Protection(削除保護)

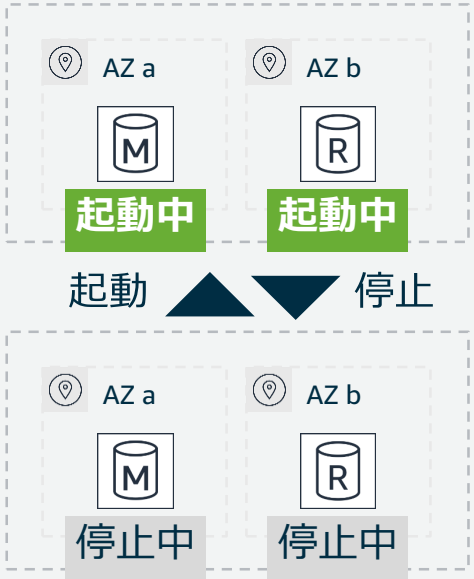
- 削除保護フラグを設定することで、
  オペミスによるDB クラスターの削除を防止することが可能
  - クラスターを削除(クラスター内に存在している最後のインスタンスを削除)しようとしても出来ないように設定

- 削除保護が有効な場合、インスタンスの削除リクエストはブロックされるため、インスタンスの削除リクエスト前に削除保護の無効化が必要



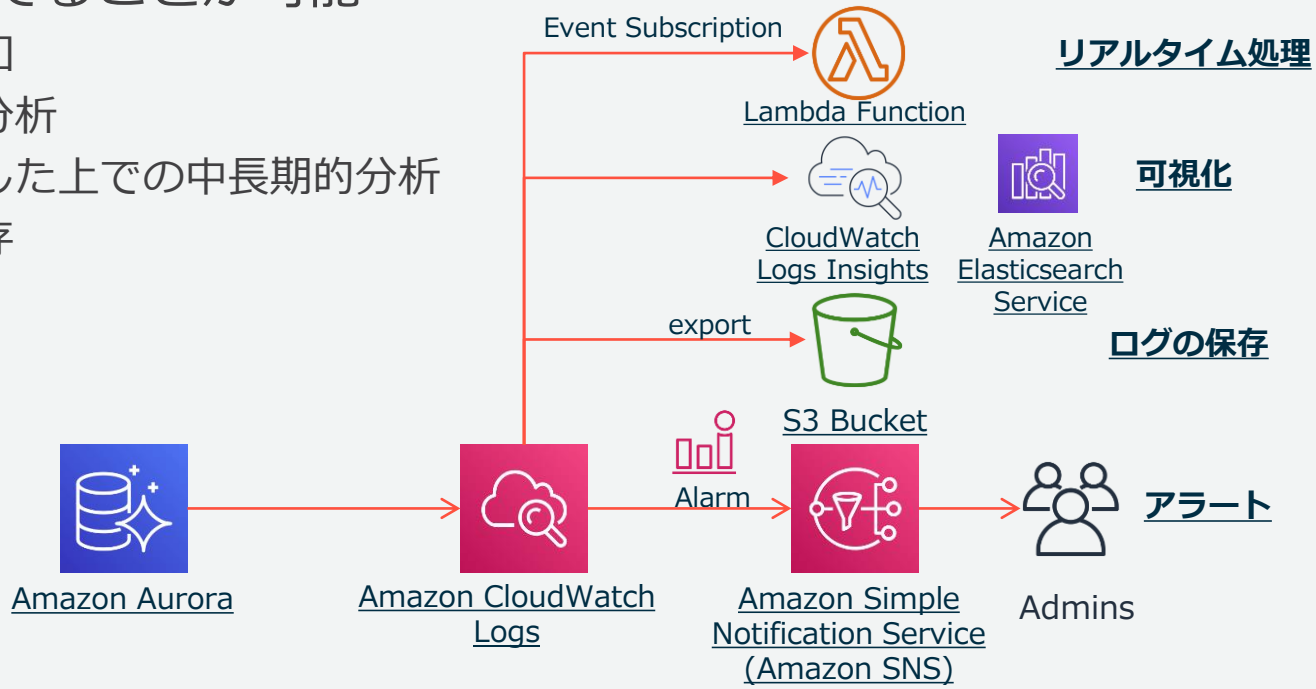
Amazon Aurora: クラスターの停止および起動をサポート

- Auroraクラスターを停止/起動可能に
- クラスターを停止すると、
  プライマリインスタンスと
  すべてのレプリカインスタンスが停止
- 停止するとインスタンス料金は
  課金されなくなるが、
  ストレージ料金は課金される
- 1回に最大7日間まで停止可能。
  7日後に自動起動



CloudWatch Logs との連携

- 出力されたログデータを Amazon CloudWatch Logs へ発行し、以下のタスクに役立てることが可能
  - アラームの通知
  - リアルタイム分析
  - ログを可視化した上での中長期的分析
  - ログの長期保存



詳細: https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/AuroraUserGuide/AuroraPostgreSQL.CloudWatch.html

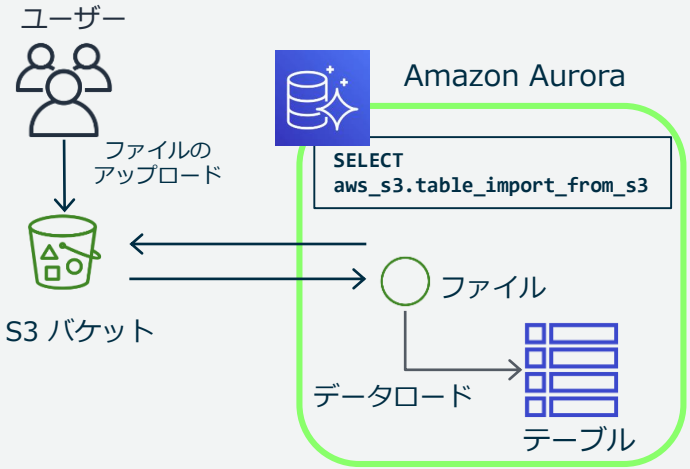
CloudWatch Logs：特定文字列検出のためのフィルター

- ログデータから特定の文字列のフィルタリングが可能
- 正規表現の利用ができない点に注意
- 一致したパターンに応じた処理が可能
  - 特定文字列の出力頻度により
    アラーム通知(メトリクスフィルタ)
  - Lambda関数を実行/
    Elasticsearch Service連携
    (サブスクリプションフィルタ)

- ユースケース
  - メッセージレベルに応じたアラーム
  - 事前に想定されるパターンに応じて
    自動処理

Load Data From S3

- S3バケットに保存されたデータを直接Aurora PostgreSQLにインポート可能
  - aws_s3 という拡張モジュールを使用
  - PostgreSQL の COPY文でサポートされるファイル形式(csvなど)が利用可能
  - RDS/Aurora に付与した IAM ロールを通じて S3 へのアクセスを許可
  - 大量データのインポート、移行などのユースケースに対応




```
=> CREATE EXTENSION aws_s3 CASCADE;
NOTICE:  installing required extension "aws_commons"
CREATE EXTENSION
=> SELECT aws_s3.table_import_from_s3(
     'aws_s3', '',
     '(format csv)',
     '(mys3bucket,path/myfile.csv,ap-northeast-1)');
                    table_import_from_s3
----------------------------------------------------------------------
30 rows imported into relation "aws_s3" from file path/myfile.csv of 912 bytes
```


詳細：https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/AuroraUserGuide/AuroraPostgreSQL.Migrating.html#USER_PostgreSQL.S3Import

New Instance Type

- R5, T3 インスタンスをサポート
  - R5: メモリ負荷の高いデータベースワークロードに最適なインスタンスタイプ
  - T3: テスト、開発などの小規模ワークロードをコスト効率よく実行
    ベースラインパフォーマンス、バースト機能について事前に確認

DBインスタンスタイプ	vCPU	メモリ(GiB)	ネットワーク
db.r5.24xlarge	96	768	25 Gbps
db.r5.12xlarge	48	384	10 Gbps
db.r5.4xlarge	16	128	最大 10 Gbps
db.r5.2xlarge	8	64	最大 10 Gbps
db.r5.xlarge	4	32	最大 10 Gbps
db.r5.large	2	16	最大 10 Gbps
db.t3.medium	2	4	最大 5 Gbps

さいごに

まとめ

- Amazon Aurora はクラウド時代に Amazon が再設計したRDBMS
  - スケーラブル、高い堅牢性、可用性をもつ設計
  - PostgreSQL 9.6, 10 との強い互換性

- 高可用性・実環境での性能向上を実現するための
  多くのチャレンジ/改善を継続して実行中
  - 高いスループット
  - 継続して新機能をリリース

Q&A

お答えできなかったご質問については
AWS Japan Blog 「https://aws.amazon.com/jp/blogs/news/」にて
後日掲載します。

AWS の日本語資料の場所「AWS 資料」で検索

https://amzn.to/JPArchive

ご視聴ありがとうございました

AWS 公式 Webinar
https://amzn.to/JPWebinar

過去資料
https://amzn.to/JPArchive