PGECons
PostgreSQL Enterprise Consortium

2024年度WG3活動成果報告
Amazon Aurora PostgreSQL Limitless Database 検証
～机上調査結果～

PostgreSQL エンタープライズコンソーシアム
WG3 パブリッククラウド検証チーム

責任範囲

- 本資料は、PGEConsが独自に検証した結果であり、
  結果はPGEConsの責任の元、公開しています。

Contents

- 全体概要
- サービス概要
  - Amazon Aurora PostgreSQL
  - Amazon Aurora Serverless v2
- Amazon Aurora PostgreSQL Limitless Database の紹介
  - 概要
  - アーキテクチャ
  - テーブル設計
  - 特徴(非機能観点)
  - ユースケース
- 非機能要求の達成度

全体概要

Amazon Aurora PostgreSQL

- Amazon Relational Database Service（以降、RDSとする）に続き Amazon Aurora PostgreSQL（以降、Auroraとする）はAWSにおいて高性能なマネージドRDBサービスである
- 2024年12月時点で以下のオプションが選択可能
  - Amazon Aurora Serverless v2（以降、Serverless v2とする）
    - 需要に応じて自動でスケールアップ/スケールダウン
    - ※ Serverless v1も存在するが、2025年3月にEOS
  - Amazon Aurora PostgreSQL Limitless Database（以降、Limitless Databaseとする）
    - 分散DB化による書き込み処理の自動スケールアウト/スケールイン
    - 選択できるスペックの上限が極めて高い
    - 2024年10月に一般提供が開始された、比較的新しいオプション

【参考】Amazon Aurora DSQL

- マルチリージョンでの分散DB化による自動スケールアウト/スケールイン
- 2025年1月時点でプレビュー段階

サービス概要

Aurora

- PostgreSQL互換をもち、ACID準拠の完全マネージドRDBサービス
  - 独自のアーキテクチャ(コンピューティング層とストレージ層の分離など)とすることで、通常のPostgreSQLよりも高いスループットを実現
- 需要が予測できる安定したワークロードに適している
  - 読み取り処理はリードレプリカをスケールアウトさせることで、書き込み処理はインスタンスをスケールアップすることで拡張可能だが、自動拡張の仕組みはサービス標準にはなく、拡張処理にも一定の時間を要する
    - とくにスケールアップ/スケールダウンを行うには、システム停止を伴うインスタンスクラスの手動変更が必要となる

Serverless v2

- 需要に応じて無停止でコンピューティング性能を増減させることができるAuroraのオプション機能の1つ
  - 処理性能はAurora Capacity Unit (ACU) と呼ばれるメモリ・CPU・ネットワークの組み合わせの容量単位で決まり、最小ACUから最大ACUの中で自動でスケールアップ/スケールダウンされる
  - Aurora同様、コンピューティング層とストレージ層が分離されているが、Auroraとはコンピューティング層が異なる
  - スケーリング時に増減するのは常にコンピューティング層であり、ストレージ層は変化しない
- システム負荷に合わせて、あらかじめ定めた範囲内で動的にリソース拡張が可能

両者の課題

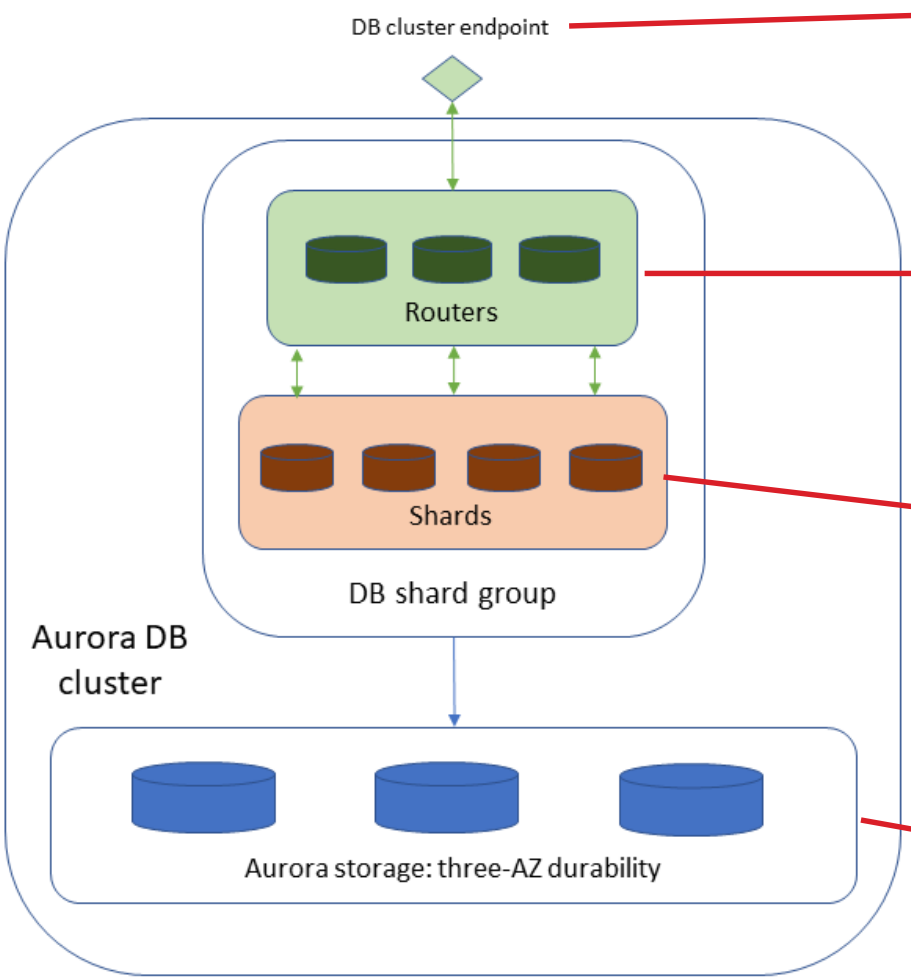
- 書き込み処理の拡張に課題がある
  - Aurora
    - 書き込み処理のスケールアウトに対応していない
    - 書き込み処理のスケールアップ時にはダウンタイムが発生する
  - Serverless v2
    - 書き込み処理のスケールアウトに対応していない
    - 書き込み処理のスケールアップは無停止で可能だが、拡張上限は低い
      - ACUの上限が256(搭載メモリサイズに換算すると512GiB相当)であり、メモリ1TiBクラスのインスタンスを提供するAuroraよりスペック上限値は低くなる
- 2つの課題を解消したサービス＝Limitless Database
  - スケールアウト
    - 分散DB化により書き込み処理のスケールアウトに対応
  - スケールアップ
    - 無停止でスケールアップ可能
    - スペック上限が極めて高い

Amazon Aurora PostgreSQL Limitless Database の紹介

Limitless Databaseの概要

- AuroraおよびServerless v2の書き込み処理における課題を解消したサービスで、以下2軸での拡張が可能
  - スケールアウト
    - 分散DB化（シャーディング）により大規模な書き込み処理が可能
  - スケールアップ
    - Serverless v2同様、ACUにより処理性能が決まる
    - 最大で6,144ACUまで設定可能（Serverless v2の最大24倍）であるため、拡張上限が極めて高い
- シャーディングはマネージドで行われる
  - ハッシュ値をもとにデータが各シャードに振り分けられる
  - 設計者はシャーディングの実装を行う必要がない

Limitless Databaseのアーキテクチャ



DB クラスターエンドポイント
クライアントからの接続先。

Aurora DB クラスター

DB シャードグループ

ルーター
クライアントからの接続を受け付け、受け取った
SQLコマンドを解析して対応するシャードに転送する。
全シャードから返ってきた結果を集約して
クライアントに結果を返却する。

シャード
それぞれがDBのデータのサブセットを保存する
Auroraインスタンスである。
ルーターからのクエリ処理要求を受け、クエリを
実行する。
クライアントから直接接続することはできない。

ストレージ
3つのAZでデータを永続的に保管する。

出典: https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/limitless-architecture.html

Limitless Databaseにおけるテーブル設計①

- データを格納する3種類のテーブルが存在する
  - シャードテーブル
    - 複数のシャードに分散されるテーブル
    - データは、テーブル内の指定された列(シャードキー)の値に基づいてシャード間で分割される
    - 頻繁に更新されるトランザクションデータに使用
  - リファレンステーブル
    - すべてのシャードに複製されるテーブル
      - リファレンステーブルとシャードテーブル間の結合クエリを単一シャード上で実行できるため、シャードとルーター間の不要なデータ移動がなくなる
    - 頻繁に変更されないマスタデータに使用
  - 標準テーブル
    - 単一シャード上に格納されるテーブル
    - デフォルトのテーブルタイプ

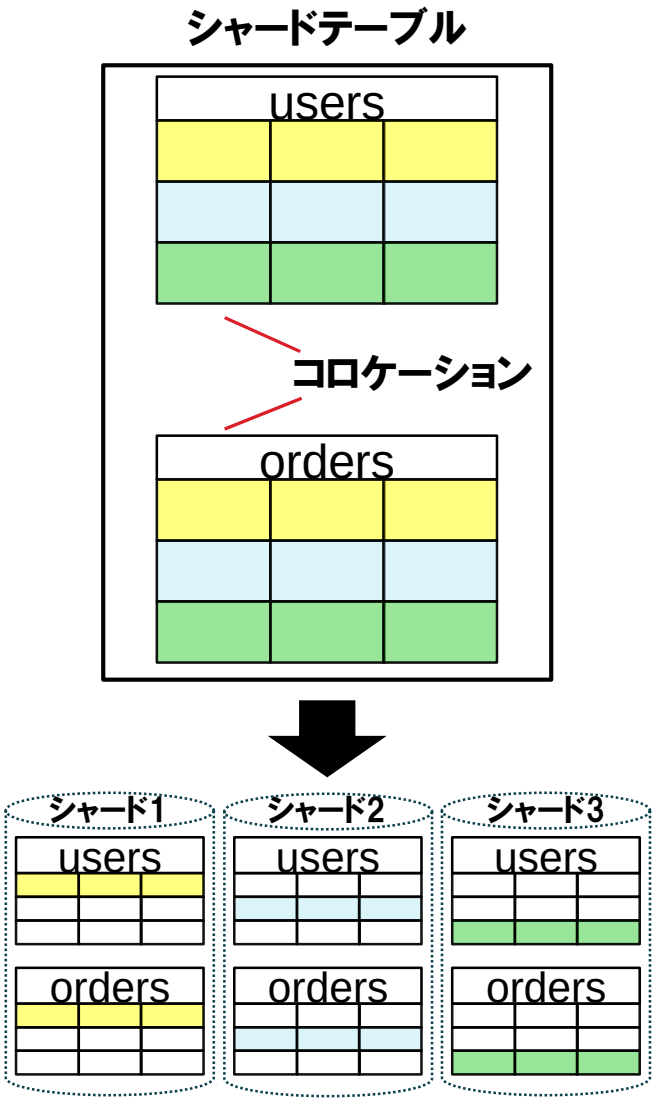
Limitless Databaseにおけるテーブル設計②

- シャードテーブル
  - 例：シャードキー user_id を使用して、シャードテーブル users を作成

```
BEGIN;
SET LOCAL rds_aurora.limitless_create_table_mode='sharded';
SET LOCAL rds_aurora.limitless_create_table_shard_key='{"user_id"}';
CREATE TABLE users(user_id BIGINT, user_name VARCHAR);
COMMIT;
```

  - 例：シャードキー user_id を使用して、シャードテーブル orders を作成
    - users テーブルとコロケーションすることで、同じシャードキーのデータは常に同一シャードに配置される
    - シャードキーで結合するクエリを単一シャード上で実行でき、シャードとルーター間の不要なデータ移動がなくなる

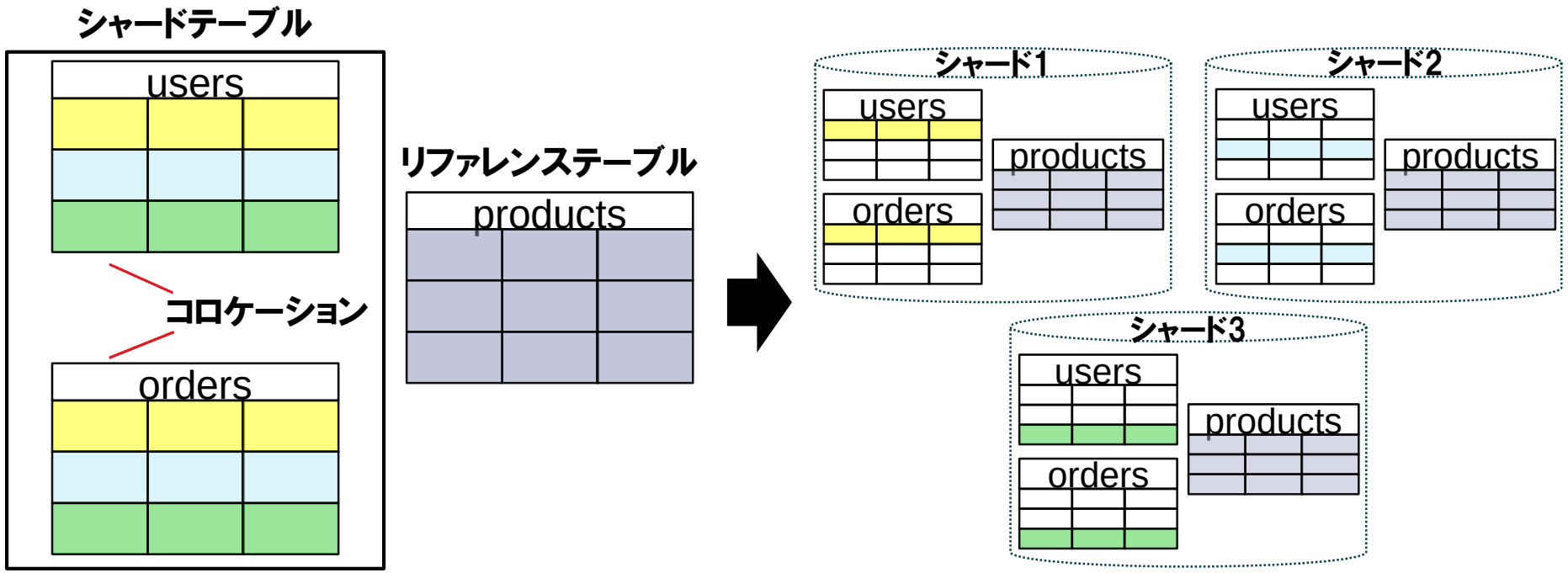
```
BEGIN;
SET LOCAL rds_aurora.limitless_create_table_mode='sharded';
SET LOCAL rds_aurora.limitless_create_table_shard_key='{"user_id"}';
SET LOCAL rds_aurora.limitless_create_table_collocate_with='users';
CREATE TABLE orders(order_id BIGINT, user_id BIGINT, product_id BIGINT);
COMMIT;
```



Limitless Databaseにおけるテーブル設計③

- リファレンステーブル
  - 例:リファレンステーブル products を作成
    - 全シャードに複製されるため、リファレンステーブルとシャードテーブル間の結合クエリを**単一シャード上で実行**でき、シャードとルーター間の不要なデータ移動がなくなる

```
BEGIN;
SET LOCAL rds_aurora.limitless_create_table_mode='reference';
CREATE TABLE products(product_id BIGINT primary key, product_name VARCHAR);
COMMIT;
```



Limitless Databaseの特徴（非機能観点）

可用性

- DBシャードグループごとに1つまたは2つのスタンバイを異なるAZに配置する設定が可能
  - フェイルオーバー時の切替時間は明記なし
- リージョン間での多重化には対応していないため、リージョン単位での障害対策は独自に行う必要がある

Limitless Databaseの特徴（非機能観点）

監視

- CloudWatch メトリクス
  - DBシャードグループごと・DBシャードグループのインスタンスごと などで、ACU・接続数・各種スループットなどのメトリクスをモニタリング
  - Auroraと同様、CloudWatch メトリクスを確認できるが、Auroraとは一部着目するメトリクスが異なる
    - CPUUtilization（Aurora） → ACUUtilization（Limitless Database） など
- CloudWatch Logs
  - PostgreSQLログ、拡張モニタリングのメトリクスが配信される
  - 拡張モニタリングの有効化およびPostgreSQLログのエクスポート設定が必須
- Performance Insights（PI）
  - 以下のメトリクスをDBシャードグループレベルでモニタリング
    - データベース負荷：平均アクティブセッションが取得できる
    - 最大CPU：DBで使用できる最大計算能力で、アクティブセッションが最大CPUを超えているかを確認できる
  - PIの保持期間を最低31日とする設定が必須

Limitless Databaseの特徴（非機能観点）

制約

- サポートされているPostgreSQL拡張機能は限定的
  - shared_preload_libraries設定を使用して拡張機能を
    Limitless Databaseにロードしても正しく機能しない場合がある
- Auroraの機能の一部がサポートされていない
  - Amazon RDS ブルー/グリーンデプロイ
  - Amazon RDS Proxy
  - Amazon Aurora Global Database
  - リードレプリカ
  - など

Limitless Databaseのユースケース

- 高い書き込み性能が必要なケース
  - 金融取引プラットフォーム
  - ECサイトの決済・在庫処理基盤
    - 大量同時トランザクションかつトランザクションの一貫性が求められるシナリオ
- レスポンスタイムの要求が厳しい場合、低レイテンシが求められる場合などは、AuroraやServerless v2のほうが適していることもある
  - Limitless Databaseはルーター・シャード間の通信が発生するため、レイテンシが高くなる場合がある
- これらの性能を確認するための実機検証を今回行った

非機能要求の達成度

概要

- 以下のサービスについて非機能要求の達成度の比較を行う
  - Aurora
  - Limitless Database
- 別紙成果物では、上記の2つにRDSとServerless v2を加えた4つのサービスの比較を行った
  - 本資料ではリンクの記載を一部除外しているが、別紙成果物ではリンクを含む完全版を記載している

可用性①

項目	比較の観点	Aurora	Limitless Database
継続性	復元可能な直近の時間は何分前か	・別途独自に継続的かつ増分的なバックアップをとっており、最も直近の復元時点は通常、現在時間の 5 分以内までである。 ※WALアーカイブを用いたPITR、とは明記されていない。 ・即時復旧を目的とする場合はMultiAZを用いる点も同様。	・Auroraと同様。
	HA構成における切り替え時間の目安	・MultiAZレプリケーションを組んでいる場合は、1分以内で完了すると規定している。仕組みとしてはスタンバイ側はリードレプリカであるが、リードのみのフラグを外し、接続先エンドポイントの付け替えを行っている。 ・Aurora Global Databaseを用いたリージョンレベルのHA構成もある。Global Databaseの切り替えには、手動/自動（スイッチオーバー）/自動（フェイルオーバー）があるが、データロストを伴わないスイッチオーバーは適用すべき（待機すべき）ログの量に依存し、障害時のフェイルオーバーはDB観点で一般的に数分で完了する。	・設定で「コンピューティング冗長性」にて利用するゾーンに跨った冗長性を0-2で選択することができる。 切り替え時間は明記なし。
	SLAとして年間何%の稼働率を規定しているか	・MultiAZの場合は月間稼働率99.99%（ダウンタイム約4.3分を超えると返金） ・SingleAZの場合は月間稼働率99.9%（ダウンタイム約43分を超えると返金）	

可用性②

項目	比較の観点	Aurora	Limitless Database
耐障害性	HAとしてサーバを多重化する仕組みは何があるか	・MultiAZによる多重化が可能。 ・Aurora Global Databaseの機能により、別リージョンへのDR構成を取ることが可能。	・MultiAZによる多重化が可能。 ・リージョン間の多重化の仕組みはない。
	データの多重化の仕組みとして何があるか	・AZをまたいで複数ストレージに書き込むようになっており、1AZあたり2か所への書き込みを3AZに対し実施する。 ・計6か所のうちQuorumで書きこみ：4、読み込み：3で整合性を担保している。	・クラスターストレージに関しては、Auroraと同様。 ・DBシャードグループに関しては設定「コンピューティング冗長性」にてスタンバイの設定を下記の3点から選択できる。 ・コンピューティング冗長性なし ・1 つのフェイルオーバーターゲットによるコンピューティング冗長性 ・2 つのフェイルオーバーターゲットによるコンピューティング冗長性
災害対策	DRの方式は何が用意されているか	・Amazon Aurora Global Databaseを用いて複数のリージョンにまたがるデータベースを構成できる。 ・DR方式としてはウォームスタンバイ・パイロットライトなどが実現できる。 ・Global Databaseの場合、コスト最適化のために、セカンダリクラスターをヘッドレス構成にしたり、AuroraとServerless v2を混在させる構成も可能。	・リージョンレベルの冗長構成は構成できない。

拡張性①

項目	比較の観点	Aurora	Limitless Database
リソースの特性と拡張性	CPUの特性と拡張性	・Intelインスタンスは、ハイパースレッドで分割されたCPUリソースを1vCPUとしているので、2vCPUで1物理CPUコアに該当する。 ※Gravitonインスタンスはハイパースレッドを使っていない。 ・Auroraのインスタンスタイプにvcpu含めメモリ・最大EBS帯域幅・ネットワーク帯域幅がセットになっている。 ・インスタンスタイプは「メモリ最適化」「バースト可能パフォーマンス」「Optimized Reads」から選択できる。 ・拡張する場合は、インスタンスタイプの変更となる。 ・変更方式は様々考えられるが最もダウンタイムが短くシンプルな方法としては、リードレプリカを拡張後のインスタンスタイプで作成し、フェイルオーバーする方法がある。	・ACUについてはServerless v2と同様。 ・ただしACUの範囲は、16-6144である。 ・ACUの最大値によって、ノード(ルーター・シャード)がスケール(アップ・ダウン)する。 ・ACUの設定だけではスケールは不十分で、ルーターの追加、シャードの分割を必要に応じて行う必要がある。
	メモリの特性と拡張性	同上	CPUと同等
	ストレージの特性と拡張性	・自動でスケールアップする仕様であり、10GiB～128TiBまで10GB単位で自動拡張する。 ・TRUNCATE TABLE、DROP TABLEなどでストレージに空きができると動的にサイズは小さくなる。	・クラスターストレージに関しては、Auroraと同様。

拡張性②

項目	比較の観点	Aurora	Limitless Database
リソースの特性と拡張性	ディスクI/O	・RDSのようなIOPSの制限はなく、確保容量によるIOPSのリミットは受けない。 ・また、Auroraはディスクストライピングという概念があり、10GB単位で分割されたストレージに分散してデータを格納する。この分散がうまく機能するデータ量、データ構造であるほど、高い性能を発揮する。 ・Auroraのバージョンにより、個々のインスタンスが持つローカルストレージがある。具体的なサイズはインスタンスタイプにより決められている。 ・temp領域をローカルのNVMeに配置できたり、共有バッファの二次キャッシュとしてローカルのNVMeに配置できるOptimized Readsは、PostgreSQLバージョン16.1以降のすべて、15.4以降、14.9以降のR6gdインスタンス、R6idインスタンスでデフォルトで利用できる。	・Auroraと同様。
	読み取り専用インスタンスの利用可否	・リードレプリカの作成可能。 ・マスタ障害時にフェイルーオーバ可能。 ・内部的には非同期レプリケーションに近いが、100ミリ秒未満のラグで反映される。	・リードレプリカの作成はできない。
	その他	・クラスターのストレージ設定は「Aurora I/O-Optimized」「Aurora Standard」から選択できる(料金にかかわる違いがあるのみ)。	・クラスターのストレージ設定は「Aurora I/O-Optimized」のみ利用可能。 ・東京リージョン利用可・大阪リージョン利用不可。 ・サポートされていない機能や、一部DDL、DMLの制限がある。

運用・保守性①

項目	比較の観点	Aurora	Limitless Database
バックアップ・リストア	実行可能なバックアップ方式について	・2種類「自動バックアップ」「スナップショット」の方式が利用可能。 ・自動バックアップは、連続的かつ増分。 ・スナップショットは、フルバックアップ。 ・任意のタイミングでスナップショットの取得が可能だが、常時ストリーミングで継続的かつ増分的なバックアップ取得しており、頻繁なスナップショットを取る必要はなく任意の時間に復元できる。 ・Auroraはストレージの構造が特殊であり、スナップショットはストリーミングバックアップのメタデータを取るだけなので高速で取得可能である。 ・論理バックアップ、pg_dumpの実行可能。 ・AWS Backupを用いてバックアップを管理することができる。	・Auroraと異なる点がある。 ・クラスターのスナップショットを手動・定期的に取得した場合には、シャードグループのデータも含まれる。 ・DB クラスター スナップショットのコピーおよび共有が行える。 ・pg_dumpまたはpg_dumpallを用いて、シャードグループ内のデータはバックアップできない。 ・DBクラスター削除時に、最終スナップショットを取得することができる。 ・DBクラスター削除時に、自動バックアップを保持することはサポートされていない。 ・Auroraと同様に、AWS CLIを用いてDBクラスターのスナップショットを作成できる。
	実行可能なリストア方式について	・スナップショットからの復元と、自動バックアップの保持期間の範囲内では任意の時刻にさかのぼり復元可能。 ・いずれも、新たなインスタンスを作成するという動作となり、既存のインスタンスをロールバックさせることはできない。 ・pg_dumpからのpg_restoreも可能。	・Auroraと異なる点がある。 ・互換性のあるエンジンバージョンのスナップショットからのみリストア可能。 ・DBクラスターの手動スナップショットからDBクラスターをリストアするとストレージ全体が対象、シャードグループも含まれる、ストレージにアクセスするにはシャードグループの作成が必要。 ・ポイントインタイムリカバリ（PITR）を用いての任意の期間内のリストアも可能、手動の場合と同様。 ・リストアする際は、拡張モニタリングとパフォーマンスインサイトを有効化する必要がある。 ・AWS CLIを用いて、DBクラスターのリストアとシャードグループの作成を行える。 ・pg_dump、pg_dumpall、pg_restoreを用いてクラスター及びシャードグループのサポートはされていない。

運用・保守性②

項目	比較の観点	Aurora	Limitless Database
バックアップ・リストア	自動バックアップの保存期間	・デフォルトでは7日間、最大35日間まで設定可能。	・Auroraと同様。
	自動バックアップによる復旧可能時点	・自動バックアップによるベースバックアップは1日1回。 ・AWS Backupを用いるPITRは最長35日前まで指定可能。	・Auroraと同様。
	バックアップ処理時間の特性	・スナップショットの取得時間は、取得時のクラスタストレージの容量に依存する。	・Auroraと同様。 ・各シャード間で整合性を持ったバックアップを取得できる。
	リストア処理時間の特性	・データはディスクストライピングにより分散保持されているため、復元時も分散処理が可能である。そのため、スナップショットの容量、復元時点の容量には依存しない。 ・スナップショット取得時間は、取得時のスナップショットの容量に依存する。	・リストアは多くのルーター、シャードのリストアを含む場合があり、Auroraより多くの時間がかかる可能性がある。

運用・保守性③

項目	比較の観点	Aurora	Limitless Database
障害監視	障害監視の方式について	・関連リソースのイベントが発行される、このイベントをユーザにより取得し監視を行う。 ※別途アプリケーション観点での監視が必要。	・Auroraと同様。
	障害監視の監視項目について	・関連リソースのイベントは下記の項目で行われる。 ・MultiAZを組んでいる場合は、以下のイベントでフェイルオーバーが自動的に発動する。 ・プライマリ利用可能ゾーンの可用性損失 ・プライマリに対するネットワーク接続の喪失 ・プライマリ上でのコンピュートユニット障害 ・プライマリへのストレージ不良	・基本的にAuroraと同様だが、コンソールからshard group内の内部クラスターのログが確認できないため、CloudWatch Logsから確認する必要がある。
	障害監視の通報方式について	・ユーザにて方式を検討する、イベントの通知を、Amazon SNSを介してメール・HTTPエンドポイントの呼び出しなどができる。	・Auroraと同様。

運用・保守性④

項目	比較の観点	Aurora	Limitless Database
性能管理（監視）	性能監視の方式について	・インスタンスの作成や修正時に「拡張モニタリング」と「Performance Insights」をそれぞれ有効にしておくことで、CloudWatch上にデータが送られ任意のメトリクスを表示させることが可能となる。 ・データベースフリートとしてのモニタリングとしてAmazon CloudWatch Database Insightsも利用可能。CloudWatchの専用画面で確認可能。 ・DevOps Guru for Auroraを用いて機械学習により具体的なボトルネックの診断・対策の推奨がなされる。	・CloudWatch メトリクスにシャードグループについてのメトリクスがレポートされる。 ・CloudWatch logsにクラスターとシャードグループのログが出力される。 ・拡張モニタリングを有効化が必要、OSレイヤの項目のメトリクスを取得できる。
	性能監視の監視項目について	・拡張モニタリングはいわゆるOS関連のメトリクスを収集可能であり、Performance Insightsではアクティブなセッション数やセッションによるCPU負荷の合計、各セッションのSQLステートメントなどを表示させることができる。	・Auroraと同様にPerformance Insightsを用いてクラスターのメトリクス、またシャードグループレベルで取得できる。
	性能監視の通報方式について	・CloudWatchにて、特定のメトリクスに対し閾値を設定すれば、値を超えた際にアラームをAmazon SNSメッセージで送信させることができる。	・Auroraと同様。

運用・保守性⑤

項目	比較の観点	Aurora	Limitless Database
保守運用	計画メンテなどについて	・関連コンポーネントのアップデートのため、計画再起動がなされることがある。 ・インスタンスの再起動により内部のコンポーネントがアップデート、メンテナンスされたものに置き換わるという運用になっている。 ・予告もある上、再起動のタイミングをずらすこともできるが、永久的に回避することは不可能である。	・Auroraと同様。
	運用保守向けにはどのようなツールが用意されているか	・データベースクラスターの運用監視としてはCloudWatchに集約されている。 ・CloudWatch Logs等でAWS全体のログを収集監視しつつ、得られた情報をもとにどのように保守するかはPJ運用次第である。 ・別途「AWS Trusted Advisor」というサービスで、AWSがユーザへのサービス提供の中で積み上げたベストプラクティスに基づくAWS環境の検査、推奨事項がある場合のガイドを受けることができる。 ・「データベースアクティビティストリーム」を用いて、データベース操作のモニタリングを行い監査に利用できる。	・Auroraと同様。 ・セキュリティの脅威検知としてGuardDuty RDS Protectionを用いることができる。 ・待機イベントは専用のものが下記のものがある。 https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/limitless-monitoring-waits.html ・専用の関数及びビューは下記のものがある。 https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/limitless-monitoring-fns-views.html
	PostgreSQLに関するパッチ情報の展開とパッチ適用のポリシー	・パッチ適用は自動的には行われない。 ・クラスターのメンテナンスタスクにパッチ適用のタスクが通知される。 ・パッチの適用はユーザにて行う。 ・条件によりZDPにて行うこともでき、適用中はベストエフォートでクライアント接続を維持する、約1分間スループットが低下する。	・基本的にはAuroraと同様だが、ZDPは対応していない

運用・保守性⑥

項目	比較の観点	Aurora	Limitless Database
バージョン管理	バージョンアップはどのように実施するか、または自動で実施されるか	●マイナーバージョン ・設定［マイナーバージョン自動アップグレードの有効化］により実行のタイミングを選択できる。 ・実行中はダウンタイムの可能性がある。 ・（パッチ適用と同様に）条件によりZDPを利用することができダウンタイムを軽減できる。 ・ZDP利用時は、適用中はベストエフォートでクライアント接続を維持する、約1分間スループットが低下する。 ※一般的にマイナーバージョンでもテストを行うことが推奨される。 ●メジャーバージョン ・自動で行われない。 ・下位互換のないDBの機能がある可能性があるため、事前に実地計画・テストを行う。 ・Amazon RDS Blue/Green Deploymentを利用することも推奨されている。	・AWSマネジメントコンソールまたはAWS CLIにて実施できる。 ・2024年12月時点で、メジャーバージョンのアップグレードはサポートされていない。
	ライフサイクル管理とEOSについて	・FAQ上では、メジャーVerはリリースから3年以上、マイナーVerは1年以上はサポートすると規定されている。 ・そこから先は不定であるが、EOSについては6か月以上の猶予期間を持って通知される。 ・バージョンはリリースカレンダーに従い計画的に利用する。 ・より長期に安定したマイナーバージョンを利用したい場合はLTSバージョンを利用する。 コミュニティサポートが終了したメジャーバージョンに対して追加で最大3年のサポートを提供する延長サポート（有償）も存在する。	・マイナーバージョンのアップグレードがサポートされている。 ・パッチ適用がサポートされている。 ・リリースポリシーについてはAuroraと同様。
	コミュニティ版PostgreSQLのリリースにどれだけ追随できているか	・メジャーバージョン・マイナーバージョンのいずれもコミュニティリリース日より数カ月以内にAuroraリリース日が設定されている。	

運用・保守性⑦

項目	比較の観点	Aurora	Limitless Database
サポート体制	用意されているサポート契約の種類とその内容、問い合わせ方法などについて	・利用量の10%程でサポート契約が可能であり、開発者/ビジネス/エンタープライズでレベルが分かれ、それぞれサポートレベルと下限となる料金が異なる。 ・RDSとAuroraについては「AWS内のサービス」と見なされ、ビジネス以上から提供されるサードパーティーSWのサポートの扱いではなく、開発者サポートから問い合わせが可能となっている。	・リリースサイクルや延長サポートの扱いはAuroraと同様。
データロード	実行可能なデータロードの方式について	・COPYコマンドはクライアントサイドにデータを用意するパターンで実行可能。 ・AmazonS3の保存されたデータをクラスターにロードすることができる。 ・AWS Database Migration Serviceも利用可能。	・PostgreSQLのCOPYコマンドを用いてローカルなどからロードできる。 ・ RDS for PostgreSQL から直接データをロードできるツールが用意されている。 ・RDS/Auroraからフルロード+CDCでロード可能なツールが用意されている。

セキュリティ

項目	比較の観点	Aurora	Limitless Database
DB接続制御	From/To でIP指定可能？ ポート指定可能？ SSL接続の強制が可能？ パブリック/プライベートの選択が可能？	・VPCやセキュリティグループなどAWSのNW設定機能でかなり細かく設定ができるため、基本的にはこちらで制御する。 ・pg_hbaは設定できないが、ポートの変更は可能。	
データの秘匿	pgcrypto関数の利用可否やその他の暗号化について	・pgcrypto関数はどちらも利用可能 ・透過的暗号化（TDE）については、Storage暗号化はどちらも利用可能。	
権限管理	権限ロール管理が出来るか？ 新しくロールを追加できるか？ スーパーユーザが利用できるか？	・PostgreSQLのSuperUserは設定/利用できない。 ・rds_superuserを最高権限として、ロール管理、ロール追加が可能。	
不正追跡・監視	セキュリティログの取得	・AWS CloudTrail ログにて、RDS含むAWSというクラウドPFに関するリクエストを記録させることができる。 ・DBアクセスに関してはpostgresql.logにあたるものをDBログとして取得可能。 ・pgAudit/Database Activity Streamも利用可能。	・pgAudit/Database Activity Streamは、2024年12月現在未サポート。
	ログ保管期間	どちらもlog_connections、log_statement等のパラメータを変更し、PostgreSQLログに情報出力は可能。pgauditの導入は未サポート。	
	ログ監視、アラート通知	・CloudTrail、DBログともにCloudWatchに送信しアラート設定することで、Amazon SNSメッセージを通して通知可能。	

コミュニティ版PostgreSQLとの違い/制約事項①

項目	比較の観点	Aurora	Limitless Database
VACUUM/ANALYZE	VACUUM/ANALYZEの実行方法、実行タイミングなど	・ユーザから見える範囲ではコミュニティ版との違いはない。autovacuumもON/OFF設定可能。	・autovacuumの利用可能。 ・Auroraと異なり、vacuumの実行タイミングは時間ベースで決定される。 ・vacuumの実行情報をデータベース統計は専用の関数が用意されている。 ・vacuumはシャードテーブルとリファレンステーブルで実行できる、制限がある関数がある。
インデックスリビルド	REINDEXやpg_repackの利用可否など	・reindex, pg_repackコマンドともに実行可能。	・reindex, pg_repack ともにサポートされていない。インデックスの再構成が必要な場合はDROP/CREATEが必要。
psql	psqlによる接続の可否、利用制約、注意事項など	・デフォルトでエンドポイント（クラスター・リーダー）が払い出される、またカスタムエンドポイントも独自に設定できる。 ・これを指定してつなぎにいけばコミュニティ版と特に変わりなく使える。 ・SSL強制などの制約もない。	・クラスターエンドポイントをが払い出される。 ・これを指定し接続を行う。
アプリケーション接続	アプリケーションからの接続方法、接続用ドライバ等	・エンドポイントURLを指定する以外、コミュニティ版PostgreSQLとの違いは特にない。 ・Javaドライバに関しては、AWSより「AWS JDBC Driver for PostgreSQL」が提供されておりこのラッパーと合わせることで、フェイルオーバの改善・高速化、IAMやAWS SecretsManagerと連携した認証機能が利用できる。 ・フェイルオーバーの高速化などであればRDS Proxyの利用も可能。	・エンドポイントのRoute53の負荷分散機能が制限されているため、その解決として下記のドライバのラッパーが必要となる。 https://github.com/aws/aws-advanced-jdbc-wrapper/blob/main/docs/using-the-jdbc-driver/using-plugins/UsingTheLimitlessConnectionPlugin.md ・2024年12月現在、下記を含む一部機能がサポートされていない点に注意。 Amazon ElastiCache Amazon RDS Proxy Aurora recommendations AWS Lambda integration AWS Secrets Manager

コミュニティ版PostgreSQLとの違い/制約事項②

項目	比較の観点	Aurora	Limitless Database
パラメータ設定	設定方法	・AWSポータル上でGUIで設定する。永続化にはパラメータグループを用いる。 ・制約のないものはSETコマンドで設定可能。 ・ALTER SYSTEMコマンドは利用不可。	・パラメータグループは、Auroraと同様に利用できる。 ・独自のクラスターパラメータがある。 ・SETはサポートされている。 ・ALTER SYSTEMコマンドはサポートされていない。
	コミュニティ版との違い/制約事項	・RDSと同様、大体のパラメータは変更可能であるが、以下の特色が適用されるパラメータが存在するとともに、クラスター単位とインスタンス単位で分類されている。 ・クラスターパラメータはAuroraのクラスター単位で統一されるパラメータとなり、インスタンスパラメータは、クラスター内のインスタンスにおいて個別に設定可能なパラメータとなる。 ①変更できないもの（WALアーカイブ関連など） ②インスタンスのスペックに応じて自動で設定してくれるもの（shared_buffersなど） ③RDS独自のパラメータ（rds.で始まる） ・この中で②については、それが変更できるものであれば任意の値を設定することもできる。 ・また、①であるが、③に連動して変更されるため、③を通してのみ変更が可能といったパラメータも存在する。 ・詳細はWeb上にリファレンスがあるためそちらで確認する、または管理画面から変更可否や設定範囲を確認できる。 ・クラスターとインスタンスに関する具体的なパラメータは下記より確認する。 https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/AuroraUserGuide/AuroraPostgreSQL.Reference.ParameterGroups.html	・Limitless用のデータベース変数 https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/limitless-reference.variables.html ・Limitless用のクラスターパラメータ https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/limitless-reference.DBCparams.html

コミュニティ版PostgreSQLとの違い/制約事項③

項目	比較の観点	Aurora	Limitless Database
EXTENSION	EXTENTIONの対応状況	・2つの方式がある。 ①通常のEXTENSIONの導入 ・利用可能な機能は下記より確認する。 https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/AuroraPostgreSQLReleaseNotes/AuroraPostgreSQL.Extensions.html ・導入の方法は下記を確認する。 https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/AuroraUserGuide/Aurora_delegated_ext.html ②TLEを用いた独自のEXTENSIONの導入 ・導入方法は下記を確認する。 https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/AuroraUserGuide/PostgreSQL_trusted_language_extension.html	・利用可能なEXTENSIONは下記から確認する。 https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/limitless-reference.DDL-limitations.html#limitless-reference.DDL-limitations.Extensions
ファイル構成の変更	PGDATAとWAL保存領域の分離	・分離不可。 ・Auroraはディスクストライピングにより10GB単位でのI/O分散が行われる設計となっており高いIOPSが発揮される。	
エクスポート	postgresql.confやlogなど、どこまでエクスポートが可能か	・postgresql.confは不可。 ・logに関しては下記の項目を出力できる。 https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/AuroraUserGuide/USER_LogAccess.html ・また、設定により監査ログpgAuditも取得できる。	・postgresql.confは不明。 ・logに関してはAuroraと同様、加えてDBシャードグループ・ルーター・ノードについても出力可能。 https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/limitless-monitoring.cwl.html
その他	停止時の休眠措置について	・インスタンスの停止は可能だが、7日以上経つと自動起動される。 ・長期停止させたい場合はスナップショットを取得し、インスタンスは削除する運用が必要。	・停止できない。

ライセンス

本作品はCC-BYライセンスによって許諾されています。ライセンスの内容を知りたい方はこちらでご確認ください。 文書の内容、表記に関する誤り、ご要望、感想等につきましては、PGEConsのサイトを通じてお寄せいただきますようお願いいたします。

- Amazon Web Services、"Powered by Amazon Web Services"ロゴ、Amazon EC2、Amazon S3、Amazon Relational Database Service (Amazon RDS) およびAmazon Auroraは、米国その他の諸国における、Amazon.com, Inc.またはその関連会社の商標です。
- PostgreSQLは、PostgreSQL Community Association of Canadaのカナダにおける登録商標およびその他の国における商標です。
- TPC, TPC Benchmark, TPC-B, TPC-C, TPC-E, tpmC, TPC-H, TPC-DS, QphHは米国Transaction Processing Performance Councilの商標です。
- その他、本資料に記載されている社名及び商品名はそれぞれ各社が 商標または登録商標として使用している場合があります 。

著者

（企業・団体名順）

版	所属企業・団体名	部署名	氏名
第1.0版 （2024年度 WG3）	日鉄ソリューションズ株式会社	流通・サービスソリューション事業本部 アドバンストテクノロジー部	伊藤　春
			野崎　幹男
			永井　光
			磯部　凌

PGECons

PostgreSQL Enterprise Consortium