EPAS 14 登場！
BDRを中心に
新機能を徹底解説

EDB技術本部長兼サービス事業部長
高鶴 勝治

2021年　12月

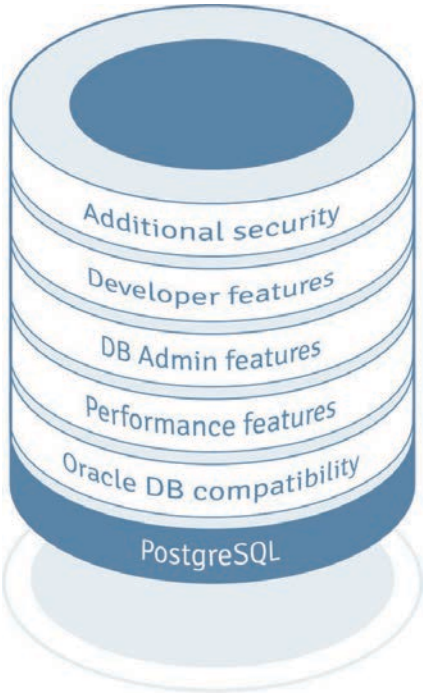
EDB

Content

- EPAS14 新機能
- EPAS14での高可用性構成
- BDR 概要
- BDR Always-On アーキテクチャー
- BDR リファレンス・アーキテクチャ

EPAS14新機能

EDB Postgres Advanced Server (EPAS)



EDB Postgres Advanced Server

- セキュリティ – パスワード・ポリシー管理, セッション・タグ監査, データ・リダクション、バーチャル・プライベート・データベース、SQL インジェクション・プロテクト, EDB*Wrap
- パフォーマンス – クエリ・オプティマイザ・ヒント句, SQL セッション/システム 待機イベント診断機能
- 開発者向け機能 – 200以上のBult-in ファンクション, ユーザ定義オブジェクト, 自律型トランザクション, ネステッド・テーブル, シノニム, アドバンスド・キューイング、DB-link
- DBA向け機能 - リソース・マネージャ、55以上のデータベース・オブジェクト/プロセス用の拡張されたカタログ・ビュー
- Oracle互換性 – 高いスキーマ互換性: データ・タイプ, 索引, ユーザ, ロール, パーティション・テーブル, Built-in パッケージ, データ・ディクショナリ・ビュー, PL/SQL トリガー, ストアド・プロシジャ、ファンクション、パッケージ, ユーティリティ(EDB*Plus EDB*Loader)

EPAS14新機能

- オブジェクトレベル監査
- 監査ログのローテーション
- オブジェクト作成/更新DDL 日付
- CONNECT BY フル・サポート
- パーティション機能拡張
- Postgres BDR へのフル対応

等　多数

オブジェクトレベル監査

- EPAS13

  EPAS13では、監査対象のステートメントしか指定できない

```
logging_collector=on
edb_audit=xml (or csv)
edb_audit_statement='insert, update, delete'
```

- EPAS14

  EPAS14では、監査を行うテーブルを指定が可能

```
alter table receivables set (edb_audit_group = 'finance');
alter system set edb_audit_statement ='insert@finance, update@finance, delete@finance';
```

```
alter table sock_drawer set (edb_audit_group = 'boring');
alter system set edb_audit_statement ='insert-boring, update-boring, delete-boring';
```

オブジェクトレベル監査 (続き)

- 監査ログの取得レイヤ
  - EPASインスタンス全体の設定(例)

```
alter system set edb_audit_statement ='insert-boring, update-boring, delete-boring';
```

  - 特定データベースに対する設定(例)

```
alter database userdb01  set edb_audit_statement ='insert-boring, update-boring, delete-boring';
```

  - 特定ユーザに対する設定(例)

```
alter user testuser01 set edb_audit_statement ='insert-boring, update-boring, delete-boring';
```

  - 特定データベース上の特定ユーザに対する設定(例)

```
alter user testuser02 in database userdb02 set edb_audit_statement ='insert-boring, update-boring, delete-boring';
```

監査ログのローテーション

- EPAS14では、指定期間を過ぎた監査ログの圧縮が可能

```
edb_audit_archiver = 'on'
edb_audit_archiver_compress_time_limit = '1d'
edb_audit_archiver_expire_time_limit = '7d'
```

オブジェクト作成/DDL最終実行日時

- EPAS14では、自動的に、DBオブジェクトの「作成日」と「DDL最終実行日時」を記録
  - テーブル
  - ビュー
  - ファンクション　　等
- all_objectsビューを参照し、確認可能

```
edb=# select created, last_ddl_time from all_objects
edb-# where object_name = 'FOO';
created                  |                  last_ddl_time
-------------------------------+-------------------------------
 21-SEP-21 15:25:07.964559 -04:00 | 21-SEP-21 15:25:07.964559 -04:00
(1 row)
```

CONNECT BY 句の強化

- EPAS13

  “Connect by” を用いたSQLでは、以下の構文のみサポート

```
CONNECT BY x = PRIOR y
CONNECT BY PRIOR x = y
```

- EPAS14
  - EPAS14では、以下の構文も更にサポート

```
CONNECT BY level <= 10
CONNECT BY x = PRIOR y AND z = t
```

  - EPAS14では更にターゲット・リストに、PRIOR を指定可能

```
SELECT empno, ename, PRIOR ename AS mgr FROM emp
START WITH mgr IS NULL CONNECT BY mgr = PRIOR empno ORDER BY 1, 2;
```

パーティション機能拡張

- サブ・パーティションのテンプレート機能

```
CREATE TABLE example (col1 NUMBER, col2 NUMBER)
PARTITION BY RANGE (col1) SUBPARTITION BY RANGE (col2)
SUBPARTITION TEMPLATE (
        SUBPARTITION s1 VALUES LESS THAN (10)
        , SUBPARTITION s2 VALUES LESS THAN (20)
)    (
PARTITION p1 VALUES LESS THAN (100)
,  PARTITION p2 VALUES LESS THAN (200)
);
```

- 特定パーティション／サブ・パーティションの参照

```
SELECT * FROM example PARTITION (p1);
SELECT * FROM example SUBPARTITION (p1_s1);
```

EPAS14でのHA構成

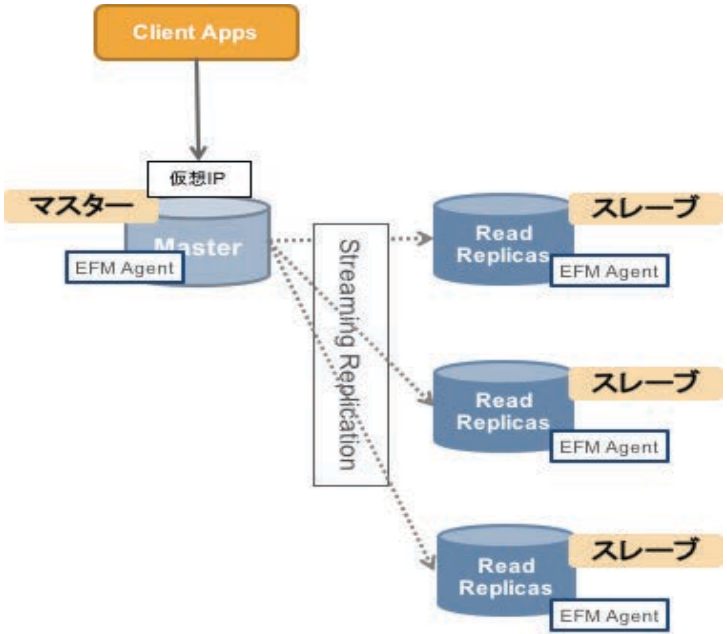
EPAS14における高可用性構成

EPAS14では、下記HA構成が可能

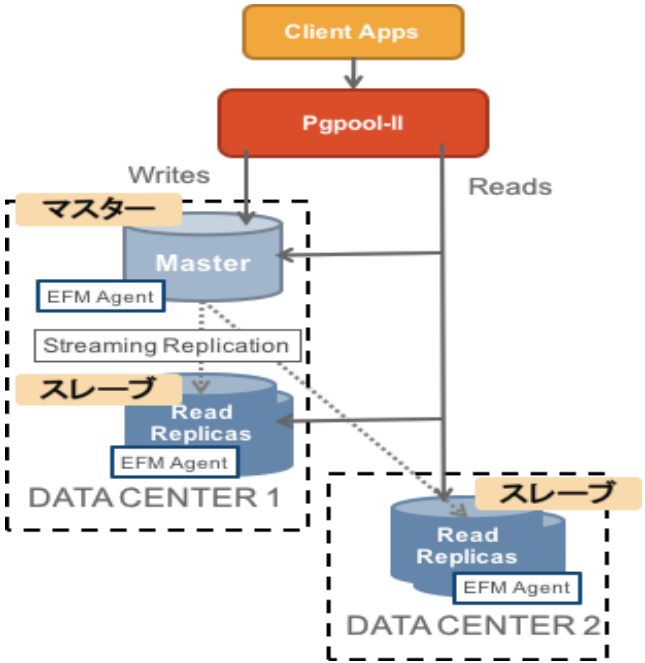
① pgpool-II による プライマリ・スタンバイ構成
② EFM によるプライマリ・スタンバイ構成
③ Pgpool-II & EFM によるプライマリ・スタンバイ構成
④ BDR によるマルチ・プライマリ構成
⑤ クラスタ・ソフトによる Active – Passive 構成

EFM によるHA構成

パターン②

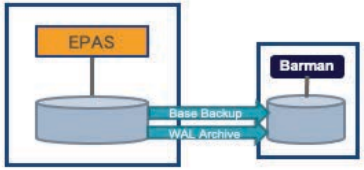


パターン③



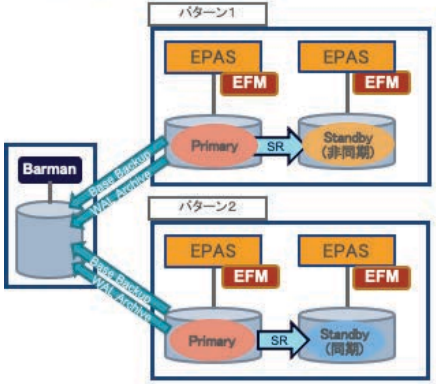
EFM によるHA構成

Bronze/ブロンズ



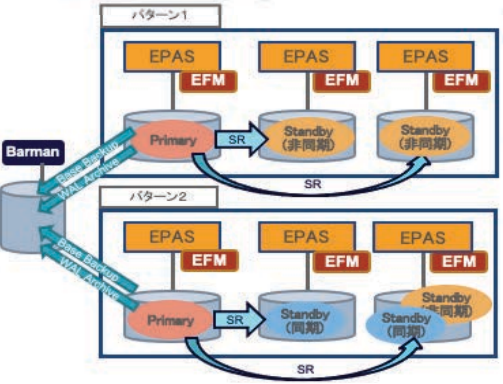
- データベースサーバのHigh Availabilityは、EDB製品等は使用せずにvSphereHAなどで対応
- 2世代以上の保持ポリシーで、Barmanを用いたデータベースのバックアップを、定期的に取得
- データベース部分の障害が発生した場合は、バックアップを用いてリカバリ。リストア中は、サービスを停止する必要がある。

Silver/シルバー



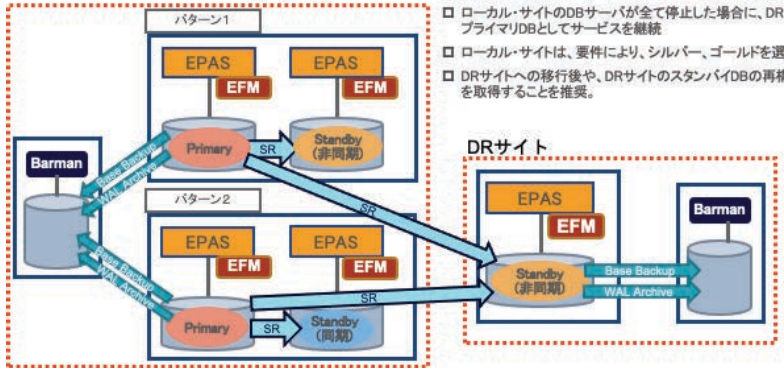
- Primary-Standby型アーキテクチャ
- 1台のDB障害に対して、サービス継続が可能
- Pgpool-II との併用により、以下が可能
  - ロード・バランシング機能によるREAD型のスケール・アウト対応
  - コネクション・プーリングによる大規模コネクション対応
- パラレル・フルバックアップ/インクリメンタル・ブロック・レベル・インクリメンタル・バック・アップを利用するため、バックアップは、プライマリDBから取得。尚、Standbyから取得することで、Primary側の負荷を軽減することも可能。
- パターン1
  - 非同期レプリケーションによる2ノード構成
  - Primaryが障害の場合、Standby(非同期)が昇格
  - データ・ロスの可能性はあるが、レプリケーションによるPrimary側のオーバヘッドは軽微
- パターン2
  - 同期レプリケーションによる2ノード構成
  - Primaryが障害の場合、Standby(同期)が昇格
  - データ・ロスはないが、同期レプリケーションによるPrimary側のオーバヘッドを考慮する必要がある

Gold/ゴールド



- Primary・Standby型アーキテクチャ
- 2台までのDB障害に対して、サービス継続が可能
- シルバーと同様、Pgpool-II との併用により、以下が可能
  - ロード・バランシング機能によるREAD型のスケール・アウト対応
  - コネクション・プーリングによる大規模コネクション対応
- バック・アップの考え方は、シルバーと同様
- パターン1
  - 非同期レプリケーションによる3ノード構成
  - Primaryが障害の場合、Standby(非同期)が昇格
  - データ・ロスの可能性はあるが、レプリケーションによるPrimary側のオーバヘッドは軽微
- パターン2
  - 同期レプリケーションによる3ノード構成
  - Primaryが障害の場合、Standby(同期)が昇格
  - データ・ロスはないが、レプリケーションによるPrimary側のオーバヘッドを考慮する必要がある
  - 2台目以降のStandbyは、同期モード/非同期モードを選択可能。但し、同期モードの台数が増えるとオーバヘッドが大きくなる傾向がある。Quorum構成、synchronous_commitの設定値を検討

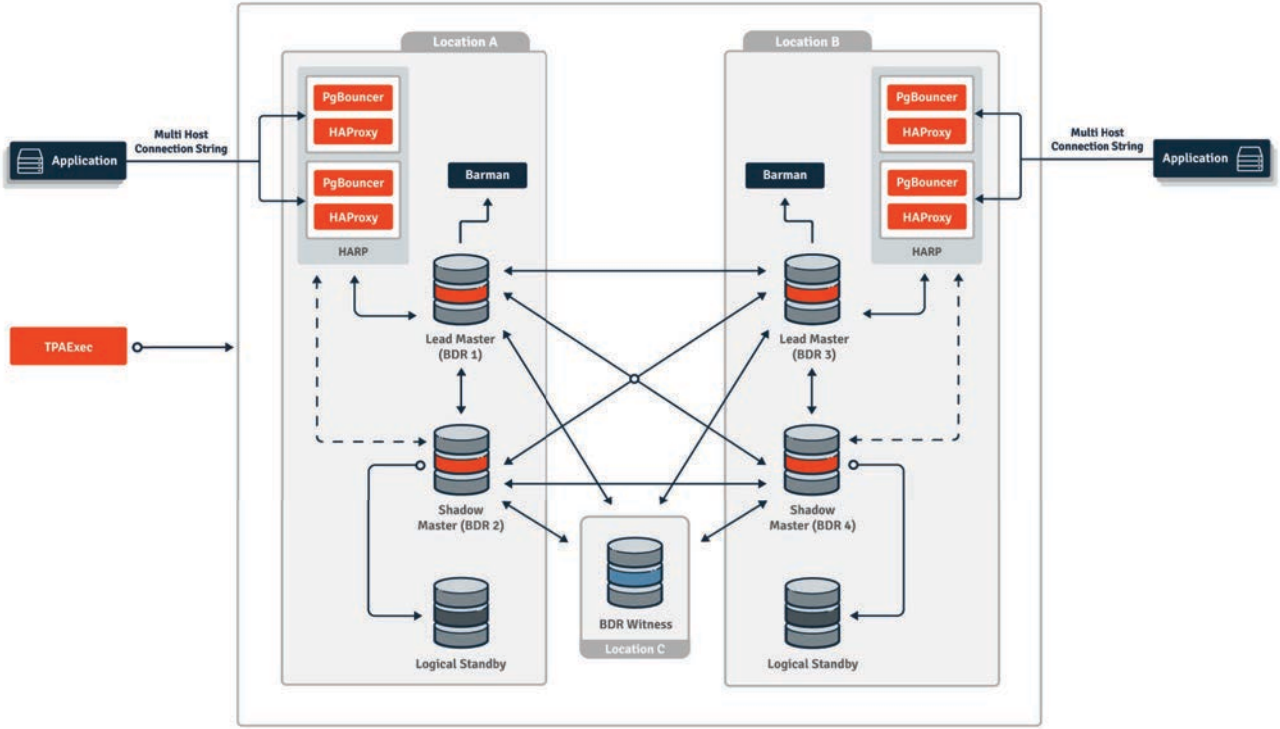
Platinum/プラチナ



- DR対応として、リモート・サイトにStandbyDB(非同期)を配置
- ローカル・サイトのDBサーバが全て停止した場合に、DRサイトのDBが昇格し、新プライマリDBとしてサービスを継続
- ローカル・サイトは、要件により、シルバー、ゴールドを選択可能
- DRサイトへの移行後や、DRサイトのスタンバイDBの再構築のため、バックアップを取得することを推奨。

BDRによるHA構成

- マルチ・マスター
- 複数拠点/センターでの更新をサポート
- 99.999% の可用性
- ノード障害対応やセンター障害に対する高い耐障害性



BDRの適用領域

要件/特徴	BDR適用領域	EFM適用領域
可用性要件	99.99 % - 99.999% (フェイル・オーバ時間： 数秒)	99.9 % - 99.99% (フェイル・オーバ時間： 1分程度)
データセンター/拠点	地理的分散での並行オペレーション (マルチ・プライマリ)	一拠点のでオペレーション (シングル・プライマリ)
アップグレード によるダウンタイム	メジャー・バージョンでの ニア・ゼロ・ダウンタイム	マイナー・バージョンでの ニア・ゼロ・ダウンタイム
想定されるアプリケーション	決済システム、 コール・ルーティング・システム、 複数拠点での連携が必要なシステム	HR、経費精算システム、CRM

BDR 概要

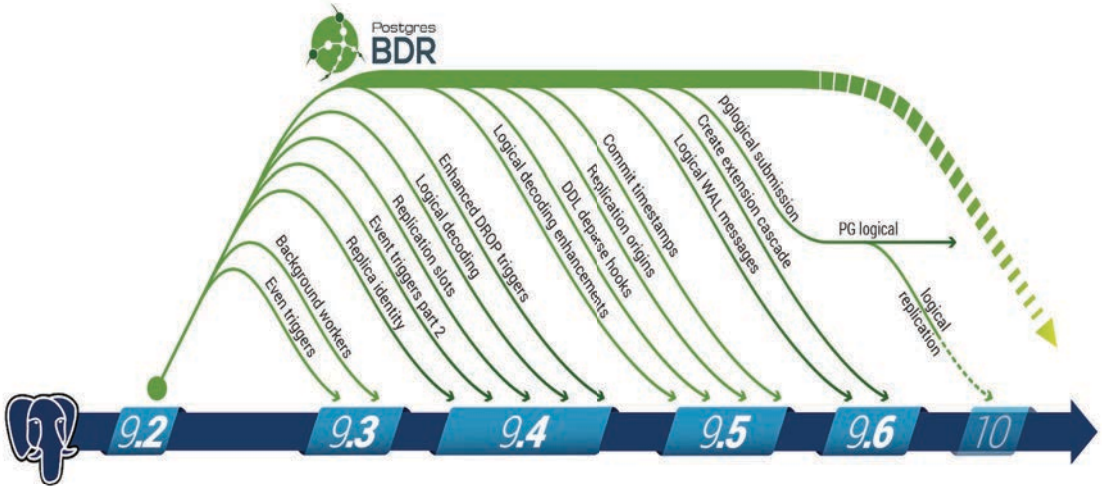
BDR とは

高可用性で地理的に分散したPostgresクラスターを可能にするマルチマスターレプリケーション

- 標準のPostgreSQL拡張機能を介して有効化されたデータとスキーマの論理レプリケーション
- 即時整合性から結果整合性までのデータ整合性オプション
- コンフリクトを管理し、パフォーマンスを監視し、一貫性を検証するための堅牢なツール
- クラウド、仮想、またはベアメタル環境にネイティブにデプロイする

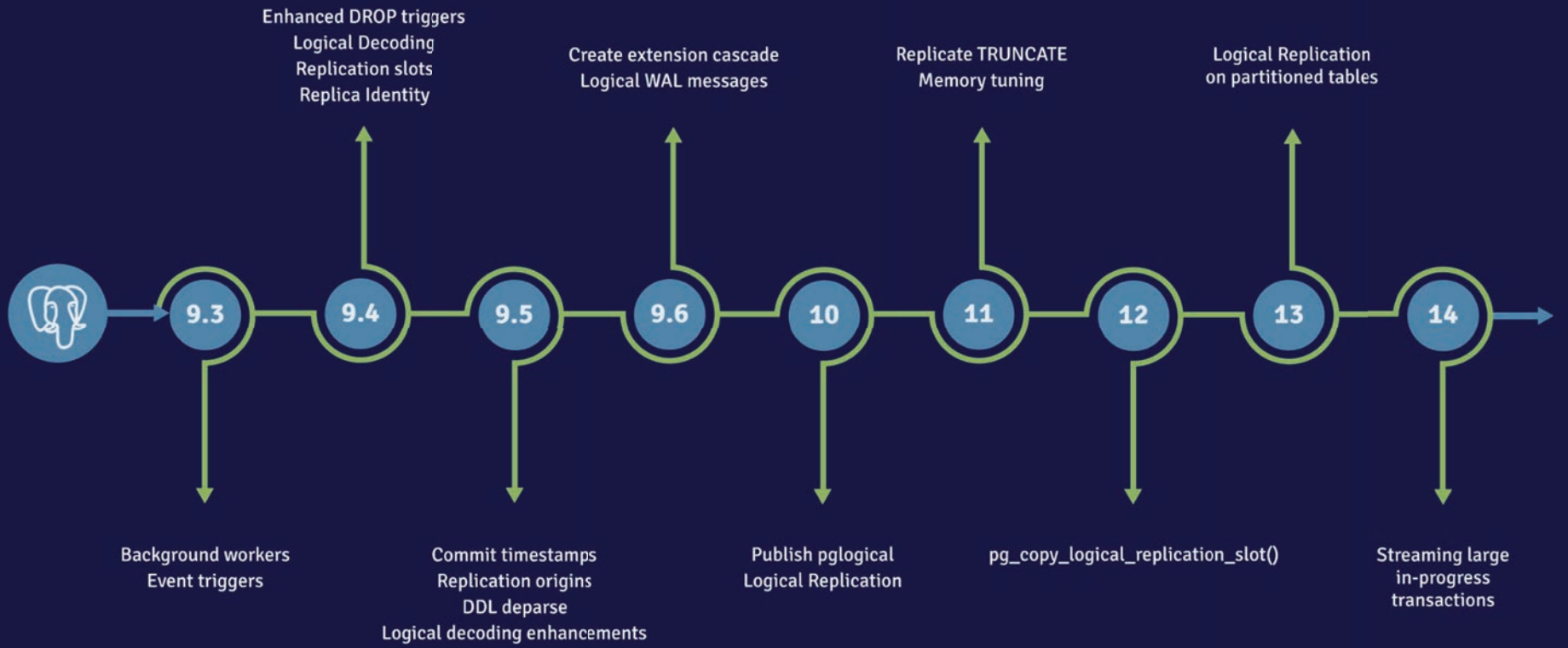
BDRの歴史

- Detailed design started in 2011
- 2014年, BDR1 リリース、 2019年デサポート
- 2018年 BDR2 リリース 、 2020年デサポート
- 2019年 BDR3 リリース



Postgres BDR

BDR project contributions to PostgreSQL



BDR 自動レプリケーション

- 自動 DML レプリケーション
  - 効率的なログ・ベースのレプリケーション
  - マスターノードのオーバーヘッドが少ない
  - 非同期/同期/CRDT オプション
- 自動 DDL レプリケーション
  - きめ細かい DDL ロック
  - 広範な DDL 管理
- 自動シーケンス処理
  - 複数のタイプの分散シーケンス ハンドラ

BDR の機能

PostgresSQLで利用可能なBDR機能

BDRは、PostgreSQLでMMRを行うための必須の機能/ツールを提供

- 一方向のロジカル・レプリケーション機能を拡張した、マルチ・マスター・レプリケーション
  - pgLogical エクステンション
  - BDR エクステンション
- MMRにおける行レベルの一貫性を保証
- ゼロ・ダウンタイムのデータベース・アップグレード

EPAS14で利用可能なBDR機能

データベース・エンジンにBDR機能をエンハンスし、更に先進的なレプリケーション能力を提供

- 列レベルのカスタマイズ可能なデータ競合解消機構
- Conflictを回避するための特別のデータ・タイプ (CRDT)
- 二重のトランザクションの実行をガードする機能 (CAMO)
- 2phase Commitによる競合のない同期レプリケーション [Eager Replication]
- パフォーマンス向上機能
  - Parallel Apply
  - Single Decode Worker

BDR事例

ACIワールドワイドは、世界の上位20行の銀行の19行を顧客とし、毎日14兆ドルを超える支払いと証券取引を処理している。大量のリアルタイムトランザクションには、多くの技術要件とそれに関連するコストが伴う。

「処理するトランザクションの財務的性質を考えると、冗長性のために複数のデータセンターが必要なだけでなく、それらのデータセンターをリアルタイムで完全に同期する必要がある」とACIワールドワイドのソフトウェアエンジニアリング担当シニアバイスプレジデント、ジャック・ブロッホは説明する。「クラウドアーキテクチャ/クラウド展開への移行に伴い、信頼性の高いレプリケーションテクノロジーがさらに必要となる。BDRは、単一の低コストソリューションで必要なものをすべて提供してくれた。」

BDR Always-On アーキテクチャ

Always-Onとは?

金融

カード決済
銀行口座へのアクセス

通信

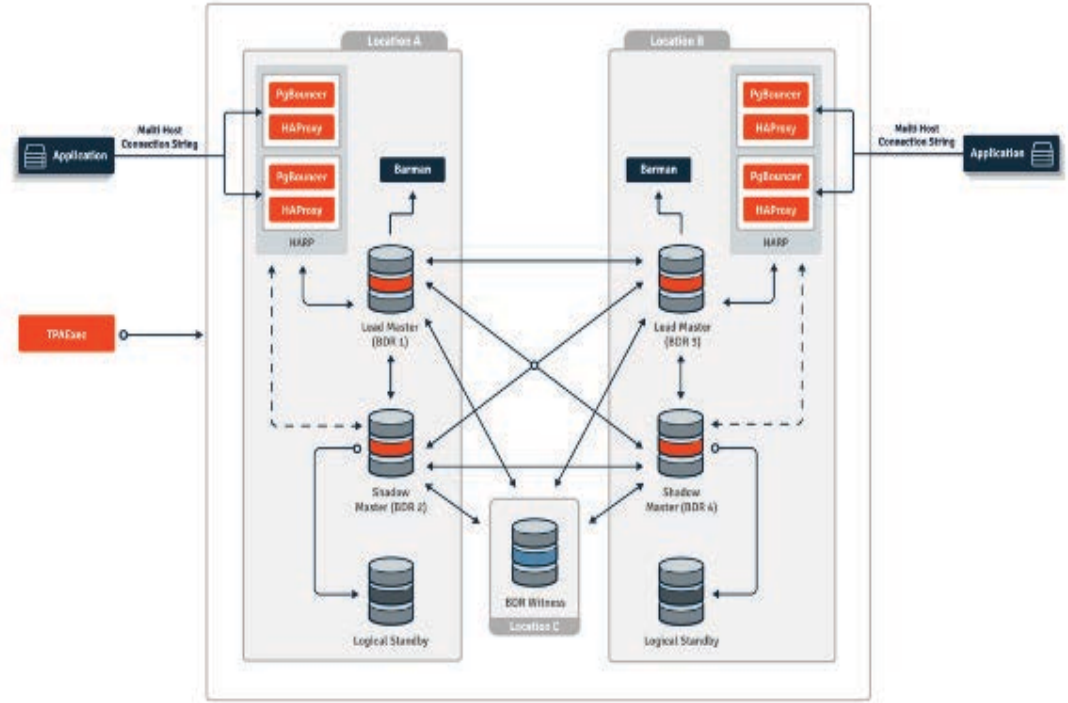
ビデオ通信
テキスト/チャット

運輸

旅行
Uber
GPS

BDR による非常に高い可用性（Always-On）

- マルチ・マスター/アクティブ- アクティブ
- ノード間の待ち時間を最小限に抑えるメッシュアーキテクチャ
- 複数のデータセンター (DC)をサポート
- Lead MasterノードへのAPPLアクセス
- 99.999%の可用性のためのShadow Masterノードへの高速フェイルオーバー
- ダウンノード時の耐性
  - ノードの停止とネットワーク パーティションからの自動回復
- 他のサービスとの統合
  - Connection Pooling、BARMAN、HA Proxy
- 更に追加可能なアーキテクチャ
  - ロジカル・スタンバイ
  - サブ・グループ



ローリングシステムアップグレード

- 自動ローリングシステムアップグレード
  - OS、データベース・サーバ、BDRの主要なソフトウェア・リリースで動作
  - 必要に応じて手動で制御可能
  - ノード間プロトコルは各リンクに対して再ネゴシエーションされ、すべてのプロトコルは下位互換性がある
- ダウンタイムの最大の原因を解決
  - セキュリティ/メンテナンス・リリース – 年間4回
  - 予定外のバグは1年に1回
  - パラメータの変更、ハードウェアの変更

BDR
リファレンス
アーキテクチャ

Postgres BDR Always Onアーキテクチャ

EDBは4つの標準アーキテクチャを設定

- Always On Bronze :
  2つのBDRノードがある単一のアクティブな場所
  (データセンターまたはクラウドリージョン)

- Always On Silver :
  3つのBDRノードを備えた単一のアクティブな場所と
  障害リカバリ（DR）の場所にバックアップ

- Always On Gold :
  2つのアクティブな場所

- Always On Platinum :
  ホットスタンバイモードで追加の冗長ハードウェアを備えた
  2つのアクティブな場所

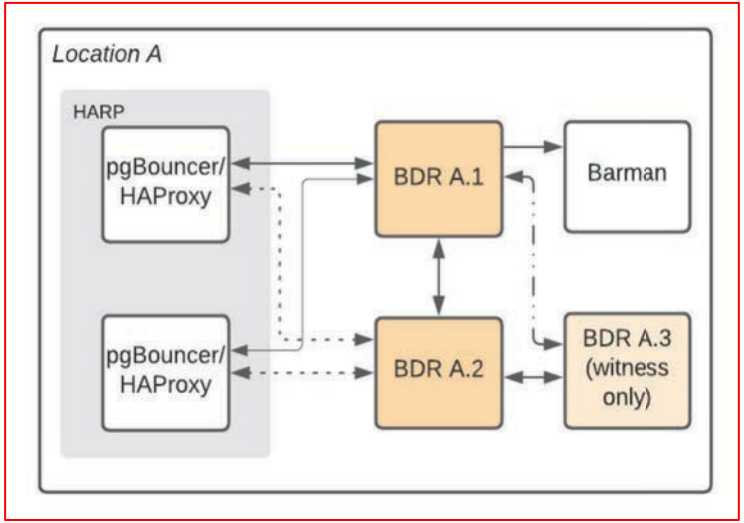
注：場所は大まかに定義されており、物理的場所（データセンター）またはクラウドリージョンである可能性があり、
実際には顧客の場所の定義または解釈に依存します。

Postgres BDR Always On Bronze

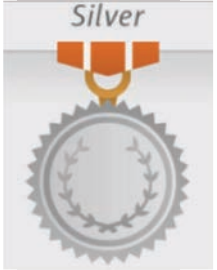
Always On Bronze （単一のアクティブな場所）

- 3つのBDRノード（1つはデータがないウィットネスのみ）
- バックアップとリカバリ用のBarmanノード
  （同じ場所に表示されますが、別の場所にオフサイトに
  することもできます）
- 2つのpgBouncer / HAProxyノード

BDRノードはアベイラビリティーゾーン全体に分散
することもできます

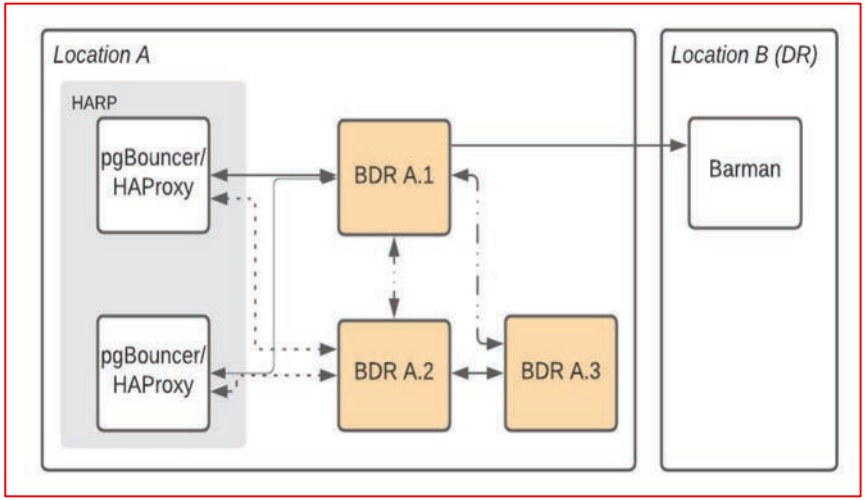


Postgres BDR Always On Silver



Always On Silver （単一のアクティブな場所、DR場所にバックアップ）

- 3つのBDRノード、各ノードにはデータがあります
- Barman offsite （offsiteはオプショナルが、お勧め）
- 2つのpgBouncer / HAProxyノード

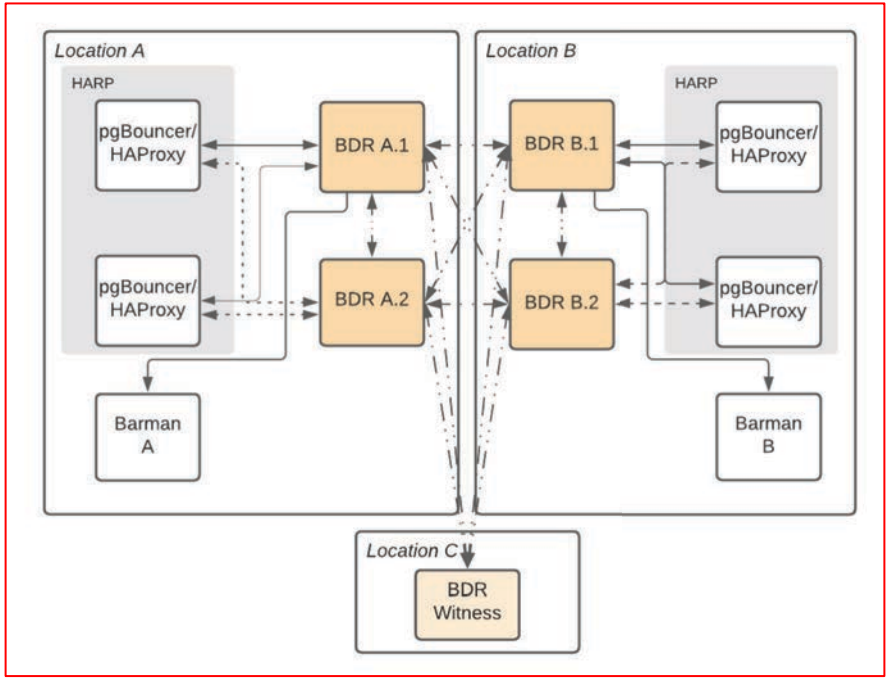


または、BDRノードをアベイラビリティーゾーンまたはロケーションに分散させ、Barmanをロケーション Aに配置することもできます

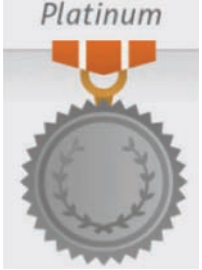
Postgres BDR Always On Gold

Always On Gold （2つのアクティブな場所）

- 4つのBDRノード（場所Aに2つ、場所Bに2つ）
- Barmanノード2つ（場所Aに1つ、場所Bに1つ）
- ロケーションCに1つの監視ノード
  （オプショナル、ただし推奨）
- 4つのpgBouncer / HAProxyノード
  （ロケーションAに2つ、ロケーションBに2つ）



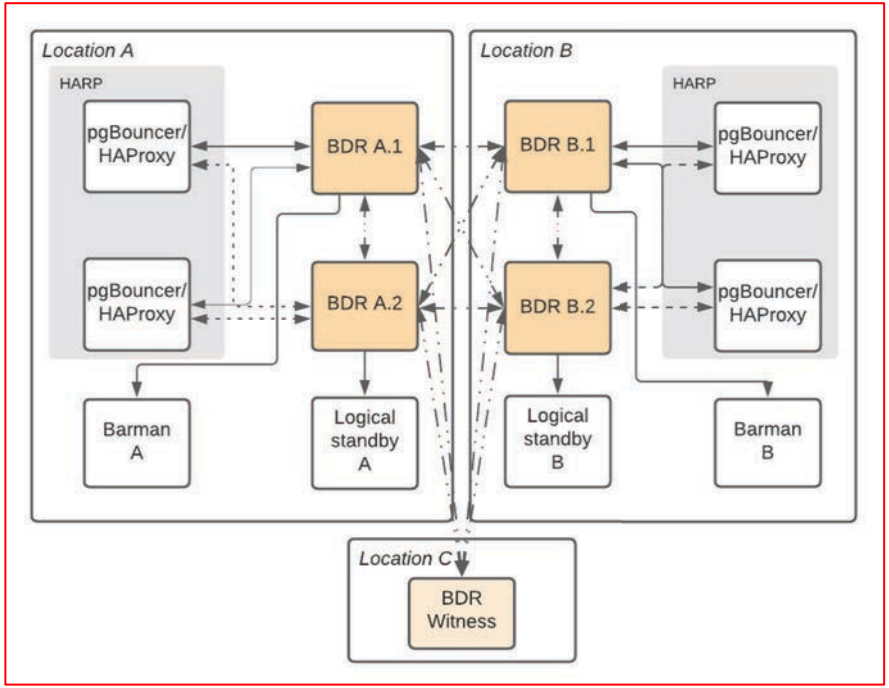
Postgres BDR Always On Platinum



Always On Platinum （2つの場所、高速HA復元）

- 4つのBDRノード（場所Aに2つ、場所Bに2つ）
- 2つのロジカルスタンバイ
  （場所Aに1つ、場所Bに1つ）
- 2つのBarmanノード（1つは場所A、1つは場所B）
- ロケーションCに1つの監視ノード
  （オプショナル、ただし推奨）
- 4つのpgBouncer / HAProxyノード
  （場所Aに2つ、場所Bに2つ）

どちらの場所にも冗長ホットスタンバイハードウェアがあり、
ハードウェア障害が発生した場合にバックアップからBDRノードの
復元を待つことなくローカルの高可用性を維持



アーキテクチャの選定ポイント

	Always On-Bronze	Always On-Silver	Always On-Gold	Always On-Plutinum
ハードウェア障害保護	○	○	○	○
ロケーション障害時のデータ保護	× (Barmanがオフサイトに移動しない限り)	○ バックアップからの回復	○	○
ロケーション障害時のサービス継続	× (Barmanがオフサイトに移動しない限り)	△ バックアップからの回復	○ DRサイトへのフェイルオーバー	○ DRサイトへのフェイルオーバー
ダウンタイムなしのアップグレード	○ (ローリング・アップグレード中1台での運用)	○	○	○
パブリック/プライベートクラウドでのAZのサポート	○	○	○	○
デバイス障害後の高可用性の高速ローカル復元	× HAを復元する時間： (1) VM prov + (2)約60分/ 500GB	○ 3つのローカルBDRノードにより、デバイス障害後にHAを維持できます	× HAを復元する時間： (1) VM prov + (2)約60分/ 500GB	○ ロジカルスタンバイを、すぐにマスタ・ノードに昇格可能
データ・センター間のネットワークトラフィック	なし	あり バックアップに関するトラフィックのみ	あり ロジカル・レプリケーションのためのトラフィック	あり ロジカル・レプリケーションのためのトラフィック
BDRライセンスコスト	2 台のBDRノード	3台のBDRノード	4台のBDRノード	4台のBDRノード

EDB

ご清聴
ありがとう
ございました！