オールインワンのPostgreSQLのクラスタ管理ツール Pgpool-IIのご紹介

PostgreSQL Conference Japan 2021
2021-11-12

SRA OSS, Inc. 日本支社
彭 博 (ペン ボ)

自己紹介

ペン ボ

- 名前： 彭 博 (Bo Peng)
  pengbo@sraoss.co.jp
- 所属：SRA OSS, Inc. 日本支社
  基盤技術グループ
- 職務：
  - OSS技術サポート、ミドルウェア構築
  - PostgreSQLクラスタ管理ツールPgpool-IIの開発者

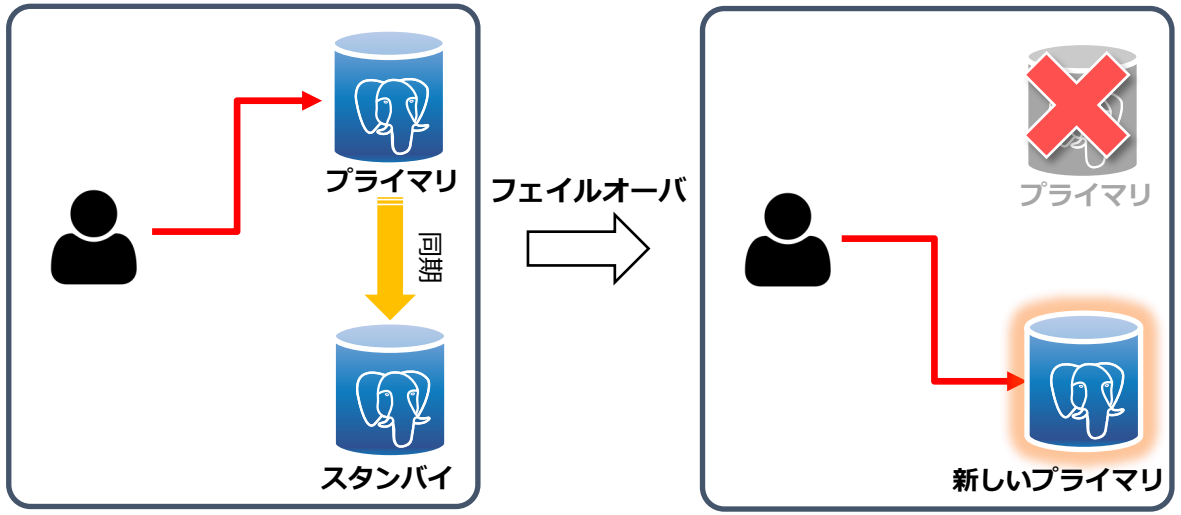
本日の流れ

- Pgpool-IIとは
- Pgpool-IIの紹介
- 新バージョン4.3の機能紹介

なぜクラスタは必要なの？

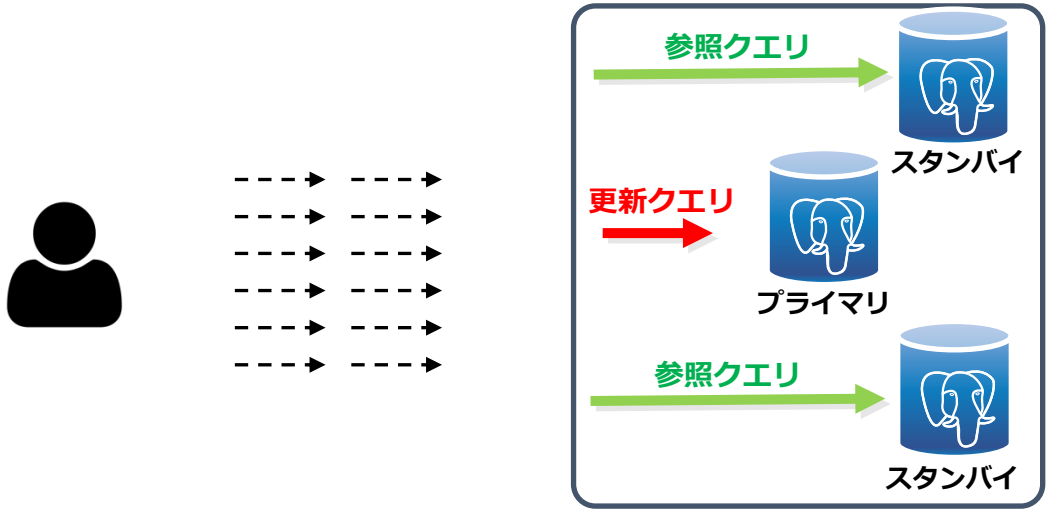
高可用性

- データの冗長化
- 稼働系に障害が発生した際に、待機系に切り替えてサービスを継続させる
- ダウンタイムを最小限に抑える



負荷分散

- レプリケーションされている複数台のPostgreSQLクラスタで、参照クエリをいずれかのPostgreSQLに振り分ける
- プライマリの負荷を下げる
- 検索性能の向上



PostgreSQLクラスタの課題

- PostgreSQLに自動フェイルオーバ/フェイルバックの仕組みがない
  - 障害が発生した際に手動での切り替えが必要
    - 障害の検知
    - 待機系の昇格
    - 残りの待機系の同期先の変更
    - クライアントの接続先情報の変更
  - 障害が発生したノードの復旧
    - 障害が発生したノードを復旧し、クラスタに再度組み込む

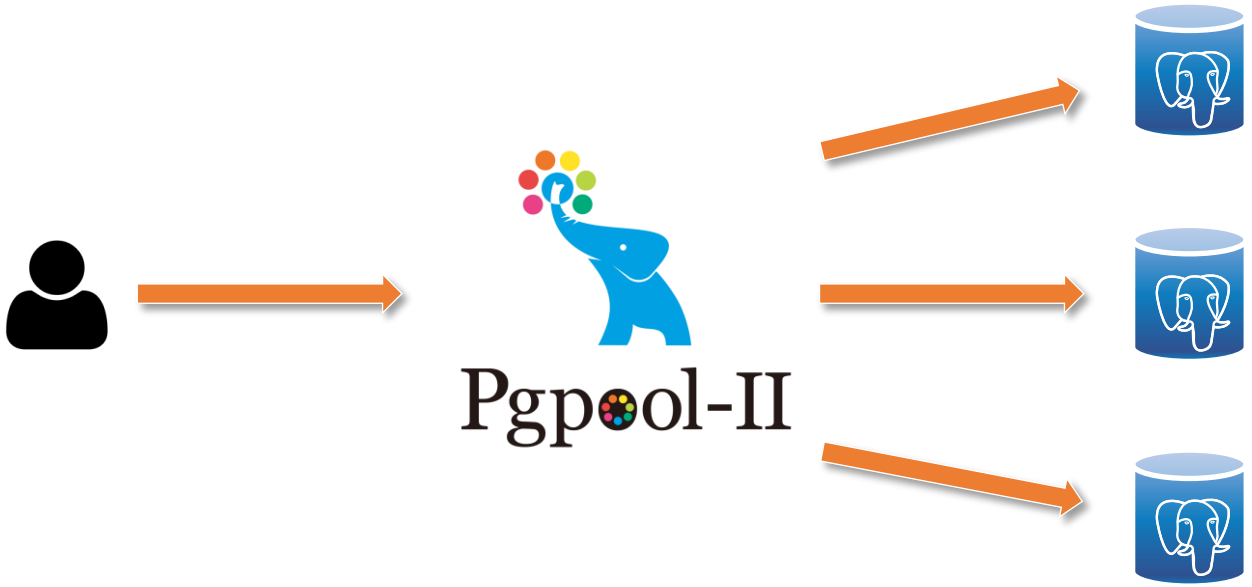
PostgreSQLクラスタの課題

- 参照クエリの振り分けの制御
  - PostgreSQLのストリーミングレプリケーション構成では、プライマリは更新クエリ/参照クエリを処理でき、スタンバイは参照クエリのみ処理できる
  - PostgreSQLは更新クエリと参照クエリを振り分ける機能を提供していない
  - アプリケーション側でクエリを振り分ける処理を実装する必要がある

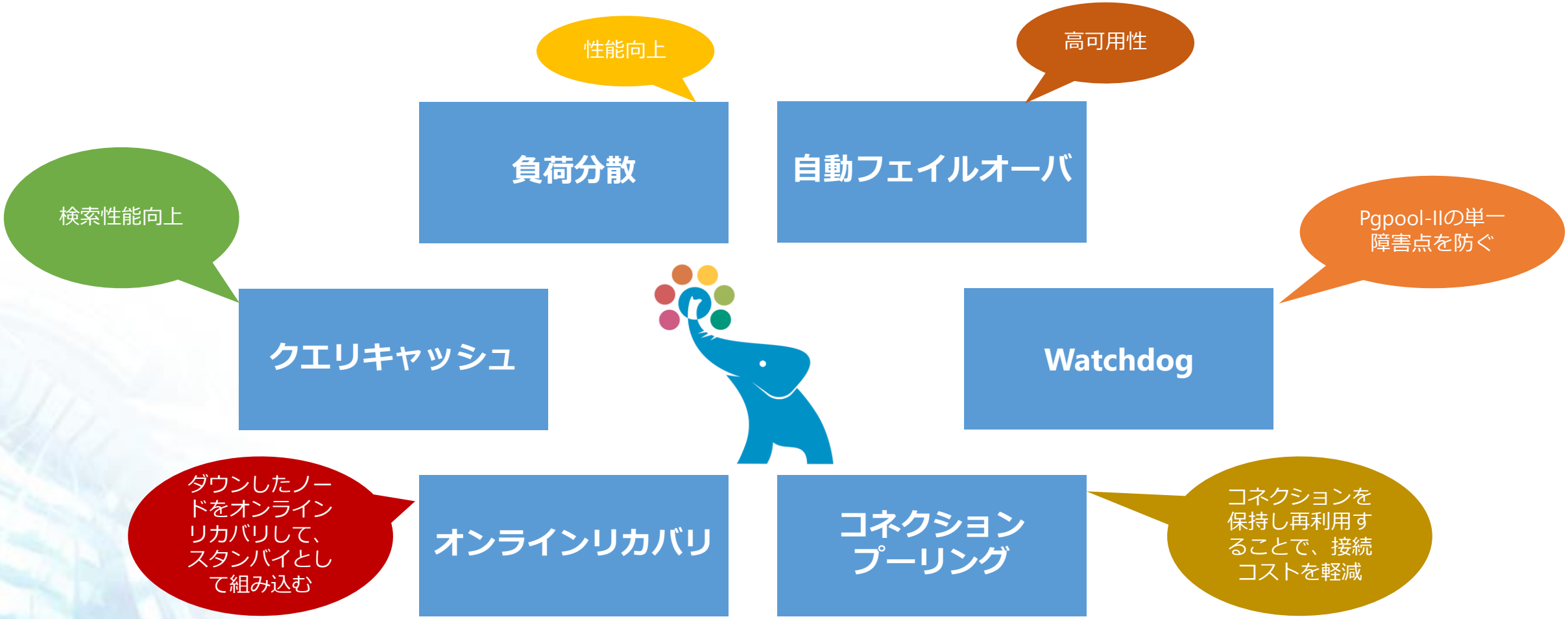
PostgreSQLクラスタの課題を解決するために

Pgpool-IIを利用することでこれらの課題を解決できる

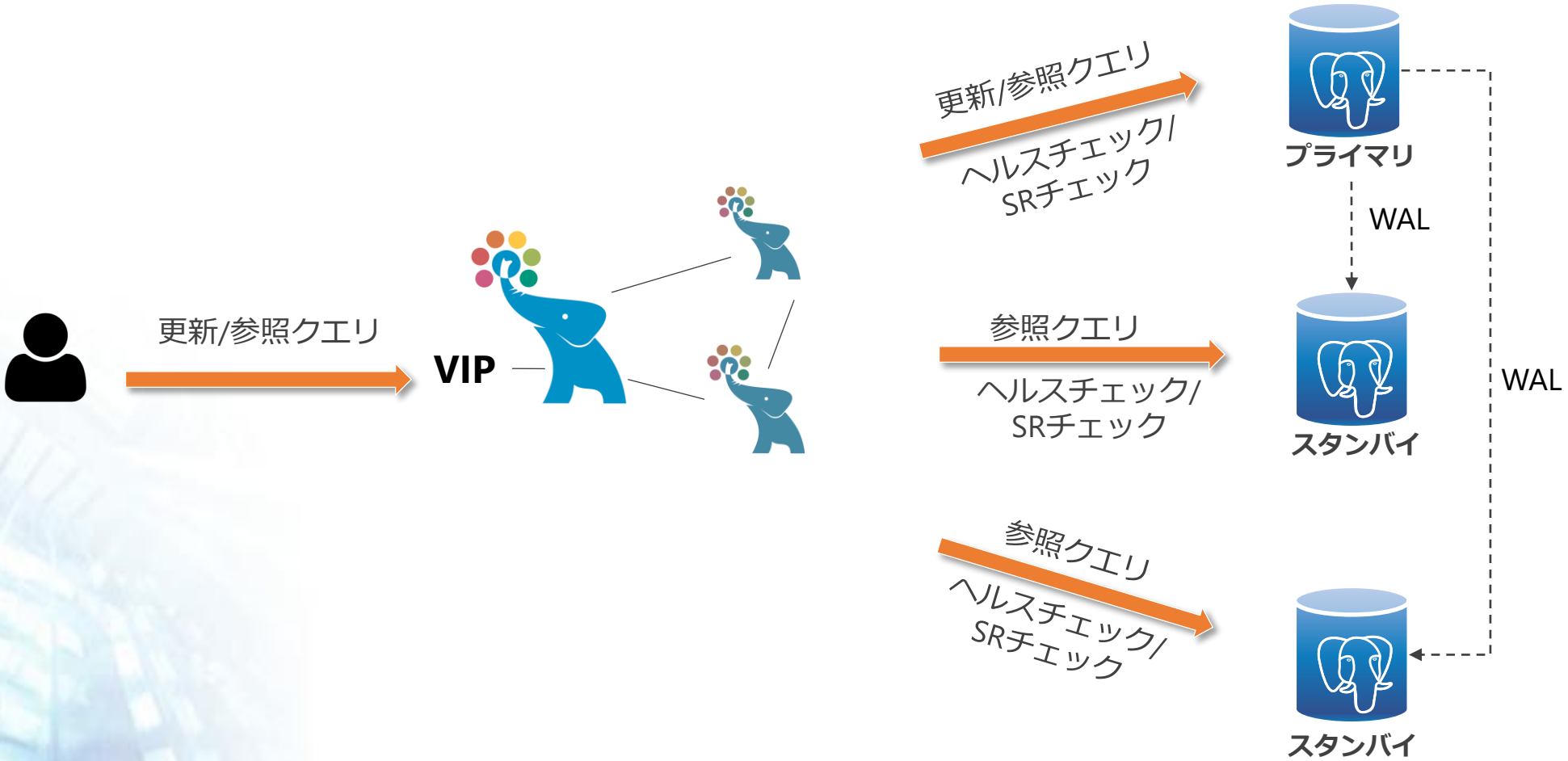
- PgpoolはクライアントとPostgreSQLの間で動作するミドルウェア
- ユーザは複数PostgreSQLサーバを意識せず、1台のように見える



Pgpool-IIの機能

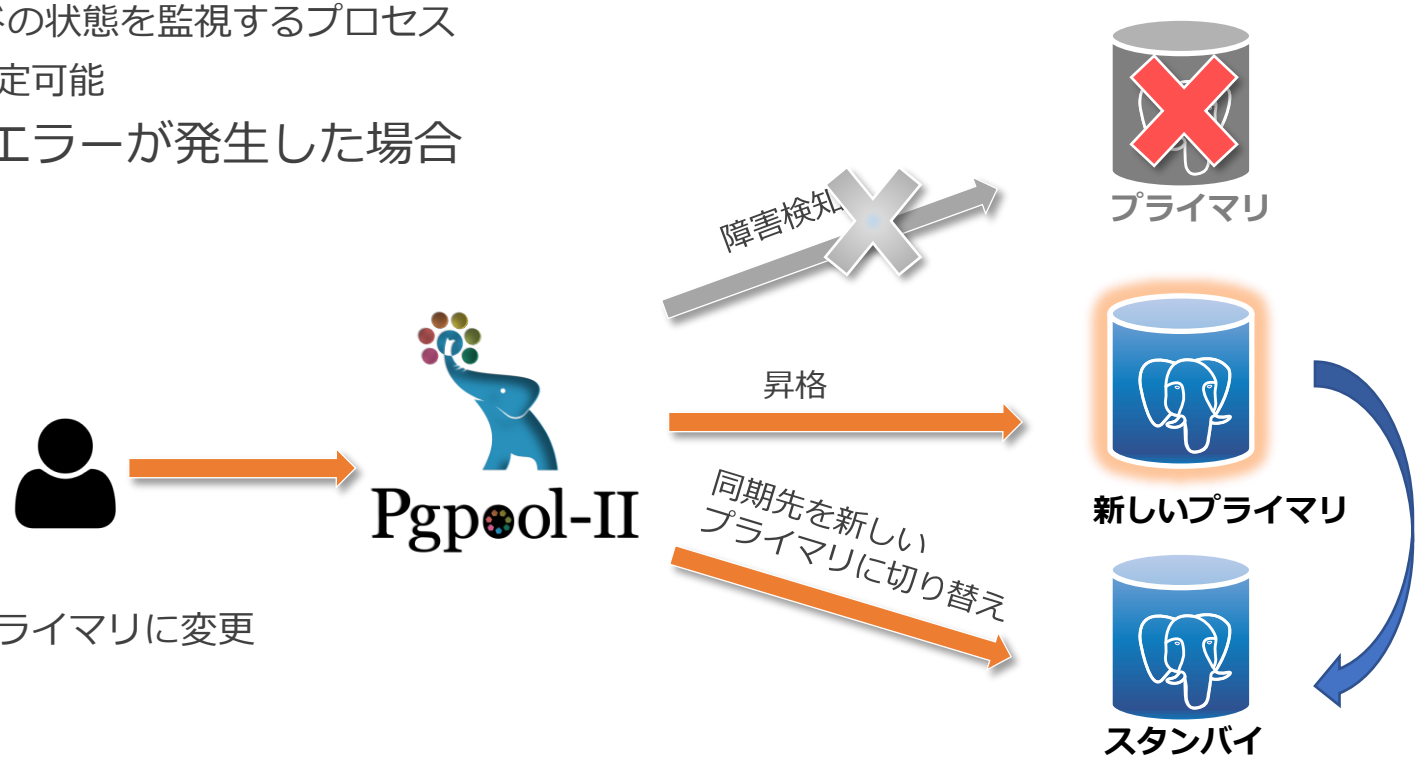


Pgpool-IIの基本構成



自動フェイルオーバ

- フェイルオーバの契機
  - ヘルスチェックでダウンと判定された場合
    - ヘルスチェック： 各PostgreSQLノードの状態を監視するプロセス
    - 実行間隔、最大リトライ回数などを設定可能
  - ノードへの接続、および接続中にエラーが発生した場合
    - failover_on_backend_error = on の場合
- フェイルオーバ処理
  1. failover_command
     - スタンバイの昇格
     - カスタムスクリプトを指定できる
  2. follow_primary_command
     - 残りのスタンバイの同期先を新しいプライマリに変更
     - カスタムスクリプトを指定できる

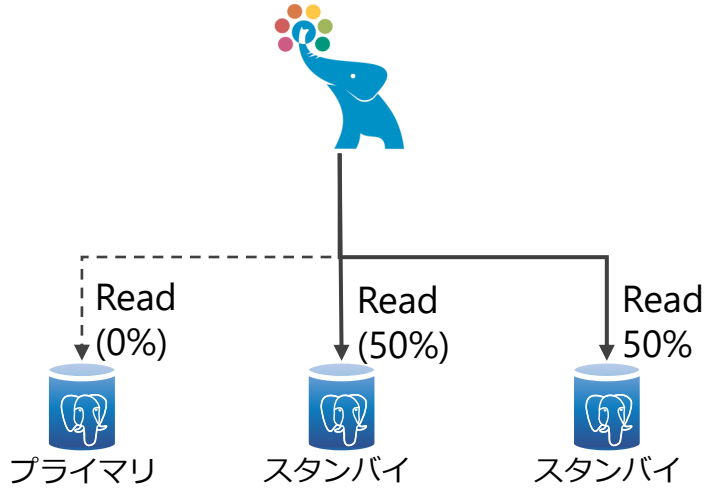


負荷分散

- SQLパーサを搭載しており、クエリを解析できる
- 更新クエリをプライマリに送り、参照クエリを複数のPostgreSQLノード間で分散
- 負荷分散モード
  - セッションレベル (デフォルト)
  - ステートメントレベル
- 負荷分散の比率が設定可能

  例えば、プライマリを更新処理専用にし、すべての参照クエリをスタンバイに振り分けたい

  backend_weight0 = 0
  backend_weight1 = 1
  backend_weight2 = 1



負荷分散

- 特定のクエリを負荷分散させない
  - 特定の文字列を含むクエリ
    - 例: primary_routing_query_pattern_list = '.*my_table*.'
    - SELECT * FROM my_table は負荷分散されない
  - クエリの先頭にコメントを付与する
    - 例: /*NO LOAD BALANCE*/SELECT * FROM t1
    - 上記クエリは負荷分散されない
  - 更新を行う関数をカンマ区切りで指定
    - 例: write_function_list = 'f1'
    - SELECT f1() は負荷分散されない

負荷分散

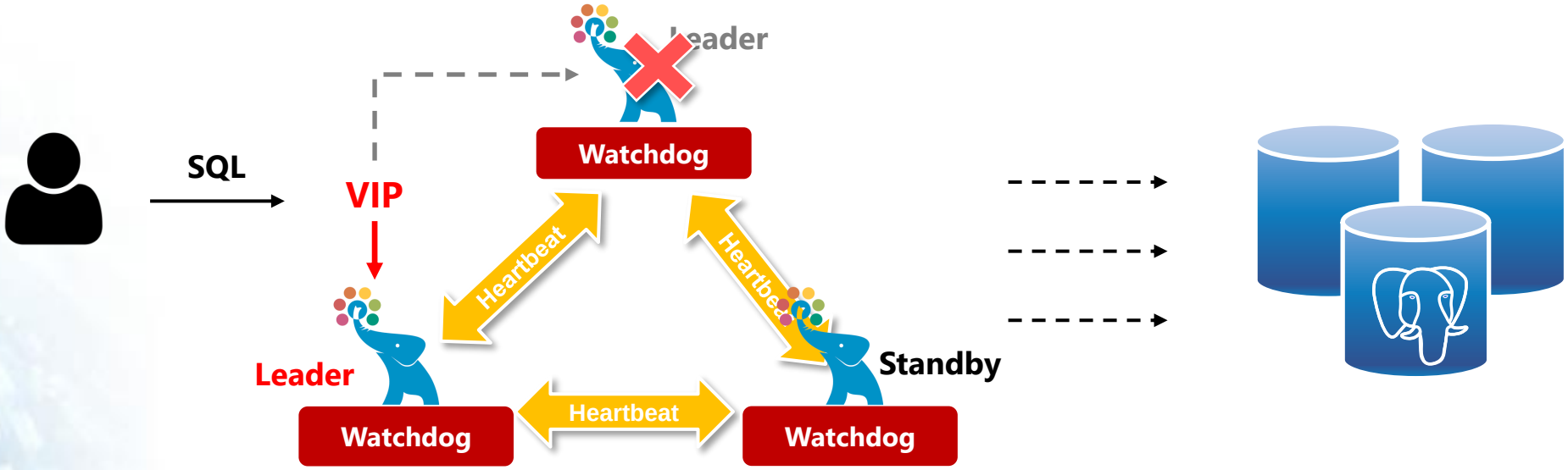
- その他の負荷分散の設定
  - database_redirect_preference_list
    - データベース名によってクエリの振り分け先を決定
  - app_name_redirect_preference_list
    - アプリケーション名によってクエリの振り分け先を決定
  - disable_load_balance_on_write
    - 更新クエリが発行された後の負荷分散の振る舞い

レプリケーションの遅延への対処

- delay_thresholdパラメータ
  - Pgpool-IIが定期的にレプリケーション遅延をチェック
  - レプリケーション遅延がdelay_thresholdに設定した値を超えた場合
    - すべてのクエリをプライマリに送信 (デフォルト)
    - 新しい負荷分散ノードの選定が行われる v4.3~
      - 1番遅延の少ないスタンバイが負荷分散ノードとして選ばれる
      - prefer_lower_delay_standby=onに設定する必要がある

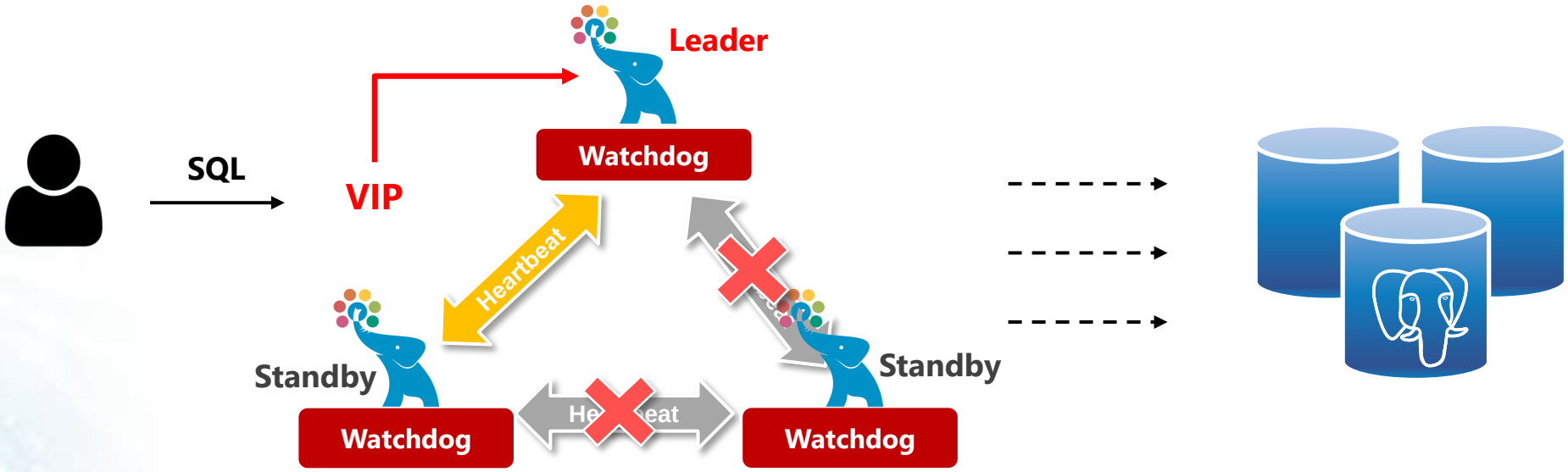
Pgpool-IIの高可用性 (Watchdog)

- Pgpool-IIの単一障害点を回避
- 複数のPgpool-IIがお互いに監視することで、Pgpool-IIを冗長化するための機能
- 定期的に他のPgpool-IIノードにハートビート信号を送信
- Pgpool-IIノードの障害が検出された際に、Watchdogは投票によって新しいリーダーを決定し、切り替える
- 仮想IPの自動切り替え
- バックエンドノードのフェイルオーバの動作を制御



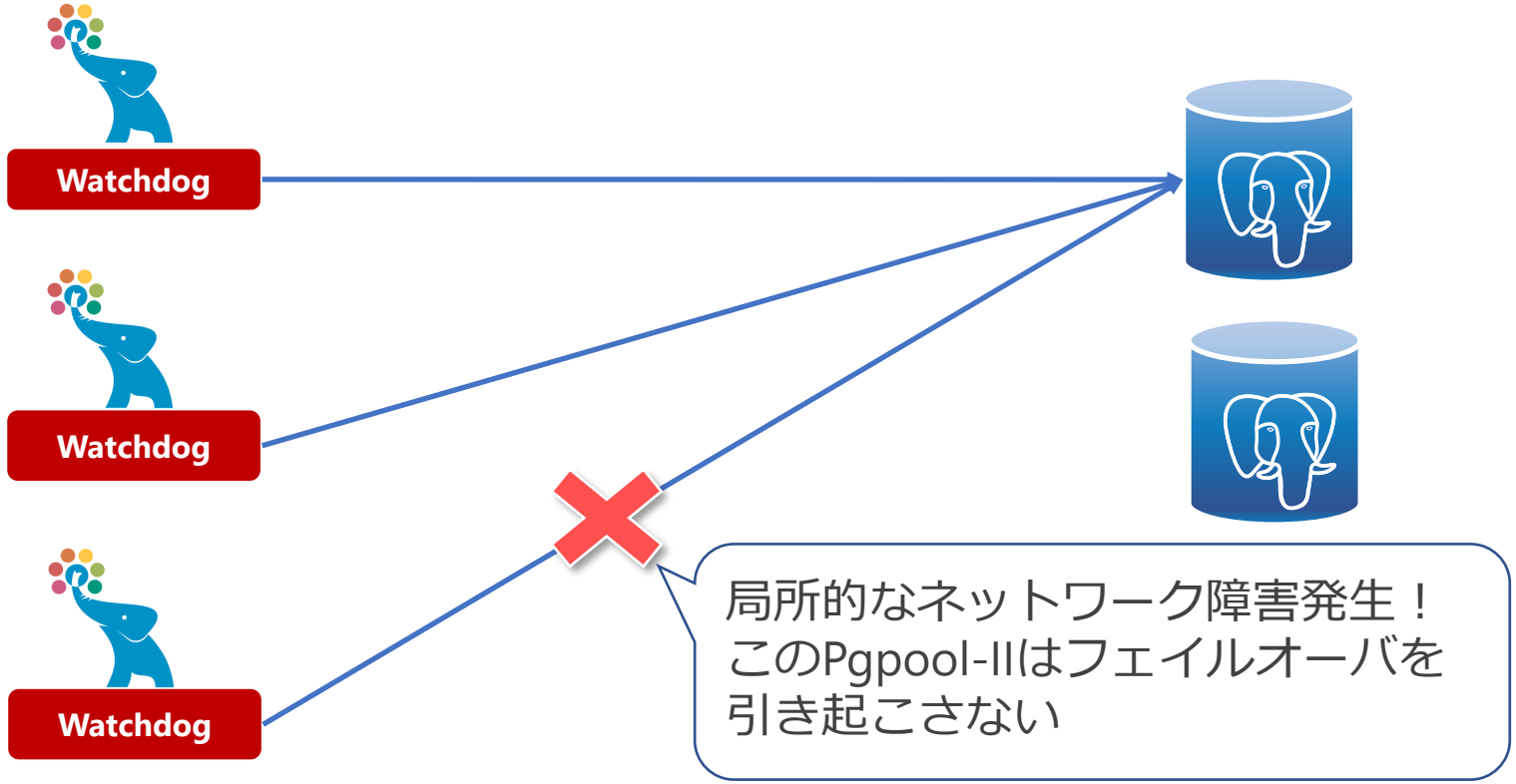
Pgpool-IIの高可用性 (Watchdog)

- スプリットブレイン対策
- Pgpool-IIを奇数台(3台以上)用意することによって、多数決でどれが本当にダウンしているのかを正しく判断できる



Pgpool-IIの高可用性 (Watchdog)

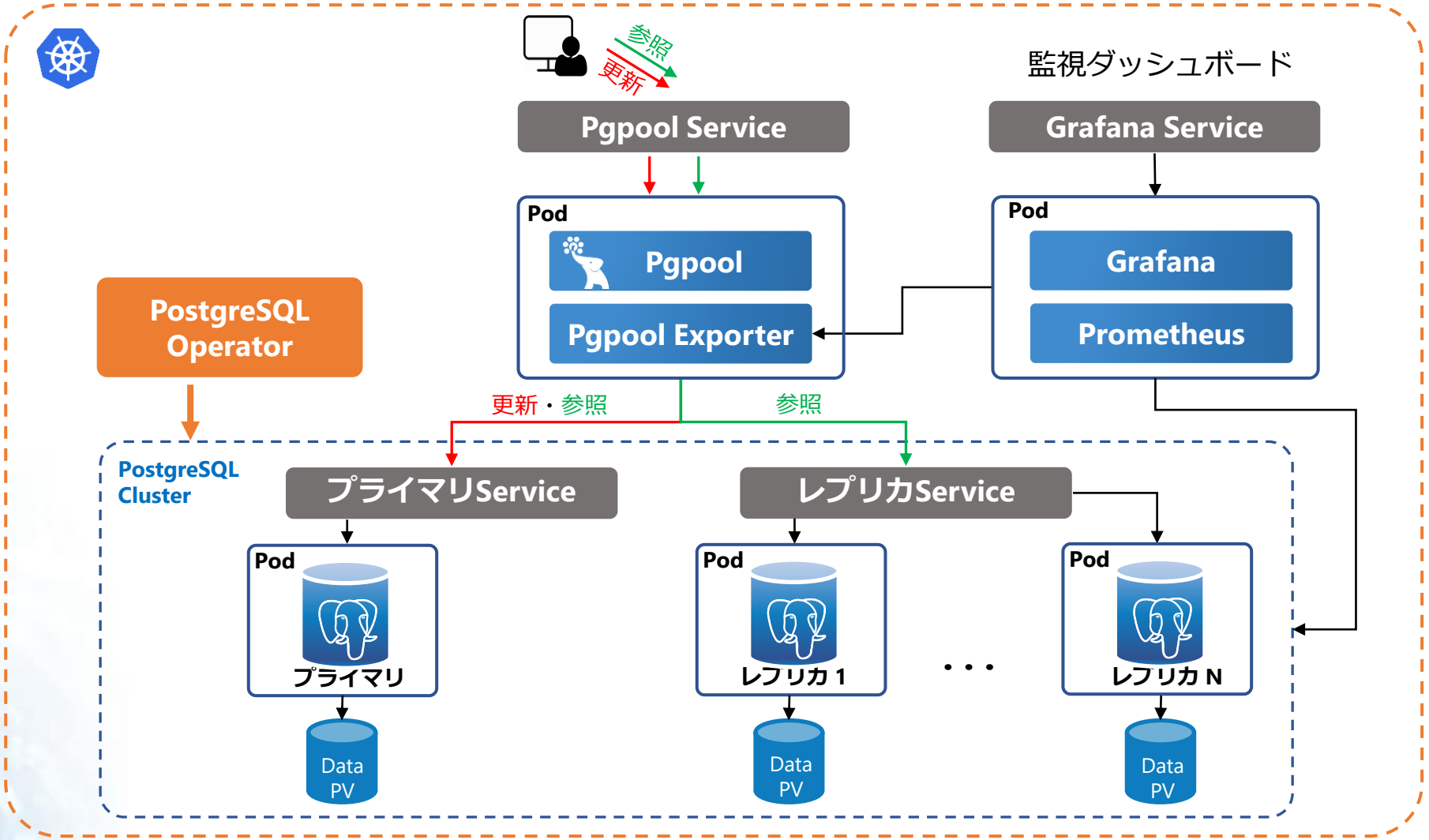
- ヘルスチェックでも多数決の原理を利用
- 局所的なネットワーク障害による障害誤検知を防止



クラウドにおけるPgpool-IIの活用

Amazon Aurora PostgreSQLやKubernetesでもPgpool-IIを利用可能

Kubernetes全体構成図



Pgpool-II 4.3の新機能紹介

v4.3の主な新機能・改良

- 設定ファイルの大幅な改善
  - pgpool.confから他の設定ファイルの読み込みが可能
  - すべてのパラメータのコメントアウト化
- フェイルオーバの機能強化
  - 予期しないフェイルオーバを回避
  - スイッチオーバ
- Pgpool-IIの独自の統計情報出力コマンドの改善
  - コネクションの統計情報の出力
  - SHOW POOL_NODESの新しいフィールドの追加
- PostgreSQL 14のSQLパーサの移植

v4.3新機能: pgpool.confから他の設定ファイルの読み込み

- pgpool.confファイルにincludeコマンドを書くことができるようになった
- pgpool.confファイルを機能ごとに複数の小さなファイルに分割することが可能

pgpool.conf

```
…
…
include = 'watchdog.conf'
include = 'query_cache.conf'
```

watchdog.conf

```
use_watchdog = off
hostname0 = ''
wd_port0 = 9000
pgpool_port0 = 9999
…
```

query_cache.conf

```
memory_cache_enabled = off
memqcache_method = 'shmem'
…
```

v4.3新機能: すべてのパラメータのコメントアウト化

- 4.2まで、コメントアウトされているパラメータとコメントアウトされていないパラメータが混在
- 4.3以降、すべてのパラメータがコメントアウト化される

```
...
#listen_addresses = 'localhost'
                                    # Host name or IP address to listen on:
                                    # '*' for all, '' for no TCP/IP connections
                                    # (change requires restart)
#port = 9999
                                    # Port number
                                    # (change requires restart)
#socket_dir = '/tmp'
                                    # Unix domain socket path
                                    # The Debian package defaults to
                                    # /var/run/postgresql
                                    # (change requires restart)
#reserved_connections = 0
                                    # Number of reserved connections.
                                    # Pgpool-II does not accept connections if over
                                    # num_init_chidlren - reserved_connections.'
```

v4.3新機能: failover_on_backend_shutdown

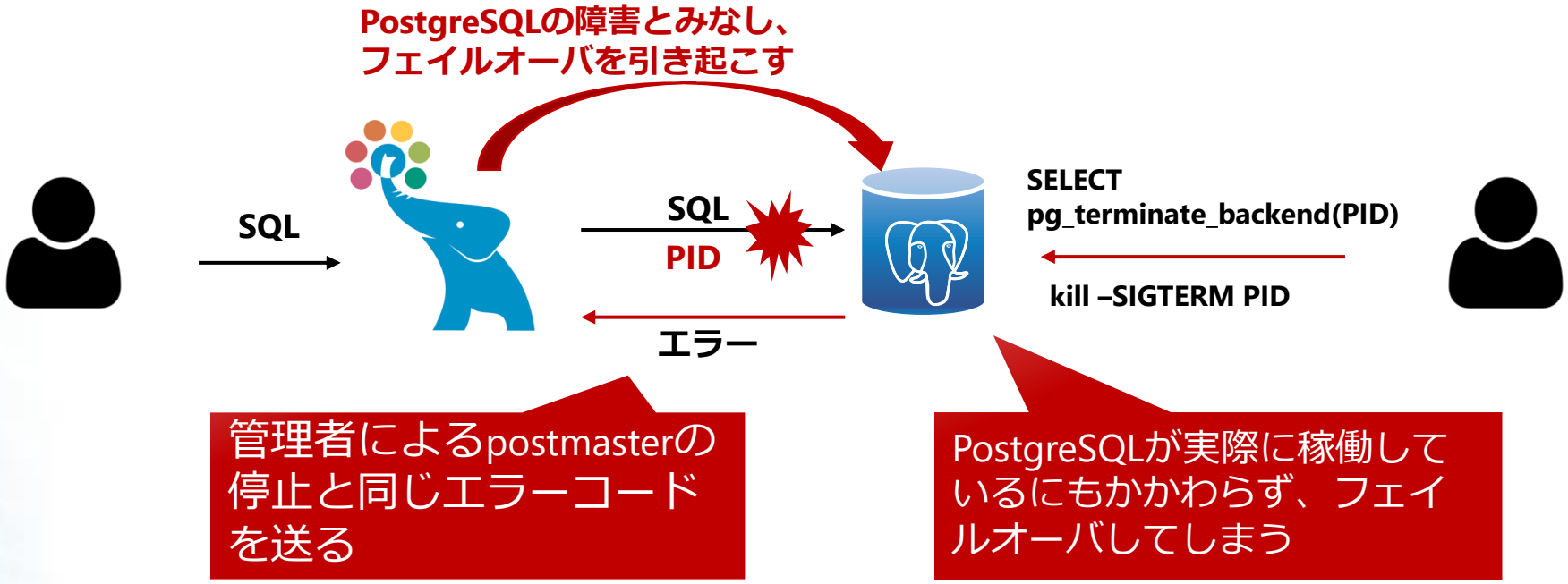
接続中のセッションを切断した場合の問題点:

予期しないフェイルオーバが発生してしまう可能性がある

以下①と②の場合は、PostgreSQLが同じエラーコードを送る

① クライアントがpgpoolに接続している間に、PostgreSQLが管理者によってシャットダウンされた場合

② pg_terminate_backend()またはSIGTERMを使って接続中のセッションを切断した場合



v4.3新機能: failover_on_backend_shutdown

failover_on_backend_shutdown =off (default) v4.3~

- 管理者によってpostmasterが停止する際に、**フェイルオーバの無効化**が可能に
- デフォルトでは、v4.2までと**異なる**挙動
- 予期しないフェイルオーバを回避したい時に便利
  - pg_terminate_backend
  - killコマンド

V4.3新機能: ノードを指定してスイッチオーバー

Pgpool-IIフェイルオーバ処理の流れ

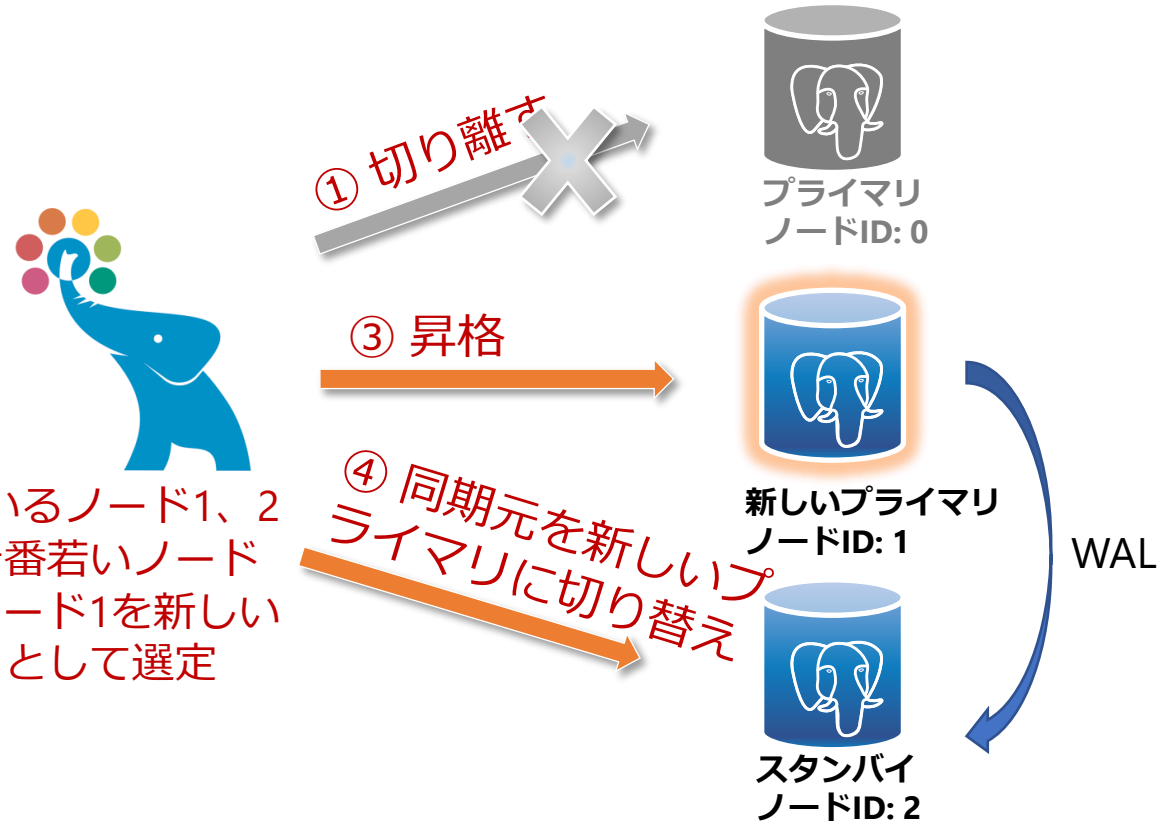
① プライマリで障害が発生/手動でPgpool-IIからプライマリを切り離す（pcp_detach_nodeにより）

② Pgpool-IIが新しいプライマリを選定
- 生きているデータベースノードの中から**一番若いノードID**を持つノード

③ failover_commandを実行
- ②で選定されたPostgreSQLノードを昇格させる

④ follow_primary_commandを実行
- 新しいプライマリからスタンバイをリカバリする

ノードを指定して昇格させることはできない



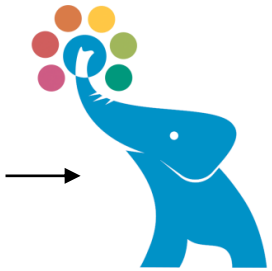
v4.3新機能: ノードを指定してスイッチオーバー

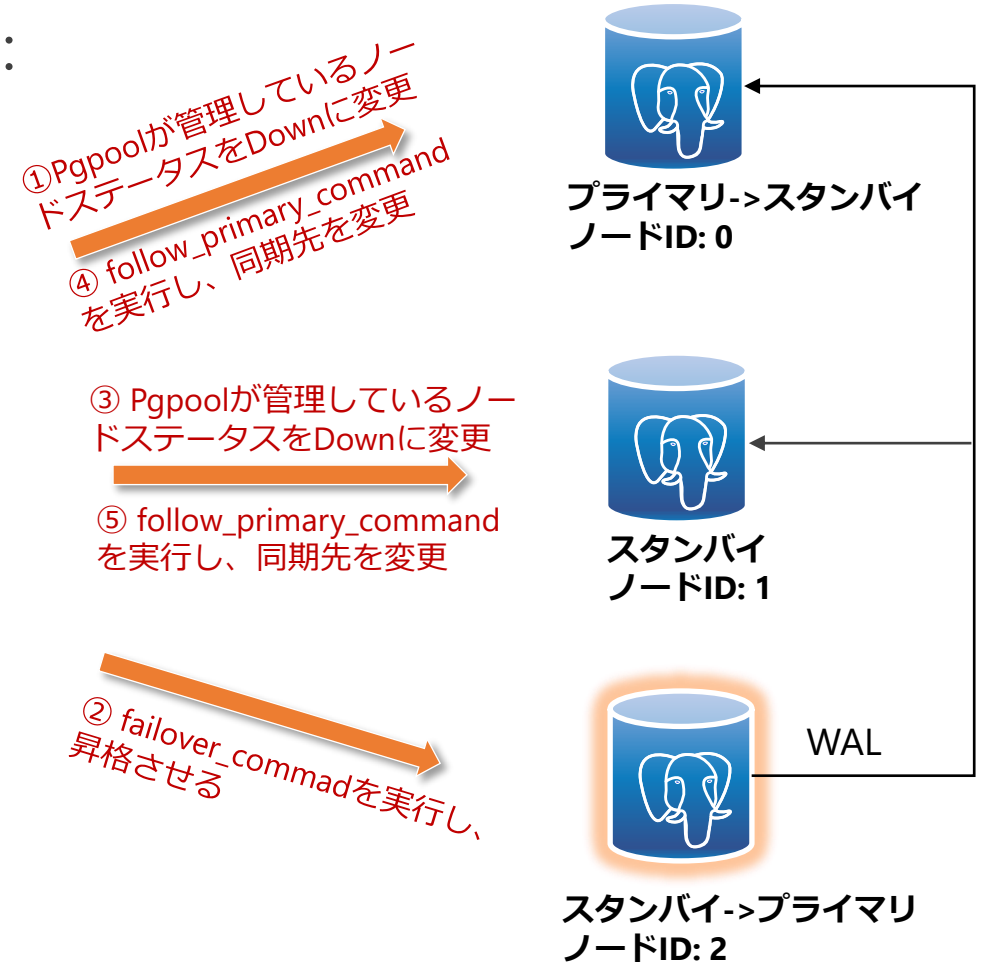
pcp_promote_nodeコマンドに新しいオプション:

-s, --switchover V4.3~

ノード2を昇格させる

```
# pcp_promote_node --switchover --node-id=2
```





v4.3新機能: SHOW POOL_NODESの新しいフィールドの追加

- Pgpool-IIは共有メモリにバックエンドのステータス(status、role)を保存している
- Pgpool-IIが管理しているバックエンドのステータスは実際のバックエンドのステータスと一致しない場合がある
  - 例えば、pcp_detach_nodeにより、バックエンドノードがクラスタから切り離されて、ステータスが「DOWN」になるが、実際は稼働している
- 4.3以降、定期的にバックエンドの情報を取得し、表示できるようになった
  - pg_status: 実際のバックエンドノードのステータス (up/down)
  - pg_role: 実際のバックエンドノードのロール (primary/standby)

v4.2~

```
postgres=> show pool_nodes;
 node_id | hostname | port | status | lb_weight |  role   |
---------+----------+------+--------+-----------+---------+-
 0       | pgpool1  | 5432 | up     | 0.333333  | primary |
 1       | pgpool2  | 5432 | up     | 0.333333  | standby |
 2       | pgpool3  | 5432 | up     | 0.333333  | standby |
```

v4.3~

```
postgres=> show pool_nodes;
 node_id | hostname | port | status | pg_status | lb_weight |  role   | pg_role |
---------+----------+------+--------+-----------+-----------+---------+---------+-
 0       | pgpool1  | 5432 | up     | up        | 0.333333  | primary | primary |
 1       | pgpool2  | 5432 | up     | up        | 0.333333  | standby | standby |
 2       | pgpool3  | 5432 | up     | up        | 0.333333  | standby | standby |
```

v4.3新機能: コネクション統計情報の出力

1. 接続先のデータベース名
2. 接続ユーザ名
3. Pgpool-IIプロセスの起動時刻
4. **プロセスの利用カウンタ** v4.3~
5. プロトコルのメジャーバージョン
6. プロトコルのマイナーバージョン
7. バックエンドへの接続時刻
8. **クライアントが最後に接続開始した時刻** v4.3~
9. **接続がidleとなっている時間(秒)** v4.3~
10. **クライアントが最後に接続終了した時刻** v4.3~
11. 接続の再利用カウンタ値
12. PostgreSQLバックエンドプロセスのプロセスID
13. フロントエンドが接続中なら1、そうでなければ0
14. pgpool子プロセスID
15. PostgreSQLバックエンドID
16. **プロセスの状態** v4.3~

child_max_connectionsに到達する前にクライアントがPgpool-IIに接続した回数を知りたいときに役立つ

client_idle_limitが0でない場合、クライアントが切断されるまでの時間

プロセスの状態:
- Wait for connection
- Execute command
- Idle
- Idle in transaction

```
$ pcp_proc_info -h localhost -p 11001 -P 23493 --verbose
Database                  : test
Username                  : postgres
Start time                : 2021-10-21 14:16:00
Client connection count   : 1
Major                     : 3
Minor                     : 0
Backend connection time   : 2021-10-21 14:16:16
Client connection time    : 2021-10-21 14:17:25
Client idle duration      : 21 (0:39 before client disconnected)
Client disconnection time : 2021-10-21 14:17:23
Pool Counter              : 2
Backend PID               : 23500
Connected                 : 1
PID                       : 23493
Backend ID                : 0
Status                    : Idle
...
```

参考情報

- Pgpool-II
  - https://pgpool.net/
  - https://www.pgpool.net/docs/latest/ja/html/
- リポジトリ
  - https://git.postgresql.org/gitweb/?p=pgpool2.git;a=summary
- ML
  - https://pgpool.net/mediawiki/index.php/Mailing_lists
- バグ報告
  - https://www.pgpool.net/mantisbt/

ご清聴ありがとうございました。

SRA OSS, INC.