pgpool-II 3.4の新機能のご紹介

JPUGコンファレンス
2014年12月5日

SRA OSS, Inc. 日本支社
石井 達夫

SRA OSS,INC.

自己紹介

- PostgreSQLの開発者(コミッタ)
- PostgreSQL用のクラスタソフト “pgpool-II” の最初の開発者、コミュニティリーダー
- SRA OSS, Inc.日本支社で支社長をやってます

SRA OSS, Inc.のご紹介

- 1999年よりPostgreSQLサポートを中心にOSSビジネスを開始、2005年に現在の形に至る
- 主なビジネス
  - PostgreSQL, Hinemos, Zabbixなどの OSSサポート
  - PowerGresファミリーの開発、販売
  - トレーニング、導入、設計コンサルティング サービス

PowerGres

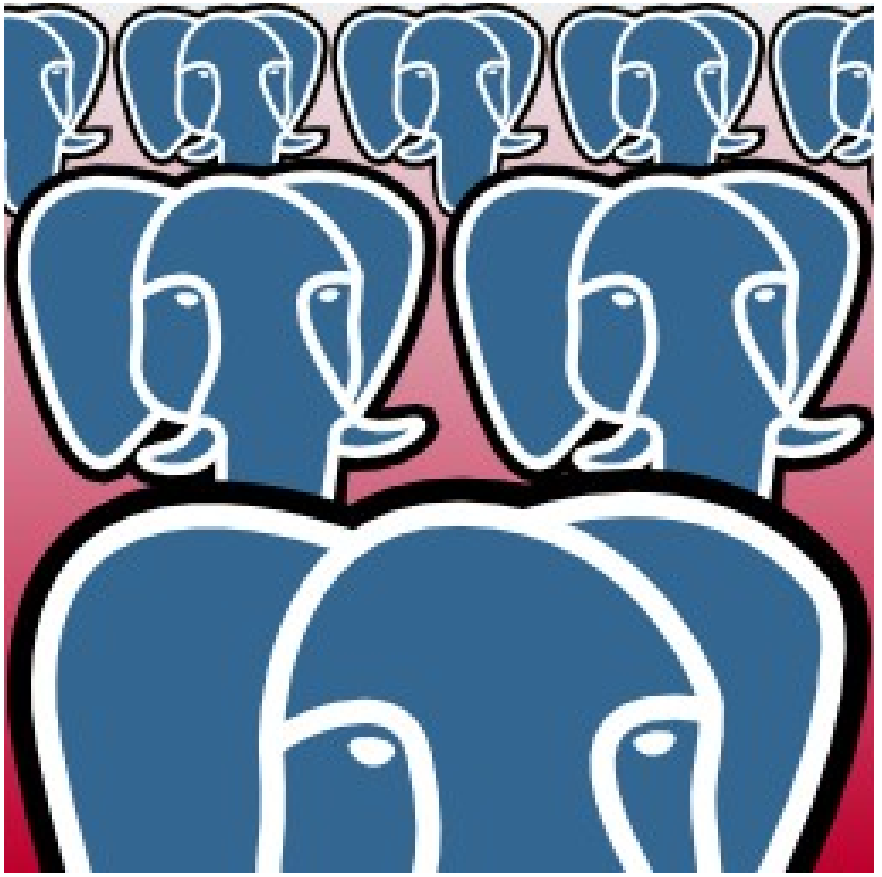
Hinemos

ZABBIX
CERTIFIED PARTNER

SRA OSS,INC.

pgpool-IIとは

- PostgreSQL及びその互換ソフトで動くproxy型のクラスタ管理サーバソフト
- OSSとして公開 (BSDライセンス)
- PostgreSQLから独立したproxyとして動作、アプリケーションの改修は最低限
- 幅広いPostgreSQLのバージョンに対応
- 性能や可用性を高める多くの機能
- メジャーバージョンアップは年に1回、マイナーバージョンアップは2-3ヶ月に1回、最新バージョンは3.4



pgpool-IIによるクラスタの概念(1)

アプリケーションからは普通のPostgreSQLに見える

検索/更新

問い合わせを負荷分散できるかどうかを判断

pgpool-II

検索処理を複数のPostgreSQLで分担することにより、性能向上(負荷分散)

検索
更新

検索

検索

PostgreSQL
primary

PostgreSQL
standby

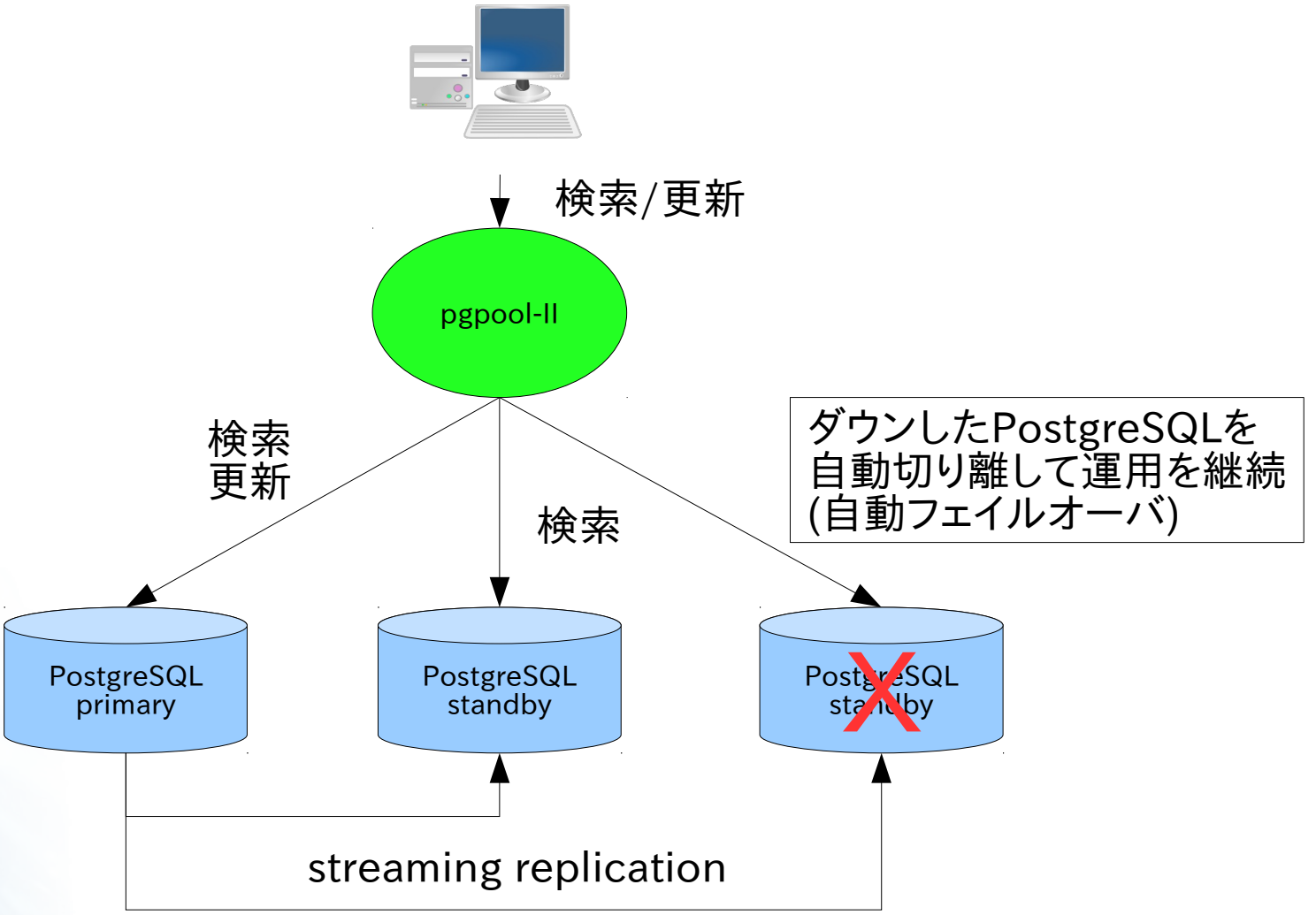
PostgreSQL
standby

更新の伝搬はPostgreSQLの機能を利用することも可能

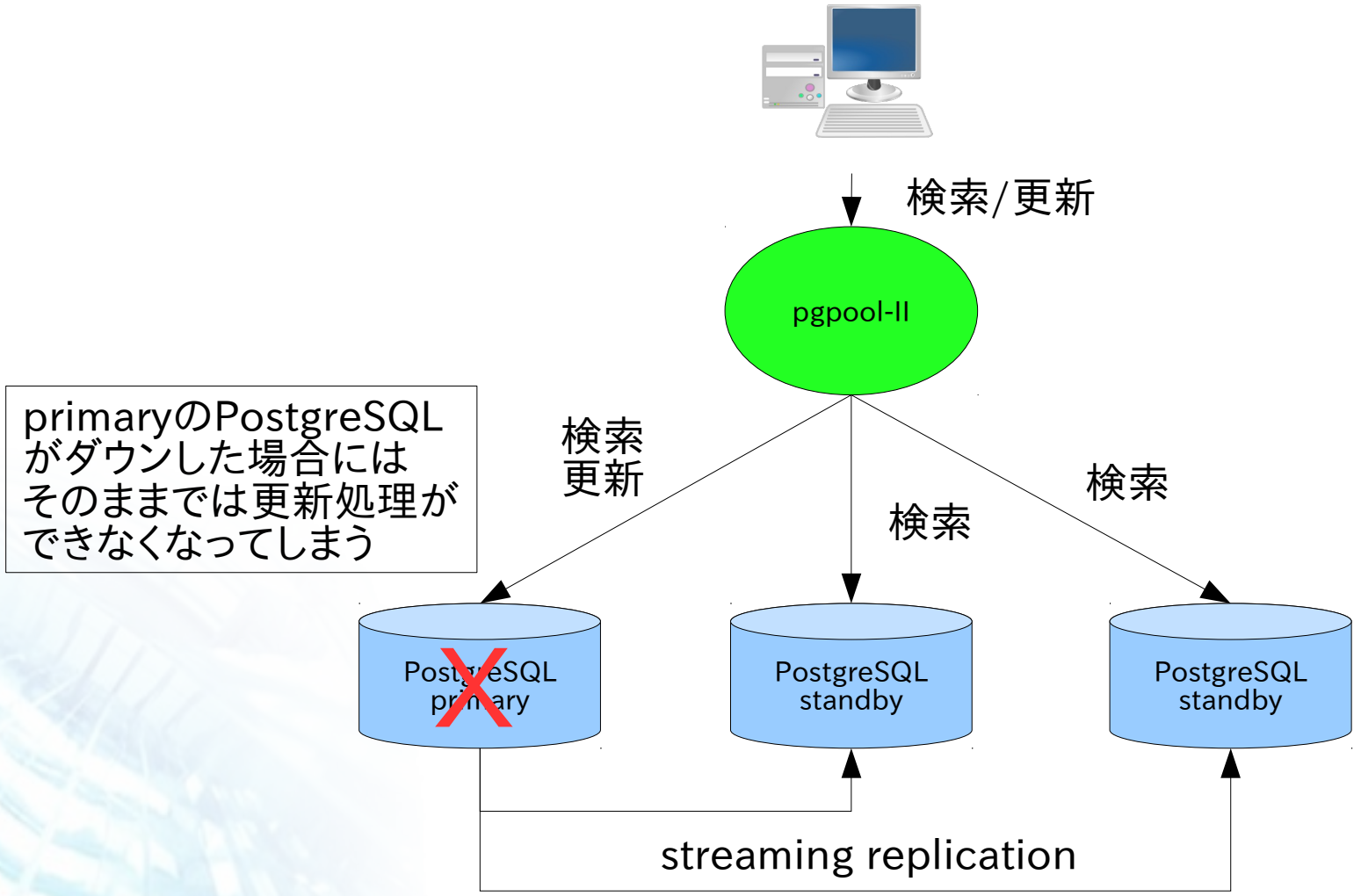
streaming replication

SRA OSS, INC.

pgpool-IIによるクラスタの概念(2)

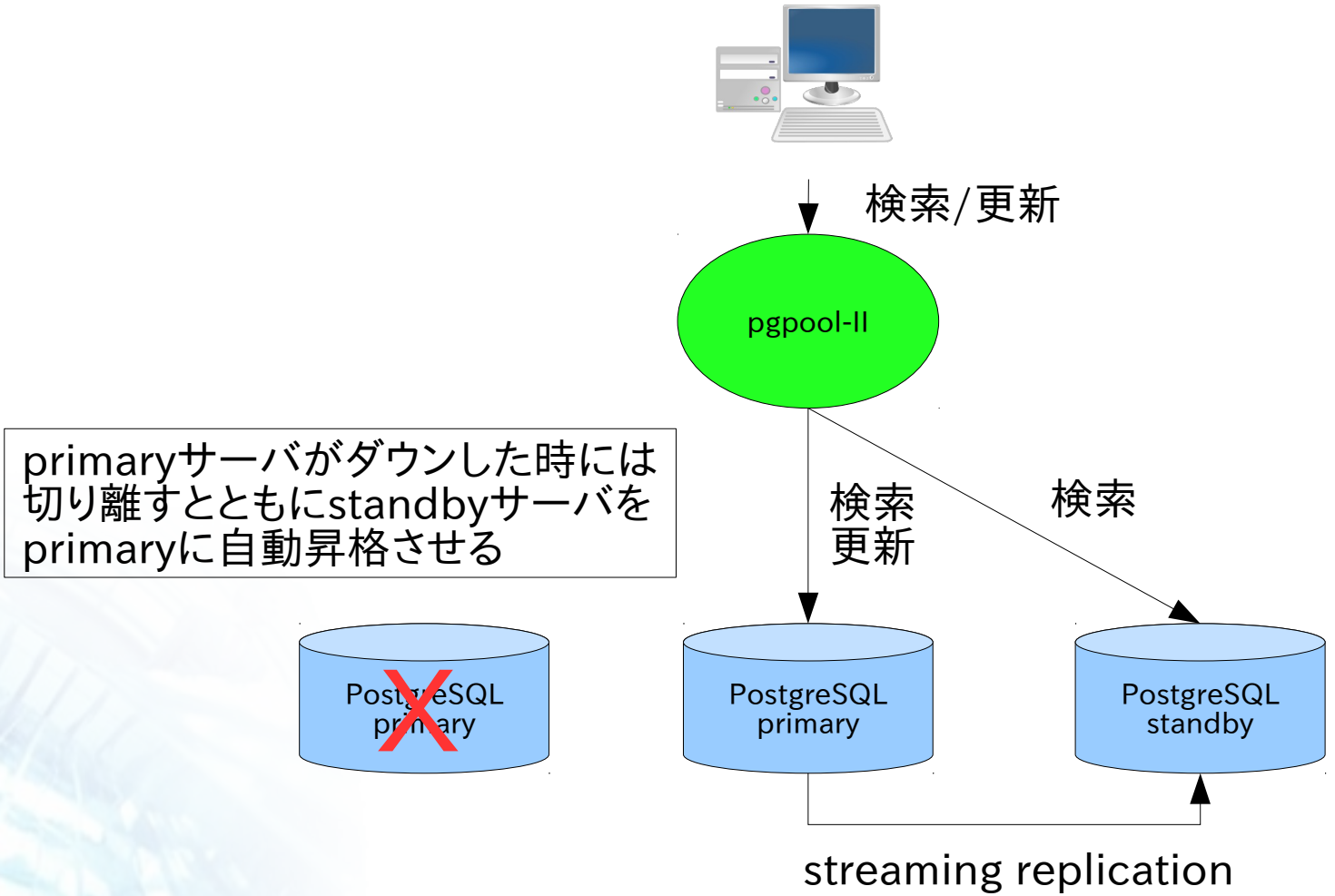


pgpool-IIによるクラスタの概念(3)

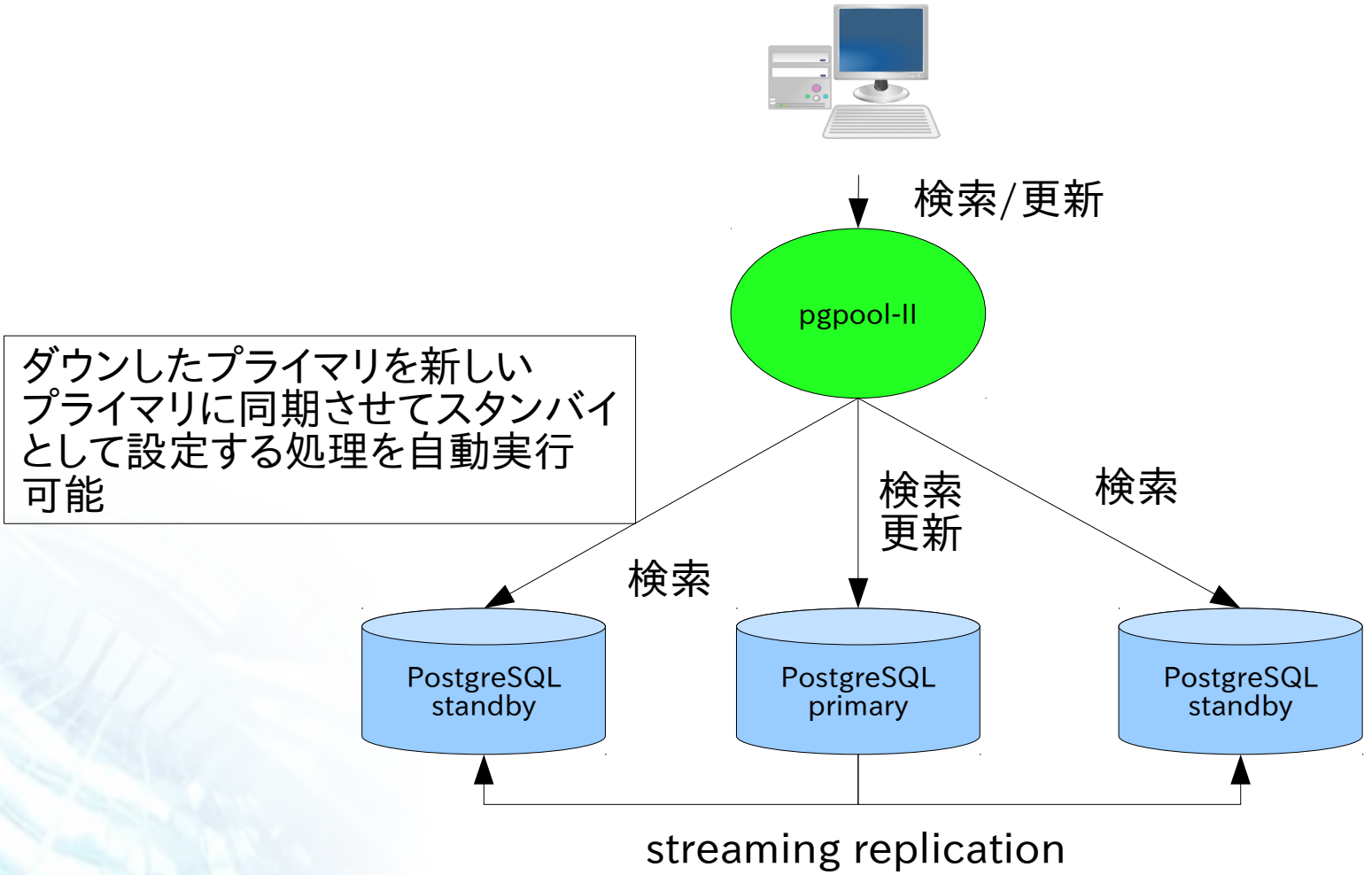


primaryのPostgreSQLがダウンした場合にはそのままでは更新処理ができなくなってしまう

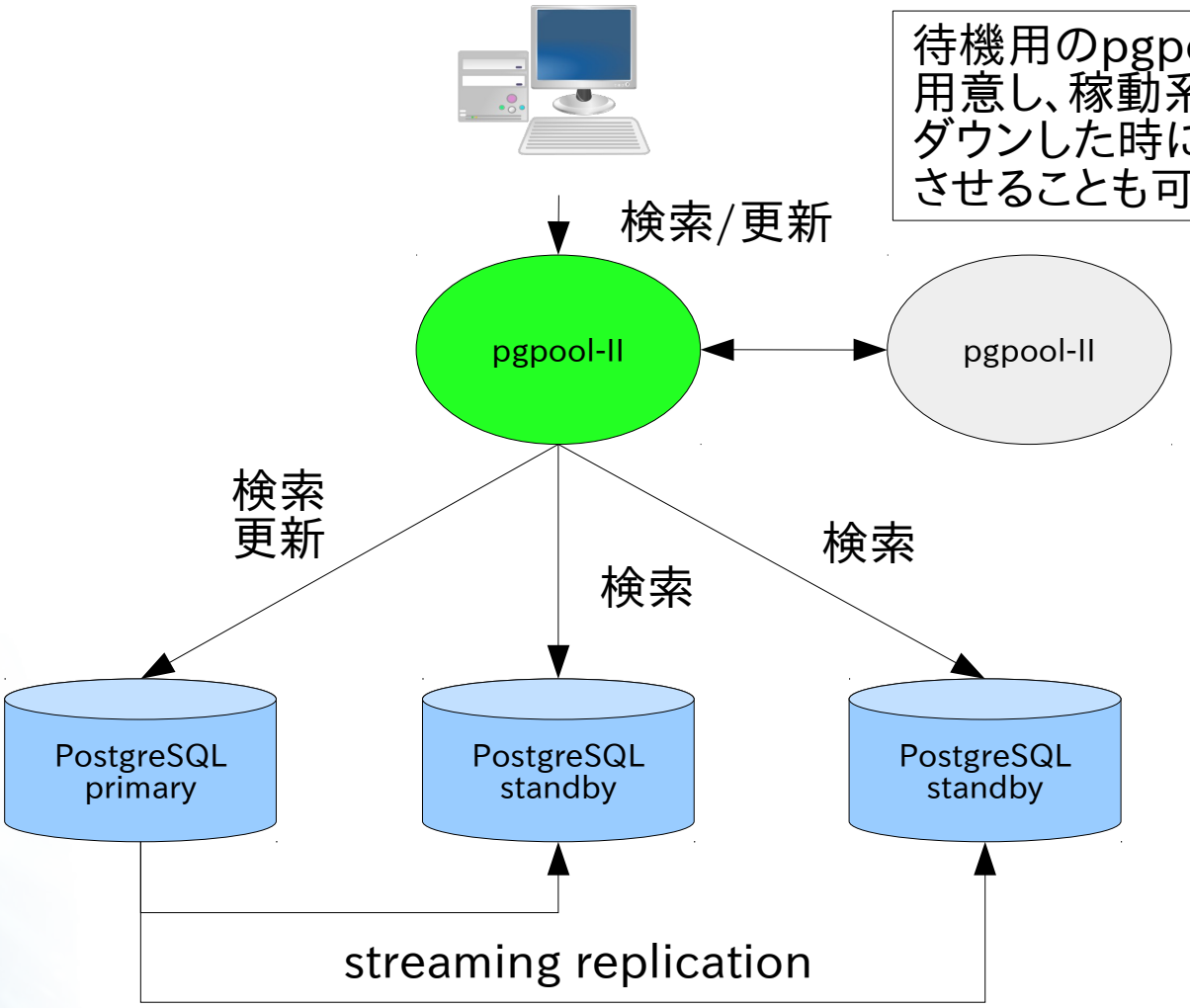
pgpool-IIによるクラスタの概念(4)



pgpool-IIによるクラスタの概念(5)

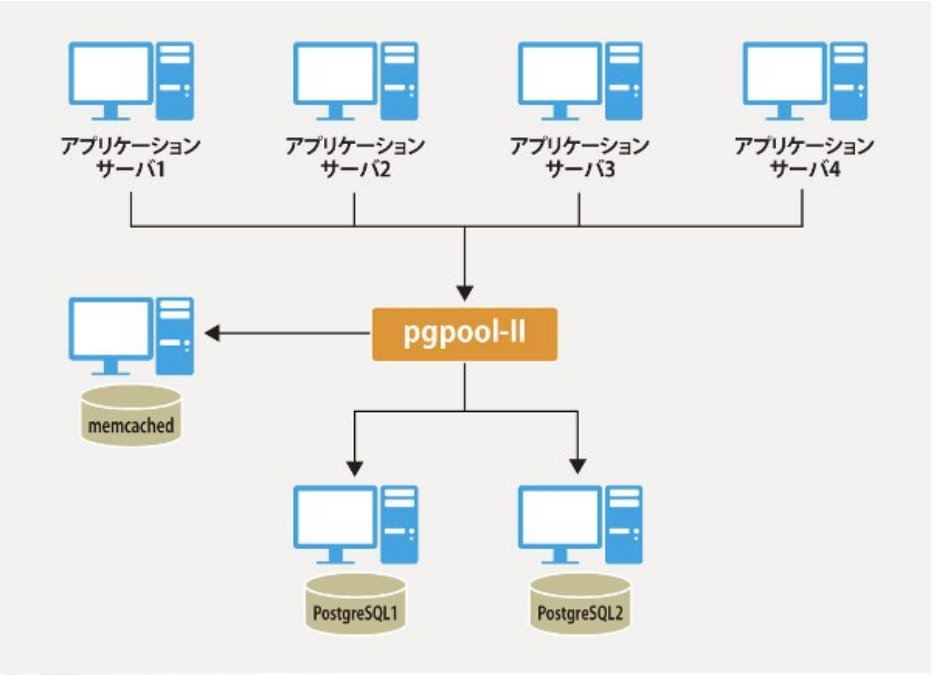


pgpool-IIによるクラスタの概念(6)



pgpool-II活用事例

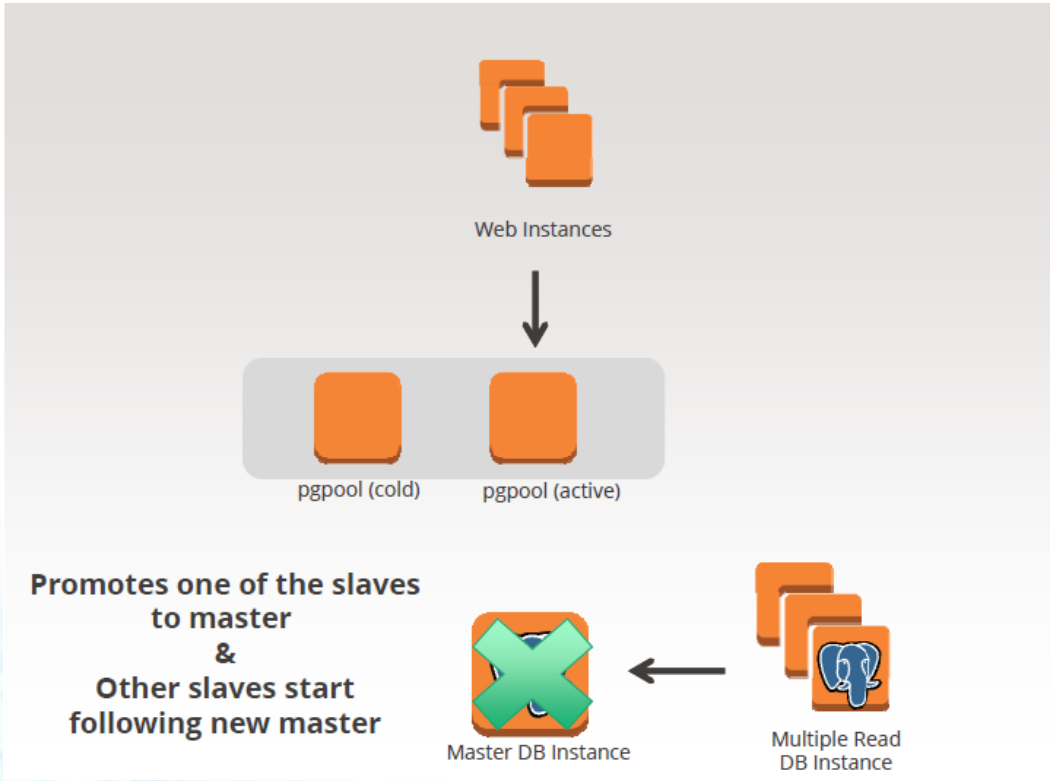
第一法規様事例



http://www.sraoss.co.jp/case_study/daiichihoki.php

- 法律や判例などを検索するシステム
- PostgreSQLのストリーミングレプリケーションを使用、pgpool-IIで負荷分散、可用性向上
- pgpool-IIのインメモリクエリキャッシュ機能を活用して更に検索性能向上
- データ更新頻度が以前は1-2ヶ月ごとだったのが毎日更新になった
- システムの利用者も増加
- 実際に一度発生したDB障害にもpgpool-IIの自動フェイルオーバでシステム全体の運用に影響なし

Gengo様事例



- 世界規模で翻訳サービスを提供
  - YouTubeの字幕翻訳も
- すべてAWS上でシステム構築
- 3万トランザクション/日
- 複数のスタンバイノードで負荷分散
- PostgreSQLのマスタが起動しなくなるトラブルをきっかけに、SPOF解消のためにpgpool-IIを導入
- pgpool-IIの導入とPostgreSQLのバージョンアップを同時にプロダクションシステムで敢行。入念なリハーサルと移行スクリプトの用意で最小限のダウンタイムで無事カットオーバ
- AWSの「リブート祭り」もシステム停止なしで無事乗り切った

pgpool-II 3.4の新機能(1)

- ようやくリリースされたPostgreSQL 9.4にいち早く対応!
  - PostgreSQL 9.4のパーサを取り込み、PostgreSQL 9.4の新しい機能、構文に対応。たとえば...
    - ALTER SYSTEM
    - REFRESH MATERIALIZED VIEW CONCURRENTLY
    - SELECT … WITH ORDINALITY
    - SELECT … FOR UPDATE NO WAIT

pgpool-II 3.4の新機能(2)

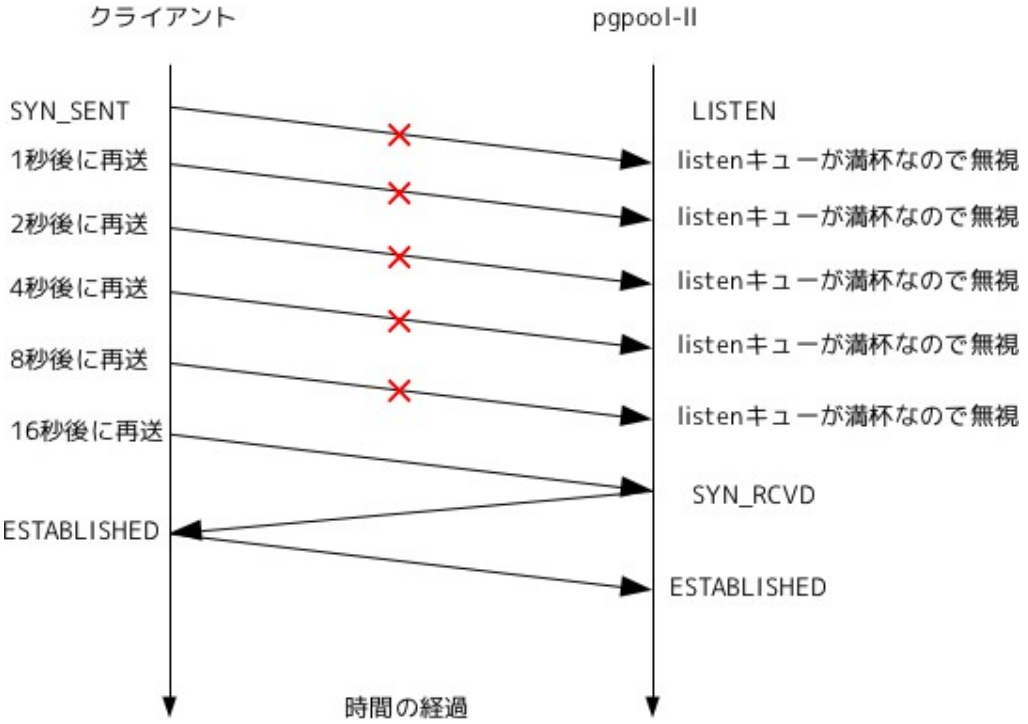
- pgpool_regclassのインストールが不要に
- pgpool_regclassとは
  - pgpool-IIは、テーブルなどの情報を得るためにシステムカタログをアクセスする。そのときにテーブル名からOIDに変換する機能が必要だが、PostgreSQLの組み込み関数regclassでは、存在しないテーブルを検索するとエラーとなり、ユーザのトランザクションがアボートしてしまう
  - pgpool_regclassはエラーを起こさず、テーブルが存在しないことを情報として返す
- PostgreSQL 9.4では、pgpool_regclassと同等の機能を持つ to_regclass が実装され、pgpool_regclass の導入が不要に
  - to_regclass はSRA OSS社員が開発しました!
- pgpool-II 3.4では、to_regclass があればそれを使うようになった
  - ちなみにこの機能は実は pgpool-II 3.3系にもバックポートされています

pgpool-II 3.4の新機能(3)

- check_unlogged_table
  - unloggedテーブルのチェックをパスすることにより、システムカタログへの問い合わせを減らして性能向上
- connect_timeout
  - pgpool-IIからPostgreSQLへの接続待ちの際のタイムアウトを指定する
    - 従来は固定1秒(3.3.3以降では10秒)
    - 仮想環境などで遅延が大きいネットワークでは、1秒ではタイムアウトが不足することがあり、フェイルオーバ発生の原因になっていた
    - 使用するネットワークの状況に応じてきめ細かく設定が可能に

pgpool-II 3.4の新機能(4)

- listen_backlog_multiplier
  - クライアントがpgpool-IIに接続される際の待ち行列(listenキュー)の長さを指定する。listenキューの長さが不足するとpgpool-IIへの接続に極端に時間がかかったり、接続エラーになることがある(右の図参照)
  - デフォルトではlistenキューの長さは、pgpool-II子プロセスの数(クライアントが同時に接続できる数を規定)の2倍。特にクライアント数が多くなるシステムでは、listen_backlog_multiplierを3以上にすることにより、listenキューの長さを3倍、4倍にして無駄な接続待ちを防ぐことができる
  - カーネルパラメータ“somaxcon”を増やすのも忘れずに!



Let's Postgresの記事より引用
http://lets.postgresql.jp/documents/technical/pgpool-II-tcp-tuning/1

pgpool-II 3.4の新機能(5)

- PostgreSQLのメモリ管理機能を導入
  - ライブラリのメモリ管理(malloc)の呼び出しを減らし、かつメモリーリークを防ぐ
  - 性能向上も期待できる
- PostgreSQLの例外処理機能を導入
  - エラーが発生した際のエラー処理を省略できる場合もあり、コーディングが簡素化
  - 良い意味での副作用でPostgreSQL同様のログ管理ができるようになった
    - 従来はERROR, LOG, DEBUGの区別しかなかったが、PostgreSQL同様、DEBUG[1-5], LOG, NOTICE, FATAL, PANICのようにきめ細かくログを管理できるようになった
    - log line prefixが使えるようになった
    - ユーザ名、アプリケーション名などもログできるようになった
    - PostgreSQL同様、log_error_verbosity, client_min_messages, log_min_messages も使えます

EnterpriseDBのMuhamad Usama氏が実装

pgpool-II 3.4の新機能(6)

- 検索負荷分散のきめ細かい指定が可能に
- 従来対応できなかった状況
  - マスタ更新のアプリケーションではレプリケーションの遅延に影響されたくないので、常にプライマリを検索したい
    - 従来はマスタ更新アプリだけでなく、他のアプリも常にプライマリの検索になってしまっていた
  - 大きなデータベースがある。特定のスタンバイノードをこのDBの検索専用にしたい
  - バックアップ(pg_dump)専用のスタンバイノードを設けたい
  - 業務分析の重いクエリを発行するアプリがある。特定のスタンバイノードをこのアプリ専用にしたい
  - マルチテナント的にスタンバイノードの利用を区分したい。アプリグループ1はスタンバイ1、アプリグループ2はスタンバイ2としたい
  - フェイルオーバにより、プライマリノードのID番号が変わってしまったが、検索処理はすべてスタンバイノードに送りたい

pgpool-II 3.4の新機能(7)

- 新しい負荷分散制御用の設定項目を追加
  - database_redirect_preference_list
    - 「DB名:ノード指定」のペアをカンマで区切って複数指定可能
    - DB名の指定には正規表現が利用可能
    - ノード指定は、ノード番号の他、“primary”と“standby”が指定可能(“standby”はスタンバイノードのうちのどれかを示す)
  - app_name_redirect_preference_list
    - 「アプリケーション名:ノード指定」のペアをカンマで区切って複数指定可能
    - アプリケーション名の指定には正規表現が利用可能
    - ノード指定はdatabase_redirect_preference_listと同様
  - database_redirect_preference_listとapp_name_redirect_preference_listの指定が矛盾する場合は、app_name_redirect_preference_listの指定が優先

pgpool-II 3.4の新機能(8)

- 利用例(1)
  - “master_app”というマスタ更新アプリは、レプリケーションの遅延を避けるために常にprimaryから検索をしたい
    - app_name_redirect_preference_list = 'master_app:primary'
  - “bigdb”という大きなDBの検索はスタンバイ2に限定したい
    - database_redirect_preference_list = 'bigdb:2'

pgpool-II 3.4の新機能(9)

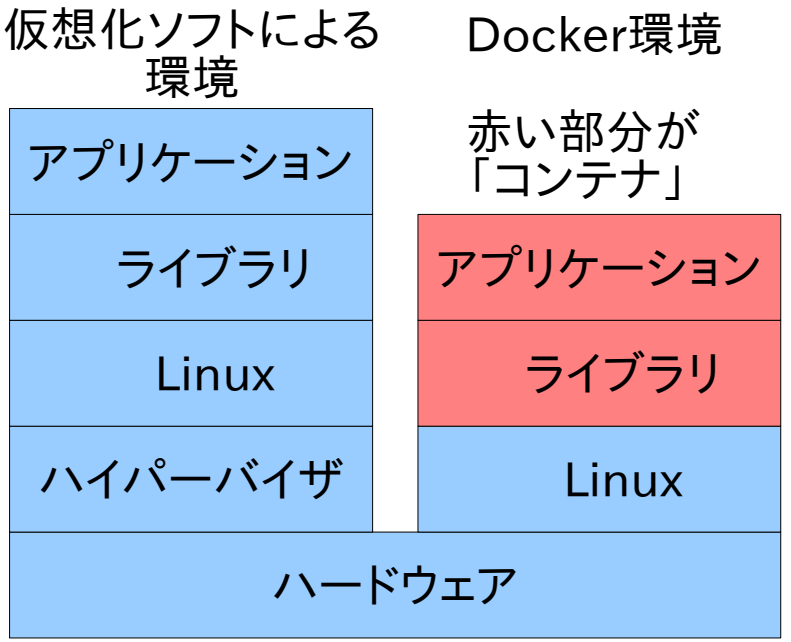
- 利用例(2)
  - バックアップ専用のDBノードをスタンバイ3にしたい
    - app_name_redirect_preference_list ='pg_dump:3'
  - 業務分析の重いクエリを発行するアプリ“heavy_app”がある。スタンバイノード1をこのアプリ専用にしたい
    - app_name_redirect_preference_list = 'heavy_app:1'
  - マルチテナント的にスタンバイノードの利用を区分したい。アプリグループ1(“app1, app2”はスタンバイ1、アプリグループ2(“app3, app4”はスタンバイ2としたい
    - app_name_redirect_preference_list = 'app[1-2]:1,app[3-4]:2'

pgpool-II 3.4の新機能(10)

- 利用例(3)
  - フェイルオーバにより、プライマリノードのID番号が変わってしまったが、検索処理はすべてスタンバイノードに送りたい
    - database_redirect_preference_list = '.*:standby'

pgpool-IIプロジェクトとDockerの関係

- Dockerとは？
  - Linuxカーネルの上で各種リソースを区分けする機能を活用して、仮想化されているように見えるような仕組みを提供する
  - 区分されたオブジェクトは「コンテナ」と呼ばれる
  - コンテナの起動は本質的にプロセスの起動と変わらないので、仮想マシンの立ち上げに比べると非常に高速かつリソース消費が少ない
  - DockerをサポートするLinuxであれば、ディストリビューションに関係なく、誰がコンテナを実行しても同じ結果が得られる
  - pgpool-II的には、使い捨ての環境を簡単に構築、気軽に捨てられる点に着目



Docker活用その1: RPMパッケージのビルド

- RPMパッケージビルドの煩わしさ
  - OSインストール直後のクリーンな環境が必要。普段使っている開発環境でRPMを作るのは良くない
  - かと言って、仮想マシンイメージをたくさん用意しておくのは大変 (pgpool-IIのバージョンxPostgreSQLのバージョンxディストリビューションの種類とバージョン)
- Dockerで解決
  - RPMビルド専用のDockerイメージを用意
  - コマンド一発(docker run)で、pgpool-IIのパッケージをクリーンな環境からビルド
  - 後は必要な組み合わせ分だけdocker runを回すだけ
  - Dockerfile etc.はここにあります
    - https://github.com/tatsuo-ishii/docker-pgpool-II-rpm

Docker活用その2: regression test (回帰テスト)

- pgpool-IIのregression test
  - pgpool-II、PostgreSQLをインストール、ストリーミングレプリケーションの設定なども行い、実際にフェイルオーバを起こさせるなど、実際のクラスタに近い環境でregression testを実施する
  - 便利だが、実行にはPostgreSQLの他色々なツールを揃えておく必要がある
    - JDBCドライバ、pgbench、memcached...
- Dockerで解決
  - regression test専用のDockerイメージを用意
  - コマンド一発(docker run)で、pgpool-IIとPostgreSQLをインストールしてregression testを実施する
  - Dockerfile etc.はここにあります
    - https://github.com/tatsuo-ishii/docker-pgpool-II-regression
    - Dockerコンテナ内の共有メモリを増やすのに苦労しました

Docker活用その3: build farm

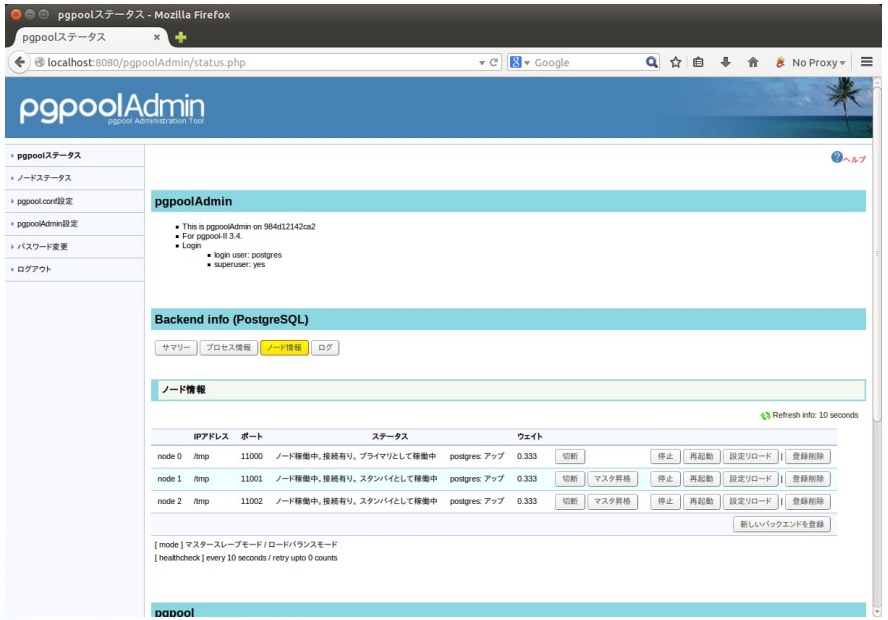
- PostgreSQLのbuild farmをご存知ですか？
  - 毎日違ったハードウェア、OSでPostgreSQLのregression testを実行、特定の環境で起こる不具合をキャッチできるという大変素晴らしい仕組み
  - pgpool-IIでも同じこと(のミニマム版)がやりたい！
- Dockerで解決
  - 前のスライドのregression test専用のDockerイメージを活用
  - 異なるpgpool-IIのバージョンを(正確にはgitのブランチ)のregression testを毎日実行
  - 今後はLinuxディストリビューションの種類を増やしていく予定

```
pgpool-II buildfarm
start:  Mon Nov 24 07:41:49 JST 2014

* Target branch: master

PostgreSQL: 9.3.5
OS: CentOS release 6.6 (Final) (3.13.0-24-generic)

** Regression test

testing 001.load_balance...ok.
testing 002.native_replication...ok.
testing 003.failover...ok.
testing 004.watchdog...ok.
testing 005.jdbc...ok.
testing 006.memqcache...ok.
testing 007.memqcache-memcached...ok.
testing 008.dbredirect...ok.
testing 009.sql_comments...ok.
testing 050.bug58...ok.
testing 051.bug60...ok.
testing 052.do_query...ok.
testing 053.insert_lock_hangs...ok.
testing 054.postgres_fdw...ok.
testing 055.backend_all_down...ok.
testing 056.bug63...ok.
testing 057.bug61...ok.
testing 058.bug68...ok.
testing 059.bug92...ok.
out of 19 ok:19 failed:0
```

Docker活用その4: 簡単検証(デモ)環境の構築

- pgpool-IIには、簡単に検証(デモ)環境を作れるように、コマンド一発でpgpool-II+PostgreSQLの環境を1台のマシン内に作る“pgpool_setup”というツールが付属しています
  - 実はregression testもこのツールを使っています
- そこでpgpool_setupを応用して、pgpool-II+PostgreSQL 3台のストリーミングレプリケーション環境+pgpoolAdmin (pgpool-II用のWeb管理ツール)を一つのコンテナに押し込んでみました

システム構成図



Dockerコンテナ

port = 8080
にマップ

pgpoolAdmin
port = 80

pgpool-II
port = 11003

PostgreSQL #1
port = 11000

PostgreSQL #2
port = 11001

PostgreSQL #3
port = 11002

psql

SRA OSS, INC.

今後の予定

- 現在次期バージョン3.5の機能を検討中
  - 性能改善
  - PostgreSQL 9.5対応
  - pcpコマンド(pgpool-IIの管理コマンド)の大改善
  - showコマンドで取得できる情報の追加
    - 負荷分散の状況
- 開発に参加したい方を募集しています。ドキュメントの整備・翻訳作業をしたい方も大歓迎!

まとめ

- pgpool-IIの概要
- pgpool-II導入事例
- pgpool-II 3.4の新機能のご紹介
- Docker vs. pgpool-II
- 今後の予定

参考URL

- SRA OSSのサイト(多数のスライドや事例、技術情報あり)
  - http://www.sraoss.co.jp
- pgpool-IIオフィシャルサイト
  - http://www.pgpool.net
  - http://www.pgpool.net/mediawiki/jp/index.php/%E3%83%A1%E3%82%A4%E3%83%B3%E3%83%9A%E3%83%BC%E3%82%B8
- PostgreSQLに関する各種技術情報
  - http://lets.postgresql.jp

ご清聴ありがとうございました