

多機能コネクションプールサーバ

pgpool

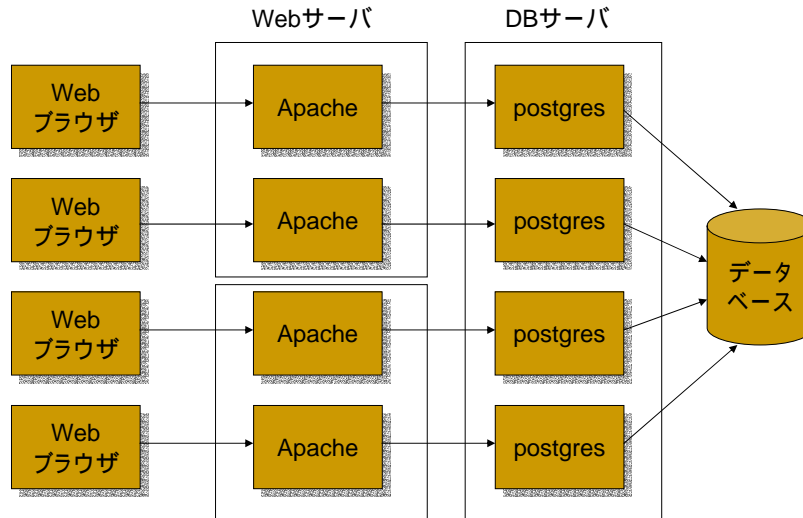
使いこなし術

(株)SRA
OSSビジネスプロジェクト
石井達夫

アジェンダ

- コネクションプーリングの必要性
- レプリケーションの必要性
- pgpoolとは
- 使い方
- システム構成例
- 今後の予定

Webアプリケーションの構成と問題点



2005/5/28札幌

Copyright(2005) Tatsuo Ishii

2

コネクションプーリングの必要性

- 性能向上
- DB負荷の軽減

2005/5/28札幌

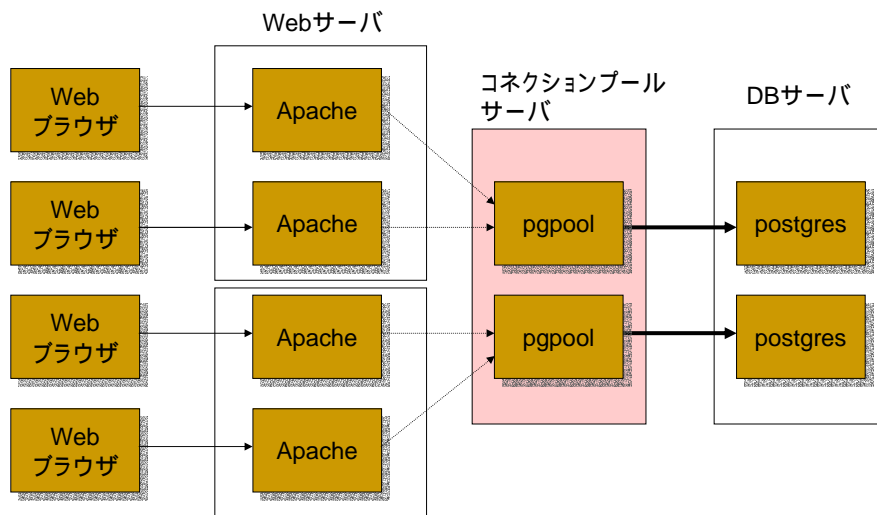
Copyright(2005) Tatsuo Ishii

3

Webアプリケーションの問題点とその解決方法

- コネクションオーバヘッド
 - コネクションプーリングにより解決
 - パーシスタントコネクションでは限界がある
- サーバプロセス数の増加による負荷増大
 - コネクションプーリングサーバにより接続数のコントロール

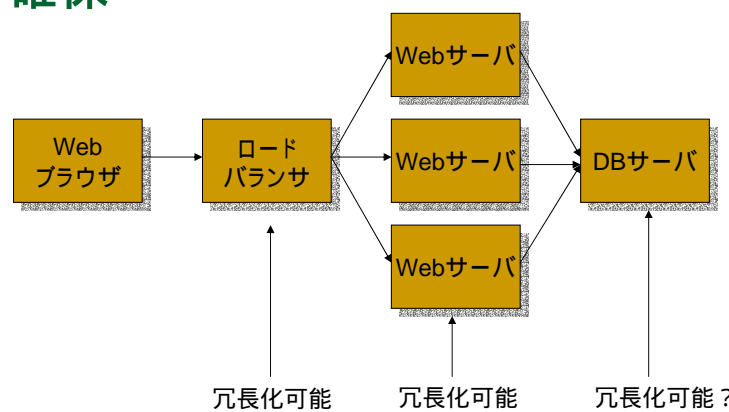
コネクションプールサーバの導入



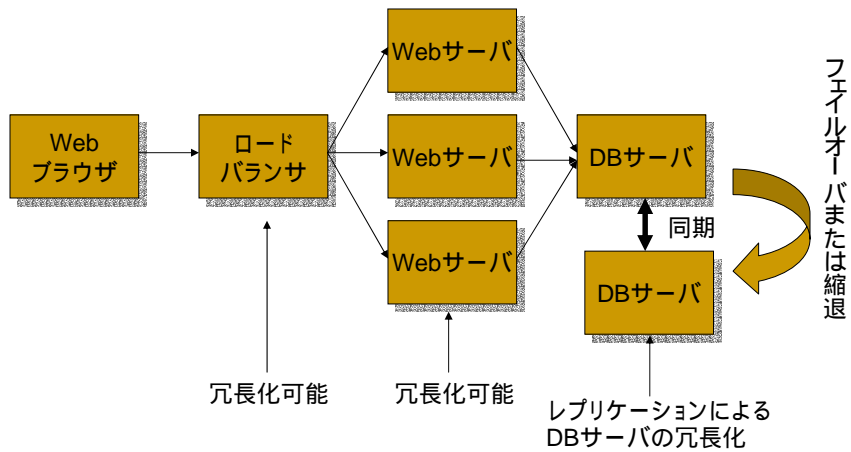
レプリケーションの必要性

- 可用性の向上
- 負荷分散

Webアプリケーションにおける可用性の確保



Webアプリケーションにおける可用性の確保

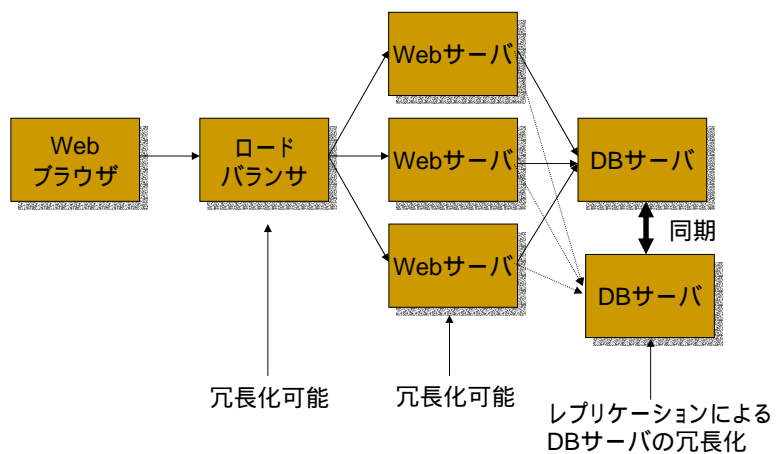


2005/5/28札幌

Copyright(2005) Tatsuo Ishii

8

レプリケーションを活用した負荷分散



2005/5/28札幌

Copyright(2005) Tatsuo Ishii

9

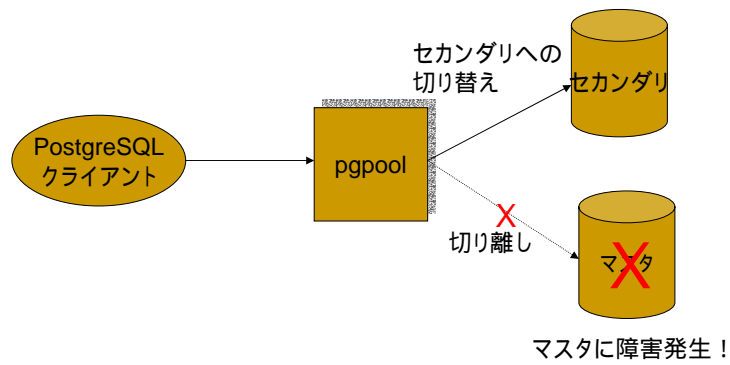
pgpoolとは

- PostgreSQL用多機能ミドルウェア
 - コネクションプール
 - フェイルオーバ
 - レプリケーション
 - ロードバランシング(負荷分散)
- APIを選ばない
- 対応PostgreSQLバージョン
 - PostgreSQL 6.4以降(8.0含む)
- トランザクション、ラージオブジェクト、DDL、一時テーブル、主キーなしテーブルに対応

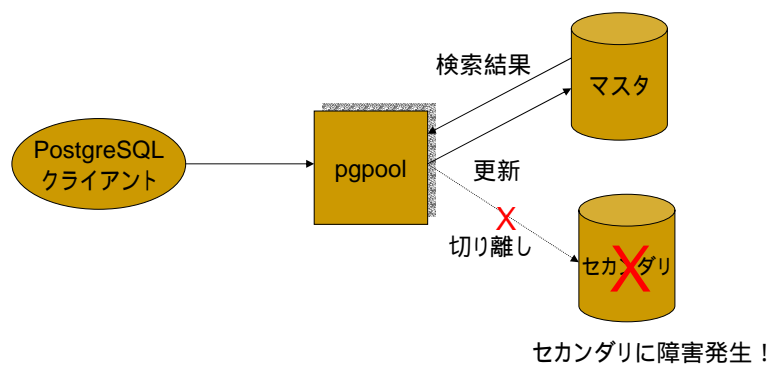
pgpoolの動作モード

- コネクションプールモード
 - シングルサーバ
 - dualサーバ(フェイルオーバ対応)
- レプリケーションモード
 - コネクションプールあり/なし
 - 負荷分散あり/なし
 - 縮退運転可能
- マスタ/スレーブモード
 - 更新はマスタのみ
 - コネクションプールあり/なし
 - SELECTは負荷分散可能

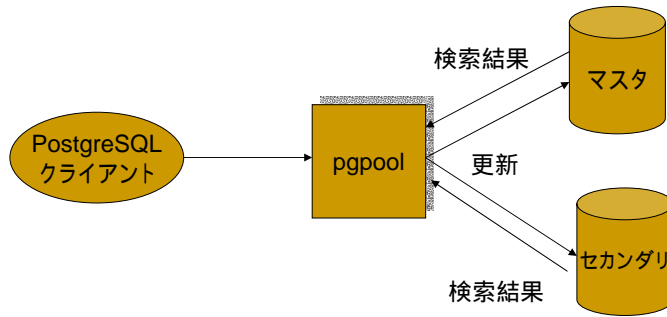
コネクションプールモード



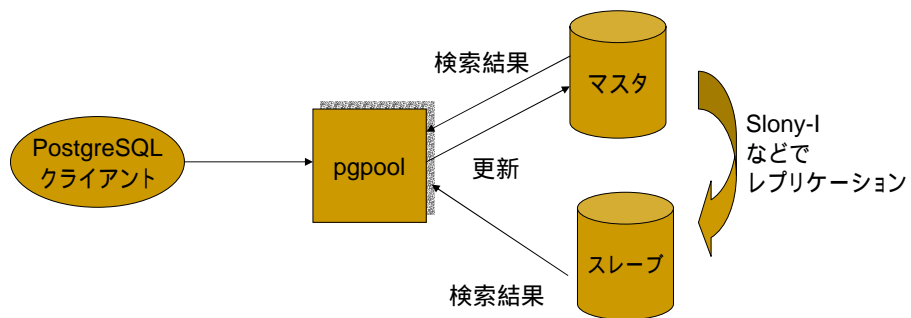
レプリケーションモード



ロードバランシング



マスタ/スレーブモード



使い方: 起動と停止

- 起動
 - pgpool [-f 設定ファイル] -n -d
- 停止
 - フロントエンドがセッションを終了するまで待つ
 - pgpool [-m smart] stop
 - 強制終了
 - pgpool -m fast stop
 - pgpool -m immedeitate stop

共通設定項目(1)

- port
 - pgpoolが使用するポート番号
- allow_inet_domain_socket
 - TCP/IP経由での接続を許可するかどうか(セキュリティ)
- logdir
 - pgpool.pidを置くディレクトリ。/tmp以外にすることを推奨
- print_timestamp
 - ログにタイムスタンプを付与するか
- num_init_children
 - 子プロセスの数 = pgpoolが同時に受け付ける接続の最大数。
これを超えると待たされる

共通設定項目(2)

- child_life_time
 - アイドル子プロセスの寿命
- reset_query_list
 - pgpoolクライアントとの接続を切るときにバックエンドに送る問い合わせリスト
- health_check_period
 - ヘルスチェックの間隔
- health_check_timeout
 - これ以上の時間応答がなければエラーと見なす
- health_check_user
 - ヘルスチェック接続のユーザ。存在しないユーザやパスワードがエラーのユーザでも構わないが、PostgreSQLにログが出る

コネクションプールモードでの設定

- backend_host_name
 - PostgreSQLホスト名もしくは"
- backend_port
 - PostgreSQLポート番号
- connection_life_time
 - アイドルコネクションの寿命
- max_pool
 - 各プロセスあたりのコネクションキャッシュの数。
PostgreSQLへの最大接続数 = num_init_children * max_pool

レプリケーションモードでの設定項目

- replication_mode
 - trueでレプリケーション有効
- secondary_backend_host_name
 - セカンダリPostgreSQLのホスト名
- secondary_backend_port
 - セカンダリPostgreSQLのポート番号
- replication_strict
 - マスタの問い合わせが完了するまでセカンダリに問い合わせを送信しない
- replication_timeout
 - replication_strictがfalseのときに発生しうるデッドロック検出タイム
- replication_stop_on_mismatch
 - PostgreSQLの検索結果の不一致を引き金に縮退する

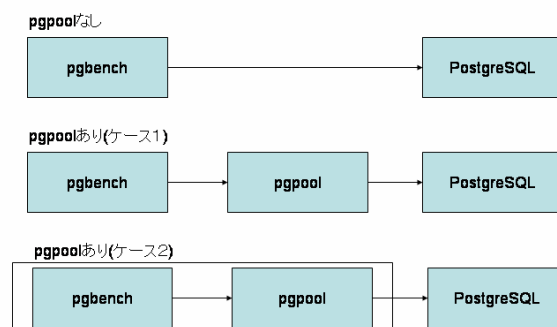
マスタスレーブモードでの設定項目

- master_slave_mode
 - trueならばマスタスレーブモード

ロードバランスモードの設定項目

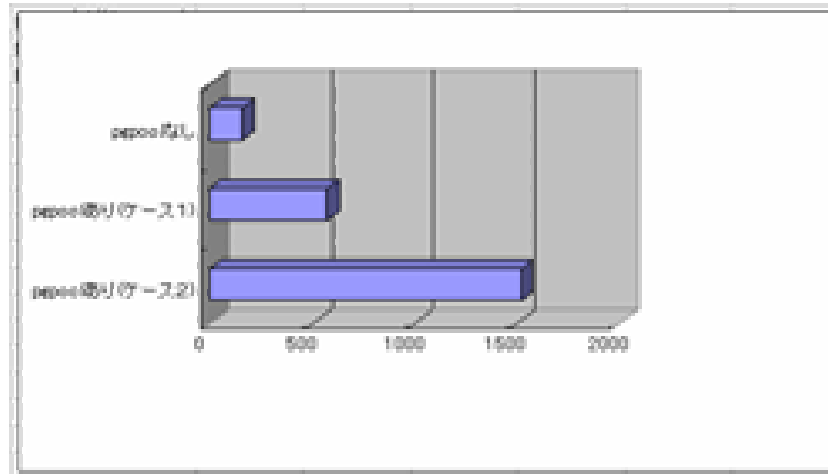
- レプリケーションモードまたはマスタスレーブモードで利用可能
- weight_master
 - マスタの負荷
- weight_secondary
 - セカンダリの負荷

TIPS 1: クライアントとpgpoolの間の接続はUNIXドメインソケットで

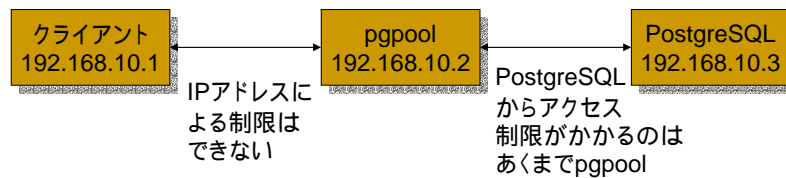


コネクションキャッシュの効果

■pgpoolを使わない場合に比べ、3倍から10倍近い性能向上効果



TIPS 2: pg_hba.confの書き方に注意



TIPS 3: 使用可能認証方式

	コネクションプール モード	レプリケーション モード	マスタスレーブモード
trust			
clear text password			
crypt		×	×
md5		×	×
ident			
ssl	×	×	×
krb4/5	×	×	×

2005/5/28札幌

Copyright(2005) Tatsuo Ishii

26

TIPS 4:レプリケーションモードにおける シーケンス/シリアル型

- そのままではレプリケーションモードのpgpoolでは使えない
 - pgpoolでは、異なるセッションどうしのトランザクションの実行順までは制御できない
 - その結果、マスタとセカンダリでシリアル値の不一致が生じる
- テーブルロックを併用することにより解決
- ただし、トランザクションの並列実行性は犠牲になる

2005/5/28札幌

Copyright(2005) Tatsuo Ishii

27

SERIAL型を含むテーブルのレプリケーション失敗例

```
セッション1      セッション2
マスタ   セカンダリ マスタ   セカンダリ
:         :           :         :
INSERT 1           :         :
:         :           INSERT 2 :
:         :           :         :
:         :           INSERT 2 :
:         INSERT 1   :         :
:         :           :         :
```

```
CREATE TABLE t1(
key SERIAL,
i INTEGER);

/* セッション1 */
INSERT INTO t1(i) VALUES(1);

/* セッション2 */
INSERT INTO t2(i) VALUES(2);
```

マスタ		セカンダリ	
key	i	key	i
1	1	1	2
2	2	2	1

テーブルロックの例

```
CREATE TABLE t1(key SERIAL, i INTEGER);

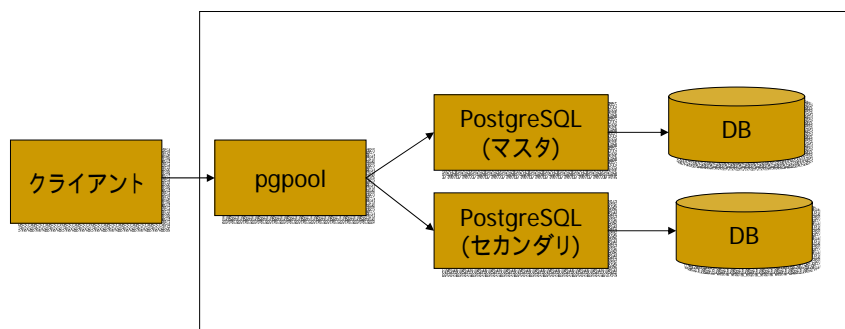
/* セッション1 */
BEGIN;
LOCK TABLE t1 IN SHARE ROW EXCLUSIVE MODE;
INSERT INTO t1(i) VALUES(1);
END;

/* セッション2 */
BEGIN;
LOCK TABLE t1 IN SHARE ROW EXCLUSIVE MODE;
INSERT INTO t2(i) VALUES(2);
END;
```

TIPS5: マスタとセカンダリで一時的にデータが一致しなくなることの回避

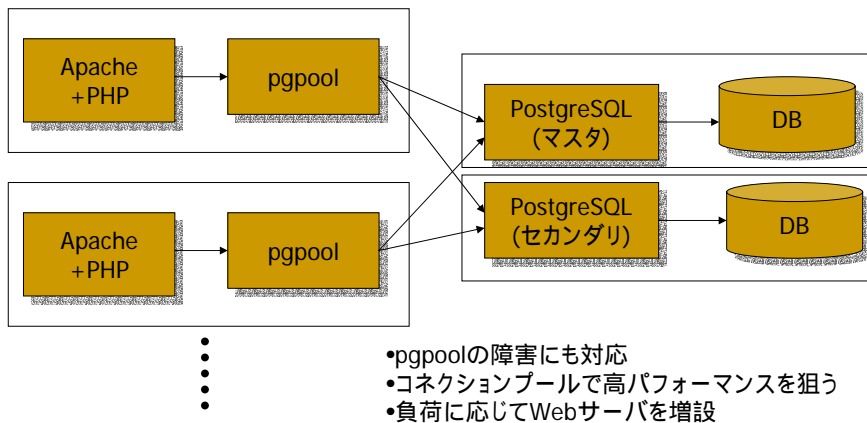
- マスタが更新されてからセカンダリが更新されるため、その間データが一致しなくなる(すべてのトランザクションが完了すればデータは一致する)
- ロードバランスモードで問題になる
- 更新中はテーブルロックをかけ、セカンダリの更新が終わるまではSELECTできないようにすることで回避
- トランザクションの並列実行性が犠牲になるのが難点

システム構成例(1)



•メディア障害に対応できる最小構成

システム構成例(2)

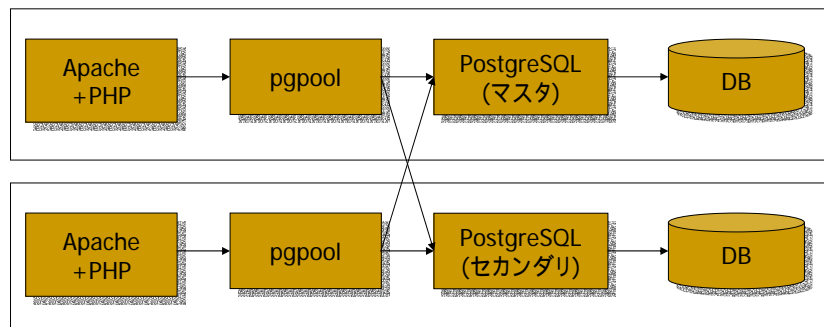


2005/5/28札幌

Copyright(2005) Tatsuo Ishii

32

システム構成例(3)



2005/5/28札幌

Copyright(2005) Tatsuo Ishii

33

今後の予定

- 3台以上のPostgreSQLのサポート
- 分散問い合わせ + パラレルクエリ
- リカバリ機能

参考文献 / URL

- まるごとPostgreSQL! Vol.1
- PostgreSQLウォッチ
 - <http://itpro.nikkeibp.co.jp/members/ITPro/oss/20040709/147053/>
 - <http://itpro.nikkeibp.co.jp/members/ITPro/oss/20040817/148690/>
- PostgreSQL公式サイト
 - <http://www.postgresql.org>
- 日本PostgreSQLユーザ会
 - <http://www.postgresql.jp>
- リリースごとのPostgreSQLの技術情報
 - <http://osb.sra.co.jp>

ご清聴ありがとうございました