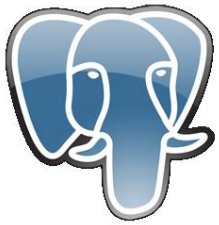


【 C3 】 PostgreSQL 運用テクニック・レベルアップ編

PostgreSQL Conference 2012

2012.2.24

NTT OSSセンタ 坂本昌彦

自己紹介

- 坂本 昌彦
  - NTT OSSセンタ勤務
    - PostgreSQLトラブルシューター
    - PostgreSQLツールメンテナ（pg_bulkloadなど）
    - PostgreSQL研修
  - 著書
    - 「PostgreSQL徹底入門 第3版」
  - 興味
    - PostgreSQL
    - Hadoop
    - NoSQL DBs（MongoDB, Redis, HBase…）
    - AppEngine
    - Python（WSGI, Twisted…）

今日の話題

- 運用について語ります
  - PostgreSQL 運用の考え方
  - 安定運用「実践」に向けて
  - 運用をサポートするツール

- ツールを使った運用のデモをします
  - 性能監視 – pg_statsinfo を使って
  - バックアップ – pg_rman を使って
  - オンライン再編成 – pg_reorg を使って
  - 高速データロード – pg_bulkload を使って

想定している聴衆

- PostgreSQL DBA
  - 企業システムにPostgreSQLを導入する技術者
- OSS-DB Silver相当
  - 以下のPostgreSQL用語を聞いて、その指し示す意味が理解できる程度
    - VACUUM, pg_dump, PITR, EXPLAIN...

その他、お知らせ

- 本日のスライド
  - JPUGサイトにアップロード予定
- 相補関係となるスライド
  - 「具体的な運用TIPS」は、以前発表したものが有用
    - 「PostgreSQL運用テクニック」
    - http://www.postgresql.jp/events/event_sozai/Summer_seminar2011_Operation_technique.pdf/view
- 本日のデモ関連のスクリプト
  - github で公開
    - git clone git://github.com/sakamotomsh/pgcon12jc3.git

PostgreSQL 運用の考え方

運用設計の考え方

- 目標の置きかた
  - 必要かつ十分な運用設計
    - コストのかけすぎ　⇒　(ﾟДﾟ) ﾏｽﾞｰ
    - 検討不足・いきあたりばったり ⇒ (ﾟДﾟ) ﾏｽﾞｰ
    - 業務要件にあった、必要十分な運用 ⇒ (ﾟдﾟ)

- 安定運用を実現するために
  - そもそも、何で失敗するのか？
    - 色々つくりこんで、バグも作りこむ
    - そもそも思いもつかなかった落とし穴にはまる
  - OSSのツールを使う
    - 作りこまれて、バグが少ない
    - 色々なケースが検討されている（ことが多い）
  - 類似プロジェクトの設計書・ツールを使う
    - ※ただし成功したPJに限る

DBAに求められる運用項目

- 監視・レポート
  - 死活監視
  - リソース監視・性能監視
  - 統計情報確認
- 性能維持
  - メンテナンスコマンド実施
- サービス管理
  - 停止 / 再起動
  - フェイルオーバ
  - プロモート
- 性能分析 / トラブルシュート
  - クエリキャンセル
  - ボトルネック調査
  - 実行計画分析
- バックアップ/リストア
  - バックアップ
  - PITR
- アップデート
  - セキュリティアップデート
  - アップグレード

デモ

デモ

デモ

バックアップを例に（１／４）

- バックアップ方法として何を選ぶ？？
  - お手軽な pg_dump ？
  - PITR対応、安心の「物理バックアップ（オンライン）」？
  - 新機能「pg_basebackup コマンド」（Ver9.1～）？
  - 付加価値つきの外部ツール「pg_rman」？
- 番外編
  - レプリケーションでよい？？オペミスは戻せないけど…

・駒をたくさん持つ
・駒の特性を把握する

お手軽　定番　カンタン

	論理バックアップ	物理バックアップ（オフライン）	物理バックアップ（オンライン）	物理バックアップ（オンライン libpq）	外部ツール	レプリケーション
バックアップコマンド	pg_dump / pg_dumpall	cp, tar rsyncなど	pg_start_backup () cp, tar,rsyncなど	pg_basebackup	pg_rman	（ストリーミングレプリケーション）
リストアコマンド	pg_restore / psql	特に無し	（PITR機能）	（PITR機能）	pg_rman	-
売り文句	・手順が容易 ・リモートで実施可 ・DB単位でバックアップ	・手順が容易	・標準的な方法	・手順が容易 ・リモートで実施可	・手順が容易 ・複数世代管理	・簡単
どこまで戻せる？	バックアップ時点	バックアップ時点	任意の時点	任意の時点	任意の時点	最新の時点のみ
レプリカになれる？	-	-	なれる	なれる	（なれる）	なれる
リモートで実施可能？	リモート可	-	-	リモート可	-	リモート可
スナップショット機能と連携可？	-	可能	可能	-	-	-
注意点など	設定ファイルなど、別途バックアップが必要	・サーバを停止させる必要がある				厳密にはバックアップではない。オペミス/改竄は防げない

バックアップを例に（２／４）

要件を確認

・要件を引き出す
・コストと天秤にかける

私「pg_dumpでバックアップを取りたいのですが…何かあったときは前日の AM12:00 の状態にDBが戻ってしまいます」

お客様「ありえないでしょ。それは。」

私「…」

私「（やっぱPITR対応のバックアップ方式が必要か…）」

バックアップを例に（３／４）

- オンラインバックアップを採用
  - なかなか難しい。スクリプト化に大きな工数

・「想定外」をなくす
・ツールで実現できるか検討する

  - ツールを利用する
    - pg_basebackup
    - pg_archivecleanup

バックアップを例に（４／４）

- もっとすごいツールを使う
  - pg_rman はすごい
    - ここがすごい
      - 簡単にオンラインバックアップ
      - 簡単にリストア（PITR前準備）
      - 複数世代管理
      - 増分バックアップ
  - 要件とマッチすれば、非常に有用です
  - 後でデモします

いったんまとめ

- 安定運用を実現するために
  - 何をするべきかを明らかにする
    - （運用項目の明確化）何があるかを把握する
    - （実現方針の決定）要件を引き出し、おとしどころを探る
    - （実現方式の決定）手持ちの駒から、実現性の高いものを選ぶ
  - 失敗しないために
    - ノウハウが詰まった、ツールを活用する。
    - ノウハウが詰まった、過去の類似PJの資料を探す。

メッセージ

- いち技術者として
  - プロジェクトでは
    - 現実路線。これは仕方ない。
  - 技術者としては…妥協しない
    - 幅を広げる
      - パターン・アンチパターンを知る
      - 過去の蓄積を類型化する
    - 知識を蓄える
      - 公式ドキュメントは隅から隅まで読む
      - Conference, 勉強会の資料を読む
      - ソースコードも読む
    - 色々知っておく
      - contrib, PgFoundry , PGXN, Sourceforge等のツール

安定運用実践に向けて

安定運用実践に向けて（１／３）

- 以下の運用要件を実現できるように技術を磨きましょう
  - 死活監視
    - 外形監視（ping/port/SQL監視）
    - H/W監視
    - プロセス監視
    - ログ監視
  - リソース監視・性能監視
    - CPU/MEM/IO/NW使用量監視
    - ディスク容量監視
    - スロークエリ監視
    - 最頻SQL監視
    - キャッシュヒット率監視
    - レプリケーション遅延監視
  - 統計情報レポート
    - トランザクション量
    - コミット数/ロールバック数
    - アーカイブ量
    - DBサイズ量
    - チェックポイント頻度
    - 自動VACUUM頻度
    - 同時接続数
    - スループット・レスポンス
    - バッチ処理時間

安定運用実践に向けて（２／３）

- 停止 / 再起動
  - 設定リロード
  - smart/fast/immediateシャットダウン
  - 再起動
  - クラッシュリカバリ時作業
- フェイルオーバ
  - HAクラスタ構築・運用
- プロモート
  - レプリケーション構築・運用
- バックアップ
  - 論理バックアップ
  - 物理バックアップ
  - アーカイブログ管理
- PITR
  - 時刻を指定したPITR
  - 最新状態までPITR
  - オペミス直前までのPITR

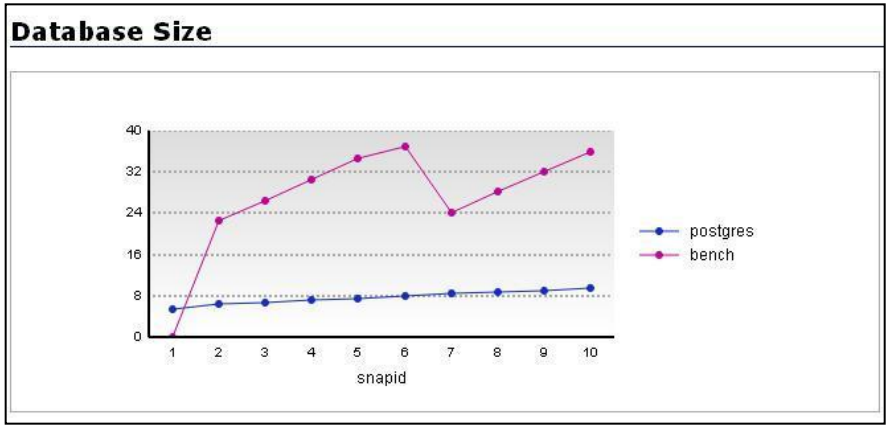
安定運用実践に向けて（３／３）

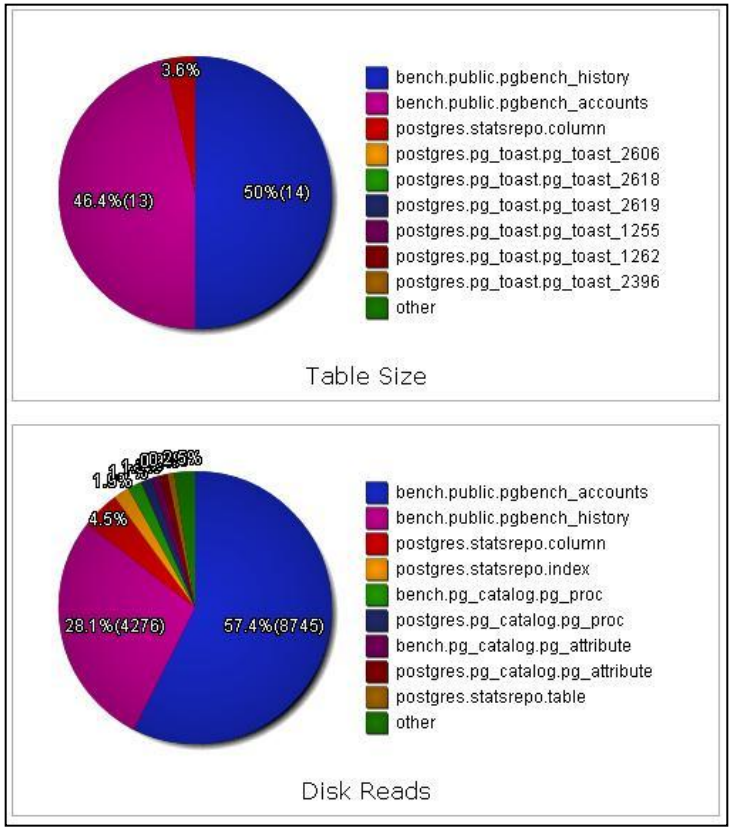
- メンテナンスコマンド実施
  - 自動VACUUMチューニング
  - 自動ANALYZEチューニング
  - REINDEX
  - CLUSTER・VACUUM FULL
- クエリキャンセル
  - 問い合わせキャンセル
  - 接続のキャンセル
- ボトルネック調査
  - リソース解析
  - ロック分析（ヘビーロック・LWロック）
  - 統計情報収集（pg_stat*）
- 実行計画分析
  - EXPLAIN（BUFFERS ON ANALYZE ON）の分析
  - 適切な対応方針の策定
- セキュリティアップデート
  - マイナーバージョンアップ
- アップグレード
  - データダンプ
  - データロード
  - メジャーバージョンアップ

運用をサポートするツール

ツール紹介 – pg_statsinfo

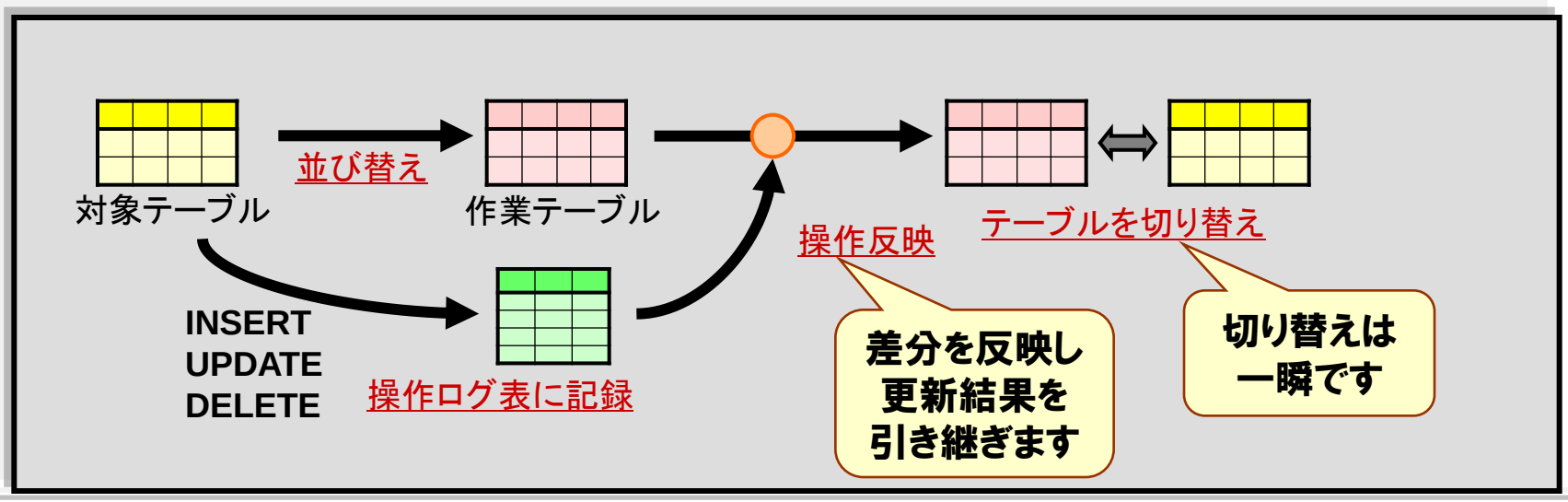
- pg_statsinfo http://pgstatsinfo.projects.postgresql.org/index_ja.html
  - 性能情報・各種統計情報をWeb画面から確認する
  - 内部状態をチェックし、アラートを上げる
- 使いどころ
  - 「グラフィカルに状態を確認したい」時
  - 「週次性能レポートをつくる」時





ツール紹介 –pg_reorg

- pg_reorg http://reorg.projects.postgresql.org/index-ja.html
  - 参照・更新を妨げずに、表・索引を再作成する
- 使いどころ
  - 「VACUUM FULL したいけど、止められない」時
  - 「REINDEX したいけど、止められない」時



ツール紹介 – pg_rman

- pg_rman http://code.google.com/p/pg-rman/
  - コマンド一つでオンラインバックアップを取得する
  - コマンド一つでPITRの前準備を実施する
- 使いどころ
  - 「とにかく簡単にバックアップ・リカバリを実現したい」時

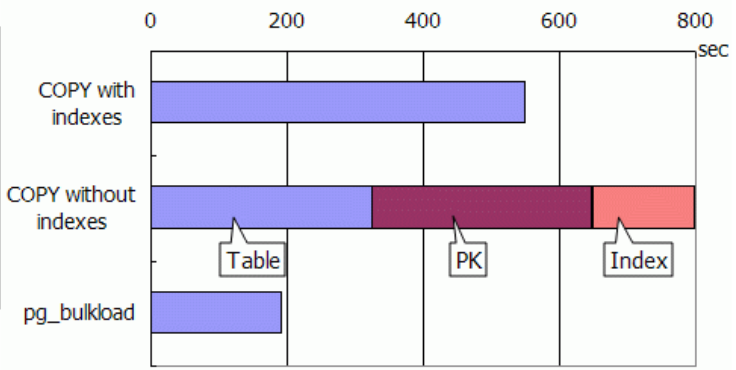
```
$ pg_ctl start
$ pg_rman backup --backup-mode=full --with-serverlog
$ pg_rman validate
$ pg_ctl stop -m immediate
$ pg_rman restore
$ pg_ctl start
```

ツール紹介 – pg_bulkload

- pg_bulkload http://pgbulkload.projects.postgresql.org/index_ja.html
  - データロードを高速にロードする
- 使いどころ
  - 「データロード時間を少しでも早くしたい」時
  - 「ロードするデータに何らかの加工をしたい」時（ETLツール）

追加ロード: COPY vs pg_bulkload

項目	version (postgres + bulkload)	
	8.4.4 + 2.4	9.0b2 + 3.0
COPY with indexes	520.4 sec	549.3 sec
COPY without indexes + CREATE INDEX	805.3 sec	799.6 sec
pg_bulkload (DIRECT) with indexes	185.2 sec	191.7 sec
時間比	35.6 %	34.9 %



デモ

お品書き

- 運用をサポートするツールを使ってデモをします。
  - 性能監視 – pg_statsinfo を使って
  - バックアップ – pg_rman を使って
  - オンライン再編成 – pg_reorg を使って
  - 高速データロード – pg_bulkload を使って

ご清聴ありがとうございました