

PostgreSQLの便利なツールのご紹介

SRA OSS, Inc. 日本支社
佐藤 友章
sato@sraoss.co.jp

- 「PostgreSQL／PowerGresサポート&保守サービス」でサポート対象になっているサードパーティ製のソフトを紹介
 - pgFouine
 - pgBadger(今後サポート対象にする予定)
 - PgFincore
 - pg_reorg
 - pg_rman
 - pg_bulkload
 - pg_statsinfo
- 各ソフトのインストールや設定は、CentOS 6にPGDGのYumリポジトリからPostgreSQL 9.2をインストールした状態が前提

```
# rpm -ivh http://yum.postgresql.org/9.2/redhat/rhel-6-x86_64/pgdg-centos92-9.2-6.noarch.rpm
# yum groupinstall "PostgreSQL Database Server PGDG"
# su - postgres
$ export PATH=/usr/pgsql-9.2/bin:$PATH
$ initdb -E UTF8 --no-locale
$ pg_ctl start -w
```

■ ログを解析し、SQLの実行に関するレポートを作るツール

- <http://pgfouine.projects.pgfoundry.org/>
- fouine(フウイン)とは
 - フランス語でテン(ムナジロテン、イタチ科の動物)
 - フランス語の動詞「fouiner(かきまわる)」の一人称単数現在形?
- PHPで書かれている
- PGDGのYumリポジトリからインストール

```
# yum install pgfouine php-gd
```

■ こんな人におすすめ(次のpgBadgerのほうがいい)

- アプリケーションを遅くしているSQLを見つけ出したい人
- 見栄えのいいレポートを作りたい人

■ ただし

- メモリの使用量が多く、解析に時間がかかる
- 開発が止まっている(2010年2月のリリースが最後)

- 解析に必要なメッセージがログに出力されるように設定

```
log_min_duration_statement = 0
log_duration = off
log_line_prefix = ' ' # syslogを使う場合
#log_line_prefix = '%t [%p]: [%l-1] ' # ロギングコレクタを使う場合
log_statement = 'none'
lc_messages = 'C'
```

- ロギングコレクタ(logging collector)だと複数行にわたるメッセージを処理できない場合があるので、syslogを使ったほうがいい
- ログがたまったら、pgfouine.phpスクリプトを実行

```
$ pgfouine.php -file /var/log/postgresql -format html-with-graphs \
  -logtype syslog -report report.html=overall,hourly,bytype
```

Firefox pgFouine: PostgreSQL log analysis ...

file:///C:/cygwin/home/tom-sato/work/pgfouine/report.html

pgFouine: PostgreSQL log analysis report

Overall statistics | Hourly statistics | Queries by type

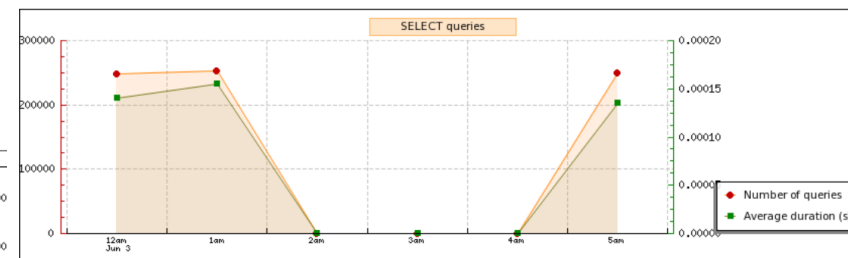
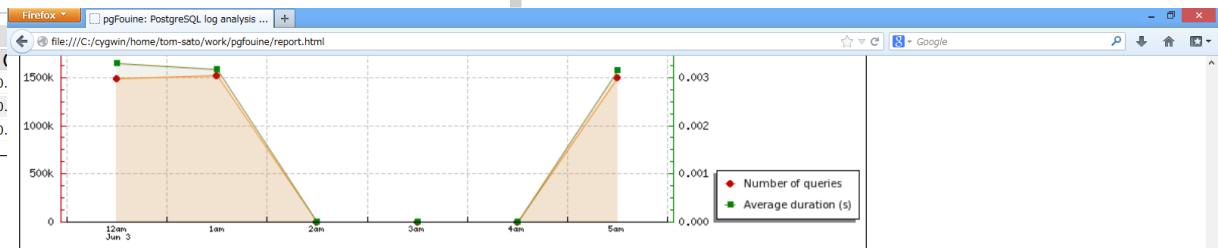
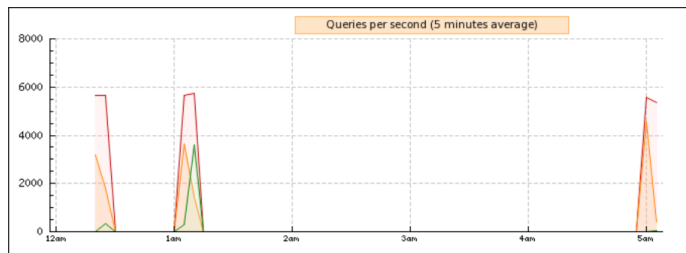
- Generated on 2013-06-03 05:24
- Parsed /home/tom-sato/opt/postgresql-9.2.4/data/pg_log/postgresql-Mon.log (10,514,273 lines) in 13m47s
- Log from 2013-06-03 00:20:31 to 2013-06-03 05:05:26
- Executed on alice

Overall statistics ^

- Number of queries: 4,506,132
- Total query duration: 3h59m55s
- First query: 2013-06-03 00:20:31
- Last query: 2013-06-03 05:05:26
- Query peak: 5,750 queries/s at 2013-06-03 01:10:39

Hourly statistics ^

Day	Time	Queries		SELECT queries	
		Count	Av. duration (s)	Count	Av. duration (s)
Jun 3	12am	1489904	0.00	248132	0.
	1am	1514762	0.00	252461	0.
	5am	1501466	0.00	250245	0.



Queries by type ^

Type	Count	Percentage
SELECT	750,838	16.7
INSERT	751,935	16.7
UPDATE	751,935	16.7

[Table of contents](#)

- ログを (pgFouineより) **高速**に解析し、SQLの実行や**サーバの動作**に関するレポートを作るツール

- <http://dalibo.github.io/pgbadger/>
- badger (バジャー) とは
 - 英語でアナグマ
 - テン (pgFouine) と同じイタチ科の動物にちなんでいる
- pgFouineでできることはpgBadgerでもできる
- **Perl**で書かれている
- PGDGのYumリポジトリからインストール

```
# yum install pgbadger
```

- こんな人におすすめ
 - pgFouineを使ってみたいと思った人 (pgBadgerのほうがいい)

- ログの設定は基本的にpgFouineと同じ

```
log_min_duration_statement = 0
log_checkpoints = on
log_connections = on
log_disconnections = on
log_duration = off
log_line_prefix = ' ' # syslogを使う場合
#log_line_prefix = '%t [%p]: [%l-1] ' # ロギングコレクタを使う場合
log_lock_waits = on
log_statement = 'none'
log_temp_files = 0
log_autovacuum_min_duration = 0
lc_messages = 'C'
```

- チェックポイントの実行、クライアントの接続や切断など、サーバの動作に関するメッセージにも対応(赤字)
- ログがたまったら、pgbadgerスクリプトを実行

```
$ pgbadger -o report.html /var/log/postgresql
```

Firefox - pgBadger: PostgreSQL log analyzer

file:///C:/cygwin/home/tom-sato/work/pgbadger/report.html

pgBadger: PostgreSQL log analyzer

Overall statistics | Hourly statistics | Queries by type | Most frequent queries (N)
Most frequent events (N) | Logs per type

Normalized reports are marked with a "(N)".

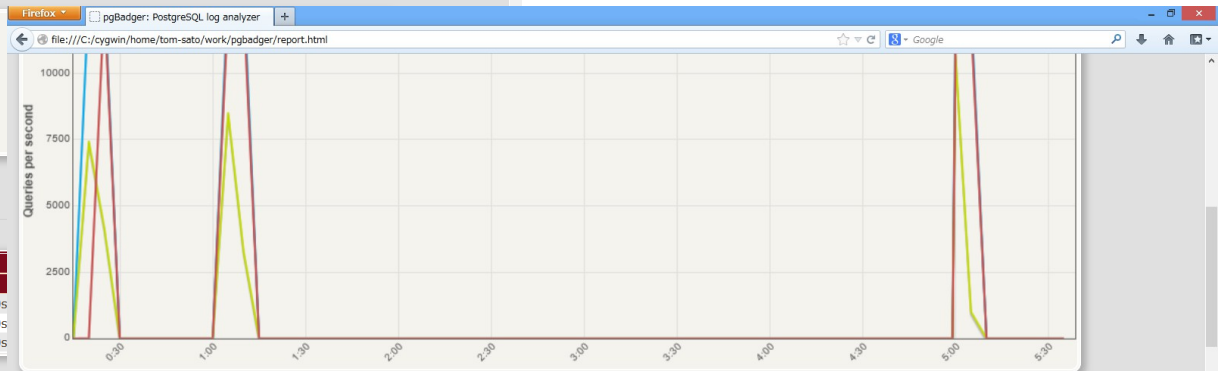
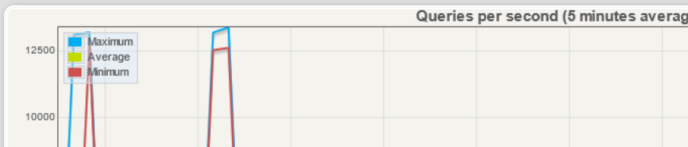
- Generated on Mon Jun 3 15:10:58 2013
- Log file: /home/tom-sato/opt/pgsql-9.2.4/data/pg_log/postgresql-Mon.log
- Parsed 10,514,277 log entries in 10m1.000s
- Log start from 2013-06-03 00:20:17 to 2013-06-03 05:33:54

Overall statistics ^

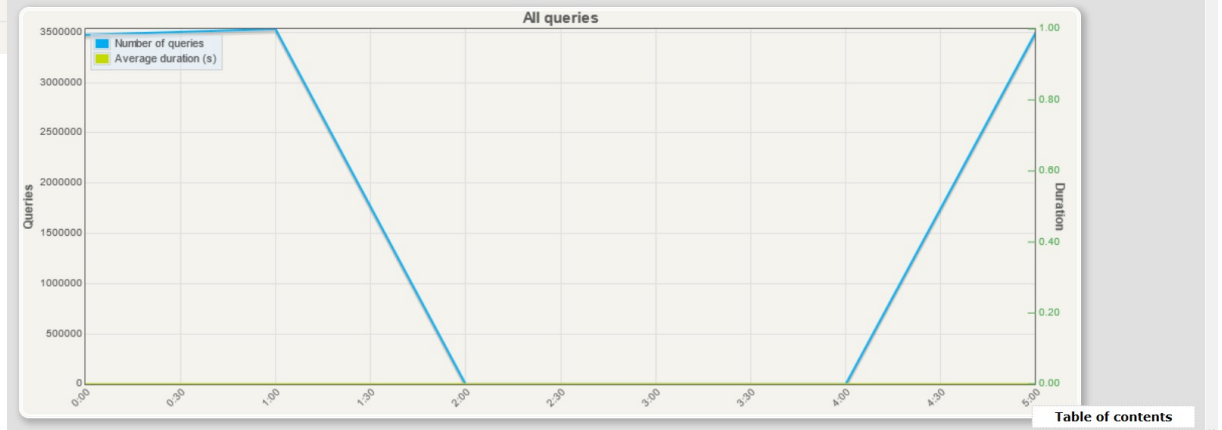
- Number of unique normalized queries: 32
- Number of queries: 10,513,958
- Total query duration: 4h49.496s
- First query: 2013-06-03 00:20:31
- Last query: 2013-06-03 05:05:26
- Query peak: 13,395 queries/s at 2013-06-03 01:10:39
- Number of events: 154
- Number of unique normalized events: 2

Hourly statistics ^

Day	Hour	Queries		SELECT queries	
		Count	Min/Max/Avg duration	Count	Avg duration
Jun 03	00	3,476,086	0.000s/20.598s/0.001s	248,132	0s
	01	3,534,448	0.000s/0.575s/0.001s	252,461	0s
	05	3,503,424	0.000s/0.554s/0.001s	250,245	0s



To Image Download Reset

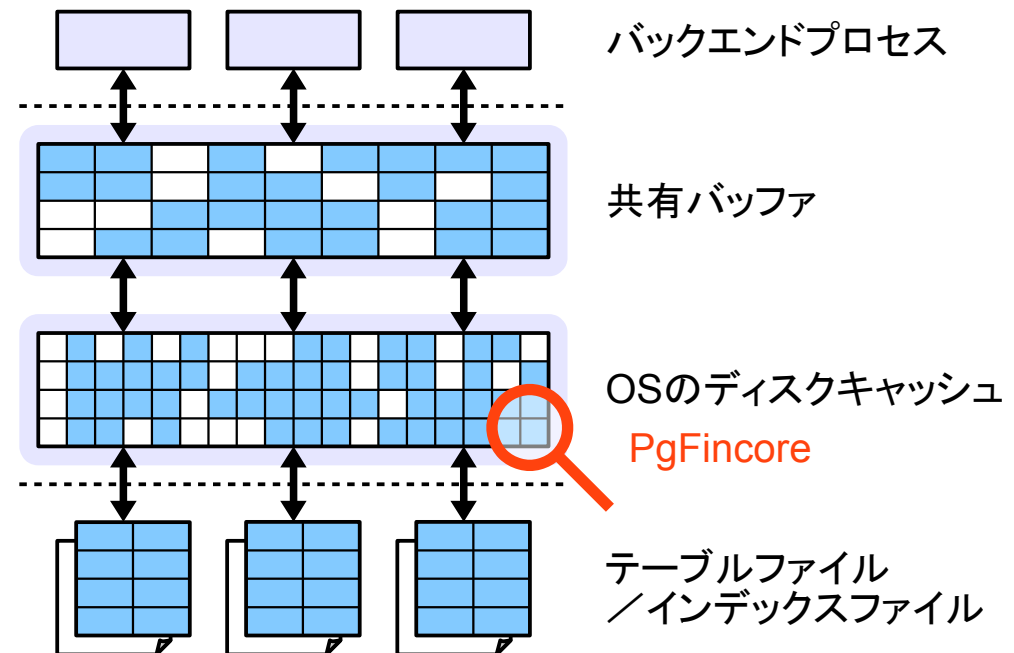


- OSのディスクキャッシュに乗ったテーブルとインデックスのページを管理する関数
 - <http://villemain.org/projects/pgfincore>
 - fincore(エフ・イン・コア)、fadvise(エフ・アドバイズ)のPostgreSQL版
 - PGDGのYumリポジトリからインストールできる

```
# yum install pgfincore92
$ psql test
=# CREATE EXTENSION pgfincore;
```

■ こんな人におすすめ

- キャッシュにどのくらいテーブルやインデックスが乗っているかを知りたい人
- 特定のテーブルやインデックスをキャッシュに乗せたい、乗りやすくしたい人



■ pgfincore関数

- キャッシュにどのくらいページが乗っているかを調べる

```
=# SELECT * FROM pgfincore('pgbench_accounts');  
-[ RECORD 1 ]-+-----  
relpath      | base/16425/16438   ファイルパス  
segment      | 0                  セグメント番号  
os_page_size | 4096               OSのページサイズ  
rel_os_pages | 262144             合計ページ数  
pages_mem    | 70758              キャッシュ内のページ数  
group_mem    | 25010              隣り合うページのグループ数  
os_pages_free | 33543              キャッシュの空きページ数  
databit      |                    どこがキャッシュされているか  
(省略)
```

■ pgfadvise_willneed/dontneed関数

- キャッシュに乗ったページをできるだけとどめる／必要なくなったら追い出すように指示する

```
=# SELECT * FROM pgfadvise_willneed('pgbench_accounts');
```

- 参照や更新を妨げずにCLUSTERやVACUUM FULLを実行するツール
 - <http://reorg.projects.pgfoundry.org/index-ja.html>
 - テーブルには主キーが必要
 - 実行中にDDLを実行してはいけない
 - とくにTRUNCATEやCREATE INDEX、ALTER TABLEの一部を実行するとデータの整合性が崩れたり、消えてしまう
 - RPMパッケージをダウンロードしてインストール

```
# rpm -ivh pg_reorg-1.1.8-1.pg92.rhel6.x86_64.rpm
$ psql test
=# CREATE EXTENSION pg_reorg;
```

- どうしてもVACUUM FULLを実行したい人におすすめ
 - (FULLでない)VACUUMをちゃんと実行しておけば、VACUUM FULLが必要になることは少ない
- pg_reorgで特定のテーブルにVACUUM FULLを実行

```
$ pg_reorg -n -t pgbench_accounts test
```

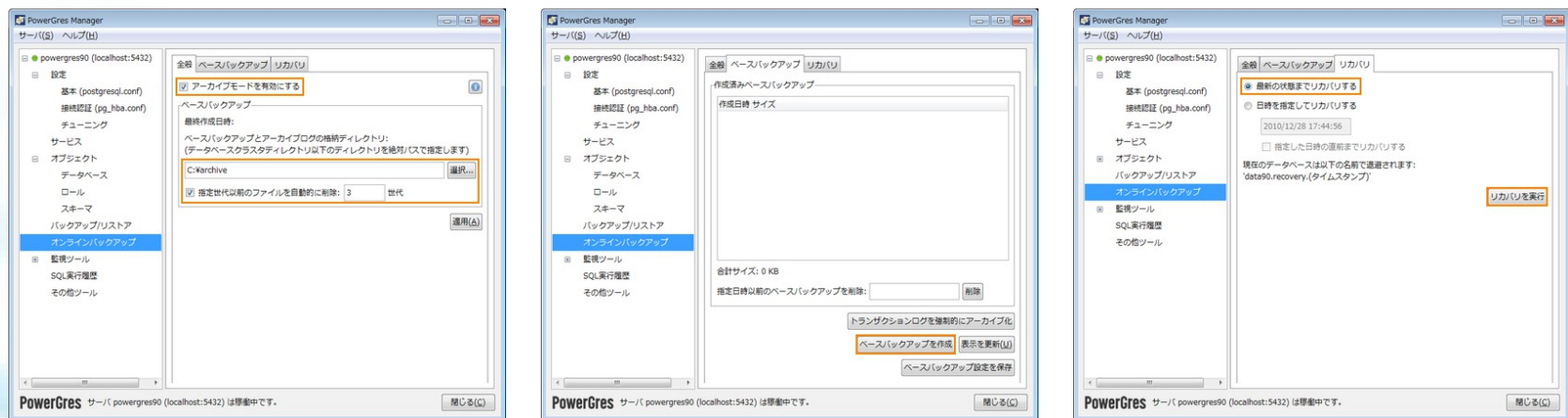
■ オンラインバックアップとPITR (point-in-time recovery) のややこしい操作を手助けするツール

- <http://code.google.com/p/pg-rman/>
- RPMパッケージをダウンロードしてインストール

```
# rpm -ivh pg_rman-1.2.5-1pg92.rhel6.x86_64.rpm
```

■ オンラインバックアップを使いたいと思っていたけど、難しそうであきらめていた人におすすめ

- ちなみに、PowerGresならオンラインバックアップはもちろん、ストリーミングレプリケーションもGUIで操作できる



- トランザクションログが保管されるように設定を行う

```
$ mkdir /mnt/disk/archive
$ vim $PGDATA/postgresql.conf

wal_level = archive
archive_mode = on
archive_command = 'cp %p /mnt/disk/archive/%f'

$ pg_ctl restart
```

- バックアップカタログを初期化

```
$ mkdir /mnt/disk/backups
$ pg_rman init -B /mnt/disk/backups
```

- フルバックアップをとる

```
$ pg_rman backup -B /mnt/disk/backups -b full
$ pg_rman validate -B /mnt/disk/backups
```

- データベースクラスタが消えてしまった状態からリカバリ

```
$ pg_ctl stop -m immediate
$ rm -fr $PGDATA
$ pg_rman restore -B /mnt/disk/backups
$ pg_ctl start
```

■ データを高速にロードするツール

- http://pgbulkload.projects.pgfoundry.org/index_ja.html
- ロード時間は、COPYに比べてデータが空の状態ですら10%程度、データが入っている状態で60%程度短くなったという報告もある
- トランザクションログへの書き込みを行わない
 - PITRの場合にはロード後にベースバックアップを取る必要がある
 - ストリーミングレプリケーションの場合には使えない
- RPMパッケージをダウンロードしてインストール

```
# rpm -ivh pg_bulkload-3.1.3-1.pg92.rhel6.x86_64.rpm
$ psql test
=# CREATE EXTENSION pg_bulkload;
```

■ データのロードに時間がかかっている人におすすめ

■ pg_bulkloadでデータをロード

```
$ pg_bulkload -i pgbench_accounts.tsv -O pgbench_accounts \
-o DELIMITER='$\t' -d test
```

- サーバの動作に関する情報を定期的に集めるツール

- http://pgstatsinfo.projects.pgfoundry.org/index_ja.html

- pg_statsinfoが集める情報

- 統計情報コレクタが集めた情報
- テーブルスペース、トランザクションログ、アーカイブログのディスク使用量
- チェックポイント、自動バキュームの実行回数、実行時間
- SQLの実行回数、合計実行時間
- OSのリソース情報

など

- 集めた情報はpg_reporterでHTML形式のレポートとして出力できる

- RPMパッケージをダウンロードしてインストール

```
# rpm -ivh pg_statsinfo-2.4.1-1.pg92.rhel6.x86_64.rpm
```

- いざというときに備えて情報をとっておきたい人におすすめ

- 設定や使い方についてはドキュメント(日本語)を参照

オープンソースとともに



SRA OSS, INC.

URL: <http://www.sraoss.co.jp/>
E-mail: pgsql-sales@sraoss.co.jp
Tel: 03-5979-2701