PostgreSQL インサイド

pg_statsinfo で統計情報を収集・蓄積する

技術を知る

FUJITSU

- ☐ 導入／環境設定
- ☐ 移行
- ☐ 性能
- ☐ チューニング
- ☐ バックアップ／リカバリー
- ☐ 冗長化／負荷分散
- ☑ 監視
- ☐ データ連携
- ☐ 災害対策
- ☐ 豆知識

PostgreSQL システムを長期的に運用する上で、データベースのレスポンスやパフォーマンスに劣化が生じていないか定期的に監視し、必要に応じてシステムを調整・改善していくことは非常に重要です。PostgreSQL の周辺ツールの 1 つである pg_statsinfo は、サーバーの統計情報や性能情報を収集・蓄積・表示することで、PostgreSQL の日々の処理傾向の把握、性能劣化などの兆候や問題発生時の原因把握に役立ちます。

1. pg_statsinfo とは

pg_statsinfo は、PostgreSQL サーバーの統計情報や稼働状況を一定の時間間隔で取得する機能、およびサーバーログを別のログに分配・蓄積する機能があります。また、収集した統計情報を確認してアラートを出す機能もあります。さらに、蓄積した情報を基にテキスト形式のレポートを出力するコマンドを提供します。pg_statsinfo が提供する機能を以下にまとめます。なお、本記事ではマニュアルと同様に pg_statsinfo で取得した統計情報をスナップショット、スナップショットの保存先のデータベースをリポジトリーDB と定義します。

表 1 pg_statsinfo の機能

機能名	説明
スナップショットの取得	PostgreSQL のスナップショットを一定の間隔で取得し、リポジトリーDB に保存します。PostgreSQL の統計情報コレクターが収集するすべての情報、性能や OS のリソースに関する情報をスナップショットとして取得できます。スナップショットは各インスタンスのデータベースクラスタ単位で取得します。
サーバーログの分配	PostgreSQL が出力するサーバーログを CSV ログ、テキストログ、syslog に分配して出力します。また、メッセージレベルや特定ユーザーによるログの出力制御などの加工ができます。
サーバーログの蓄積	PostgreSQL が出力するサーバーログを収集し、リポジトリーDB に蓄積します。また、メッセージレベルや特定ユーザーによるログの蓄積制御ができます。蓄積したログは蓄積ログと定義します。
アラート	スナップショット取得時にリポジトリーDB 内にアラート設定テーブルが生成され、そのテーブルの値を基に監視対象インスタンスの状態をチェックします。問題を検知した場合にアラートメッセージをテキストログに出力します。
自動メンテナンス	1 日 1 回、任意の時刻に、古いスナップショットの削除、蓄積ログの削除、サーバーログのログファイルの整理を自動で実行します。
簡易レポートの出力	リポジトリーDB に保存されたスナップショットからテキスト形式でレポートを出力するコマンドを提供します。
運用管理コマンド	pg_statsinfo の運用管理を行うためのコマンドを提供します。

pg_statsinfo は、監視対象のサーバーにエージェントをインストールして、PostgreSQL の設定ファイルの値やユーザーのコマンド実行により動作します。以下に pg_statsinfo の機能の仕組みを図解します。

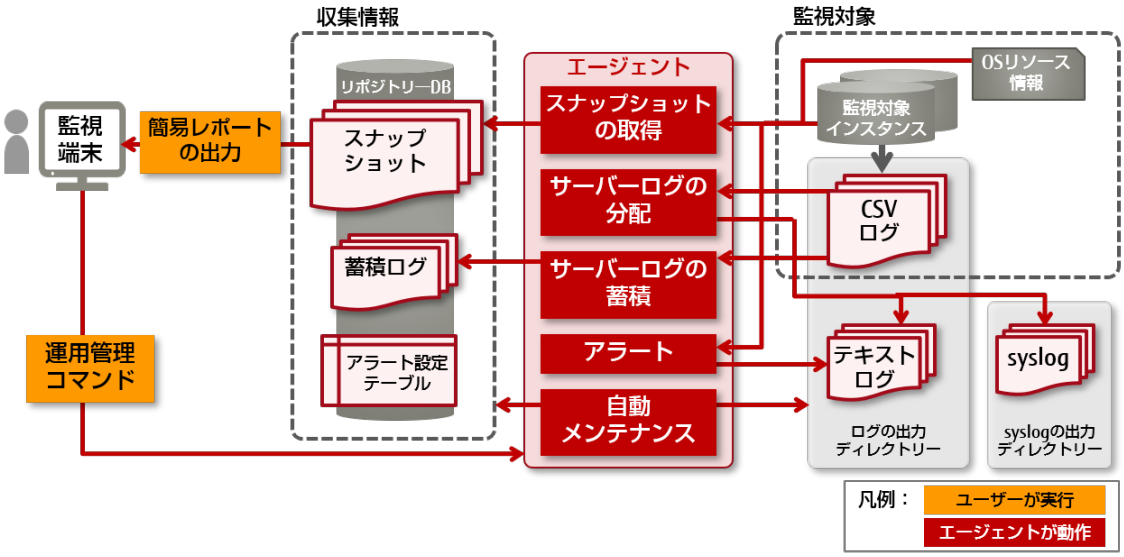


図 1 pg_statsinfo の機能概要

参考

- リポジトリーDB は、監視対象インスタンスと同一インスタンスでも、別インスタンス、または別のサーバーでも設定可能です。また、1 つのリポジトリーDB に複数の監視対象インスタンスのスナップショットを格納することもできます。
- pg_statsinfo の仕様については、オープンソース・ソフトウェアに同梱されているマニュアルを参照してください。
- pg_statsinfo は便利な機能ですが、注意点もあります。ご利用の前には必ず「4. pg_statsinfo 利用時の注意点」をお読みください。
- pg_statsinfo で収集したスナップショットは、別の周辺ツールである pg_stats_reporter を利用してグラフィカルな形で解析・出力することがきます。

2. 設定の準備

pg_statsinfo を利用するには、まず、必要な情報がログファイルに出力されるよう、PostgreSQL の設定ファイルである postgresql.conf にパラメーターを設定します。なお、pg_statsinfo は Linux 上で動作します。今回は、pg_statsinfo 12.0 をベースに説明します。

PostgreSQL および pg_statsinfo のインストールは完了し、監視対象 DB とリポジトリーDB は同一インスタンスになるよう構成します。インスタンスのエンコーディングは UTF8 です。

1. postgresql.conf のパラメーターを設定します。図 1 に赤色で示した pg_statsinfo のエージェントの機能はこのパラメーターの値によって動作しますので、事前に設計する必要があります。表 2 の設定例は、CSV ログを出力する設定値とし、代表的なパラメーターを挙げて説明しています。デフォルト値およびその他についてはオープンソース・ソフトウェアに同梱されているマニュアルを参照してください。

表 2 pg_statsinfo に必要な postgresql.conf のパラメーター

機能	パラメーター	設定例	説明
スナップショットの取得	shared_preload_libraries	'pg_statsinfo'	サーバー起動時にプリロードされる共有ライブラリを指定する。
	track_functions	all	関数の呼び出しに関する統計情報を収集するには pl または all を指定する。
	pg_statsinfo.snapshot_interval	30min	スナップショットの取得間隔（分）を指定する。
	pg_statsinfo.repository_server	'port=5432 dbname=repo user=postgres'	リポジトリーDB への接続文字列を指定する。
	pg_statsinfo.target_server	'port=5432 dbname=mydb'	監視対象 DB の接続文字列を指定する。
	pg_statsinfo.excluded_dbnames	'template0, template1,postgres,repo'	監視対象から除外するデータベース名を指定する。
サーバーログの分配	logging_collector	on	ログの収集を指定する。強制的に on が設定される。
	log_destination	'csvlog'	ログの出力方法を指定する。強制的に'csvlog'が設定され、'stderr'は削除される。'syslog'をカンマ区切りで追加できる。
	log_filename	'stats-%Y-%m-%d_%H%M%S.log'	CSV ログおよびテキストログのファイル名を指定する。
	log_min_messages	warning	ログに出力するメッセージレベルを指定する。（注 1）

機能	パラメーター	設定例	説明
	log_checkpoints	on	on を指定することで、チェックポイントの実行をログに残す。
	log_autovacuum_min_duration	0	自動バキューム状況で、指定された時間（ミリ秒）以上かかったログを残す。 0：すべて -1：無効
	lc_messages	'ja_JP.UTF-8'	メッセージが表示される言語を設定。エンコーディングの値と統一。
	pg_statsinfo.textlog_min_messages	error	テキストログへ出力するメッセージレベルを指定する。無効は disable。（注 1）
	pg_statsinfo.textlog_filename	'pg_statsinfo.log'	テキストログのファイル名を指定する。空文字はエラー。
	pg_statsinfo.textlog_line_prefix	'%t %p %c-%l %x %q(%u, %d, %r, %a) '	テキストログの各行の先頭に追加される文字列を指定する。
	pg_statsinfo.syslog_min_messages	disable	syslog へ出力するメッセージレベルを指定する。無効は disable。（注 1）
サーバーログの蓄積	pg_statsinfo.repolog_min_messages	disable	蓄積ログへ出力するメッセージレベルを指定する。無効は disable。（注 1）（注 2）

機能	パラメーター	設定例	説明
アラート	pg_statsinfo.enable_alert	on	アラート機能の有効/無効を指定する。 ・on：アラート機能有効 ・off：アラート機能無効
自動メンテナンス	pg_statsinfo.enable_maintenance	'on'	自動メンテナンス機能の設定を以下より選択する。1）から 3）はカンマ区切りで複数指定できる。 1）'snapshot'：スナップショット削除を実行 2）'repolog'：蓄積ログ削除を実行 3）'log'：ログファイル整理コマンドを実行 4）'on'：1）から 3）をすべて実行 5）'off'：自動メンテナンス無効
	pg_statsinfo.maintenance_time	'00:02:00'	自動メンテナンスの実行時刻（時：分：秒）を指定する。実行は 1 日に 1 回。
	pg_statsinfo.repository_keepday	7	スナップショットの保持期間（日）を指定する。
	pg_statsinfo.repolog_keepday	7	蓄積ログの保持期間（日）を指定する。

- 注 1　設定する値は他のパラメーターとの組合せが必要です。詳細については表 4 を参照してください。
- 注 2　監視対象 DB とリポジトリーDB が同一インスタンスの場合は、disable（無効）が推奨です。

2. pg_hba.conf を設定します。データベースへの接続ユーザーが localhost からのアクセスでパスワードの入力が不要になるよう設定します。認証方式は「ident」を推奨します。

```
# TYPE   DATABASE     USER           CIDR-ADDRESS    METHOD  [for UNIX]
local    all          postgres                       ident
```

3. PostgreSQL を起動または再起動します。PostgreSQL の起動と連動して pg_statsinfo のエージェントが起動されます。

3. pg_statsinfo の使い方

実際に pg_statsinfo を動作させて、統計情報を収集・蓄積・確認する方法について順を追って見ていきましょう。また、アラート値の変更やサーバーログの加工方法も説明します。 リポジトリーDB として repo を作成し、監視対象 DB として mydb を作成すると、各データベース作成直後のスキーマ構成は以下になります。

データベース名	スキーマ名	補足
repo	public,statsrepo	表 2 の pg_statsinfo.repository_server パラメーターに「repo」をリポジトリーDB として設定することで、pg_statsinfo が statsrepo スキーマを自動生成
mydb	public,statsinfo	表 2 の pg_statsinfo.target_server パラメーターに「mydb」を監視対象 DB として設定することで、pg_statsinfo が statsinfo スキーマを自動生成

サンプルデータの収集には、PostgreSQL に同梱されているベンチマークツール「pgbench」を利用します。ここでは、mydb というデータベース上で、50 クライアント、1000 トランザクションで pgbench を実行してみます。

```
$ pgbench -i mydb
$ pgbench -c 50 -t 1000 mydb
```

3.1 スナップショットのレポート出力

pg_statsinfo が取得したスナップショットをレポートに出力して状況を確認します。スナップショットの情報は、pg_statsinfo のコマンドを使うことにより、リポジトリーDB に保存されたスナップショットから任意の期間のレポートを、テキスト形式で標準出力に表示します。

1. スナップショットが取得できているかを確認します。

```
$ pg_statsinfo -h localhost -d repo -l   ←スナップショットの一覧表示
-------------------------------------
Snapshot List
-------------------------------------
SnapshotID  InstanceID  Host                    Port            Timestamp  Comment          Execute Time      Size
-------------------------------------------------------------------------------------------------------------------
         1           1  VM012345.localdomain    5432  2020-11-10 16:00:00                     00:00:01   344 KiB
         2           1  VM012345.localdomain    5432  2020-11-10 16:10:00                     00:00:01   144 KiB
         3           1  VM012345.localdomain    5432  2020-11-10 16:20:00                     00:00:01   136 KiB
         5           1  VM012345.localdomain    5432  2020-11-10 16:29:24  test (10337)       00:00:01   128 KiB
         6           1  VM012345.localdomain    5432  2020-11-10 16:30:00                     00:00:01   144 KiB
         7           1  VM012345.localdomain    5432  2020-11-10 16:40:00                     00:00:01   168 KiB
         8           1  VM012345.localdomain    5432  2020-11-10 16:50:00                     00:00:01   144 KiB
         9           1  VM012345.localdomain    5432  2020-11-10 17:00:00                     00:00:00   152 KiB
        10           1  VM012345.localdomain    5432  2020-11-13 10:20:00                     00:00:01   184 KiB
        ...
```

2. 指定した日以降のスナップショットの概要を表示する場合、レポート種別に「-r summary」、開始日に「-B 2020-11-14」を指定します。

```
$ pg_statsinfo -h localhost -d repo -r summary -B 2020-11-14
-------------------------------------
/* Summary */                        ↑概要を指定  ↑レポートの開始日を指定
-------------------------------------
Database System ID   : 6847735629115694388
Host                 : VM012345.localdomain
Port                 : 5432
PostgreSQL Version   : 12.1
Snapshot Begin       : 2020-11-14 00:00:00
Snapshot End         : 2020-11-15 10:30:00
Snapshot Duration    : 1 days 10:30:00
Total Database Size  : 159 MiB
Total Commits        : 29531
Total Rollbacks      : 0
```

ヒント

pg_statsinfo コマンドの-r オプションの後に指定できるレポート種別 ID は以下です。大文字小文字は区別なく、頭文字から最短一致で指定可能です。レポート種別 ID とレポートの内容についてはオープンソース・ソフトウェアに同梱されているマニュアルを参照してください。以下は、レポート種別 ID として「Summary」を小文字で最短で指定した例です。

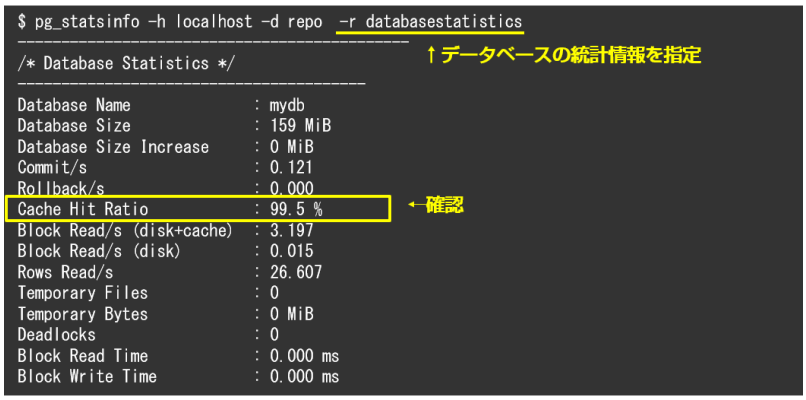
例）$ pg_statsinfo -r **su**

- Summary：スナップショットの概要
- Alert：アラートの出力情報
- DatabaseStatistics：データベースの統計情報
- InstanceActivity：トランザクションログ（WAL）に関する情報
- OSResourceUsage：CPU、IO、メモリー使用量などの情報
- DiskUsage：ディスク使用量
- LongTransactions：トランザクションに関する情報
- NotableTables：テーブルに関する情報
- CheckpointActivity：チェックポイントに関する情報
- AutovacuumActivity：自動バキュームに関する情報
- QueryActivity：SQL 文の問い合わせに関する情報
- LockConflicts：ロック待ちに関する情報
- ReplicationActivity：レプリケーションに関する情報
- SettingParameters：パラメーターの設定値
- SchemaInformation：テーブルとインデックスに関する情報
- Profiles：処理名
- All：上記の全情報

3.2 運用状態の監視

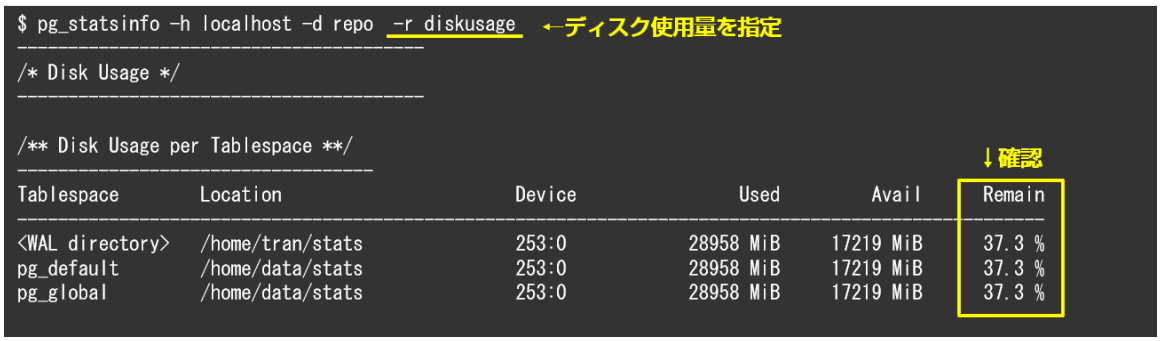
スナップショットの情報を使って PostgreSQL の運用状態を確認してみます。

1. スナップショットの「データベースの統計情報」を表示して、「Cache Hit Ratio（キャッシュヒット率）」を確認します。



状況を見てチューニングします。「Cache Hit Ratio」の値が 90%を下回っていたら、メモリー不足と推測できます。postgresql.conf の shered_buffers パラメーターの値を追加するなど検討してください。メモリーのチューニングの詳細については、「データベースチューニング」の「2.2 メモリーに関するパラメーター」を参照してください。

2. スナップショットの「ディスク使用量」を表示して「Remain（テーブルスペースのディスク空き容量）」を確認します。



テーブルスペースのディスク空き容量が少なくなっていることが分かります。スナップショットで取得する情報の一部（1 秒間のコミット数、テーブルスペースのディスク空き容量など）をアラート機能が監視対象としています。アラート機能を使って早めにアラームをあげることも可能です。

3.3 アラートメッセージの出力

pg_statsinfo は、スナップショット取得時に監視対象インスタンスの状態を定期的にチェックし、問題を検知した場合にアラートメッセージをテキストログに出力します。アラートメッセージは、メッセージレベルが「ALERT」で出力されます。なお、アラート機能で検出したアラートの内容は、リポジトリーDB にも蓄積されます。アラート機能による出力情報は性能ボトルネックを特定し、改善に役立てることができます。

3.3.1 アラート値（閾値）の確認

表 2 の pg_statsinfo.enable_alert パラメーターでアラート機能を有効（on）に設定した場合、初回のスナップショットが完了した監視対象インスタンスに対して自動的に有効になります。初期状態のアラート条件（閾値）はアラート設定テーブルの初期値が適用されます。アラート設定テーブルは、リポジトリーDB 内に「statsrepo.alert」テーブルとして生成されます。テーブルのカラムを表 3 に示します。アラート条件（閾値）を変更する場合は、各カラムの値を UPDATE コマンドで変更します。各カラムの値を「-1」に設定すると、カラム単位にアラート機能を無効化できます。

表 3 アラート設定テーブル

カラム名	デフォルト値	説明
instid		監視対象インスタンス ID（変更不可）
rollback_tps	100	1 秒間のロールバック数の閾値
commit_tps	1000	1 秒間のコミット数の閾値
garbage_size	-1	監視対象インスタンス中の不要領域のサイズの閾値（MB）
garbage_percent	30	監視対象インスタンスに占める不要領域の割合の閾値（%）
garbage_percent_table	30	各テーブルに占める不要領域の割合の閾値（%）

カラム名	デフォルト値	説明
response_avg	10	クエリの平均レスポンス時間の閾値（秒）
response_worst	60	クエリの最長レスポンス時間の閾値（秒）
backend_max	100	テーブルの断片化率（correlation）の閾値（%）
correlation_percent	70	各テーブルの相関係数（correlation）（%）
disk_remain_percent	20	テーブルスペースのディスク空き容量の閾値（%）
loadavg_1min	7	過去 1 分間のロードアベレージの閾値
loadavg_5min	6	過去 5 分間のロードアベレージの閾値
loadavg_15min	5	過去 15 分間のロードアベレージの閾値
swap_size	1000000	スワップ使用量の閾値（KB）
rep_flush_delay	100	レプリケーションのマスタとスタンバイ間の WAL 書き込み遅延量の閾値（MB）
rep_replay_delay	200	レプリケーションのスタンバイのリカバリー遅延量の閾値（MB）
enable_alert	true	アラート対象判定フラグ（true：有効、false：無効）

ヒント

アラート機能全体を無効化する場合は、表 2 の pg_statsinfo.enable_alert パラメーターに「off」を設定します。監視対象インスタンス単位でアラート機能を無効化する場合は、その監視対象インスタンスの表 3 の enable_alert カラムに「false」を設定します。

3.3.2 アラート値の変更

テーブルスペースのディスク空き容量のアラート値を変更して、テキストログにアラートメッセージを出力してみます。

1. まず、監視対象インスタンス ID を確認します。


```
$ pg_statsinfo -h localhost -d repo  -s   ←スナップショットのサイズ情報を指定
----------------------------------------
Snapshot Size Information
----------------------------------------
Instance ID                : 1  ←監視対象インスタンスID
Database System ID         : 6847735629115694388
Host                       : VM012345.localdomain
Port                       : 5432
Number Of Snapshots        : 251
Snapshot Size              : 45 MiB
Latest Snapshot ID         : 252
Latest Snapshot Timestamp  : 2020-11-15 17:00:00

Total Snapshot Size  : 45 MiB
```

2. アラート設定テーブルのディスク空き容量の閾値を確認します。

```
$ psql -d repo -c "SELECT disk_remain_percent FROM statsrepo.alert"
disk_remain_percent
---------------------
                  20  ←初期値の20%
(1 行)ance ID
```

3. 監視対象インスタンス ID「1」のアラート設定テーブルのディスク空き容量の閾値の「20%」を「40%」に変更します。

```
$ psql -d repo -c "UPDATE statsrepo.alert SET disk_remain_percent = 40 WHERE instid = 1"
UPDATE 1
                        ↑40%に変更                        ↑
                                           監視対象インスタンスIDが1
```

4. アラート設定テーブルのディスク空き容量の閾値が変更されたかを確認します。

```
$ psql -d repo -c "SELECT disk_remain_percent FROM statsrepo.alert"
disk_remain_percent
---------------------
                  40  ←40%に変更されたことを確認
(1 行)ance ID
```

5. スナップショットの取得間隔に合わせて「Alert」のレポートが出力されているかを確認します。

```
$ pg_statsinfo -h localhost -d repo -r alert   ←アラートの出力を指定
----------------------------------------
/* Alert */
----------------------------------------
DateTime             Message                                                                                                   ↓確認
------------------------------------------------------------------------------------------
2020-11-15 17:30:00  free disk space ratio at 'pg_default' fell below threshold in snapshot '2020-11-15 17:30:00' --- 37.3 % (threshold = 40 %)
2020-11-15 17:30:00  free disk space ratio at 'pg_global' fell below threshold in snapshot '2020-11-15 17:30:00' --- 37.3 % (threshold = 40 %)
2020-11-15 17:30:00  free disk space ratio at '<pg_xlog>' fell below threshold in snapshot '2020-11-15 17:30:00' --- 37.3 % (threshold = 40 %)
```

6. テキストログに「ALERT」が出力されているかを確認します。

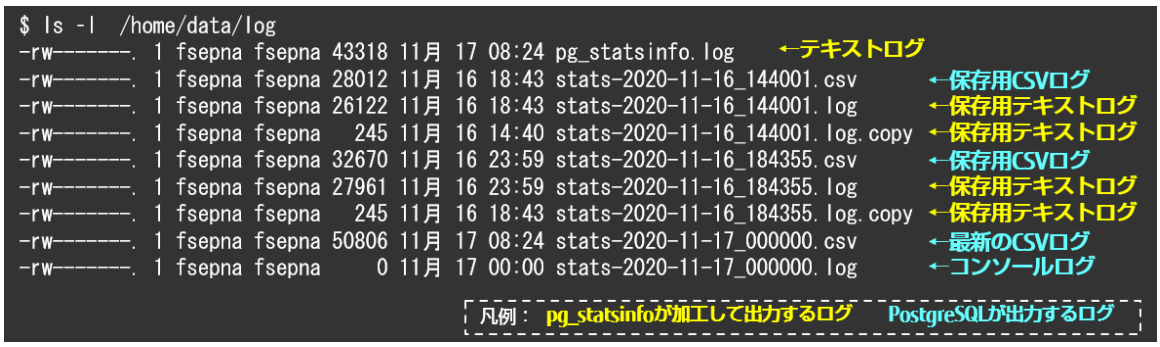
```
$ cd /home/data/log
$ vi pg_statsinfo.log

 . . . . .
2020-11-15 17:07:26 JST 13718 5f0eaa34.3596-28 0 LOG:  checkpoint complete: wrote 3 buffers (0.0%); 0 WAL file(s) added, 0 removed, 0 recycled: write=0.000 s, sync=0.007 s, total=0.069 s; sync files=3, longest=0.007 s, average=0.002 s; distance=9 kB, estimate=152 kB
2020-11-15 17:12:25 JST 13718 5f0eaa34.3596-29 0 LOG:  checkpoint starting: time
2020-11-15 17:12:25 JST 13718 5f0eaa34.3596-30 0 LOG:  checkpoint complete: wrote 4 buffers (0.0%); 0 WAL file(s) added, 0 removed, 0 recycled: write=0.402 s, sync=0.001 s, total=0.421 s; sync files=4, longest=0.001 s, average=0.000 s; distance=9 kB, estimate=138 kB
2020-11-15 17:17:25 JST 13718 5f0eaa34.3596-31 0 LOG:  checkpoint starting: time
2020-11-15 17:17:25 JST 13718 5f0eaa34.3596-32 0 LOG:  checkpoint complete: wrote 3 buffers (0.0%); 0 WAL file(s) added, 0 removed, 0 recycled: write=0.301 s, sync=0.001 s, total=0.324 s; sync files=3, longest=0.001 s, average=0.000 s; distance=9 kB, estimate=125 kB
2020-11-15 17:22:26 JST 13718 5f0eaa34.3596-33 0 LOG:  checkpoint starting: time
2020-11-15 17:22:26 JST 13718 5f0eaa34.3596-34 0 LOG:  checkpoint complete: wrote 4 buffers (0.0%); 0 WAL file(s) added, 0 removed, 0 recycled: write=0.402 s, sync=0.002 s, total=0.417 s; sync files=4, longest=0.002 s, average=0.000 s; distance=9 kB, estimate=113 kB
2020-11-15 17:27:26 JST 13718 5f0eaa34.3596-35 0 LOG:  checkpoint starting: time
2020-11-15 17:27:26 JST 13718 5f0eaa34.3596-36 0 LOG:  checkpoint complete: wrote 3 buffers (0.0%); 0 WAL file(s) added, 0 removed, 0 recycled: write=0.301 s, sync=0.001 s, total=0.325 s; sync files=3, longest=0.001 s, average=0.000 s; distance=9 kB, estimate=103 kB
2020-11-15 17:30:00 JST 13725 -  (pg_statsinfod, , , pg_statsinfod) ALERT:  pg_statsinfo: free disk space ratio at '<WAL directory>' fell below threshold in snapshot '2020-07-15 17:30:00' --- 74.7 % (threshold = 80 %)
2020-11-15 17:30:00 JST 13725 -  (pg_statsinfod, , , pg_statsinfod) ALERT:  pg_statsinfo: free disk space ratio at 'pg_default' fell below threshold in snapshot '2020-07-15 17:30:00' --- 74.7 % (threshold = 80 %)
2020-11-15 17:30:00 JST 13725 -  (pg_statsinfod, , , pg_statsinfod) ALERT:  pg_statsinfo: free disk space ratio at 'pg_global' fell below threshold in snapshot '2020-07-15 17:30:00' --- 74.7 % (threshold = 80 %)
2020-11-15 17:32:26 JST 13718 5f0eaa34.3596-37 0 LOG:  checkpoint starting: time
2020-11-15 17:32:31 JST 13718 5f0eaa34.3596-38 0 LOG:  checkpoint complete: wrote 42 buffers (0.3%); 0 WAL file(s) added, 0 removed, 0 recycled: write=4.222 s, sync=0.002 s, total=4.249 s; sync files=40, longest=0.002 s, average=0.000 s; distance=196 kB, estimate=196 kB
2020-11-15 17:37:26 JST 13718 5f0eaa34.3596-39 0 LOG:  checkpoint starting: time
2020-11-15 17:37:26 JST 13718 5f0eaa34.3596-40 0 LOG:  checkpoint complete: wrote 6 buffers (0.0%); 0 WAL file(s) added, 0 removed, 0 recycled: write=0.602 s, sync=0.001 s, total=0.623 s; sync files=5, longest=0.001 s, average=0.000 s; distance=17 kB, estimate=178 kB
```


3.4 サーバーログの加工

PostgreSQL では大量のサーバーログの中から必要な情報に絞って参照したい場合があります。pg_statsinfo は PostgreSQL が出力するサーバーログをフィルタリングして pg_statsinfo 独自のログに加工できます。pg_statsinfo を使うことで、CSV ログ、テキストログ、syslog のそれぞれが出力するメッセージレベルを変更したり、特定のユーザーのログを出力制御することができます。また、テキストログのファイル名は表 2 の pg_statsinfo.textlog_filename パラメーターで固定（デフォルトは pg_statsinfo.log）できるため、システム監視用のソフトウェアを使ってログ監視に利用できます。

1. サーバーログに出力するメッセージレベルの変更は、表 4 に示す 4 つのパラメーターに設定します。
2. 生成されているサーバーログを確認します。



pg_statsinfo が加工して出力するログには以下があります。

- テキストログ：PostgreSQL が出力する最新の CSV ログの情報を元に pg_statsinfo が加工したログです。
- 保存用テキストログ：上記のテキストログを log_filename パラメーターで定義された名前にリネームしたファイルです。ログのローテーションの際に生成されます。ログのローテーションについては、postgresql.conf の値に従います。

3.5 運用に必要な作業

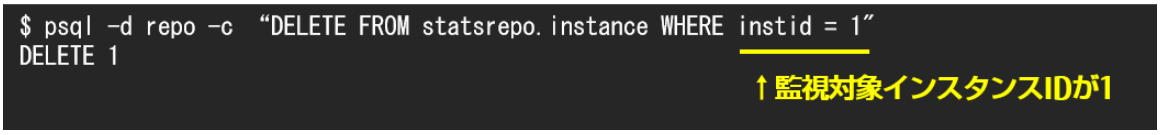
pg_statsinfo を使った監視運用時に必要な作業について説明します。

サーバーログの削除

自動メンテナンス機能を設定することで、不要になったスナップショットや蓄積ログは定時に削除されますが、ログファイル（CSV ログとテキストログ）は圧縮アーカーブされるだけで、削除されません。不要になったログファイルは、手動で定期的に削除してください。

監視対象インスタンス情報の削除

不要になった監視対象インスタンスの情報を削除する場合は、リポジトリーDB を直に操作します。なお、インスタンス情報を削除すると、関連するすべてのスナップショットが削除されるので、注意してください。以下の例では監視対象インスタンス ID は「1」とします。



注意

リポジトリーDB の管理する情報を操作する場合は、リポジトリーDB のテーブル構成を知る必要があります。リポジトリーDB のテーブルの詳細については、オープンソース・ソフトウェアに同梱されているマニュアルを参照してください。

ログのローテーション

postgresql.conf に指定したログ保持に関するパラメーターと異なる任意のタイミングでログをローテーションさせたい場合は、監視対象インスタンスで以下を実行します。

```
$ psql -d repo -c  “SELECT pg_rotate_logfile()”
SELECT 1
```

異常終了時の対処

pg_statsinfo のみが異常終了した場合、PostgreSQL のインスタンスには影響ありませんが、pg_statsinfo の機能は停止したままになります。pg_statsinfo を再起動するには、PostgreSQL のインスタンスを再起動します。なお、pg_statsinfo が異常終了してから再起動するまでの間に出力されたサーバーログは、再起動時にサーバーログの分配機能のみ作動して分配されます。

4. pg_statsinfo 利用時の注意点

ここでは、実際に pg_statsinfo を利用する際に発生する可能性のある課題とその対策について紹介します。

複数の監視対象インスタンスを 1 つのリポジトリーDB で管理する場合

複数の監視対象インスタンスのスナップショットを同一のリポジトリーDB に蓄積する構成では、自動メンテナンス設定のスナップショット保持期間の設定に注意が必要です。複数の監視対象インスタンスで自動メンテナンスのスナップショット削除を有効とする場合、すべてのインスタンスに対して、最も期間の短いスナップショット保持期間が有効となります。

- 例）　インスタンス 1：pg_statsinfo.repository_keepday = 7
  インスタンス 2：pg_statsinfo.repository_keepday = 5 <-- インスタンス 1 と 2 のスナップショット保持期間としてこちらが採用される

スナップショットのサイズ

スナップショットのサイズは、監視対象インスタンス内のオブジェクト数、スナップショットの取得間隔およびスナップショットの保持期間に依存して増加します。そのため、スナップショットの取得は、それらのパラメーター値とリポジトリーDB のディスク容量を考慮して設定する必要があります。なお、スナップショットのサイズ概算については「pg_statsinfo のサイジング」を参照してください。

サーバーログへ出力するメッセージレベルのパラメーター

サーバーログの分配機能に関する以下のパラメーターは相互に影響しあうので注意が必要です。log_min_messages パラメーターの設定値が pg_statsinfo で出力されるメッセージレベルの基準となります。他の 3 つのパラメーターの設定値は、log_min_messages パラメーターに設定した値と同じ、またはそれよりもログへの出力情報が少ない値（表 4 の右側）を設定する必要があります。

表 4 ログへ出力するメッセージレベルのパラメーター

No.	postgresql.confのパラメーター	設定値（注3）									
1	log_min_messages（注4）	debug5～1	info	notice	warning	error	log				
2	pg_statsinfo.textlog_min_messages	debug	info	notice	warning	error	log	fatal	panic	alert	disable
3	pg_statsinfo.syslog_min_messages	debug	info	notice	warning	error	log	fatal	panic	alert	disable
4	pg_statsinfo.repolog_min_messages	debug	info	notice	warning	error	log	fatal	panic	alert	disable

下線はデフォルト。赤字は本記事の設定値。

- 注 3　右側がよりメッセージ深刻度レベルが高く、ログへの出力情報が少なくなります。
- 注 4　pg_statsinfo と連携する場合、「fatal」と「panic」は指定できません。

エンコーディングの扱い

監視対象インスタンスでは、エンコーディングと lc_messages パラメーターの値を統一する必要があります。pg_statsinfo は PostgreSQL がサポートするエンコーディングやメッセージ・ロケールに対応していますが、別々の値を設定することはできません。

二重化環境での pg_statsinfo のリカバリーについて

ダウンした PostgreSQL サーバーを pg_basebackup や pg_rewind を利用して復旧すると、pg_statsinfo が正常に動作しなくなる場合があります。 具体的に、pg_statsinfo の管理ファイルである pg_statsinfo.control が pg_rewind によってスタンバイサーバに 転送されたことによりスタンバイサーバ側での参照の整合性が取れなくなったために起こります。 この問題を解決するには、リカバリー時、復旧対象のサーバーを起動する前に pg_statsinfo.control を削除する必要があります。

その他、HA 構成や複数の監視対象インスタンス構成での注意点などは、オープンソース・ソフトウェアに同梱されているマニュアルを参照してください。

ここまで説明してきましたが、pg_statsinfo は比較的簡単な手順で利用できることがおわかりいただけたと思います。「PostgreSQL の拡張機能でよく使う周辺 OSS の一覧」では、監視の分類として pg_statsinfo や pgBadger などを紹介しています。pgBadger と比較すると、スナップショットの対応範囲が狭いですが、システムポリシー上、データベース内に処理やオブジェクトを追加してのパフォーマンス分析が難しいケースにおいて利用可能です。監視の用途に合わせて、ご利用を検討してください。pgBadger と pg_statsinfo の機能比較については「pgBadger でログファイルを解析し、統計レポートを作成する」を参照してください。

参考

Fujitsu Enterprise Postgres では、バージョン 10 から pg_statsinfo を同梱しており、インストールは不要で、Fujitsu Enterprise Postgres が動作する Linux のバージョンをサポートしています。Fujitsu Enterprise Postgres に同梱している pg_statsinfo は、富士通の 24 時間 365 日保守サポートにより、PostgreSQL 本体および pg_statsinfo を含む周辺ツールのご質問、トラブル対応およびバグ修正にも迅速に対応しますので、ご利用を検討ください。

2025 年 1 月 27 日