

# Estatística, monitoramento e diagnóstico através do catálogo do PostgreSQL



## O que é o catálogo?

Os catálogos são os locais onde os SGBDs armazenam os metadados do esquema, tais como informações sobre tabelas, colunas, e informações de controle interno.

É uma excelente fonte de informações, mas nem sempre é fácil de encontrar.

# Porque conhecer o catálogo é importante?

Resumidamente, o catálogo pode ser descrito como um banco de dados que contém informações sobre o seu banco de dados e o seu servidor. Portanto ele pode ser um grande aliado tanto no desenvolvimento quanto na manutenção e monitoramento de uma base de dados

## Information Schema

O esquema de informações consiste em um conjunto de visões contendo informações sobre os objetos definidos no banco de dados corrente. O esquema de informações é definido no padrão SQL e, portanto, pode-se esperar que seja portátil e permaneça estável — isto é diferente dos catálogos do sistema, que são específicos do PostgreSQL e modelados segundo interesses da implementação. Entretanto, as visões do esquema de informações não contêm informações sobre funcionalidades específicas do PostgreSQL; para obter este tipo de informação, é necessário consultar os catálogos do sistema e outras visões específicas do PostgreSQL.

## System Catalogs

Os catálogos do sistema do PostgreSQL são tabelas comuns. Estas tabelas podem ser removidas e recriadas, podem ser adicionadas colunas, podem ser inseridos e atualizados valores, mas isso não é recomendável. Normalmente, os catálogos do sistema não devem ser modificados manualmente, sempre existe um comando SQL para fazê-lo (Por exemplo, `CREATE DATABASE` insere uma linha no catálogo `pg_database` — e cria realmente o banco de dados no disco).

# Information Schema – PostgreSQL 8.4 / 9.1

www.postgresql.org/docs/8.4/ir www.postgresql.org/docs/9.1/interactive/information-schema.html

## Chapter 33. The Information Schema

### Table of Contents

- 33.1. [The Schema](#)
- 33.2. [Data Types](#)
- 33.3. [information schema catalog name](#)
- 33.4. [administrable role authorizations](#)
- 33.5. [applicable roles](#)
- 33.6. [attributes](#)
- 33.7. [check constraint routine usage](#)
- 33.8. [check constraints](#)
- 33.9. [column domain usage](#)
- 33.10. [column privileges](#)
- 33.11. [column udt usage](#)
- 33.12. [columns](#)
- 33.13. [constraint column usage](#)
- 33.14. [constraint table usage](#)
- 33.15. [data type privileges](#)
- 33.16. [domain constraints](#)
- 33.17. [domain udt usage](#)
- 33.18. [domains](#)
- 33.19. [element types](#)
- 33.20. [enabled roles](#)
- 33.21. [foreign data wrapper options](#)
- 33.22. [foreign data wrappers](#)
- 33.23. [foreign server options](#)
- 33.24. [foreign servers](#)
- 33.25. [key column usage](#)
- 33.26. [parameters](#)
- 33.27. [referential constraints](#)
- 33.28. [role column grants](#)
- 33.29. [role routine grants](#)
- 33.30. [role table grants](#)
- 33.31. [role usage grants](#)
- 33.32. [routine privileges](#)
- 33.33. [routines](#)
- 33.34. [schemas](#)
- 33.35. [sequences](#)
- 33.36. [sql features](#)
- 33.37. [sql implementation info](#)
- 33.38. [sql languages](#)
- 33.39. [sql packages](#)
- 33.40. [sql parts](#)
- 33.41. [sql sizing](#)
- 33.42. [sql sizing profiles](#)
- 33.43. [table constraints](#)
- 33.44. [table privileges](#)
- 33.45. [tables](#)
- 33.46. [triggers](#)
- 33.47. [usage privileges](#)
- 33.48. [user mapping options](#)
- 33.49. [user mappings](#)
- 33.50. [view column usage](#)
- 33.51. [view routine usage](#)
- 33.52. [view table usage](#)
- 33.53. [views](#)

## Chapter 34. The Information Schema

### Table of Contents

- 34.1. [The Schema](#)
- 34.2. [Data Types](#)
- 34.3. [information schema catalog name](#)
- 34.4. [administrable role authorizations](#)
- 34.5. [applicable roles](#)
- 34.6. [attributes](#)
- 34.7. [character sets](#)
- 34.8. [check constraint routine usage](#)
- 34.9. [check constraints](#)
- 34.10. [collations](#)
- 34.11. [collation character set applicability](#)
- 34.12. [column domain usage](#)
- 34.13. [column privileges](#)
- 34.14. [column udt usage](#)
- 34.15. [columns](#)
- 34.16. [constraint column usage](#)
- 34.17. [constraint table usage](#)
- 34.18. [data type privileges](#)
- 34.19. [domain constraints](#)
- 34.20. [domain udt usage](#)
- 34.21. [domains](#)
- 34.22. [element types](#)
- 34.23. [enabled roles](#)
- 34.24. [foreign data wrapper options](#)
- 34.25. [foreign data wrappers](#)
- 34.26. [foreign server options](#)
- 34.27. [foreign servers](#)
- 34.28. [foreign table options](#)
- 34.29. [foreign tables](#)
- 34.30. [key column usage](#)
- 34.31. [parameters](#)
- 34.32. [referential constraints](#)
- 34.33. [role column grants](#)
- 34.34. [role routine grants](#)
- 34.35. [role table grants](#)
- 34.36. [role usage grants](#)
- 34.37. [routine privileges](#)
- 34.38. [routines](#)
- 34.39. [schemas](#)
- 34.40. [sequences](#)
- 34.41. [sql features](#)
- 34.42. [sql implementation info](#)
- 34.43. [sql languages](#)
- 34.44. [sql packages](#)
- 34.45. [sql parts](#)
- 34.46. [sql sizing](#)
- 34.47. [sql sizing profiles](#)
- 34.48. [table constraints](#)
- 34.49. [table privileges](#)
- 34.50. [tables](#)
- 34.51. [triggered update columns](#)
- 34.52. [triggers](#)
- 34.53. [usage privileges](#)
- 34.54. [user mapping options](#)
- 34.55. [user mappings](#)
- 34.56. [view column usage](#)
- 34.57. [view routine usage](#)
- 34.58. [view table usage](#)
- 34.59. [views](#)





The world's most  
advanced open  
source database.

# System Catalogs – PostgreSQL 8.4 / 9.1

← → ↺ ↻ [www.postgresql.org/docs/8.4](http://www.postgresql.org/docs/8.4)

## Table of Contents

- 44.1. [Overview](#)
- 44.2. [pg\\_aggregate](#)
- 44.3. [pg\\_am](#)
- 44.4. [pg\\_amop](#)
- 44.5. [pg\\_amproc](#)
- 44.6. [pg\\_attrdef](#)
- 44.7. [pg\\_attribute](#)
- 44.8. [pg\\_authid](#)
- 44.9. [pg\\_auth\\_members](#)
- 44.10. [pg\\_cast](#)
- 44.11. [pg\\_class](#)
- 44.12. [pg\\_constraint](#)
- 44.13. [pg\\_conversion](#)
- 44.14. [pg\\_database](#)
- 44.15. [pg\\_depend](#)
- 44.16. [pg\\_description](#)
- 44.17. [pg\\_enum](#)
- 44.18. [pg\\_foreign\\_data\\_wrapper](#)
- 44.19. [pg\\_foreign\\_server](#)
- 44.20. [pg\\_index](#)
- 44.21. [pg\\_inherits](#)
- 44.22. [pg\\_language](#)
- 44.23. [pg\\_largeobject](#)
- 44.24. [pg\\_listener](#)
- 44.25. [pg\\_namespace](#)
- 44.26. [pg\\_opclass](#)
- 44.27. [pg\\_operator](#)
- 44.28. [pg\\_opfamily](#)
- 44.29. [pg\\_pltemplate](#)
- 44.30. [pg\\_proc](#)
- 44.31. [pg\\_rewrite](#)
- 44.32. [pg\\_shdepend](#)
- 44.33. [pg\\_shdescription](#)
- 44.34. [pg\\_statistic](#)
- 44.35. [pg\\_tablespace](#)
- 44.36. [pg\\_trigger](#)
- 44.37. [pg\\_ts\\_config](#)
- 44.38. [pg\\_ts\\_config\\_map](#)
- 44.39. [pg\\_ts\\_dict](#)
- 44.40. [pg\\_ts\\_parser](#)
- 44.41. [pg\\_ts\\_template](#)
- 44.42. [pg\\_type](#)
- 44.43. [pg\\_user\\_mapping](#)
- 44.44. [System Views](#)
- 44.45. [pg\\_cursors](#)
- 44.46. [pg\\_group](#)
- 44.47. [pg\\_indexes](#)
- 44.48. [pg\\_locks](#)
- 44.49. [pg\\_prepared\\_statements](#)
- 44.50. [pg\\_prepared\\_xacts](#)
- 44.51. [pg\\_roles](#)
- 44.52. [pg\\_rules](#)
- 44.53. [pg\\_settings](#)
- 44.54. [pg\\_shadow](#)
- 44.55. [pg\\_stats](#)
- 44.56. [pg\\_tables](#)
- 44.57. [pg\\_timezone\\_abbrevs](#)
- 44.58. [pg\\_timezone\\_names](#)
- 44.59. [pg\\_user](#)
- 44.60. [pg\\_user\\_mappings](#)
- 44.61. [pg\\_views](#)

← → ↺ ↻ [www.postgresql.org/docs/9.1/interactive/catalogs.html](http://www.postgresql.org/docs/9.1/interactive/catalogs.html)

## Table of Contents

- 45.1. [Overview](#)
- 45.2. [pg\\_aggregate](#)
- 45.3. [pg\\_am](#)
- 45.4. [pg\\_amop](#)
- 45.5. [pg\\_amproc](#)
- 45.6. [pg\\_attrdef](#)
- 45.7. [pg\\_attribute](#)
- 45.8. [pg\\_authid](#)
- 45.9. [pg\\_auth\\_members](#)
- 45.10. [pg\\_cast](#)
- 45.11. [pg\\_class](#)
- 45.12. [pg\\_constraint](#)
- 45.13. [pg\\_collation](#)
- 45.14. [pg\\_conversion](#)
- 45.15. [pg\\_database](#)
- 45.16. [pg\\_db\\_role\\_setting](#)
- 45.17. [pg\\_default\\_acl](#)
- 45.18. [pg\\_depend](#)
- 45.19. [pg\\_description](#)
- 45.20. [pg\\_enum](#)
- 45.21. [pg\\_extension](#)
- 45.22. [pg\\_foreign\\_data\\_wrapper](#)
- 45.23. [pg\\_foreign\\_server](#)
- 45.24. [pg\\_foreign\\_table](#)
- 45.25. [pg\\_index](#)
- 45.26. [pg\\_inherits](#)
- 45.27. [pg\\_language](#)
- 45.28. [pg\\_largeobject](#)
- 45.29. [pg\\_largeobject\\_metadata](#)
- 45.30. [pg\\_namespace](#)
- 45.31. [pg\\_opclass](#)
- 45.32. [pg\\_operator](#)
- 45.33. [pg\\_opfamily](#)
- 45.34. [pg\\_pltemplate](#)
- 45.35. [pg\\_proc](#)
- 45.36. [pg\\_rewrite](#)
- 45.37. [pg\\_seclabel](#)
- 45.38. [pg\\_shdepend](#)
- 45.39. [pg\\_shdescription](#)
- 45.40. [pg\\_statistic](#)
- 45.41. [pg\\_tablespace](#)
- 45.42. [pg\\_trigger](#)
- 45.43. [pg\\_ts\\_config](#)
- 45.44. [pg\\_ts\\_config\\_map](#)
- 45.45. [pg\\_ts\\_dict](#)
- 45.46. [pg\\_ts\\_parser](#)
- 45.47. [pg\\_ts\\_template](#)
- 45.48. [pg\\_type](#)
- 45.49. [pg\\_user\\_mapping](#)
- 45.50. [System Views](#)
- 45.51. [pg\\_available\\_extensions](#)
- 45.52. [pg\\_available\\_extension\\_versions](#)
- 45.53. [pg\\_cursors](#)
- 45.54. [pg\\_group](#)
- 45.55. [pg\\_indexes](#)
- 45.56. [pg\\_locks](#)
- 45.57. [pg\\_prepared\\_statements](#)
- 45.58. [pg\\_prepared\\_xacts](#)
- 45.59. [pg\\_roles](#)
- 45.60. [pg\\_rules](#)
- 45.61. [pg\\_seclabels](#)
- 45.62. [pg\\_settings](#)
- 45.63. [pg\\_shadow](#)
- 45.64. [pg\\_stats](#)
- 45.65. [pg\\_tables](#)
- 45.66. [pg\\_timezone\\_abbrevs](#)
- 45.67. [pg\\_timezone\\_names](#)
- 45.68. [pg\\_user](#)
- 45.69. [pg\\_user\\_mappings](#)
- 45.70. [pg\\_views](#)

## pg\_catalog – Tabelas

**pg\_class** - Contém informações de outras tabelas e objetos, é uma das principais tabelas do catálogo, geralmente buscamos informações dela diretamente ou através de views do próprio PostgreSQL, o que vem sendo cada vez mais comum.

**pg\_database** - Armazena informações sobre os bancos de dados disponíveis.

Diferente da maioria dos catálogos do sistema, é compartilhado por todos os bancos de dados do cluster.

**pg\_extension** - Armazena informações sobre as extensões instaladas

**pg\_language** – Armazena as linguagens instaladas que podem ser usadas para criar funções

**pg\_proc** - O catálogo pg\_proc armazena informações sobre as funções (ou procedimentos).

**pg\_statistic** - Armazena dados estatísticos sobre o conteúdo do banco de dados. Entradas são criadas por ANALYZE e posteriormente utilizado pelo planejador de comandos

**pg\_tablespace** - Armazena informações sobre os tablespaces disponíveis, é compartilhado entre todas as bases de um cluster



**Complementando o catálogo do PostgreSQL existem as system views que fornecem acesso rápido e fácil a diversas informações sobre o banco de dados e também ao estado interno do servidor.**

## 7.4 - 8.0

View Name	Purpose
<a href="#">pg_indexes</a>	indexes
<a href="#">pg_locks</a>	currently held locks
<a href="#">pg_rules</a>	rules
<a href="#">pg_settings</a>	parameter settings
<a href="#">pg_stats</a>	planner statistics
<a href="#">pg_tables</a>	tables
<a href="#">pg_user</a>	database users
<a href="#">pg_views</a>	views

## 8.1

View Name	Purpose
<a href="#">pg_group</a>	groups of database users
<a href="#">pg_indexes</a>	indexes
<a href="#">pg_locks</a>	currently held locks
<a href="#">pg_prepared_xacts</a>	currently prepared transactions
<a href="#">pg_roles</a>	database roles
<a href="#">pg_rules</a>	rules
<a href="#">pg_settings</a>	parameter settings
<a href="#">pg_shadow</a>	database users
<a href="#">pg_stats</a>	planner statistics
<a href="#">pg_tables</a>	tables
<a href="#">pg_user</a>	database users
<a href="#">pg_views</a>	views

## 8.2 - 8.3

View Name	Purpose
<a href="#">pg_cursors</a>	open cursors
<a href="#">pg_group</a>	groups of database users
<a href="#">pg_indexes</a>	indexes
<a href="#">pg_locks</a>	currently held locks
<a href="#">pg_prepared_statements</a>	prepared statements
<a href="#">pg_prepared_xacts</a>	prepared transactions
<a href="#">pg_roles</a>	database roles
<a href="#">pg_rules</a>	rules
<a href="#">pg_settings</a>	parameter settings
<a href="#">pg_shadow</a>	database users
<a href="#">pg_stats</a>	planner statistics
<a href="#">pg_tables</a>	tables
<a href="#">pg_timezone_abbrevs</a>	time zone abbreviations
<a href="#">pg_timezone_names</a>	time zone names
<a href="#">pg_user</a>	database users
<a href="#">pg_views</a>	views

## 8.4 - 9.0

View Name	Purpose
<a href="#">pg_cursors</a>	open cursors
<a href="#">pg_group</a>	groups of database users
<a href="#">pg_indexes</a>	indexes
<a href="#">pg_locks</a>	currently held locks
<a href="#">pg_prepared_statements</a>	prepared statements
<a href="#">pg_prepared_xacts</a>	prepared transactions
<a href="#">pg_roles</a>	database roles
<a href="#">pg_rules</a>	rules
<a href="#">pg_settings</a>	parameter settings
<a href="#">pg_shadow</a>	database users
<a href="#">pg_stats</a>	planner statistics
<a href="#">pg_tables</a>	tables
<a href="#">pg_timezone_abbrevs</a>	time zone abbreviations
<a href="#">pg_timezone_names</a>	time zone names
<a href="#">pg_user</a>	database users
<a href="#">pg_user_mappings</a>	user mappings
<a href="#">pg_views</a>	views

## 9.1

View Name	Purpose
<a href="#">pg_available_extensions</a>	available extensions
<a href="#">pg_available_extension_versions</a>	available versions of extensions
<a href="#">pg_cursors</a>	open cursors
<a href="#">pg_group</a>	groups of database users
<a href="#">pg_indexes</a>	indexes
<a href="#">pg_locks</a>	currently held locks
<a href="#">pg_prepared_statements</a>	prepared statements
<a href="#">pg_prepared_xacts</a>	prepared transactions
<a href="#">pg_roles</a>	database roles
<a href="#">pg_rules</a>	rules
<a href="#">pg_seclabels</a>	security labels
<a href="#">pg_settings</a>	parameter settings
<a href="#">pg_shadow</a>	database users
<a href="#">pg_stats</a>	planner statistics
<a href="#">pg_tables</a>	tables
<a href="#">pg_timezone_abbrevs</a>	time zone abbreviations
<a href="#">pg_timezone_names</a>	time zone names
<a href="#">pg_user</a>	database users
<a href="#">pg_user_mappings</a>	user mappings
<a href="#">pg_views</a>	views

View Name	Purpose
<a href="#"><u>pg_available_extensions</u></a>	available extensions
<a href="#"><u>pg_available_extension_versions</u></a>	available versions of extensions
<a href="#"><u>pg_cursors</u></a>	open cursors
<a href="#"><u>pg_group</u></a>	groups of database users
<a href="#"><u>pg_indexes</u></a>	indexes
<a href="#"><u>pg_locks</u></a>	currently held locks
<a href="#"><u>pg_prepared_statements</u></a>	prepared statements
<a href="#"><u>pg_prepared_xacts</u></a>	prepared transactions
<a href="#"><u>pg_roles</u></a>	database roles
<a href="#"><u>pg_rules</u></a>	rules
<a href="#"><u>pg_seclabels</u></a>	security labels
<a href="#"><u>pg_settings</u></a>	parameter settings
<a href="#"><u>pg_shadow</u></a>	database users
<a href="#"><u>pg_stats</u></a>	planner statistics
<a href="#"><u>pg_tables</u></a>	tables
<a href="#"><u>pg_timezone_abbrevs</u></a>	time zone abbreviations
<a href="#"><u>pg_timezone_names</u></a>	time zone names
<a href="#"><u>pg_user</u></a>	database users
<a href="#"><u>pg_user_mappings</u></a>	user mappings
<a href="#"><u>pg_views</u></a>	views

# Coletor de Estatísticas

O coletor de estatísticas do PostgreSQL é um subsistema de apoio a coleta e relatório de informações sobre as atividades do servidor.

postgresql.conf

# RUNTIME STATISTICS

#-----

# - Query/Index Statistics Collector -

#track\_activities = on

#track\_counts = on

track\_functions = pl

# none, pl, all

#track\_activity\_query\_size = 1024 # (change requires restart)

#update\_process\_title = on

#stats\_temp\_directory = 'pg\_stat\_tmp' [1]

# - Statistics Monitoring - [2]

#log\_parser\_stats = off

#log\_planner\_stats = off

#log\_executor\_stats = off

#log\_statement\_stats = off

[1] - File System em RAM diminui o impacto de I/O

[2] - Informações detalhas após cada execução

# Views pré-definidas Coletor de Estatísticas – PostgreSQL 9.1

## **pg\_stat\_activity**

pg\_stat\_bgwriter

## **pg\_stat\_database**

pg\_stat\_database\_conflicts

pg\_stat\_replication

pg\_stat\_all\_tables

pg\_stat\_sys\_tables

## **pg\_stat\_user\_tables**

pg\_stat\_xact\_all\_tables

pg\_stat\_xact\_sys\_tables

pg\_stat\_xact\_user\_tables

pg\_stat\_all\_indexes

pg\_stat\_sys\_indexes

## **pg\_stat\_user\_indexes**

pg\_statio\_all\_tables

pg\_statio\_sys\_tables

## **pg\_statio\_user\_tables**

pg\_statio\_all\_indexes

pg\_statio\_sys\_indexes

## **pg\_statio\_user\_indexes**

pg\_statio\_all\_sequences

pg\_statio\_sys\_sequences

pg\_statio\_user\_sequences

## **pg\_stat\_user\_functions**

pg\_stat\_xact\_user\_functions

<http://www.postgresql.org/docs/9.1/interactive/monitoring-stats.html#MONITORING-STATS-VIEWS-TABLE>

As views `pg_statio*` são úteis para determinar a eficácia de uso do cache, quando o número de `blks_read` (disk reads) é menor do que `blks_hit` (buffer hits) o uso do cache é satisfatório, no entanto essas estatísticas não devem servir como única medida para o acerto ideal do sistema.

O uso de ferramentas do sistema operacional, estatísticas do servidor e medição de cache de I/O tanto de controladoras de disco quanto do próprio kernel devem ser levados em conta.



Além das views também podemos obter informações do coletor de estatísticas através de funções pré definidas.

Uma relação de todas as funções é listada na documentação.

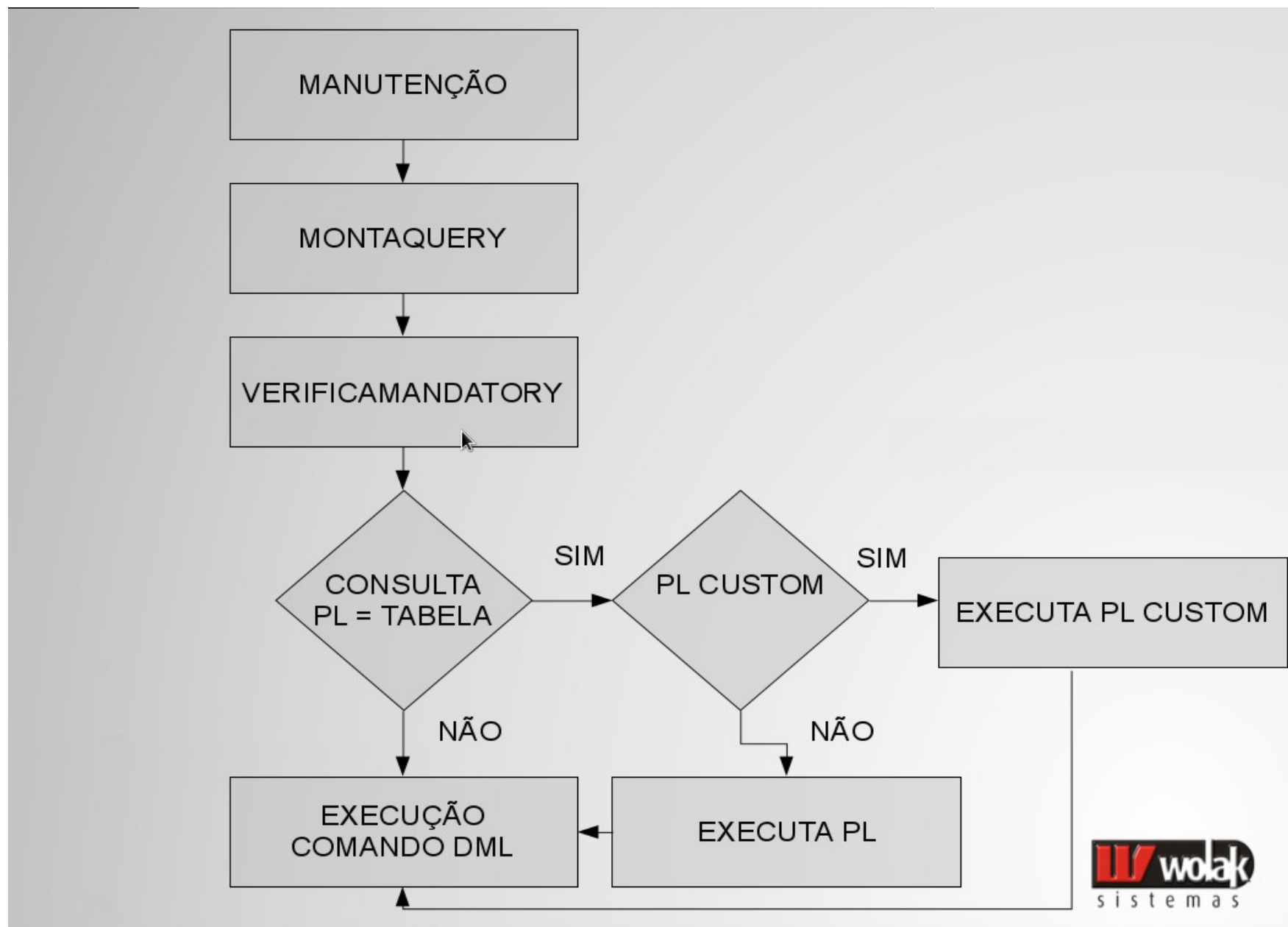
<http://www.postgresql.org/docs/9.1/interactive/monitoring-stats.html#MONITORING-STATS-FUNCS-TABLE>

Exemplos, dicas e uso prático do catálogo.

Versão utilizada PostgreSQL 9.1!

## Exemplo de informação usada no desenvolvimento:

### Exemplo 1: SQL1.sql



Front-End chama uma PL passando um array com campo e conteúdo.

```
SELECT pk as
pknotafiscalentrada,mandatory,mandatorymensagem,mensagem
FROM fnc.manutencao(
1, -- pempresa
2, -- pfilial
3, -- punidade
0, -- pusuario
array['notafiscalentrada','update','NotaFiscalEntrada_Form',
'MODIF'], -- pacao
ARRAY['fknotafiscalentradaxml' , '84','fkordemcompra' ,
NULL,'fktipooperacao_naturezaoperacao' ,
'22','fknaturezaoperacao' , '130','fkcadastro_documento' ,
'251','fkcadastro' , '244','fkcadastro_vinculo' ,
'213','fktipooperacao' , '11','fkdocumentofiscalicms' ,
'30','fkseriedocumentofiscal' , '2','numero' ,
'238085','dataemissao' , '2011-10-13','dataentradasaida' ,
'2011-10-26 10:43:00.000','valorbasecalculoicms' ,
'4528.34','valoricms' ,
'543.4','valorbasecalculoicmssubstituicaotributaria' ,
```

```
CREATE OR REPLACE FUNCTION fnc.manutencao(  
    pempresa          INTEGER,  
    pfilial            INTEGER,  
    punidade          INTEGER,  
    pusuario          INTEGER,  
    pcacao             VARCHAR[],  
    pccampoconteudo   VARCHAR[],  
    pccampoconteudopk VARCHAR[])  
RETURNS fnc.manutencao_retorno AS
```

Pesquisando o catalogo "pg\_proc" para verificar se existe uma PL com o nome da tabela que foi passada como parâmetro, caso exista, a PL manutenção executa uma outra PL com o mesmo nome do parâmetro passada em fnc.manutencao.

```
IF (SELECT TRUE FROM pg_proc JOIN pg_namespace ON nspname = 'fnc'  
AND pg_namespace.oid = pg_proc.pronamespace  
    WHERE proname = LOWER(pcAcao[1]) LIMIT 1) = TRUE AND  
UPPER(pcAcao[2]) IN('UPDATE','UPDATE_ONLY','DELETE','LOCK') THEN  
    IF (rFuncaoCustom.nometabela IS NOT NULL AND  
rFuncaoCustom.bloquearfnc = 2) OR (rFuncaoCustom.nometabela IS  
NULL) THEN  
        FOR rResultadoFuncaoBefore IN EXECUTE 'SELECT fnc.' ||  
pcAcao[1] || '(' || QUOTE_LITERAL('BEFORE') || ', ' ||  
pcCampoConteudoPK[2]::integer || ', ' ||  
QUOTE_LITERAL(ARRAY_TO_STRING(pcAcao, ',')) || ')' AS mensagem'  
        LOOP  
            cResultadoFuncaoBefore =  
rResultadoFuncaoBefore.mensagem;  
        END LOOP;  
    END IF;  
END IF;
```



Executa a funcao fnc.montaquery para escrever o comando DML;

```
IF UPPER(pcAcao[2]) <> 'LOCK' THEN
    FOR rResultado IN EXECUTE
        fnc.montaquery(pEmpresa,pFilial,pUnidade,pUsuario,pcAcao[2],pcAcao[
            1],pcCampoConteudo,pcCampoConteudoPK) LOOP END LOOP;
END IF;
```

**Verifica se o campo é nulo, caso não seja busca a propriedade  
DEFAULT da coluna, monta a coluna com a sua propriedade original de  
comando.**

```
cCampoConteudo = pcCampoConteudo;
FOR i IN 2..ARRAY_UPPER(pcCampoConteudo,1) BY 2
LOOP
    SELECT INTO cDefault,cIsNull column_default,is_nullable FROM
    information_schema.columns WHERE table_name = QUOTE_IDENT(pcTabela)
    AND column_name = QUOTE_IDENT(pcCampoConteudo[i-1]);
    IF cIsNull = 'NO' AND pcCampoConteudo[i] IS NULL THEN
        cCampoConteudo[i] = cDefault;
    END IF;
    pcCampoConteudoAlterado[i] =
    QUOTE_LITERAL(TRIM(fnc.extraiconteudo(pcCampoConteudo[i-
    1],cCampoConteudo,NULL))) || COALESCE('::' ||
        (SELECT UPPER(udt_name) FROM information_schema.columns WHERE
        table_name = QUOTE_IDENT(pcTabela) AND column_name =
        QUOTE_IDENT(pcCampoConteudo[i-1])), ''));
END LOOP;
```

## Resultado da função MONTAQUERY.

```
UPDATE notafiscalentrada SET
controle.fkusuarioregistrobloqueado =
NULL,controle.datainativo = NULL,controle.confirmainclusao =
NULL,controle.fkusuarioinclusao = 0,controle.datainclusao =
'2011-10-26 10:43:33.734',controle.fkusuarioalteracao =
0,controle.dataalteracao = '2011-10-29
22:04:03.514',fknotafiscalentradaxml =
'84'::INT4,fkordemcompra =
NULL,fktipooperacao_naturezaoperacao =
'22'::INT4,fknaturezaoperacao =
'130'::INT4,fkcadastro_documento = '251'::INT4,fkcadastro =
...
...

WHERE pknotafiscalentrada = '148'::INT4 RETURNING
pknotafiscalentrada AS pk,0 AS inclusao;
```

# Exemplo de informação usada no monitoramento:

## Exemplo 2: SQL2.sql

```
SELECT * FROM pg_stat_activity;
```

```
SELECT datname,username,client_addr,waiting,current_query  
FROM pg_stat_activity ;
```

```
SELECT datname,username,client_addr,waiting,current_query  
FROM pg_stat_activity WHERE current_query != '<IDLE>' ;
```

```
SELECT (current_timestamp - query_start) as tempoquery,  
datname, username,waiting,current_query  
FROM pg_stat_activity WHERE current_query != '<IDLE>'  
ORDER BY 1 DESC;
```

# Exemplo de informação usada no monitoramento (Bloqueios)

## Exemplo 3: SQL3.sql

```
SELECT locktype, pg_database.datname, pg_class.relname, transactionid,  
       classid, objid, objsubid, virtualtransaction, pid, mode, granted  
FROM pg_locks  
JOIN pg_database ON pg_database.oid = pg_locks.database  
JOIN pg_class ON pg_class.oid = pg_locks.relation
```

```
SELECT  
esperando.current_query AS query_esperando,  
esperando.procpid AS pid_esperando,  
esperando.username AS user_esperando,  
parado.current_query AS query_parado,  
parado.procpid AS pid_parado,  
parado.username AS user_parado,  
tabelas_travadas.schemaname||'.'||tabelas_travadas.relname AS tabelas_travadas  
FROM pg_stat_activity esperando  
JOIN pg_locks locks_esperando ON locks_esperando.pid = esperando.procpid  
JOIN pg_locks locks_parado ON locks_parado.relation = locks_esperando.relation  
JOIN pg_stat_activity parado ON parado.procpid = locks_parado.pid  
JOIN pg_stat_user_tables tabelas_travadas ON tabelas_travadas.relid =  
locks_esperando.relation  
WHERE esperando.waiting IS TRUE  
AND NOT locks_esperando.granted  
AND locks_parado.granted IS TRUE
```

# Exemplo de informação sobre tamanho do banco e das tabelas:

## Exemplo 4: SQL4.sql

--Tamanho dos bancos

```
SELECT pg_database.datname,  
pg_size_pretty(pg_database_size(pg_database.datname))  
FROM pg_database  
ORDER BY pg_database_size(pg_database.datname) DESC,  
pg_database.datname;
```

--Tamanho das tabelas do banco atual

```
SELECT retorno.schemaname, retorno.tablename,  
pg_size_pretty(pg_total_relation_size(schemaname||'.'||tablename)) AS  
tamanho  
FROM (SELECT tablename, schemaname  
FROM pg_tables  
WHERE schemaname NOT IN ('pg_catalog', 'information_schema',  
'pg_toast') ) retorno  
ORDER BY pg_total_relation_size(schemaname||'.'||tablename) DESC
```

# Exemplo de informação de varreduras nas tabelas

## Exemplo 5: SQL5.sql

--Leituras Sequencias/Indexadas

```
SELECT schemaname,relname,seq_scan,seq_tup_read,idx_scan,idx_tup_fetch  
FROM pg_stat_user_tables  
WHERE (idx_scan + seq_scan) <> 0  
ORDER BY seq_scan DESC
```

--Tabela que tiveram maiores índices de Insert/Update/Delete

```
SELECT relname,n_tup_ins AS insert, n_tup_upd AS update, n_tup_del AS delete  
FROM pg_stat_user_tables  
--ORDER BY relname;  
--ORDER BY n_tup_ins  
--ORDER BY n_tup_upd  
ORDER BY n_tup_del  
DESC
```

--Tabelas que tiveram maiores buscas no cache ou no disco/kernel

```
SELECT relname,heap_blks_hit,heap_blks_read FROM pg_statio_user_tables  
WHERE (heap_blks_hit + heap_blks_read)<>0  
ORDER BY heap_blks_hit  
--ORDER BY heap_blks_read  
DESC
```

<http://www.postgresql.org/docs/9.1/interactive/monitoring-stats.html#MONITORING-STATS-VIEWS-TABLE>



## Exemplo de informação de funções (PL)

### Exemplo 6: SQL6.sql

```
SELECT  
pg_proc.proname,  
pg_proc.pronargs,  
pg_stat_user_functions.*  
FROM pg_stat_user_functions  
JOIN pg_proc ON pg_proc.oid = pg_stat_user_functions.funcid
```

```
postgresql.conf  
track_functions = pl # none, pl, all
```

# Exemplo de informação dos índices

```
SELECT * FROM pg_stat_user_indexes;
```

idx\_tup\_read – Número de varreduras por index scan

idx\_tup\_fetch – Número de linhas buscadas por index scan usando o índice

Leituras bitmap scan não atualizam a coluna idx\_tup\_fetch, geralmente porque nessas condições dois ou mais índices são usados, avaliados e combinados, portanto é difícil eleger qual o índice que realmente foi o “responsável” pelo retorno da informação, por isso apenas a coluna idx\_tup\_read é incrementada, portanto valores menores em idx\_tup\_fetch são normais

Considere idx\_tup\_read como a melhor alternativa em caso de comparações entre índices.

# Exemplo de informação dos índices

## Exemplo 7: SQL7.sql

```
SELECT indexrelname,idx_scan  
FROM pg_stat_user_indexes  
--WHERE relname='tabela'  
WHERE idx_scan = 0  
AND indexrelname NOT LIKE '%pk%' AND indexrelname NOT LIKE  
'%uk%' --Se houver padrão de nome de chave pode utilizar para não  
buscar UKs e PKs  
ORDER BY idx_scan DESC
```

```
SELECT schemaname, relname, indexrelname, idx_scan,  
indnatts AS numerodecolunas  
FROM pg_stat_user_indexes  
JOIN pg_index ON pg_index.indexrelid =  
pg_stat_user_indexes.indexrelid  
WHERE pg_index.indisunique IS FALSE --Não seleciona índices  
únicos  
AND indisprimary IS FALSE --Não seleciona chaves primárias  
--ORDER BY idx_scan,relname  
ORDER BY idx_scan DESC
```

# Exemplo de informação dos índices

## Exemplo 8: SQL8.sql

### Verificando o uso do cache nas varreduras de índices

```
SELECT relname,indexrelname,idx_blks_hit,idx_blks_read  
FROM pg_statio_user_indexes  
JOIN pg_index ON pg_index.indexrelid =  
pg_statio_user_indexes.indexrelid  
WHERE (idx_blks_hit + idx_blks_read)<> 0  
AND pg_index.indisunique IS FALSE  
AND indisprimary IS FALSE
```

# Exemplo de informação dos índices

## Desabilitando um índice

```
UPDATE pg_index  
SET indisvalid = false  
WHERE indexrelid = ?
```

**Não faça isso para índices únicos!**

**O índice continua sendo atualizado por INSERT e UPDATE mas não será utilizado em nenhuma consulta.**

**Desta maneira é possível testar índices antes de remover.**

# Ferramentas de monitoramento para PostgreSQL

Existem várias ferramentas disponíveis no mercado, tanto livres quanto proprietárias.

O uso delas aliado ao conhecimento do catálogo e das estruturas internas do PostgreSQL facilita a tarefa de administrar uma ou várias bases de dados, segue uma lista resumida das ferramentas mais comuns.





The world's most  
advanced open  
source database.

Zabbix  
Pgwatch  
PgFouine  
Collectd  
Cedrus



The world's most  
advanced open  
source database.

# Perguntas?

Muito Obrigado!

Fabiano Machado Dias  
[fabiano@wolaksistemas.com.br](mailto:fabiano@wolaksistemas.com.br)

Eduardo Wolak  
[wolak@wolaksistemas.com.br](mailto:wolak@wolaksistemas.com.br)

[www.wolaksistemas.com.br](http://www.wolaksistemas.com.br)  
[www.erpws.com.br](http://www.erpws.com.br)