بسمه تعالی

رویدادنگاری، ممیزی و جرم‌شناسی در پایگاه داده‌ی PostgreSQL

فهرست مطالب

۱ مقدمه

با توجه به ذخیره‌ی حجم زیادی از اطلاعات حساس در پایگاه‌های داده لازم است که عملیات و تراکنش‌های رخ داده روی پایگاه‌های داده، مورد بررسی قرار گیرد تا مشخص شود هرکسی، چه عملیاتی را در چه زمانی انجام داده‌است. از این‌رو، ثبت رویدادها و تهیه‌ی ممیزی از اهمیت زیادی برخوردار است.

در این گزارش در بخش ۲، انواع فایل‌های رویدادنگاری در پایگاه داده PostgreSQL مورد بررسی قرار می‌گیرند. پس از آن در فصل ۳، سیستم ثبت رخدادها برای استفاده در ممیزی شرح داده می‌شود و یکی از معروف‌ترین افزونه‌های[1] ممیزی برای نسخه‌ی متن‌باز PostgreSQL به نام pgAudit معرفی گردد. در آخر نیز ابزار edb_audit در EnterpriseDB Postgres Advanced server برای ثبت ممیزی در این پایگاه داده توصیه می‌گردد. لازم به ذکر است که پایگاه داده PostgreSQL ابزار مناسبی برای جرم‌شناسی و تجزیه‌وتحلیل رویدادهای ثبت‌شده ندارد.

۲ انواع فایل‌های رویدادنگاری در پایگاه داده PostgreSQL

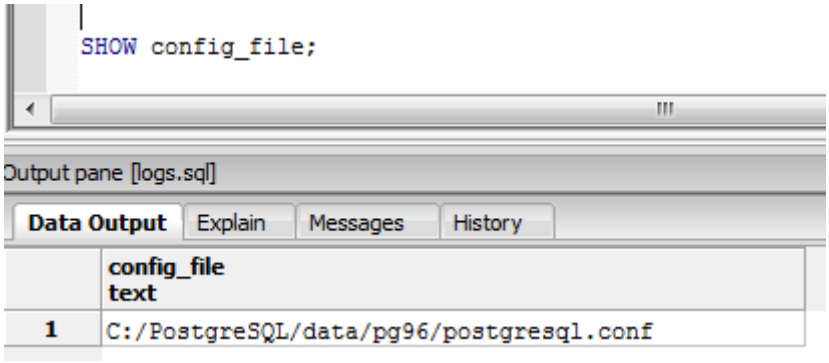
در این بخش انواع فایل‌های رویدادنگاری موردبررسی قرارگرفته‌اند. لازم به ذکر است که بررسی‌های انجام شده در این گزارش بر روی سیستم‌عامل ویندوز ۷ و سیستم مدیریت پایگاه داده‌ی PostgreSQL 9.6.5 صورت گرفته است.

۲-۱ ثبت فعالیت‌های پایگاه داده

ثبت فعالیت‌های پایگاه داده شامل پیغام‌های خطا، پرسمان‌ها و پیغام‌های راه‌اندازی/توقف است. این دسته از فایل‌های رویدادنگاری اولین مکانی هستند که در صورت عدم راه‌اندازی PostgreSQL، باید به دنبال اطلاعات در آن‌ها گشت [۱]. تنظیمات موجود در فایل postgresql.conf، ثبت فعالیت‌های پایگاه داده را کنترل می‌کنند. در ابتدا برای یافتن محل فایل پیکربندی postgresql.conf، دستور زیر اجرا می‌شود (Error! Reference source not found.):

```
SHOW config_file;
```

[1] Extension



شکل ۱ یافتن محل فایل پیکربندی postgresql.conf

حال می‌توان این فایل را با ویرایشگر متن دلخواه باز کرد و تغییرات موردنظر در تنظیمات را در قسمت ERROR REPORTING AND LOGGING اعمال کرد (**Error! Reference source not found.**). پارامترهای مختلفی در این فایل برای تنظیم ثبت رخدادها وجود دارند. در جدول ۱ این پارامترها معرفی‌شده‌اند [۲]

```
#------------------------------------------------------------------------------
# ERROR REPORTING AND LOGGING
#------------------------------------------------------------------------------

# - Where to Log -

#log_destination = 'stderr'     # Valid values are combinations of
                    # stderr, csvlog, syslog, and eventlog,
                    # depending on platform.  csvlog
                    # requires logging_collector to be on.

# This is used when logging to stderr:
logging_collector = on
                    # into log files. Required to be on for
                    # csvlogs.
                    # (change requires restart)

# These are only used if logging_collector is on:
log_directory = 'C:/POSTGR~1/data/logs/pg96'
                    # can be absolute or relative to PGDATA
log_filename = 'postgresql-%a.log'
                    # can include strftime() escapes
#log_file_mode = 0600           # creation mode for log files,
                    # begin with 0 to use octal notation
log_truncate_on_rotation = on
                    # same name as the new log file will be
                    # truncated rather than appended to.
                    # But such truncation only occurs on
                    # time-driven rotation, not on restarts
                    # or size-driven rotation.  Default is
                    # off, meaning append to existing files
                    # in all cases.
#log_rotation_age = 1d          # Automatic rotation of logfiles will
                    # happen after that time.  0 disables.
#log_rotation_size = 10MB       # Automatic rotation of logfiles will
                    # happen after that much log output.
                    # 0 disables.
```

شکل ۲ فایل postgresql.conf

جدول ۱ پارامترهای موجود در فایل postgresql.conf

پارامتر	مقادیر	توضیح
log_destination	stderr، csvlog ، syslog یا eventlog	PostgreSQL از چندین روش ازجمله stderr، csvlog، syslog و evenlog در ویندوز، برای ثبت پیغام‌های کارگزار پشتیبانی می‌کند. این پارامتر فهرستی از این مقادیر را دریافت می‌کند. مقدار پیش‌فرض stderr است. درصورتی‌که csvlog به عنوان مقدار پارامتر log_destination انتخاب شود، رکوردهای ثبت، فرمت CSV[۲] خواهند داشت.
logging_collector	on یا off	این پارامتر logging collector را فعال می‌کند. Logging collector در حقیقت فرآیند پس‌زمینه‌ای است که پیغام‌های ثبت ارسالی به stderr را دریافت و آن‌ها را به فایل‌های رویدادنگاری ارسال می‌کند.
log_directory	مسیر مطلق یا نسبی نسبت به دایرکتوری داده، برای ایجاد فایل‌های رویدادنگاری	در صورت فعال بودن logging_collector کاربرد دارد.
log_filename	الگویی برای نام فایل‌های	در صورت فعال بودن logging_collector کاربرد دارد.

[2] Comma separated value

	رویدادنگاری	
log_file_mode	مجوزهای لازم برای فایل‌های رویدادنگاری	در صورت فعال بودن logging_collector کاربرد دارد. بر روی سیستم‌های یونیکسی کاربرد داشته و برای ویندوز این پارامتر نادیده گرفته می‌شود.
log_rotation_age	حداکثر زمان حیات هر فایل رویدادنگاری	در صورت فعال بودن logging_collector کاربرد دارد. پس از گذشتن این زمان، فایل رویدادنگاری جدیدی ایجاد می‌شود. در صورت انتخاب مقدار صفر، ایجاد فایل‌های رویدادنگاری بر اساس زمان غیرفعال می‌گردد.
log_rotation_size	حداکثر سایز هر فایل رویدادنگاری	در صورت فعال بودن logging_collector کاربرد دارد. در صورت نوشته شدن مقدار کیلوبایت تعیین‌شده در فایل رویدادنگاری، فایل جدیدی ایجاد می‌شود. مقدار صفر، ایجاد فایل‌های رویدادنگاری جدید بر اساس سایز را غیرفعال می‌کند.
log_truncate_on_rotation	On یا off	درصورتی‌که logging_collector فعال باشد، با فعال بودن این گزینه و rotation بر اساس زمان، به جای پیوستن رویدادها به فایل رویدادنگاری با نام یکسان، بر روی آن مجدداً نوشته می‌شود.
syslog_facility	LOCAL0، LOCAL1، ... و LOCAL7	در صورت تعیین مقدار syslog برای log_destination کاربرد دارد. تعیین مقدار facility
syslog_ident	نام برنامه برای تعیین پیغام‌های	در صورت تعیین مقدار syslog برای log_destination کاربرد دارد. مقدار پیش‌فرض

	PostgreSQL در فایل‌های رویدادنگاری syslog	برای این پارامتر postgres است.
syslog_sequence_numbers	On یا off	در صورت تعیین مقدار syslog برای log_destination کاربرد دارد. در صورت فعال بودن این پارامتر، شماره متوالی افزایشی، پیشوند هر پیغام می‌شود.
syslog_split_messages	On یا off	در صورت تعیین مقدار syslog برای log_destination کاربرد دارد. این پارامتر چگونگی تحویل پیغام‌ها به syslog را مشخص می‌کند. در صورت فعال بودن پارامتر، پیغام‌ها با خط‌ها تقسیم شده و خط‌های طولانی نیز تقسیم می‌شوند ولی در صورت غیرفعال بودن پارامتر، پیغام‌ها همان‌گونه که هستند به سرویس syslog تحویل داده می‌شوند.
event_source	نام برنامه برای تعیین پیغام‌های PostgreSQL در رویدادهای ثبت‌شده	درصورتی‌که مقدار پارامتر log_destination، eventlog باشد این پارامتر کاربرد دارد. مقدار پیش‌فرض PostgreSQL است.
client_min_messages	DEBUG5، DEBUG4، DEBUG3، DEBUG2، DEBUG1، LOG، NOTICE،	سطوح پیغام‌هایی که به کلاینت ارسال می‌شوند را کنترل می‌کند. هر سطح شامل تمامی سطوحی است که به دنبال آن آمده‌اند. مقدار پیش‌فرض NOTICE است.

	WARNING، ERROR، FATAL و PANIC	
log_min_messages	DEBUG5، DEBUG4، DEBUG3، DEBUG2، DEBUG1، INFO، NOTICE، WARNING، ERROR، LOG، FATAL و PANIC	سطوح پیغام‌هایی که در server log نوشته می‌شوند را کنترل می‌کند. هر سطح شامل تمامی سطوحی است که به دنبال آن آمده‌اند.
log_min_error_statement	DEBUG5، DEBUG4، DEBUG3، DEBUG2، DEBUG1، INFO، NOTICE، WARNING، ERROR، LOG، FATAL و PANIC	این پارامتر کنترل می‌کند که کدام عبارات SQL که باعث ایجاد شرایط خطا شده‌اند، در server log ثبت شوند. مقدار پیش‌فرض ERROR است یعنی تمامی عباراتی که باعث ایجاد ERROR، پیغام‌های LOG، خطاهای FATAL یا PANIC شده‌اند، ثبت شوند.
Log_min_duration_statement	حداقل زمان برای کامل اجراشدن عبارت	تمامی عباراتی که به اندازه زمان مشخص‌شده یا بیشتر، زمان برای اجرا نیاز داشته باشند، ثبت می‌شوند. درصورتی‌که این مقدار صفر باشد، تمامی

		عبارات ثبت می‌شوند.
debug_print_parse debug_print_rewritten debug_print_plan	On یا off	در صورت فعال شدن این گزینه‌ها، برای هر پرسمانی که اجرا می‌شود موارد زیر چاپ می‌شوند: - نتیجه‌ی درخت تجزیه[3] - خروجی rewriter - طرح اجرایی[4]
Debug_pretty_print	On یا off	در صورت فعال بودن این گزینه، خروجی سه گزینه‌ای که در سطر بالا توضیح داده شد، خواناتر و با فرمت نمایشی مناسب‌تری نشان داده می‌شوند.
log_checkpoints	On یا off	باعث می‌شود که نقاط بررسی[5] و نقاط بازآغازی[6] ثبت شوند.
log_connections	On یا off	سبب می‌شود هر تلاش برای اتصال به کارگزار ثبت شود.
log_disconnections	On یا off	سبب می‌شود رویداد قطع نشست، ثبت شود.
log_duration	On یا off	باعث ثبت مدت زمان خاتمه‌ی هر عبارت می‌شود.
log_error_verbosity	Terse، default یا verbose	میزان جزئیات نوشته شده در فایل‌های رویدادنگاری را کنترل می‌کند.
log_hostname	On یا off	در کنار آدرس IP ی میزبان اتصال[7]، نام میزبان[8] را

[3] Parse tree

[4] Execution plan

[5] Checkpoints

[6] Restartpoints

		نیز ثبت می‌کند.
log_line_prefix	الگوی شروع هر خط ثبت رخداد	به‌عنوان‌مثال الگویی شبیه الگوی زیر تعیین می‌شود: %t [%p]: [%l-1] user=%u,db=%d,app=%a,client=%h'
log_lock_waits	On یا off	تعیین می‌کند، درصورتی‌که نشستی بیشتر از deadlock_timeout برای گرفتن قفل صبر کند، آیا پیام ثبتی ایجاد شود یا خیر.
log_statement	None (به معنی off)، ddl، mod (تمامی کتابخانه‌های ddl و تمامی عباراتی که داده را تغییر می‌دهند) و all (به معنی تمامی عبارات)	کنترل می‌کند که کدامین عبارات SQL ثبت شوند.
log_replication_commands	On یا off	سبب می‌شود هر دستور replication در server log ثبت شود.
log_temp_files	سایزی که درصورتی‌که فایل موقت[9] آن سایز یا بیشتر را داشته	فایل‌های موقت برای اهداف مختلفی ایجاد می‌شوند. زمانی که فایل موقت حذف می‌شود، برای آن یک رکورد ثبت تهیه می‌شود. مقدار صفر، تمامی اطلاعات مربوط به فایل موقت را ثبت

[7] Connection host

[8] Host name

[9] Temporary file

	باشد، رویدادی ثبت می‌شود.	می‌کند و مقدار 1- این نوع ثبت رخداد را غیرفعال می‌کند.
log_timezone	منطقه زمانی	برای مهر زمانی[10] که در server log نوشته می‌شود، منطقه‌ی زمانی[11] تعیین می‌شود.

نمونه‌ای از رویدادهای ثبت‌شده توسط تنظیمات اعمال‌شده در فایل پیکربندی postgresql.conf در شکل ۲ دیده می‌شود. در این شکل سعی شده است، خطاهایی تولید و ثبت رخدادهای آن‌ها نشان داده شوند.

```
s,app=pgAdmin3 LTS by BigSQL - Query Tool,client=127.0.0.1 STATEMENT:  create table test;
 LOG:  checkpoint starting: time
 LOG:  checkpoint complete: wrote 0 buffers (0.0%); 0 transaction log file(s) added, 0 removed, 0 recycled;
s,app=pgAdmin3 LTS by BigSQL - Query Tool,client=127.0.0.1 ERROR:  unterminated quoted string at or near "'


s,app=pgAdmin3 LTS by BigSQL - Query Tool,client=127.0.0.1 STATEMENT:  insert into test2 values ('aaaaaaaaa


s,app=pgAdmin3 LTS by BigSQL - Query Tool,client=127.0.0.1 ERROR:  unterminated quoted string at or near "'


s,app=pgAdmin3 LTS by BigSQL - Query Tool,client=127.0.0.1 STATEMENT:  create table test2 (a varchar(10));


aaaaaa);


s,app=pgAdmin3 LTS by BigSQL - Query Tool,client=127.0.0.1 ERROR:  value too long for type character varyir
s,app=pgAdmin3 LTS by BigSQL - Query Tool,client=127.0.0.1 STATEMENT:  insert into test2 values ('aaaaaaaaa


s,app=pgAdmin3 LTS by BigSQL - Query Tool,client=127.0.0.1 ERROR:  value too long for type character varyir
es,app=pgAdmin3 LTS by BigSQL - Query Tool,client=127.0.0.1 STATEMENT:  insert into test2 values ('aaaaaaaa
```

شکل ۲ نمونه‌ای از فایل رویدادنگاری

[10] Timestamp

[11] Time zone

۲-۲ pg_xlog

pg_xlog در حقیقت ثبت رخدادهای تراکنش در PostgreSQL است. این مجموعه از فایل‌های رویدادنگاری دودیی[۱۲] با نام‌هایی شبیه '000000010000000000000001'، شامل تصاویری[۱۳] از داده‌های تراکنش‌های اخیر هستند. این دسته از فایل‌های رویدادنگاری، برای تکرار دودویی نیز استفاده می‌شوند[۱]. دایرکتوری pg_xlog که در دایرکتوری داده قرار دارد فایل‌های WAL[۱۴] را در خود نگه می‌دارد. فایل‌های WAL شامل رکوردهایی از تمامی تغییرات درون پایگاه داده هستند. یک مشکل عمومی در PostgreSQL پرشدن فضای دیسک در دایرکتوری pg_xlog است؛ به همین دلیل توصیه می‌شود که این دایرکتوری در دیسکی جداگانه نسبت به داده‌ها نگهداری شود [۳] .

۲-۳ pg_clog

pg_clog شامل ثبت رویداد از فراداده‌ی تراکنش است. این دسته از فایل‌های رویدادنگاری به PostgreSQL اطلاع می‌دهند که کدام‌یک از تراکنش‌ها پایان‌یافته‌اند و کدام‌یک پایان نیافته‌اند. درصورتی‌که از دایرکتوری داده، فایل‌های پشتیبان تهیه شود، باید فایل‌های پشتیبان شامل pg_clog و pg_xlog هم باشند، در غیر این صورت فایل‌های پشتیبان قابل‌استفاده نخواهند بود [۱]. به دلیل آنکه فایل‌های رویدادنگاری در دایرکتوری pg_clog برای استفاده داخلی PostgreSQL است، قابل‌خواندن توسط کاربران نیستند [۴].

۳ نحوه‌ی انجام ممیزی در پایگاه داده PostgreSQL

در سازمان‌ها و شرکت‌های مختلف نیاز به ممیزی[۱۵] در سطوح مختلفی احساس می‌شود. در برخی از شرکت‌ها اطلاعاتی با جزئیات بیشتر موردنیاز است و در برخی دیگر تنها از کسانی که به پایگاه داده متصل می‌شوند، ممیزی تهیه می‌شود [۵]. به صورت پیش‌فرض بر روی نسخه‌ی متن‌باز[۱۶] پایگاه داده‌ی PostgreSQL، قابلیت ممیزی وجود ندارد؛ بنابراین در ادامه ابتدا سیستم ثبت رخدادها برای استفاده در

---

[12] Binary

[13] Images

[14] Write ahead log

[15] Auditing

[16] Open source

ممیزی توضیح داده شده است. پس از آن، یکی از معروف‌ترین افزونه‌های[17] ممیزی برای نسخه‌ی متن‌باز PostgreSQL به نام pgAudit معرفی می‌شود. در آخر، edb_audit در EnterpriseDB Postgres Advanced server برای ثبت ممیزی نیز توضیح داده می‌شود.

۳-۱ سیستم ثبت رخدادها

برخی از پارامترهایی که در جدول ۱ به آن‌ها اشاره شده است، برای تهیه‌ی ممیزی مورداستفاده قرار می‌گیرند (همچون log_connections و log_disconnections، برای تهیه ممیزی از ورود و خروج از پایگاه داده).

پارامتر دیگری که برای ممیزی پرکاربرد است پارامتر log_statement است. درصورتی‌که مقدار این پارامتر برابر ddl تنظیم شود، تمامی عبارات DDL ثبت می‌شوند و در صورت تنظیم مقدار mod، تمامی عبارات DDL به همراه عبارات تغییردهنده‌ی داده ثبت می‌شوند. به‌منظور ثبت همه‌ی عبارات، مقدار all برای این پارامتر تنظیم می‌شود [۵].

۳-۲ افزونه‌ی ممیزی در PostgreSQL

برخی نیازمندی‌ها مانند تهیه‌ی ممیزی در سطح شیئ سبب می‌شوند که تنها استفاده از سیستم ثبت رخدادها کافی نباشد. در این صورت می‌توان از افزونه‌ی pgAudit استفاده کرد [۵]. pgAudit امکان تهیه‌ی ممیزی با جزئیات، از شیئ و/یا نشست را فراهم می‌کند [۶].

ثبت عبارات با استفاده از پارامتر log_statement = all و تهیه‌ی فهرستی از تمامی عملیات انجام‌شده در پایگاه داده، برای نظارت و سایر کاربردها قابل‌قبول است، اما برای نیازمندی‌های مرتبط با ممیزی کافی نیست. در ممیزی، امکان یافتن عبارات موردنظر ممیزی گیرنده[18]، نیز لازم است. ثبت رخدادها نشان می‌دهد که کاربر چه درخواستی داشته است ولی pgAudit بر روی جزئیات آنچه در حین انجام درخواست رخ داده است، تمرکز دارد [۶].

[17] Extension

[18] Auditor

۳-۲-۱ نصب افزونه‌ی pgAudit

به‌منظور نصب pgAudit مراحل زیر طی می‌شوند:

- ابتدا در خط فرمان[19]، وارد فولدر نصب PostgreSQL شده (cd C:\PostgreSQL) و دستورات زیر به ترتیب اجرا می‌شوند [۷]:

```
pgc update
pgc install pgaudit
```

- پس‌ازآن، postgresql.conf را با ویرایشگر متن دلخواهی باز کرده و خط زیر در آن وارد می‌شود:

```
shared_preload_libraries='pgaudit'
```

- در آخر، کارگزار را مجدداً راه‌اندازی کرده و به psql وصل شده و دستور زیر اجرا می‌شود:

```
CREATE EXTENSION pgaudit;
```

۳-۲-۲ انواع ثبت ممیزی با افزونه‌ی pgAudit

پس از نصب pgAudit، می‌توان از آن برای ثبت ممیزی نشست و ثبت ممیزی شیئ استفاده کرد. در ادامه این بخش هر یک از آن‌ها به تفضیل بیان‌شده‌اند.

۳-۲-۲-۱ ثبت ممیزی نشست[20]

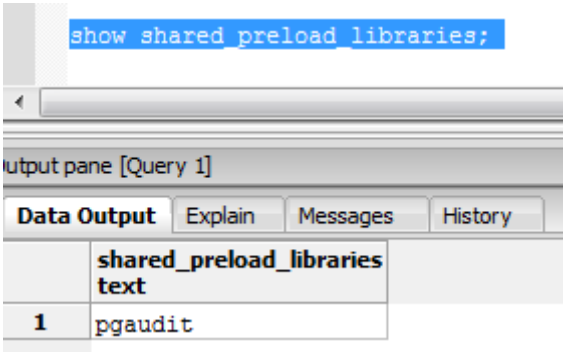
ثبت ممیزی نشست، تمامی عبارات اجراشده توسط کاربر را با جزئیات ثبت می‌کند [۷].

- ابتدا نسبت به فعال بودن pgAudit با دستور زیر اطمینان حاصل می‌شود (شکل ۳)[۷]:

```
SHOW shared_preload_libraries;
```

[19] Command line

[20] Session audit logging



شکل ۳ بررسی فعال بودن pgAudit

- ثبت ممیزی نشست با تنظیم pgaudit.log فعال می‌شود. به عنوان نمونه دستور زیر عبارات خواندن و ddl را برای ثبت ممیزی فعال می‌کند.

```
SET pgaudit.log = 'read, ddl';
```

در حقیقت در pgaudit.log کلاس‌های عبارات برای ثبت توسط ثبت ممیزی نشست، مشخص می‌شوند. کلاس‌های عبارات شامل موارد زیر هستند [۶]:

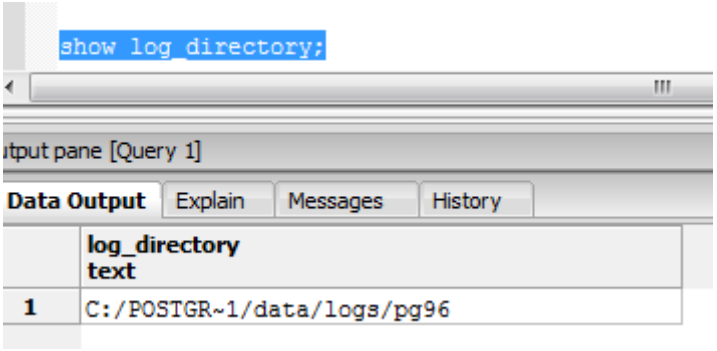
- READ: SELECT و COPY درصورتی‌که مبدأ یک رابطه یا یک پرسمان باشد.
- WRITE: INSERT، UPDATE، DELETE، TRUNCATE و COPY درصورتی‌که مقصد یک رابطه باشد.
- FUNCTION: فراخوانی تابع و بلاک‌های DO
- ROLE: عبارات مرتبط با نقش‌ها و مجوزها شامل GRANT، REVOKE، CREATE/ALTER/DROP ROLE
- DDL: تمامی عبارات DDL که در کلاس ROLE شامل نمی‌شوند
- MISC: دستورات متفرقه همچون DISCARD، FETCH، CHECKPOINT و VACUUM

- به‌منظور آزمودن ممیزی، دستورات زیر اجرا می‌شوند:

```
CREATE TABLE industry(id int,name text);
INSERT INTO industry (id,name)VALUES (11,'polo');
SELECT * FROM industry;
```

- حال می‌توان رخدادهای ثبت‌شده را در فایل‌های موجود در دایرکتوری خروجی دستور زیر مشاهده کرد (شکل ۴):

```
SHOW log_directory;
```



شکل ۴ دایرکتوری مربوط به فایل‌های رویدادنگاری

نمونه‌ای از این فایل، مربوط به دستورات اجراشده، در شکل ۵ نشان داده شده است.

```
AUDIT: SESSION,3,1,DDL,CREATE TABLE,TABLE,public.industry,"create table industry(id int,name text);

AUDIT: SESSION,4,1,READ,SELECT,,,"create table industry(id int,name text);

AUDIT: SESSION,5,1,READ,SELECT,,,"SELECT format_type(oid,-1) as typname FROM pg_type WHERE oid = 23",
AUDIT: SESSION,6,1,READ,SELECT,,,"SELECT CASE WHEN typbasetype=0 THEN oid else typbasetype END AS bas

AUDIT: SESSION,7,1,READ,SELECT,,,"SELECT format_type(oid,-1) as typname FROM pg_type WHERE oid = 25",
AUDIT: SESSION,8,1,READ,SELECT,,,"SELECT CASE WHEN typbasetype=0 THEN oid else typbasetype END AS bas

  relation "industry" already exists
MENT:  create table industry(id int,name text);

AUDIT: SESSION,9,1,READ,SELECT,,,select * from industry;,<not logged>
```

شکل ۵ ثبت ممیزی نشست

۲-۲-۲-۳ ثبت ممیزی شیئ

ثبت ممیزی شیئ[۲۱]، عباراتی که بر روی یک رابطه‌ی[۲۲] (همچون جدول و دید) خاص تاثیر می‌گذارند را ثبت می‌کند. در ثبت ممیزی شیئ، تنها دستورات SELECT، INSERT، UPDATE و DELETE پشتیبانی می‌شوند. در حقیقت این نوع ممیزی گونه‌ی ریزدانه‌تر از pgaudit.log = 'read, write' است [۷].

ثبت ممیزی در سطح شیئ با استفاده از سیستم نقش‌ها پیاده‌سازی شده است. با تنظیم pgaudit.role نقشی تعریف می‌شود که برای ثبت ممیزی مورداستفاده قرار می‌گیرد. درصورتی‌که نقش تعریف‌شده دارای مجوزهای لازم بر روی دستور اجراشده باشد یا مجوزها را از نقش دیگری به ارث ببرد، از رابطه، ممیزی تهیه می‌شود. برای واضح‌تر شدن مطالب در ادامه مثالی از ثبت ممیزی شیئ آورده شده است [۷].

- ابتدا نقشی با دستور زیر تعریف می‌شود:

```
CREATE ROLE auditor;
```

- سپس pgaudit.role تنظیم می‌شود:

```
SET pgaudit.role = 'auditor';
```

- پس‌ازآن، دو جدول با ساختار و داده‌های یکسان ایجاد و به نقش auditor تنها دسترسی‌های لازم برای یک جدول اعطا می‌شود:

```
CREATE TABLE test1 (a int);
CREATE TABLE test2 (a int);


INSERT INTO test1 VALUES(10);
INSERT INTO test1 VALUES(20);


INSERT INTO test2 VALUES(10);
INSERT INTO test2 VALUES(20);
```

[21] Object audit logging

[22] Relation

```
GRANT select,insert,update,delete ON test1 TO auditor;
```

- با توجه به آنکه نقش auditor تنها بر روی test1 مجوز دسترسی دارد، پرسمان SELECT بـر روی test2 در فایل رویدادنگاری، ثبت نشده و تنها پرسمان SELECT بر روی test1، ثبت می‌شود:

```
2017-09-10 23:23:14 +0430 [7128]: [30-1] user=postgres,db=postgres,app=pgAdmin3 LTS by BigSQL - Query Tool,client=127.0.0.1 LOG: AUDIT: OBJECT,20,1,READ,SELECT,TABLE,public.test1,"select * from test1;",<not logged>
```

- می‌توان تنظیماتی بر اساس ستون‌ها داشت. بدین منظور دستور زیر اجرا می‌شود:

```
GRANT select (a) ON test2 TO auditor;
```

- حال پرسمان SELECT بر روی ستون تعیین‌شده، باعث ثبت رویدادی مشابه مورد زیر می‌شود:

```
2017-09-10 23:30:28 +0430 [7128]: [33-1] user=postgres,db=postgres,app=pgAdmin3 LTS by BigSQL - Query Tool,client=127.0.0.1 LOG: AUDIT:
OBJECT,21,1,READ,SELECT,TABLE,public.test2,select a from test2;,<not logged>
```

۳-۲-۳ ساختار رکوردهای ممیزی

رکوردهای ممیزی شامل ستون‌های زیر هستند؛ این ستون‌ها با کاما از هم جدا می‌شوند [۶].

- نوع ممیزی: SESSION یا OBJECT
- شناسه یکتایی برای عبارت در یک نشست
- شناسه‌ی ترتیبی[۲۳] برای هر زیر عبارت داخل عبارت اصلی
- کلاس مثل READ، ROLE و ... (موارد مربوط به pgaudit.log)
- دستور مثل ALTER TABLE، SELECT

[23] Sequential ID

- نوع شیئ شامل جدول، شاخص، دید و غیره. برای عبارات SELECT، DML و اکثر عبارات DDL نوع شیئ تعیین می‌شود.
- نام کامل شیئ. برای عبارات SELECT، DML و اکثر عبارات DDL نام شیئ تعیین می‌شود.
- عبارت اجراشده
- پارامترهای تعیین‌شده با pgaudit.log_parameter

در مثال زیر نمونه‌ای از اجزای یک رکورد ممیزی دیده می‌شود:

```
OBJECT,21,1,READ,SELECT,TABLE,public.test2,select a from test2;,<not logged>
```

۳-۳ Edb_autit در EDB Postgres Advanced Server

Postgres Advanced Server امکان ردگیری و تحلیل فعالیت‌های پایگاه داده را با استفاده از ثبت ممیزی‌ها فراهم می‌کند. ثبت ممیزی برای به دست آوردن اطلاعاتی مشابه اطلاعات زیر تنظیم می‌شود [۸]:

- چه زمانی یک نقش به Advanced server متصل شده است.
- یک نقش هنگامی‌که به Advanced server متصل شده است چه اشیایی را ایجاد، تغییر و حذف کرده است.
- چه زمانی تلاش برای تصدیق اصالت شکست خورده است.

در ادامه برخی از پارامترهایی که در فایل postgresql.conf برای فعال و تنظیم نمودن edb_audit مورداستفاده قرار می‌گیرند، شرح داده می‌شوند:

- edb_audit: فعال و غیرفعال کردن ممیزی پایگاه داده. مقادیر xml یا csv ثبت ممیزی را فعال و مقدار none ممیزی را غیرفعال می‌کند.
- edb_audit_directory: با این پارامتر دایرکتوری موردنظر برای ایجاد فایل‌های رویدادنگاری مشخص می‌شود.

- edb_audit_filename: نام فایل‌های ممیزی که اطلاعات ممیزی در آن‌ها ذخیره می‌شوند را مشخص می‌کند.
- edb_audit_connect: به‌منظور ثبت تلاش کاربران برای اتصال به پایگاه داده از این پارامتر استفاده می‌شود. برای غیرفعال کردن تهیه‌ی ممیزی از تلاش‌های اتصال، مقدار این پارامتر none تنظیم می‌شود. با تنظیم مقدار failed، تلاش‌های ناموفق برای اتصال ثبت می‌شوند و با تنظیم all تمامی تلاش‌ها برای اتصال، ثبت خواهند شد.
- edb_audit_disconnect: به‌منظور ثبت قطع اتصال کاربران از پایگاه داده مقدار این پارامتر all و برای غیرفعال کردن ثبت این دسته از رویدادها مقدار none، ثبت می‌شود.
- edb_audit_statement: به‌منظور ممیزی دسته‌های مختلف عبارات SQL این پارامتر باید به‌درستی تنظیم شود. مقادیر مجاز برای این پارامتر در ادامه بیان شده است:
  - error: برای تهیه ممیزی از عباراتی که منجر به خطا می‌شوند، مقدار error برای پارامتر تنظیم می‌شود.
  - all: به‌منظور تهیه ممیزی از کلیه‌ی عبارات
  - ddl
  - dml
  - rollback
  - None

در شکل ۶ نمونه‌ای از فایل postgresql.conf و تنظیمات مربوط به edb_audit نشان داده شده است.

```
#------------------------------------------------------------------------------
# EDB AUDIT
#------------------------------------------------------------------------------

edb_audit = 'csv'            # none, csv or xml

# These are only used if edb_audit is not none:
edb_audit_directory = 'edb_audit'    # Directory where log files are written
                    # Can be absolute or relative to PGDATA

edb_audit_filename = 'audit-%Y-%m-%d_%H%M%S' # Audit file name pattern.
                    # Can include strftime() escapes

edb_audit_rotation_day = 'every'     # Automatic rotation of logfiles based
                    # on day of week. none, every, sun,
                    # mon, tue, wed, thu, fri, sat

edb_audit_rotation_size = 0      # Automatic rotation of logfiles will
                    # happen after this many megabytes (MB)
                    # of log output.  0 to disable.

edb_audit_rotation_seconds = 0       # Automatic log file rotation will
                    # happen after this many seconds.

edb_audit_connect = 'failed'         # none, failed, all
edb_audit_disconnect = 'none'        # none, all
edb_audit_statement = 'ddl, error'  # none, dml, ddl, select, error, rollback, all
#edb_audit_tag = ''          # Audit log session tracking tag.
```

شکل ۶ تنظیمات مربوط به edb_audit

نمونه‌ای از رویدادهای ثبت‌شده توسط edb_audit در شکل ۷ نمایش داده شده است.

```
"statement: CREATE TEMP TABLE pga_tmp_zombies(jagpid int4)",,,,,,,,,"",""
,"statement: DROP TABLE pga_tmp_zombies",,,,,,,,,"",""
,"statement: CREATE TABLE test (a int);",,,,,,,,,"pgAdmin 4 - CONN:2631542",""
,"statement: CREATE TABLE test (a int);",,,,,,,,,"pgAdmin 4 - CONN:4819283",""
,"relation ""test"" already exists",,,,,,"CREATE TABLE test (a int);",,,"pgAdmin 4 - CONN:4819283",
,"statement: CREATE TABLE test (a int);",,,,,,,,,"pgAdmin 4 - CONN:4819283",""
,"relation ""test"" already exists",,,,,,"CREATE TABLE test (a int);",,,"pgAdmin 4 - CONN:4819283",
,"statement: CREATE TABLE test (a int);",,,,,,,,,"pgAdmin 4 - CONN:4819283",""
,"relation ""test"" already exists",,,,,,"CREATE TABLE test (a int);",,,"pgAdmin 4 - CONN:4819283",
```

شکل ۷ نمونه‌ای از رویدادهای ثبت‌شده توسط edb_audit

٤ جمع‌بندی

پایگاه داده‌ی PostgreSQL دارای فایل پیکربندی جامعی برای ثبت رویدادها است. برخی نیازمندی‌ها مانند تهیه‌ی ممیزی در سطح شیئ سبب می‌شوند که تنها استفاده از سیستم ثبت رخدادها کافی نباشد. به صورت پیش‌فرض بر روی نسخه‌ی متن‌باز پایگبشکاه داده‌ی PostgreSQL، قابلیت ممیزی وجود ندارد؛ هرچند، برای این پایگاه داده، افزونه‌های ممیزی مختلفی وجود دارد که نمونه‌ای از آن به نام pgAudit در گزارش توضیح داده شد. لازم به ذکر است که پایگاه داده‌ی PostgreSQL ابزار مناسبی برای جرم‌شناسی و تجزیه‌وتحلیل رویدادهای ثبت‌شده ندارد.

٥ منابع

[1]. http://it.toolbox.com/blogs/database-soup/pg_log-pg_xlog-and-pg_clog-45611

[2]. https://www.postgresql.org/docs/9.6/static/runtime-config-logging.html

[3]. http://blog.endpoint.com/2014/09/pgxlog-disk-space-problem-on-postgres.html

[4]. http://www.dbrnd.com/2016/12/postgresql-difference-between-pg_log-pg_clog-and-pg_xlog-log-directories-transaction-log-binary-log/

[5]. https://blog.dbi-services.com/auditing-in-postgresql/

[6]. https://github.com/pgaudit/pgaudit

[7]. https://www.openscg.com/bigsql/docs/security/pgaudit

[8]. https://www.enterprisedb.com/docs/en/9.4/eeguide/Postgres_Plus_Enterprise_Edition_Guide.1.020.html