

Take Your pgBadger Usage to the Next Level

Discover the Potential of Log Analysis Like Never Before

DRII

Hettie Dombrovskaya
Database Architect

PG Day Lowland 2024

Who Am I

pgBadger New Level

Hettie Dombrovskaya

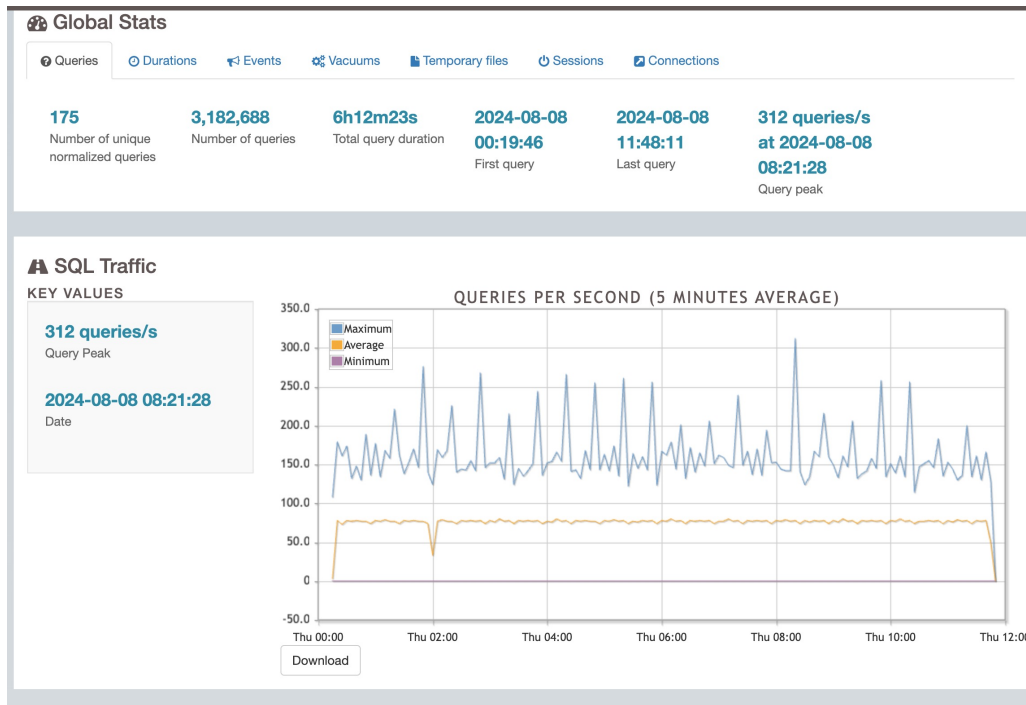
Database Architect at DRW
Local Organizer of the Chicago PostgreSQL User Group

PG Day Chicago Organizer



Why we love pgBadger

Report Example



🕒 Time consuming queries (N)

Rank	Total duration	Times executed	Min duration	Max duration	Avg duration	Query
1	6h2m38s	8,543 Details	2s483ms	2s569ms	2s546ms	🔗 DELETE FROM customer_survey_response WHERE response -> ? ->> ? = ?; Examples User(s) involved App(s) involved
2	3m53s	33,054 Details	6ms	17ms	7ms	🔗 SELECT ct.customer_id, p.item_price, ct.num_items, ct.sales_point_id, p.item_id FROM customers_sales ct CROSS JOIN products p; Examples User(s) involved App(s) involved
3	1m1s	6,210 Details	9ms	10ms	9ms	🔗 SELECT response_text, row_number FROM customer_survey_response WHERE created_at > ? AND response_text -> ? ->> ? = ? ORDER BY row_number; Examples User(s) involved App(s) involved
4	47s271ms	18,831 Details	2ms	4ms	2ms	🔗 SELECT product_id, calories FROM nutrition; Examples User(s) involved App(s) involved

🕒 Normalized slowest queries (N)

Rank	Min duration	Max duration	Avg duration	Times executed	Total duration	Query
1	2s483ms	2s569ms	2s546ms	8,543 Details	6h2m38s	🔗 DELETE FROM customer_survey_response WHERE response -> ? ->> ? = ?; Examples User(s) involved
2	9ms	10ms	9ms	6,210 Details	1m1s	🔗 SELECT response_text, row_number FROM customer_survey_response WHERE created_at > ? AND response_text -> ? ->> ? = ? ORDER BY row_number; Examples User(s) involved
3	6ms	17ms	7ms	33,054 Details	3m53s	🔗 SELECT ct.customer_id, p.item_price, ct.num_items, ct.sales_point_id, p.item_id FROM customers_sales ct CROSS JOIN products p; Examples User(s) involved
4	0ms	11ms	3ms	1,029 Details	3s820ms	🔗 SELECT current_database() datname, schemaname, relname, seq_scan, seq_tup_read, idx_scan, idx_tup_fetch, n_tup_ins, n_tup_upd, n_tup_del, n_tup_hot_upd, n_live_tup, n_dead_tup, n_mod_since_analyze, coalesce(last_vacuum, ?), coalesce(last_vacuum, ?) AS last_vacuum, coalesce(last_autovacuum, ?) AS last_autovacuum, coalesce(last_analyze, ?) AS last_analyze, coalesce(last_autoanalyze, ?) AS last_autoanalyze, vacuum_count, autovacuum_count, analyze_count, autoanalyze_count FROM pg_stat_user_tables; Examples User(s) involved

What is was missing?



Open Questions

Most frequent queries (N)

Rank	Times executed	Total duration	Min duration	Max duration	Avg duration	Query
1	904,927 Details	1s724ms	0ms	0ms	0ms	<code>BEGIN;</code> Examples User(s) involved App(s) involved
2	904,504 Details	4s192ms	0ms	8ms	0ms	<code>COMMIT;</code> Examples User(s) involved App(s) involved
3	199,238 Details	8s501ms	0ms	0ms	0ms	<code>SELECT id, address_line1, city, zipcode, plant_type_id, is_active FROM manufactures WHERE name = ?;</code> Examples User(s) involved App(s) involved
4	126,310 Details	2s93ms	0ms	0ms	0ms	<code>SELECT active, category, flavor, campaign, campaign_start, campaign_end, promotion_code, excluded_items, customer_items, customer_participation FROM settings, customer_settings, campaign_settings;</code> Examples User(s) involved App(s) involved
5	88,520 Details	8s690ms	0ms	0ms	0ms	<code>SELECT customer_id, salespoint_id FROM customer_salespoints;</code> Examples User(s) involved App(s) involved
6	81,874 Details	2s471ms	0ms	0ms	0ms	<code>SELECT product_id, product_type, promotion_active FROM product_promotions;</code> Examples User(s) involved App(s) involved
7	81,874 Details	1s15ms	0ms	0ms	0ms	<code>SELECT id FROM salespoints WHERE is_current = TRUE;</code>

I knew there is an answer,

I was just unsure where...

logs_fdw?



Parsing???

THANK YOU GILLS DAROLD!!!



Now we are ready to present...

New pgBadger Magic!

What will be covered

- Changes made to pgBadger by Gills Darold
- How we can use it
- Logs processing automation
- Load Example
- Problem-solving example
- Logs processing by the numbers
- What's next?

pgBadger New Level

New pgBadger features

New pgBadger options

`--dump-raw-csv` – produces
parsed log, in a csv format

`--csv-separator` – assigns separator different
from ‘,’

```
pgbadger /Users/hettie/chocolate.log --dump-raw-csv --csv-separator # >  
chocolate.csv
```


Loading csv into Postgres database

```

9. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.002 ms bind <unnamed>: BEGIN
10. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.004 ms execute <unnamed>: BEGIN
11. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.023 ms parse <unnamed>: SELECT prod
12. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.028 ms bind <unnamed>: SELECT prod
13. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.034 ms execute <unnamed>: SELECT p
14. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.003 ms parse <unnamed>: COMMIT
15. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.004 ms bind <unnamed>: COMMIT
16. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.005 ms execute <unnamed>: COMMIT
17. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.008 ms parse <unnamed>:
18. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.002 ms bind <unnamed>:
19. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.004 ms parse <unnamed>: BEGIN
20. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.002 ms bind <unnamed>: BEGIN
21. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.003 ms execute <unnamed>: BEGIN
22. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.033 ms parse <unnamed>: select id
23. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.003 ms parse <unnamed>: BEGIN
24. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.001 ms bind <unnamed>: BEGIN
25. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.001 ms execute <unnamed>: BEGIN
26. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.031 ms bind <unnamed>: select id f
27. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.017 ms parse <unnamed>: SELECT cam
28. er=sales_app,db=chocolate_db,app=PostgreSQL JDBC Driver,client=99.88.010.27 LOG: duration: 0.020 ms execute <unnamed>: select i

```

SELECT * FROM CHOCOLATE_LOG

sqlstate text	duration numeric	query text
[null]	0.022	SELECT active, category, flavor, campaign, campaign_start, campaign_end, promo
[null]	0.005	COMMIT
[null]	0.004	BEGIN
[null]	0.034	SELECT product_id, product_type, promotion_active FROM product_promotions
[null]	0.005	COMMIT
[null]	0.003	BEGIN
[null]	0.001	BEGIN
[null]	0.009	SELECT campaign_id, campaign_code, campaign_start_date, campaign_end_date
[null]	0.005	COMMIT
[null]	0.003	COMMIT
[null]	0.632	SELECT product_id, ingr_id, ingredient_name, weight, calories FROM product_ingr
[null]	0.483	SELECT p.product_id, promotion_code, discount, base_price effective FROM prom
[null]	0.003	BEGIN
[null]	0.045	SELECT counterparty_id, amount_in_usd FROM counterparty_notional_exposure
[null]	0.004	COMMIT
[null]	0.002	BEGIN
[null]	0.165	SELECT salespoint_id, user_id, product_id, num_sold FROM users_sales
[null]	[null]	p.item_price, ct.num_items, ct.sales_point_id, p.item_id
[null]	0.001	BEGIN
[null]	0.005	COMMIT
[null]	0.125	SELECT salespoint_id, user_id, group FROM users

4	-05	sales_app	chocolate_db	35146	99.88.010.27	[null]	LOG
5	-05	sales_app	chocolate_db	35146	99.88.010.27	[null]	LOG
6	-05	sales_app	chocolate_db	27951	99.88.010.27	[null]	LOG
7	-05	sales_app	chocolate_db	33338	99.88.010.27	[null]	LOG
8	-05	sales_app	chocolate_db	33338	99.88.010.27	[null]	LOG
9	-05	sales_app	chocolate_db	27951	99.88.010.27	[null]	LOG
10	-05	sales_app	chocolate_db	33338	99.88.010.27	[null]	LOG
11	-05	sales_app	chocolate_db	2911453	99.88.010.27	[null]	LOG
12	-05	sales_app	chocolate_db	2911453	99.88.010.27	[null]	LOG
13	-05	sales_app	chocolate_db	34170	99.88.010.27	[null]	LOG
14	-05	sales_app	chocolate_db	34170	99.88.010.27	[null]	LOG
15	-05	sales_app	chocolate_db	34170	99.88.010.27	[null]	LOG
16	-05	sales_app	chocolate_db	36525	99.88.010.27	[null]	LOG
17	-05	sales_app	chocolate_db	36525	99.88.010.27	[null]	LOG
18		[null]	[null]	[null]	[null]	[null]	[null]
19	-05	sales_app	chocolate_db	35146	99.88.010.27	[null]	LOG
20	-05	sales_app	chocolate_db	36525	99.88.010.27	[null]	LOG
21	-05	sales_app	chocolate_db	35146	99.88.010.27	[null]	LOG

How can we use this table?

- Session tracing
- Tracing events which happened at the same time
- Performance dynamics
- Precise access control

pgBadger New Level

Design Details

Log_id and Partitioning

- Why we need log_id
- Partitioning
- Problems
- Reload
- Further partitioning
- Building log_id

```
log_id bigserial  
partition by range(log_sample)  
what if we miss one log file?!  
we do not want duplicates!  
partition by list  
(log_timestamp)  
(epoch::bigint)*1000000000+  
(row_number() over (
```


Indexing

- Obvious indexes:
 - PK: `log_id`, `log_sample`, `log_timestamp`
- Obvious non-indexed fields
 - `client`, `username`
- Pattern search
 - `(substr(lower(query,1,1000) text_pattern_ops)`
- Generating search functions for users

Security and Monitoring

- Security model:
 - read-only access, one schema per customer
- `partition_creation` table
- `processed_logfiles` table

Example



pgBadger New Level

Tracing Application Session

Traditional pgBadger Report

Most frequent queries (N)

Rank	Times executed	Total duration	Min duration	Max duration	Avg duration	Query
1	904,927 Details	1s724ms	0ms	0ms	0ms	<code>BEGIN;</code> Examples User(s) involved App(s) involved
2	904,504 Details	4s192ms	0ms	8ms	0ms	<code>COMMIT;</code> Examples User(s) involved App(s) involved
3	199,238 Details	8s501ms	0ms	0ms	0ms	<code>SELECT id, address_line1, city, zipcode, plant_type_id, is_active FROM manufactures WHERE name = ?;</code> Examples User(s) involved App(s) involved
4	126,310 Details	2s93ms	0ms	0ms	0ms	<code>SELECT active, category, flavor, campaign, campaign_start, campaign_end, promotion_code, excluded_items, customer_items, customr_participation FROM settings, customer_settings, campaign_settings;</code> Examples User(s) involved App(s) involved
5	88,520 Details	8s690ms	0ms	0ms	0ms	<code>SELECT customer_id, salespoint_id FROM customer_salespoints;</code> Examples User(s) involved App(s) involved
6	81,874 Details	2s471ms	0ms	0ms	0ms	<code>SELECT product_id, product_type, promotion_active FROM product_promotions;</code> Examples User(s) involved App(s) involved
7	81,874 Details	1s15ms	0ms	0ms	0ms	<code>SELECT id FROM salespoinS WHERE is_current = TRUE;</code> Examples User(s) involved App(s) involved

Where these BEGIN comes from?!

```
select pid, count(*)  
  from sweets_logs.chocolate_log  
 where log_sample_time='2024-08-08'  
       and lower (substr(query,1,1000)) like 'begin%'  
 group by 1 order by 2 desc
```


Query Query History





```



1  select pid, count(*)
2     from sweets_logs.chocolate_log
3     where log_sample_time='2024-08-08'
4           and lower (substr(query,1,1000)) like '%begin%'
5     group by 1 order by 2 desc
6

```

Data Output Messages Notifications

	pid integer	count bigint
1	336248	5363
2	415891	5357
3	396066	5351
4	199605	5345
5	180081	5345
6	234410	5345
7	140527	5327
8	435563	5327
9	466473	5321
10	273262	5321
11	160701	5315
12	54878	5297

```
select
  log_id,
  log_time,
  pid,
  duration,
  query
from sweets_logs.chocolate_log
where pid =336248
order by log_id
```


log_id bigint	log_time timestamp with time zone	pid integer	duration numeric	query text
1723094386002008077	2024-08-08 07:12:07-05	336248	0.006	BEGIN
1723094386002008078	2024-08-08 07:12:07-05	336248	0.132	SELECT id, address_line1, city, zipcode, plant_type_id, is_active FROM manufactures WHERE
1723094386002008079	2024-08-08 07:12:07-05	336248	0.005	COMMIT
1723094386002008080	2024-08-08 07:12:07-05	336248	0.001	BEGIN
1723094386002008081	2024-08-08 07:12:07-05	336248	0.041	SELECT active, category, flavor, campaign, campaign_start, campaign_end, promotion_code
1723094386002008082	2024-08-08 07:12:07-05	336248	0.005	COMMIT
1723094386002008083	2024-08-08 07:12:07-05	336248	0.001	BEGIN
1723094386002008084	2024-08-08 07:12:07-05	336248	0.131	SELECT salespoint_id, user_id, group FROM users
1723094386002008085	2024-08-08 07:12:07-05	336248	0.005	COMMIT
1723094386002008086	2024-08-08 07:12:07-05	336248	0.002	BEGIN
1723094386002008087	2024-08-08 07:12:07-05	336248	0.108	SELECT customer_id, salespoint_id FROM customer_salespoints
1723094386002008088	2024-08-08 07:12:07-05	336248	0.004	COMMIT
1723094386002008089	2024-08-08 07:12:07-05	336248	0.001	BEGIN
1723094386002008090	2024-08-08 07:12:07-05	336248	0.029	SELECT product_id, product_type, promotion_active FROM product_promotions
1723094386002008091	2024-08-08 07:12:07-05	336248	0.005	COMMIT
1723094386002008092	2024-08-08 07:12:07-05	336248	0.001	BEGIN
1723094386002008093	2024-08-08 07:12:07-05	336248	0.012	select id from salespoints where iis_current = true
1723094386002008094	2024-08-08 07:12:07-05	336248	0.004	COMMIT
1723094386002008228	2024-08-08 07:12:09-05	336248	0.003	BEGIN
1723094386002008229	2024-08-08 07:12:09-05	336248	0.085	SELECT id, address_line1, city, zipcode, plant_type_id, is_active FROM manufactures WHERE
1723094386002008230	2024-08-08 07:12:09-05	336248	0.004	COMMIT

pgBadger New Level

“Sometimes, it’s slow!”

Query Query History Scratch Pad X

```

1 select * from example_logs.icecream_log_08_14_2024
2 where query = 'CREATE INDEX IF NOT EXISTS invoices_created_at_idx ON sales_data
3

```

Data Output Messages Notifications

	log_id [PK] bigint	log_time timestamp with time zone	user_name text	database_name text	pid integer	client text	sessionid text	loglevel text
1	1723612800000000289	2024-08-14 00:23:10-05	icecream_admin	icecream_db	3507585	icecream_sales_app	[null]	LOG
2	1723615800000000683	2024-08-14 01:18:07-05	icecream_admin	icecream_db	3509870	icecream_sales_app	[null]	LOG
3	1723619400000000670	2024-08-14 02:17:00-05	icecream_admin	icecream_db	3511822	icecream_sales_app	[null]	LOG
4	1723623000000000671	2024-08-14 03:17:34-05	icecream_admin	icecream_db	3513763	icecream_sales_app	[null]	LOG
5	1723626600000000679	2024-08-14 04:18:04-05	icecream_admin	icecream_db	3515743	icecream_sales_app	[null]	LOG
6	1723630800000000001	2024-08-14 05:20:03-05	icecream_admin	icecream_db	3517839	icecream_sales_app	[null]	LOG

```

1 select * from example_logs.icecream_log_08_17_2024
2 where query = 'CREATE INDEX IF NOT EXISTS invoices_created_at_idx ON sales_data
3

```

Data Output Messages Notifications

	log_id [PK] bigint	log_time timestamp with time zone	user_name text	database_name text	pid integer	client text	sessionid text	loglevel text
1	1723872000000000144	2024-08-17 00:20:57-05	icecream_admin	icecream_db	3718939	icecream_sales_app	[null]	LOG
2	1723875000000000690	2024-08-17 01:17:44-05	icecream_admin	icecream_db	3721279	icecream_sales_app	[null]	LOG
3	1723879800000000603	2024-08-17 02:38:25-05	icecream_admin	icecream_db	3723239	icecream_sales_app	[null]	LOG
4	1723885800000000681	2024-08-17 04:17:29-05	icecream_admin	icecream_db	3727143	icecream_sales_app	[null]	LOG
5	1723889400000000674	2024-08-17 05:17:55-05	icecream_admin	icecream_db	3729200	icecream_sales_app	[null]	LOG

```

1  select
2  log_id, log_time, duration, query
3  from example_logs.icecream_log
4  where
5  pid=3517839
6  order by 1
7

```

3723239

Data Output Messages Notifications

	log_id bigint	log_time timestamp with time zone	duration numeric	query text
13	1723629600000000592	2024-08-14 05:07:06-05	0.220	\n SELECT inhrelid::regclass::text AS partitions\n FROM pg_catalog.pg_inherits\r
14	1723629600000000593	2024-08-14 05:07:06-05	0.210	\n SELECT inhrelid::regclass::text AS partitions\n FROM pg_catalog.pg_inherits\r
15	1723629600000000594	2024-08-14 05:07:06-05	0.225	\n SELECT inhrelid::regclass::text AS partitions\n FROM pg_catalog.pg_inherits\r
16	1723629600000000595	2024-08-14 05:07:06-05	0.191	\n SELECT inhrelid::regclass::text AS partitions\n FROM pg_catalog.pg_inherits\r
17	1723629600000000596	2024-08-14 05:07:06-05	0.238	\n SELECT inhrelid::regclass::text AS partitions\n FROM pg_catalog.pg_inherits\r
18	1723629600000000597	2024-08-14 05:07:06-05	0.052	COMMIT
19	1723629600000000598	2024-08-14 05:07:06-05	0.058	BEGIN READ WRITE
20	1723629600000000599	2024-08-14 05:07:06-05	58.148	DROP INDEX IF EXISTS sales_data.orders_created_at_idx
21	1723629600000000600	2024-08-14 05:07:07-05	9.050	ALTER TABLE sales_data.orders DROP CONSTRAINT IF EXISTS orders_file_id_fkey;
22	1723629600000000601	2024-08-14 05:07:07-05	8.559	COMMIT
23	1723629600000000602	2024-08-14 05:07:15-05	0.059	BEGIN READ WRITE
24	1723629600000000603	2024-08-14 05:07:15-05	31.345	SELECT COALESCE(is_resolved, true) FROM files.processed_file LEFT OUTER JOIN files.error
25	1723629600000000605	2024-08-14 05:07:22-05	0.461	SELECT COALESCE(is_resolved, true) FROM files.processed_file LEFT OUTER JOIN files.error
26	1723629600000000606	2024-08-14 05:07:22-05	5.273	DROP INDEX IF EXISTS sales_data.invoices_created_at_idx
27	1723629600000000607	2024-08-14 05:07:22-05	0.740	ALTER TABLE sales_data.invoices DROP CONSTRAINT IF EXISTS invoices_file_id_fkey
28	1723629600000000608	2024-08-14 05:07:22-05	28.701	COMMIT
29	1723629600000000754	2024-08-14 05:08:40-05	0.102	BEGIN READ WRITE

```

1  select
2  log_id, log_time, duration, query
3  from example_logs.icecream_log
4  where pid=3723239
5  order by 1
6

```

3723239

Data Output
Messages
Notifications

+ 📄 🗑️ 📄 📄 📄 📄 📄 SQL

		log_time <small>timestamp with time zone</small>	duration <small>numeric</small>	query <small>text</small>
10	00000588	2024-08-17 02:07:06-05	0.125	\n SELECT inhrelid::regclass::text AS partitions\n FROM pg_catalog.pg_inherits\n WHER
11	00000589	2024-08-17 02:07:06-05	0.077	\n SELECT inhrelid::regclass::text AS partitions\n FROM pg_catalog.pg_inherits\n WHER
12	00000590	2024-08-17 02:07:06-05	0.090	\n SELECT inhrelid::regclass::text AS partitions\n FROM pg_catalog.pg_inherits\n WHER
13	00000591	2024-08-17 02:07:06-05	0.087	\n SELECT inhrelid::regclass::text AS partitions\n FROM pg_catalog.pg_inherits\n WHER
14	00000592	2024-08-17 02:07:06-05	0.076	\n SELECT inhrelid::regclass::text AS partitions\n FROM pg_catalog.pg_inherits\n WHER
15	00000593	2024-08-17 02:07:06-05	0.088	\n SELECT inhrelid::regclass::text AS partitions\n FROM pg_catalog.pg_inherits\n WHER
16	00000594	2024-08-17 02:07:06-05	0.051	\n SELECT inhrelid::regclass::text AS partitions\n FROM pg_catalog.pg_inherits\n WHER
17	00000595	2024-08-17 02:07:06-05	0.077	\n SELECT inhrelid::regclass::text AS partitions\n FROM pg_catalog.pg_inherits\n WHER
18	00000596	2024-08-17 02:07:06-05	0.051	COMMIT
19	00000597	2024-08-17 02:07:06-05	0.015	BEGIN READ WRITE
20	00000598	2024-08-17 02:07:06-05	245.424	DROP INDEX IF EXISTS sales_data.orders_created_at_idx
21	00000599	2024-08-17 02:07:06-05	18.486	ALTER TABLE sales_data.orders DROP CONSTRAINT IF EXISTS orders_file_id_fkey
22	00000600	2024-08-17 02:07:06-05	92.819	COMMIT
23	00000601	2024-08-17 02:07:14-05	0.058	BEGIN READ WRITE
24	00000602	2024-08-17 02:07:14-05	167.905	SELECT COALESCE(is_resolved, true) FROM files.processed_file LEFT OUTER JOIN files.error USING (file_ic
25	00000604	2024-08-17 02:07:19-05	33.882	SELECT COALESCE(is_resolved, true) FROM files.processed_file LEFT OUTER JOIN files.error USING (file_ic
26	00000605	2024-08-17 02:07:19-05	93.053	DROP INDEX IF EXISTS sales_data.invoices_created_at_idx
27	00000606	2024-08-17 02:07:19-05	1.179	ALTER TABLE sales_data.invoices DROP CONSTRAINT IF EXISTS invoices_file_id_fkey
28	00000607	2024-08-17 02:07:19-05	79.466	COMMIT
29	00000634	2024-08-17 02:08:24-05	0.086	BEGIN READ WRITE
30	00000635	2024-08-17 02:08:24-05	21.626	SELECT COALESCE(is_resolved, true) FROM files.processed_file LEFT OUTER JOIN files.error USING (file_ic


```

1 select
2 log_id, log_time, pid, duration, query
3 from example_logs.icecream_log
4 where (query like '%processed_file%'
5 or query like '%sales_data.orders%'
6 or query like '%sale_data.invoices%')
7 and log_time between '2024-08-17 02:07:00 AM' and '2024-08-
8 order by 1
  
```

Data Output Messages Notifications

	lock	log_time	lock	pid	lock	duration	lock	query
		timestamp with time zone		integer		numeric		text
1		2024-08-17 02:07:06-05		3723239		245.424		DROP INDEX IF EXISTS sales_data.orders_created_at_idx
2		2024-08-17 02:07:06-05		3723239		18.486		ALTER TABLE sales_data.orders DROP CONSTRAINT IF EXIST
3		2024-08-17 02:07:14-05		3723239		167.905		SELECT COALESCE(is_resolved, true) FROM files.processed_f
4		2024-08-17 02:07:19-05		3723239		33.882		SELECT COALESCE(is_resolved, true) FROM files.processed_f
5		2024-08-17 02:08:24-05		3723241		21.626		SELECT COALESCE(is_resolved, true) FROM files.processed_f
6		2024-08-17 02:09:33-05		3723241		14.605		SELECT COALESCE(is_resolved, true) FROM files.processed_f
7		2024-08-17 02:09:35-05		3723239		26.896		DROP INDEX IF EXISTS sales_data.orders_created_at_idx;\n
8		2024-08-17 02:10:07-05		3723239		0.381		SELECT COALESCE(is_resolved, true) FROM files.processed_f
9		2024-08-17 02:10:43-05		3723239		0.641		SELECT COALESCE(is_resolved, true) FROM files.processed_f
10		2024-08-17 02:10:50-05		3723242		0.601		INSERT INTO files.processed_file (file_name, created_at, file_
11		2024-08-17 02:10:57-05		3723242		0.395		SELECT COALESCE(is_resolved, true) FROM files.processed_f
12		2024-08-17 02:11:04-05		3723239		0.792		SELECT COALESCE(is_resolved, true) FROM files.processed_f

Automation



Steps to set up logs loading

1. Create logging: create a new schema, RO user, monitoring tables, user functions
2. Create logging instance: create a new table for a specified database
3. Create log partitions: create a partition for a specified date
4. Start transferring log files to pgBadger server

```
call logs_meta.load_log_file(  
    'sweets',  
    'chocolate',  
    '2024-08-08',  
    'chocolate_1723094386')
```

We have over 200 production instances and everybody wants logging!

pgBadger New Level

Ultimate Automation Using Python

- pgbadger server: host, port, dbname
- client: host, port, dbname
- new: yes/no
- start time (last X minutes)
- run for specified time interval/run until stop
- New schema is created if not found
- New log table is created if not found
- Day partitions created as needed
- If the same log file is loaded, partition is dropped and recreated

By the Numbers



Disk usage and processing

- Log size: 2.2GB
- Logging time: 11.5 hours
- # rows: 3.3 M
- Total table size: 1 GB
- Load: real-time

Future work

- More standardized reports
- Unit tests
- Archiving strategy
- Wait events?!



Q&A

Hettie Dombrovskaya
Database Architect DRW

hdombrovska@drwholdings.com

www.drw.com