Scaling the Wall of Text

Best Practices for Logging in PostgreSQL

SCaLE 18x | March 5, 2020 | Richard Yen
About Me

• Support Engineer at EnterpriseDB
• Previously a DBA and Web Developer
• Been using PostgreSQL since v. 7.4
Setting Up a New Postgres Instance
Setting Up a New Postgres Instance

- `listen_addresses`
Setting Up a New Postgres Instance

- listen_addresses
- port
Setting Up a New Postgres Instance

- `listen_addresses`
- `port`
- `shared_buffers`
Setting Up a New Postgres Instance

- `listen_addresses`
- `port`
- `shared_buffers`
- `work_mem`
Setting Up a New Postgres Instance

- `listen_addresses`
- `port`
- `shared_buffers`
- `work_mem`
- `autovacuum (costs)`
Setting Up a New Postgres Instance

• listen_addresses
• port
• shared_buffers
• work_mem
• autovacuum (costs)
• random_page_cost
HALT!

What about logging?
What are the consequences?
What are the consequences?

• Slows down diagnosis and recovery when there's an outage or performance problem
What are the consequences?

- Slows down diagnosis and recovery when there's an outage or performance problem
- Reflects poorly when your boss asks you what happened last night when the database went down
What are the consequences?

- Slows down diagnosis and recovery when there's an outage or performance problem
- Reflects poorly when your boss asks you what happened last night when the database went down
- Prevents you from planning for the future
What the log can tell you
What the log can tell you

• When a query started, finished, and how long it took
What the log can tell you

• When a query started, finished, and how long it took

• Indicators of data corruption
What the log can tell you

• When a query started, finished, and how long it took

• Indicators of data corruption

• Any errors with a query, or with a connection attempt
What the log can tell you

• When a query started, finished, and how long it took
• Indicators of data corruption
• Any errors with a query, or with a connection attempt
• Statistics about queries and maintenance activity
What the log can tell you

• When a query started, finished, and how long it took
• Indicators of data corruption
• Any errors with a query, or with a connection attempt
• Statistics about queries and maintenance activity
• Information about temp files
What the log can tell you

• When a query started, finished, and how long it took
• Indicators of data corruption
• Any errors with a query, or with a connection attempt
• Statistics about queries and maintenance activity
• Information about temp files
• Who connected to the database
What the log can tell you

• When a query started, finished, and how long it took
• Indicators of data corruption
• Any errors with a query, or with a connection attempt
• Statistics about queries and maintenance activity
• Information about temp files
• Who connected to the database
• Where the connection came from
What the log can tell you

• When a query started, finished, and how long it took
• Indicators of data corruption
• Any errors with a query, or with a connection attempt
• Statistics about queries and maintenance activity
• Information about temp files
• Who connected to the database
• Where the connection came from
• What queries ran before a transaction was rolled back
What the log can tell you

• When a query started, finished, and how long it took
• Indicators of data corruption
• Any errors with a query, or with a connection attempt
• Statistics about queries and maintenance activity
• Information about temp files
• Who connected to the database
• Where the connection came from
• What queries ran before a transaction was rolled back
• And more!
More log means...
More log means...

- Faster diagnosis
More log means...

- Faster diagnosis
- Faster time to recovery
More log means...

- Faster diagnosis
- Faster time to recovery
- More happy users
More log means…

- Faster diagnosis
- Faster time to recovery
- More happy users
- More sleep for DBAs
More log means…

- Faster diagnosis
- Faster time to recovery
- More happy users
- More sleep for DBAs
- Better auditing
More log means...

- Faster diagnosis
- Faster time to recovery
- More happy users
- More sleep for DBAs
- Better auditing
- Better capacity planning
Log = 😱💰.hero️
PostgreSQL Logging Parameters

- `log_destination`
- `log_directory`
- `log_filename`
- `log_file_mode`
- `log_line_prefix`
- `log_min_duration_statement`
- `log_statement`
- `log_connections`
- `log_disconnections`
- `log_rotation_age`
- `log_rotation_size`
- `log_autovacuum_min_duration`
- `log_checkpoints`
- `log_temp_files`
- `log_truncate_on_rotation`
- `log_min_messages`
- `log_min_error_statement`
- `log_error_verbosity`
- `log_hostname`
- `log_lock_waits`
- `log_replication_commands`
- `log_timezone`
- `log_parser_stats`
- `log_planner_stats`
- `log_executor_stats`
- `log_statement_stats`
log_line_prefix
your starting point
• printf-style string at the beginning of each log line

• As of v.10, defaults to %m [%p] (timestamp + backend PID)

• Before v.10, default is blank

• The more you log, the more chances you can catch a culprit
log_line_prefix

- printf-style string at the beginning of each log line
- As of v.10, defaults to `%m [%p]` (timestamp + backend PID)
- Before v.10, default is blank
- The more you log, the more chances you can catch a culprit

<table>
<thead>
<tr>
<th>Escape</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>%a</td>
<td>Application name</td>
</tr>
<tr>
<td>%u</td>
<td>User name</td>
</tr>
<tr>
<td>%d</td>
<td>Database name</td>
</tr>
<tr>
<td>%r</td>
<td>Remote host name or IP address, and remote port</td>
</tr>
<tr>
<td>%h</td>
<td>Remote host name or IP address</td>
</tr>
<tr>
<td>%p</td>
<td>Process ID</td>
</tr>
<tr>
<td>%t</td>
<td>Time stamp without milliseconds</td>
</tr>
<tr>
<td>%m</td>
<td>Time stamp with milliseconds</td>
</tr>
<tr>
<td>%n</td>
<td>Time stamp with milliseconds (as a Unix epoch)</td>
</tr>
<tr>
<td>%i</td>
<td>Command tag: type of session's current command</td>
</tr>
<tr>
<td>%e</td>
<td>SQLSTATE error code</td>
</tr>
<tr>
<td>%c</td>
<td>Session ID: see below</td>
</tr>
<tr>
<td>%l</td>
<td>Number of the log line for each session or process, starting at 1</td>
</tr>
<tr>
<td>%s</td>
<td>Process start time stamp</td>
</tr>
<tr>
<td>%v</td>
<td>Virtual transaction ID (backendID/localXID)</td>
</tr>
<tr>
<td>%x</td>
<td>Transaction ID (0 if none is assigned)</td>
</tr>
<tr>
<td>%q</td>
<td>Produces no output, but tells non-session processes to stop at this point in the string; ignored by session processes</td>
</tr>
<tr>
<td>%$</td>
<td>Literal %</td>
</tr>
</tbody>
</table>
In a sample of 130 confs

- 5% of users don’t change `log_line_prefix`
- 38% only log timestamps (and nothing else)
- Why?
The Wall of Text (sorted)

2020-02-29 00:11:15.330 UTC [2411] LOG: statement: BEGIN;
2020-02-29 00:11:15.332 UTC [2411] LOG: statement: END;
2020-02-29 00:11:15.332 UTC [2411] LOG: statement: INSERT INTO pgbench_history (tid, bid, aid, delta, mtime) VALUES (8, 1, 19334, -4410, CURRENT_TIMESTAMP);
2020-02-29 00:11:15.331 UTC [2411] LOG: statement: SELECT abalance FROM pgbench_accounts WHERE aid = 19334;
2020-02-29 00:11:15.332 UTC [2411] LOG: statement: UPDATE pgbench_accounts SET abalance = abalance + -4410 WHERE aid = 19334;
2020-02-29 00:11:15.331 UTC [2411] LOG: statement: UPDATE pgbench_branches SET bbalance = bbalance + -4410 WHERE bid = 1;
2020-02-29 00:11:15.331 UTC [2411] LOG: statement: UPDATE pgbench_tellers SET tbalance = tbalance + -4410 WHERE tid = 8;
2020-02-29 00:11:15.332 UTC [2412] LOG: statement: BEGIN;
2020-02-29 00:11:15.334 UTC [2412] LOG: statement: END;
2020-02-29 00:11:15.334 UTC [2412] LOG: statement: INSERT INTO pgbench_history (tid, bid, aid, delta, mtime) VALUES (8, 1, 57806, -1442, CURRENT_TIMESTAMP);
2020-02-29 00:11:15.333 UTC [2412] LOG: statement: SELECT abalance FROM pgbench_accounts WHERE aid = 57806;
2020-02-29 00:11:15.332 UTC [2412] LOG: statement: UPDATE pgbench_accounts SET abalance = abalance + -1442 WHERE aid = 57806;
2020-02-29 00:11:15.333 UTC [2412] LOG: statement: UPDATE pgbench_branches SET bbalance = bbalance + -1442 WHERE bid = 1;
2020-02-29 00:11:15.333 UTC [2412] LOG: statement: UPDATE pgbench_tellers SET tbalance = tbalance + -1442 WHERE tid = 8;
2020-02-29 00:11:15.332 UTC [2413] LOG: statement: BEGIN;
2020-02-29 00:11:15.337 UTC [2413] LOG: statement: END;
2020-02-29 00:11:15.334 UTC [2413] LOG: statement: INSERT INTO pgbench_history (tid, bid, aid, delta, mtime) VALUES (8, 1, 57894, 4310, CURRENT_TIMESTAMP);
2020-02-29 00:11:15.332 UTC [2413] LOG: statement: SELECT abalance FROM pgbench_accounts WHERE aid = 57894;
2020-02-29 00:11:15.332 UTC [2413] LOG: statement: UPDATE pgbench_accounts SET abalance = abalance + 4310 WHERE aid = 57894;
2020-02-29 00:11:15.336 UTC [2413] LOG: statement: UPDATE pgbench_branches SET bbalance = bbalance + 4310 WHERE bid = 1;
2020-02-29 00:11:15.335 UTC [2413] LOG: statement: UPDATE pgbench_tellers SET tbalance = tbalance + 4310 WHERE tid = 8;
2020-02-29 00:11:15.332 UTC [2414] LOG: statement: BEGIN;
2020-02-29 00:11:15.337 UTC [2414] LOG: statement: END;
2020-02-29 00:11:15.334 UTC [2414] LOG: statement: INSERT INTO pgbench_history (tid, bid, aid, delta, mtime) VALUES (9, 1, 75471, 2644, CURRENT_TIMESTAMP);
2020-02-29 00:11:15.332 UTC [2414] LOG: statement: SELECT abalance FROM pgbench_accounts WHERE aid = 75471;
2020-02-29 00:11:15.332 UTC [2414] LOG: statement: UPDATE pgbench_accounts SET abalance = abalance + 2644 WHERE aid = 75471;
2020-02-29 00:11:15.335 UTC [2414] LOG: statement: UPDATE pgbench_branches SET bbalance = bbalance + 2644 WHERE bid = 1;
2020-02-29 00:11:15.334 UTC [2414] LOG: statement: UPDATE pgbench_tellers SET tbalance = tbalance + 2644 WHERE tid = 9;
2020-02-29 00:11:15.332 UTC [2415] LOG: statement: BEGIN;
2020-02-29 00:11:15.337 UTC [2415] LOG: statement: END;
2020-02-29 00:11:15.334 UTC [2415] LOG: statement: UPDATE pgbench_accounts SET abalance = abalance + 4798 WHERE aid = 30074;
2020-02-29 00:11:15.345 UTC [2415] STATEMENT: UPDATE pgbench_accounts SET abalance = abalance + 4798 WHERE aid = 30074;
2020-02-29 00:11:15.345 UTC [2416] ERROR: relation "pgbench_accounts" does not exist at character 8
2020-02-29 00:11:15.332 UTC [2416] LOG: statement: BEGIN;
2020-02-29 00:11:15.337 UTC [2416] LOG: statement: UPDATE pgbench_accounts SET abalance = abalance + 524 WHERE aid = 23231;
2020-02-29 00:11:15.345 UTC [2416] STATEMENT: UPDATE pgbench_accounts SET abalance = abalance + 524 WHERE aid = 23231;
2020-02-29 00:11:15.345 UTC [2417] ERROR: relation "pgbench_accounts" does not exist at character 8
2020-02-29 00:11:15.332 UTC [2417] LOG: statement: BEGIN;
2020-02-29 00:11:15.337 UTC [2417] LOG: statement: UPDATE pgbench_accounts SET abalance = abalance + -4793 WHERE aid = 23424;
2020-02-29 00:11:15.345 UTC [2417] STATEMENT: UPDATE pgbench_accounts SET abalance = abalance + -4793 WHERE aid = 23424;
2020-02-29 00:11:15.345 UTC [2418] ERROR: relation "pgbench_accounts" does not exist at character 8
2020-02-29 00:11:15.332 UTC [2418] LOG: statement: BEGIN;
2020-02-29 00:11:15.337 UTC [2418] LOG: statement: UPDATE pgbench_accounts SET abalance = abalance + -118 WHERE aid = 37745;
2020-02-29 00:11:15.352 UTC [2418] STATEMENT: UPDATE pgbench_accounts SET abalance = abalance + -118 WHERE aid = 37745;
2020-02-29 00:11:21.589 UTC [2419] LOG: automatic analyze of table "postgres.public.pgbench_branches" system usage: CPU: user: 0.00 s, system: 0.00 s, elapsed: 0.02 s
2020-02-29 00:11:21.603 UTC [2419] LOG: automatic analyze of table "postgres.public.pgbench_history" system usage: CPU: user: 0.00 s, system: 0.00 s, elapsed: 0.00 s
2020-02-29 00:11:21.616 UTC [2419] LOG: automatic analyze of table "postgres.public.pgbench_tellers" system usage: CPU: user: 0.00 s, system: 0.00 s, elapsed: 0.00 s
2020-02-29 00:11:21.666 UTC [2419] LOG: automatic vacuum of table "postgres.public.pgbench_branches": index scans: 0
2020-02-29 00:11:15.337 UTC [658] LOG: statement: drop table pgbench_accounts;
My Personal Favorites
My Personal Favorites

• %a - Application Name - Allows quick reference and filtering
My Personal Favorites

• \%a - Application Name - Allows quick reference and filtering

• \%u - User Name - Allows filter by user name
My Personal Favorites

• %a - Application Name - Allows quick reference and filtering

• %u - User Name - Allows filter by user name

• %d - Database Name - Allows filter by database name
My Personal Favorites

- %a - Application Name - Allows quick reference and filtering
- %u - User Name - Allows filter by user name
- %d - Database Name - Allows filter by database name
- %r - Remote Host IP/Name (w/ port) - Helps identify suspicious activity from a host
My Personal Favorites

- %a - Application Name - Allows quick reference and filtering
- %u - User Name - Allows filter by user name
- %d - Database Name - Allows filter by database name
- %r - Remote Host IP/Name (w/ port) - Helps identify suspicious activity from a host
- %p - Process ID - Helps identify specific problematic sessions
My Personal Favorites

- %a - Application Name - Allows quick reference and filtering
- %u - User Name - Allows filter by user name
- %d - Database Name - Allows filter by database name
- %r - Remote Host IP/Name (w/ port) - Helps identify suspicious activity from a host
- %p - Process ID - Helps identify specific problematic sessions
- %l - Session/Process Log Line - Helps identify what a session has done
My Personal Favorites

• %a - Application Name - Allows quick reference and filtering
• %u - User Name - Allows filter by user name
• %d - Database Name - Allows filter by database name
• %r - Remote Host IP/Name (w/ port) - Helps identify suspicious activity from a host
• %p - Process ID - Helps identify specific problematic sessions
• %l - Session/Process Log Line - Helps identify what a session has done
• %v/%x - Transaction IDs - Helps identify what queries a transaction ran
Other Useful Parameters

- `log_min_duration_statement` -- helpful in identifying slow queries
- `log_statement` -- good for auditing purposes
- `log_[dis]connections` -- good for auditing purposes
- `log_rotation_age/log_rotation_size` -- good for organization
- `log_autovacuum_min_duration` -- gives insight into autovacuum behavior
- `log_checkpoints` -- know what queries happened between checkpoints
- `log_temp_files` -- helps identify work_mem shortages, I/O spikes
The Wall of Text (with log_connections)

LOG: connection received: host=127.0.0.1 port=45214
LOG: connection authorized: user=postgres database=postgres
LOG: statement: END;
LOG: connection authorized: user=postgres database=postgres
LOG: statement: BEGIN;
LOG: statement: UPDATE pgbench_accounts SET abalance = abalance + 4305 WHERE aid = 47364;
LOG: statement: BEGIN;
LOG: statement: SELECT abalance FROM pgbench_accounts WHERE aid = 47364;
LOG: connection authorized: user=postgres database=postgres
LOG: statement: UPDATE pgbench_accounts SET abalance = abalance + -2684 WHERE aid = 53855;
LOG: statement: UPDATE pgbench_tellers SET tbalance = tbalance + 4305 WHERE tid = 10;
LOG: statement: UPDATE pgbench_branches SET bbalance = bbalance + 4305 WHERE bid = 1;
LOG: statement: INSERT INTO pgbench_history (tid, bid, aid, delta, mtime) VALUES (10, 1, 47364, 4305, CURRENT_TIMESTAMP);
LOG: statement: END;
LOG: statement: SELECT abalance FROM pgbench_accounts WHERE aid = 53855;
LOG: connection received: host=127.0.0.1 port=45216
LOG: statement: drop table pgbench_accounts
LOG: statement: UPDATE pgbench_tellers SET tbalance = tbalance + -2684 WHERE tid = 10;
LOG: statement: UPDATE pgbench_branches SET bbalance = bbalance + -2684 WHERE bid = 1;
LOG: statement: INSERT INTO pgbench_history (tid, bid, aid, delta, mtime) VALUES (10, 1, 53855, -2684, CURRENT_TIMESTAMP);
LOG: statement: END;
LOG: connection authorized: user=postgres database=postgres
LOG: connection received: host=127.0.0.1 port=45218
LOG: connection authorized: user=postgres database=postgres
LOG: statement: BEGIN;
LOG: statement: UPDATE pgbench_accounts SET abalance = abalance + -3198 WHERE aid = 51840;
LOG: statement: BEGIN;
LOG: statement: UPDATE pgbench_accounts SET abalance = abalance + 4395 WHERE aid = 62616;
ERROR: relation "pgbench_accounts" does not exist at character 8
STATEMENT: UPDATE pgbench_accounts SET abalance = abalance + 4395 WHERE aid = 62616;
ERROR: relation "pgbench_accounts" does not exist at character 8
STATEMENT: UPDATE pgbench_accounts SET abalance = abalance + -3198 WHERE aid = 51840;
LOG: connection received: host=127.0.0.1 port=45220
LOG: connection received: host=127.0.0.1 port=45222
LOG: connection authorized: user=postgres database=postgres
LOG: connection authorized: user=postgres database=postgres
LOG: statement: BEGIN;
LOG: statement: UPDATE pgbench_accounts SET abalance = abalance + -148 WHERE aid = 98357;
ERROR: relation "pgbench_accounts" does not exist at character 8
STATEMENT: UPDATE pgbench_accounts SET abalance = abalance + -148 WHERE aid = 98357;
LOG: statement: BEGIN;
LOG: statement: UPDATE pgbench_accounts SET abalance = abalance + -3241 WHERE aid = 26657;
ERROR: relation "pgbench_accounts" does not exist at character 8
STATEMENT: UPDATE pgbench_accounts SET abalance = abalance + -3241 WHERE aid = 26657;
LOG: automatic vacuum of table "postgres.public.pgbench_branches": index scans: 0
LOG: automatic analyze of table "postgres.public.pgbench_branches" system usage: CPU: user: 0.00 s, system: 0.00 s, elapsed: 0.02 s
LOG: automatic analyze of table "postgres.public.pgbench_history" system usage: CPU: user: 0.00 s, system: 0.00 s, elapsed: 0.00 s
LOG: automatic vacuum of table "postgres.public.pgbench_tellers": index scans: 0
LOG: automatic analyze of table "postgres.public.pgbench_tellers" system usage: CPU: user: 0.00 s, system: 0.00 s, elapsed: 0.00 s
Found the culprit!
How much is too much?
How much is too much?

- Log as much, as long, as your system can handle
How much is too much?

• Log as much, as long, as your system can handle
• Most logging params can be turned on/off with a HUP/reload
How much is too much?

• Log as much, as long, as your system can handle
• Most logging params can be turned on/off with a HUP/reload
• Use a separate partition
How much is too much?

- Log as much, as long, as your system can handle
- Most logging params can be turned on/off with a HUP/reload
- Use a separate partition
- Get familiar with `grep` and `awk` (or even `sed`)
How much is too much?

• Log as much, as long, as your system can handle
• Most logging params can be turned on/off with a HUP/reload
• Use a separate partition
• Get familiar with `grep` and `awk` (or even `sed`)
• Compress your log files (using a cronjob)
How much is too much?

- Log as much, as long, as your system can handle
- Most logging params can be turned on/off with a HUP/reload
- Use a separate partition
- Get familiar with `grep` and `awk` (or even `sed`)
- Compress your log files (using a cronjob)
- If you know what to look for, you can filter out the noise
A word on verbosity

- **log_min_messages**
  - This indicates *generally* how much to log.
  - Valid values are:
    - DEBUG\{5,4,3,2,1\}
    - INFO
    - NOTICE
    - WARNING
    - ERROR
    - LOG
    - FATAL
    - PANIC
  - WARNING is the default, but you shouldn’t ever have to do more than INFO
What to look for: Logging Levels

• WARNING
  • Not necessarily urgent, but a proactive notice from PG

• ERROR
  • Failure to execute a command; often a syntax error, but can indicate worse (i.e., corruption)

• FATAL
  • Session terminated due to some error (ex., wrong password, bad auth)

• PANIC
  • Some anomaly encountered in the database engine (segfault, corruption, etc.)
  • All sessions terminated for safety
What to look for: Common Issues

- Don’t ignore DETAIL and HINT messages
- A spike in FATAL messages often correlates to misconfigured application
- A spike in ERROR messages often correlates to an application bug
- A spike in ERROR messages can indicate broken replication
- Autovacuum problems? Look for long-running idle transactions
What to look for: hints

2016-03-04 10:13:51.746 CET [15340]: [80091-1] user=postgres,db=postgres,10.0.1.1(49015) WARNING: database "postgres" must be vacuumed within 9763669 transactions
   You might also need to commit or roll back old prepared transactions.
2016-03-04 10:13:51.746 CET [15337]: [79775-1] user=postgres,db=postgres,10.0.1.1(49012) WARNING: database "postgres" must be vacuumed within 9763668 transactions
   You might also need to commit or roll back old prepared transactions.
2016-03-04 10:13:51.746 CET [15332]: [80655-1] user=postgres,db=postgres,10.0.1.1(49007) WARNING: database "postgres" must be vacuumed within 9763667 transactions
   You might also need to commit or roll back old prepared transactions.
2017-01-11 16:19:24 IST FATAL: the database system is starting up
2017-01-11 16:19:25 IST FATAL: the database system is starting up
2017-01-11 16:19:26 IST FATAL: the database system is starting up
2017-01-11 16:19:27 IST FATAL: the database system is starting up
2017-01-11 16:19:28 IST FATAL: the database system is starting up
2017-01-11 16:19:29 IST FATAL: the database system is starting up
2017-01-11 16:19:30 IST FATAL: the database system is starting up
2017-01-11 16:19:31 IST FATAL: the database system is starting up
2017-01-11 16:19:32 IST FATAL: the database system is starting up
2017-01-11 16:19:33 IST FATAL: the database system is starting up
2017-01-11 16:19:33 IST LOG: startup process (PID 15153) was terminated by signal 6: Aborted
2017-01-11 16:19:33 IST LOG: aborting startup due to startup process failure
2017-01-11 16:20:00 IST LOG: database system was shut down at 2017-01-11 15:34:43 IST
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST PANIC: could not locate a valid checkpoint record
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST FATAL: the database system is starting up
2017-01-11 16:20:00 IST FATAL: the database system is starting up
What to look for: Application Bugs

2017-01-11 00:02:18 IST WARNING: there is no transaction in progress
2017-01-11 00:02:18 IST WARNING: there is no transaction in progress
2017-01-11 00:02:18 IST WARNING: there is no transaction in progress
2017-01-11 00:02:18 IST WARNING: there is no transaction in progress
2017-01-11 00:15:07 IST WARNING: there is no transaction in progress
2017-01-11 00:16:48 IST WARNING: there is no transaction in progress
2017-01-11 00:25:11 IST WARNING: there is no transaction in progress
2017-01-11 00:45:01 IST WARNING: there is no transaction in progress
2017-01-11 00:48:33 IST WARNING: there is no transaction in progress
2017-01-11 00:56:23 IST WARNING: there is no transaction in progress
2017-01-11 01:01:04 IST LOG: unexpected EOF on client connection
2017-01-11 01:01:04 IST LOG: unexpected EOF on client connection
2017-01-11 01:01:04 IST LOG: unexpected EOF on client connection
2017-01-11 01:01:04 IST LOG: unexpected EOF on client connection
2017-01-11 01:01:04 IST LOG: unexpected EOF on client connection
2017-01-11 01:01:04 IST LOG: unexpected EOF on client connection
2017-01-11 01:01:04 IST LOG: unexpected EOF on client connection
2017-01-11 01:01:04 IST LOG: unexpected EOF on client connection
2017-01-11 01:01:04 IST LOG: unexpected EOF on client connection
2017-01-11 01:01:04 IST LOG: unexpected EOF on client connection
2017-01-11 01:05:14 IST WARNING: there is no transaction in progress
2017-01-11 01:15:10 IST WARNING: there is no transaction in progress
2017-01-11 01:25:29 IST WARNING: there is no transaction in progress
2017-01-11 01:45:01 IST WARNING: there is no transaction in progress
What to look for: Broken Replication

2019-02-19 19:35:57.222 UTC > ERROR: requested WAL segment 00000001000000000000000000000000 has already been removed
2019-02-19 19:36:02.217 UTC > ERROR: requested WAL segment 00000001000000000000000000000000 has already been removed
2019-02-19 19:36:07.235 UTC > ERROR: requested WAL segment 00000001000000000000000000000000 has already been removed
2019-02-19 19:36:12.232 UTC > ERROR: requested WAL segment 00000001000000000000000000000000 has already been removed
2019-02-19 19:36:17.235 UTC > ERROR: requested WAL segment 00000001000000000000000000000000 has already been removed
What to look for: autovacuum

2020-02-25 00:29:29.062 UTC [66340] LOG: automatic vacuum of table "postgres.public.pgbench_accounts": index scans: 1
  pages: 0 removed, 8694 remain, 0 skipped due to pins, 0 skipped frozen
  tuples: 198 removed, 125983 remain, 99944 are dead but not yet removable, oldest xmin: 95278
  buffer usage: 25163 hits, 0 misses, 0 dirtied
  avg read rate: 0.000 MB/s, avg write rate: 0.000 MB/s
  system usage: CPU: user: 0.17 s, system: 0.01 s, elapsed: 3.00 s
What to look for: temp files

What to look for: locking & blocking

2020-02-14 09:20:24 EST [43118]: [64706-1] db=immcare,user=svc-mmpphmp,app=PostgreSQL JDBC Driver,client=10.8.37.178LOG: process 43118 still waiting for AccessExclusiveLock on tuple (0,2) of relation 17305 of database 16493 after 1000.117 ms


Process 7715 waits for AccessShareLock on relation 34670707 of database 9122478; blocked by process 6652.
Process 6652: SELECT * FROM myschema.proc_mv_refresh()
Process 7715: SELECT c.relname AS view_name, c.relkind AS view_type, c.relispopulated AS ispopulated, spc.spcname AS tablespace_name, pg_get_userbyid(c.relowner) AS view_owner, pg_get_viewdef(c.oid, true) AS definition FROM pg_catalog.pg_class c LEFT OUTER JOIN pg_tablespace spc ON spc.oid=c.reltablespace LEFT OUTER JOIN pg_namespace n ON n.oid=c.relnamespace WHERE (c.relkind = 'v' OR c.relkind = 'm') AND c.relnamespace = n.oid AND n.nspname = $1","See server log for query details.

"SQL statement ""REFRESH MATERIALIZED VIEW ENTRP_SERVS.MV_8001_AU_PRSN"
Digesting the log: pgbadger

- Analyzes your logs
- Generates statistics of your query traffic
- Helps identify problematic queries
More Knowledge, More Power

- **auto_explain**
  - Adds EXPLAIN output to your log
  - Can show the runtimes of each plan node
  - Can show caching behavior and memory usage
- **pg_stat_statements**
  - Logs statistics on each statement executed
  - Doesn’t go into your log, but in a table within Postgres
  - Can help alleviate log traffic (YMMV)
Some gotchas

- **logging_collector**
  - needs to be turned on if you want separate log files (i.e., not using syslog)
- **log_filename**
  - needs to include the relevant escapes for notation to happen
  - otherwise, it just appends (unless **log_truncate_on_rotation** is turned on)
Some gotchas

- `log_min_duration_statement` vs `log_duration`
  - `log_duration`
    - only logs the time it took for a query to run
    - does not log the query
    - Applies to all queries
  - `log_min_duration_statement`
    - logs the time it took for a query to run
    - logs the query
    - Applies to only queries exceeding the threshold
- Why `log_duration`?
  - Allows you to gather stats on high-workload environments, without the added overhead of printing the full query
Some gotchas

- `log_min_duration_statement` vs `log_statement`
  - `log_min_duration_statement`
    - logs the statement and duration AFTER the query has finished
    - does not log parameters
  - `log_statement`
    - logs the statement BEFORE the query is executed
    - logs parameters
Summary
Summary

- Be sure to set your **log_line_prefix**
Summary

• Be sure to set your `log_line_prefix`

• Turn on `log_connections` to track down the origin of your queries
Summary

• Be sure to set your log_line_prefix

• Turn on log_connections to track down the origin of your queries

• Log as much as you can, for as long as you can
Summary

• Be sure to set your `log_line_prefix`

• Turn on `log_connections` to track down the origin of your queries

• Log as much as you can, for as long as you can

• Run your logs through pgbadger for visibility and forecasting
Be sure to set your `log_line_prefix`

Turn on `log_connections` to track down the origin of your queries

Log as much as you can, for as long as you can

Run your logs through pgbadger for visibility and forecasting

Monitor for spikes in FATAL/ERROR/WARNING messages
Questions?