Progress reporting in Postgres

Álvaro Herrera
PostgreSQL developer
PgConf.Brasil, July 2019

Progress Reporting

2ndQuadrant® PostgreSQL

- Reporting of what?
- How does it look?
- How to use it?
- What commands are supported?
- How can I implement more?

Progress Reporting

2ndQuadrant® PostgreSQL

- Reporting of what?
- How does it look?
- How to use it?
- What commands are supported?
- How can I implement more?

DDL progress reporting

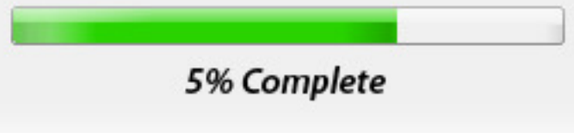
2ndQuadrant® PostgreSQL

- Many DDL commands take very long time to execute
  - VACUUM, CREATE INDEX, etc
- It's useful to have insight as to:
  - How much total work there is
  - How much work we have already done
- Allows to extrapolate

DDL progress reporting

2ndQuadrant® PostgreSQL

- Many DDL commands take very long time to execute
  - VACUUM, CREATE INDEX, etc
- It's useful to have insight as to:
  - How much total work there is
  - How much work we have already done
- Allows to extrapolate
- ... with caveats



Feature design principles

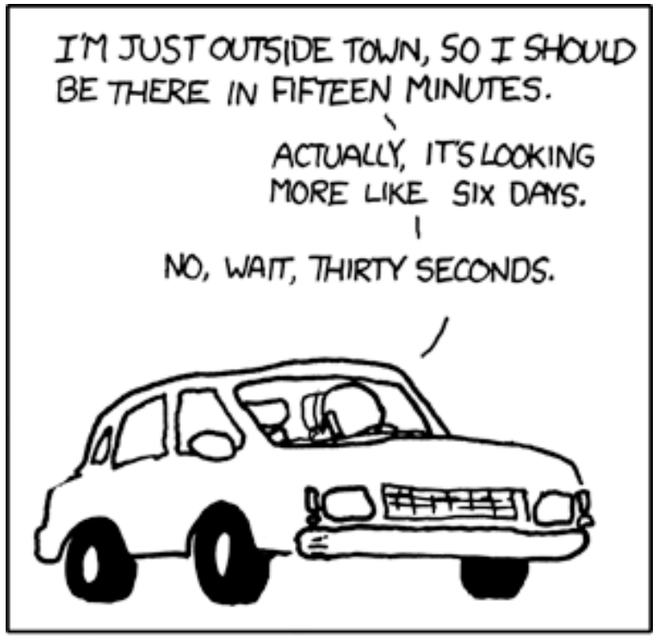
2ndQuadrant® PostgreSQL

- We want to present hard facts
- Not fiction
  - No guessing
  - No busted percentages
    - 0% – 95% in one minute ... *then a slow crawl to 99%*
    - *... 245% done*
    - *progress bars going backwards*

Feature design principles

2ndQuadrant® PostgreSQL

- We want to present hard facts
- Not fiction
  - No guessing
  - No busted percentages
    - 0% – 95% in one minute ... *then a slow crawl to 99%*
    - *... 245% done*
    - *progress bars going backwards*
- ... Preferrably, detailed and useful facts



Reporting VACUUM progress

2ndQuadrant® PostgreSQL

- PostgreSQL 10
- Add a generic command progress reporting facility.
  - `http://git.postgresql.org/pg/commitdiff/b6fb6471f6af`
  - Vinayak Pokale, Rahila Syed, Amit Langote, Robert Haas.
- Add simple VACUUM progress reporting.
  - `http://git.postgresql.org/pg/commitdiff/c16dc1aca5e0`
  - Amit Langote, Robert Haas, Vinayak Pokale, Rahila Syed.

Reporting vacuum progress (2)

2ndQuadrant® PostgreSQL

```
alvherre=# SELECT * FROM pg_stat_progress_vacuum;
```

Record 1

pid	4204
datid	12386
datname	alvherre
relid	234754
phase	scanning heap
heap_blks_total	89759
heap_blks_scanned	61181
heap_blks_vacuumed	0
index_vacuum_count	0
max_dead_tuples	291
num_dead_tuples	0

Reporting vacuum progress (3)

2ndQuadrant® PostgreSQL

```
alvherre=# SELECT now(), pid, relid::regclass as table, phase,
           heap_blks_total, heap_blks_scanned, heap_blks_vacuumed,
           index_vacuum_count, max_dead_tuples, num_dead_tuples
           FROM pg_stat_progress_vacuum WHERE datname = current_database();
```

Record 1

now	2019-08-01 11:32:15.300526-04
pid	4204
table	esquema.tabela
phase	scanning heap
heap_blks_total	134007
heap_blks_scanned	105442
heap_blks_vacuumed	0
index_vacuum_count	0
max_dead_tuples	291
num_dead_tuples	0

Reporting vacuum progress (4)

2ndQuadrant® PostgreSQL

```
alvherre=# \t
alvherre=# \pset tuples_only on
alvherre=# SELECT .. FROM pg_stat_progress_vacuum \watch 0,1
```

VACUUM: operation phases

2ndQuadrant® PostgreSQL

1. initializing
2. scanning heap
3. vacuuming indexes
4. vacuuming heap
5. cleaning up indexes
6. truncating heap
7. performing final cleanup

VACUUM: operation phases

2ndQuadrant® PostgreSQL

1. initializing
2. scanning heap
   → `dead_tuples[0 ... maintenance_work_mem]`
3. vacuuming indexes
4. vacuuming heap
5. cleaning up indexes
6. truncating heap
7. performing final cleanup

Record 1

now	...
pid	...
table	...
phase	...
heap_blks_total	134007
heap_blks_scanned	105442
heap_blks_vacuumed	0
index_vacuum_count	0
max_dead_tuples	291
num_dead_tuples	0

VACUUM: operation phases

2ndQuadrant® PostgreSQL

1. initializing
2. scanning heap
3. vacuuming indexes
4. vacuuming heap
5. cleaning up indexes
6. truncating heap
7. performing final cleanup
   → FSM update

VACUUM: operation phases

2ndQuadrant® PostgreSQL

1. initializing
2. scanning heap
3. vacuuming indexes
4. vacuuming heap
5. cleaning up indexes
6. truncating heap
   → requires access exclusive lock
   → step not done if unavailable
7. performing final cleanup
   → FSM update

Reporting CREATE INDEX / REINDEX progress

2ndQuadrant® PostgreSQL

- PostgreSQL 12
- Report progress of CREATE INDEX operations
  - `http://git.postgresql.org/pg/commitdiff/ab0dfc961b6a`
  - Álvaro Herrera
- Report progress of REINDEX operations
  - `http://git.postgresql.org/pg/commitdiff/03f9e5cba0ee`
  - Peter Eisentraut

Report of CREATE INDEX

2ndQuadrant® PostgreSQL

```
SELECT ... FROM pg_stat_progress_create_index ... \watch 1
```

pid	1209
relid	esquema.tabela
index_relid	35684
command	CREATE INDEX CONCURRENTLY
phase	building index: scanning table
lockers_total	0
lockers_done	0
current_locker_pid	0
blocks_total	44248
blocks_done	17627
tuples_total	0
tuples_done	0
partitions_total	0
partitions_done	0

Operation phases of CREATE INDEX / REINDEX

2ndQuadrant® PostgreSQL

1. initializing
2. waiting for writers before build
3. building index
4. waiting for writers before validation
5. index validation: scanning index
6. index validation: sorting tuples
7. index validation: scanning table
8. waiting for old snapshots
9. waiting for readers before marking dead
10. waiting for readers before dropping

Operation phases of CREATE INDEX / REINDEX

2ndQuadrant® PostgreSQL

1. initializing
2. waiting for writers before build
3. building index
4. waiting for writers before validation
5. index validation: scanning index
6. index validation: sorting tuples
7. index validation: scanning table
8. waiting for old snapshots
9. waiting for readers before marking dead
10. waiting for readers before dropping

phase	....
lockers_total	0
lockers_done	0
current_locker_pid	0
blocks_total	44248
blocks_done	17627
tuples_total	0
tuples_done	0
partitions_total	0
partitions_done	0

Build phases for B-Tree indexes

2ndQuadrant® PostgreSQL

1. initializing
2. scanning table
3. sorting live tuples
4. sorting dead tuples
5. loading tuples in tree

Reporting CLUSTER / VACUUM FULL progress

2ndQuadrant® PostgreSQL

- PostgreSQL 12
- Add progress reporting for CLUSTER and VACUUM FULL.
  - `http://git.postgresql.org/pg/commitdiff/6f97457e0ddd`
  - Tatsuro Yamada

Reporting cluster progress

2ndQuadrant® PostgreSQL

```
alvherre=# SELECT * FROM pg_stat_progress_cluster \watch 1
```

pid	1209
table	esquema.tabela
command	VACUUM FULL
phase	seq scanning heap
cluster_index_relid	0
heap_tuples_scanned	8064358
heap_tuples_written	8064358
heap_blks_total	44248
heap_blks_scanned	35684
index_rebuild_count	0

CLUSTER: operation phases

2ndQuadrant® PostgreSQL

1. initializing
2. seq scanning heap
3. index scanning heap
4. sorting tuples
5. writing new heap
6. swapping relation files
7. rebuilding index
8. performing final cleanup

Reporting ANALYZE progress

2ndQuadrant® PostgreSQL

- Patch submitted for PostgreSQL 13
- `https://postgr.es/m/20190621185207.GA27929@alvherre.pgsql`

Questions?

2ndQuadrant® PostgreSQL

Thanks for listening!

Appendix: Implementation

2ndQuadrant® PostgreSQL

- Set-returning function `pg_stat_get_progress_info(text)`
- Returns raw metrics
- View definitions (`pg_stat_progress_vacuum` etc) transform metrics into user-readable parameters
- PostgreSQL C code injects metrics into `pgstat` system

```
pgstat_progress_start_command(command type);
pgstat_progress_update_param(4, 158);
pgstat_progress_update_param(PROGRESS_ANALYZE_PHASE,
                             PROGRESS_ANALYZE_PHASE_SCAN_TABLE);
pgstat_progress_end_command();
```