Progress reporting in Postgres

Álvaro Herrera
PostgreSQL developer
PgConf.Brasil, July 2019
Progress Reporting

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

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

• 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

• 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

• 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

- 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*
- ... Preferably, detailed and useful facts
Reporting VACUUM progress

- 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)

```
alvherre=# SELECT * FROM pg_stat_progress_vacuum;

<table>
<thead>
<tr>
<th>pid</th>
<th>datid</th>
<th>datname</th>
<th>relid</th>
<th>phase</th>
<th>heap_blks_total</th>
<th>heap_blks_scanned</th>
<th>heap_blks_vacuumed</th>
<th>index_vacuum_count</th>
<th>max_dead_tuples</th>
<th>num_dead_tuples</th>
</tr>
</thead>
<tbody>
<tr>
<td>4204</td>
<td>12386</td>
<td>alvherre</td>
<td>234754</td>
<td>scanning heap</td>
<td>89759</td>
<td>61181</td>
<td>0</td>
<td>0</td>
<td>291</td>
<td>0</td>
</tr>
</tbody>
</table>
```
### Reporting vacuum progress (3)

```sql
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();
```

<table>
<thead>
<tr>
<th>now</th>
<th>2019-08-01 11:32:15.300526-04</th>
</tr>
</thead>
<tbody>
<tr>
<td>pid</td>
<td>4204</td>
</tr>
<tr>
<td>table</td>
<td>esquema.tabela</td>
</tr>
<tr>
<td>phase</td>
<td>scanning heap</td>
</tr>
<tr>
<td>heap_blks_total</td>
<td>134007</td>
</tr>
<tr>
<td>heap_blks_scanned</td>
<td>105442</td>
</tr>
<tr>
<td>heap_blks_vacuumed</td>
<td>0</td>
</tr>
<tr>
<td>index_vacuum_count</td>
<td>0</td>
</tr>
<tr>
<td>max_dead_tuples</td>
<td>291</td>
</tr>
<tr>
<td>num_dead_tuples</td>
<td>0</td>
</tr>
</tbody>
</table>
alvherre=# \t
alvherre=# \pset tuples_only on
alvherre=# SELECT .. FROM pg_stat_progress_vacuum \watch 0,1
VACUUM: operation phases

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

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

<table>
<thead>
<tr>
<th>now</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>pid</td>
<td>...</td>
</tr>
<tr>
<td>table</td>
<td>...</td>
</tr>
<tr>
<td>phase</td>
<td>...</td>
</tr>
<tr>
<td>heap_blks_total</td>
<td>134007</td>
</tr>
<tr>
<td>heap_blks_scanned</td>
<td>105442</td>
</tr>
<tr>
<td>heap_blks_vacuumed</td>
<td>0</td>
</tr>
<tr>
<td>index_vacuum_count</td>
<td>0</td>
</tr>
<tr>
<td>max_dead_tuples</td>
<td>291</td>
</tr>
<tr>
<td>num_dead_tuples</td>
<td>0</td>
</tr>
</tbody>
</table>
VACUUM: operation phases

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

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

- PostgreSQL 12
- Report progress of CREATE INDEX operations
  - [http://git.postgresql.org/pg/commitdiff/ab0dfc961b6a](http://git.postgresql.org/pg/commitdiff/ab0dfc961b6a)
  - Álvaro Herrera
- Report progress of REINDEX operations
  - [http://git.postgresql.org/pg/commitdiff/03f9e5cbe0ee](http://git.postgresql.org/pg/commitdiff/03f9e5cbe0ee)
  - Peter Eisentraut
### Report of CREATE INDEX

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

<table>
<thead>
<tr>
<th>pid</th>
<th>1209</th>
</tr>
</thead>
<tbody>
<tr>
<td>relid</td>
<td>esquema.tabela</td>
</tr>
<tr>
<td>index_relid</td>
<td>35684</td>
</tr>
<tr>
<td>command</td>
<td>CREATE INDEX CONCURRENTLY</td>
</tr>
<tr>
<td>phase</td>
<td>building index: scanning table</td>
</tr>
<tr>
<td>lockers_total</td>
<td>0</td>
</tr>
<tr>
<td>lockers_done</td>
<td>0</td>
</tr>
<tr>
<td>current_locker_pid</td>
<td>0</td>
</tr>
<tr>
<td>blocks_total</td>
<td>44248</td>
</tr>
<tr>
<td>blocks_done</td>
<td>17627</td>
</tr>
<tr>
<td>tuples_total</td>
<td>0</td>
</tr>
<tr>
<td>tuples_done</td>
<td>0</td>
</tr>
<tr>
<td>partitions_total</td>
<td>0</td>
</tr>
<tr>
<td>partitions_done</td>
<td>0</td>
</tr>
</tbody>
</table>

PGConf.Brasil 2019
Operation phases of CREATE INDEX / REINDEX

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

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

<table>
<thead>
<tr>
<th>phase</th>
<th>....</th>
</tr>
</thead>
<tbody>
<tr>
<td>lockers_total</td>
<td>0</td>
</tr>
<tr>
<td>lockers_done</td>
<td>0</td>
</tr>
<tr>
<td>current_locker_pid</td>
<td>0</td>
</tr>
<tr>
<td>blocks_total</td>
<td>44248</td>
</tr>
<tr>
<td>blocks_done</td>
<td>17627</td>
</tr>
<tr>
<td>tuples_total</td>
<td>0</td>
</tr>
<tr>
<td>tuples_done</td>
<td>0</td>
</tr>
<tr>
<td>partitions_total</td>
<td>0</td>
</tr>
<tr>
<td>partitions_done</td>
<td>0</td>
</tr>
</tbody>
</table>
Build phases for B-Tree indexes

1. initializing
2. scanning table
3. sorting live tuples
4. sorting dead tuples
5. loading tuples in tree
Reporting CLUSTER / VACUUM FULL progress

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

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

<table>
<thead>
<tr>
<th>pid</th>
<th>1209</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>esquema.tabela</td>
</tr>
<tr>
<td>command</td>
<td>VACUUM FULL</td>
</tr>
<tr>
<td>phase</td>
<td>seq scanning heap</td>
</tr>
<tr>
<td>cluster_index_relid</td>
<td>0</td>
</tr>
<tr>
<td>heap_tuples_scanned</td>
<td>8064358</td>
</tr>
<tr>
<td>heap_tuples_written</td>
<td>8064358</td>
</tr>
<tr>
<td>heap_blks_total</td>
<td>44248</td>
</tr>
<tr>
<td>heap_blks_scanned</td>
<td>35684</td>
</tr>
<tr>
<td>index_rebuild_count</td>
<td>0</td>
</tr>
</tbody>
</table>
CLUSTER: operation phases

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

- Patch submitted for PostgreSQL 13
- https://postgr.es/m/20190621185207.GA27929@alvherre.pgsq
Questions?

Thanks for listening!
Appendix: Implementation

- 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

```c
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();
```