

SQLreduce - Minimizing SQL Queries

Less is more

Christoph Berg <christoph.berg@credativ.de>
credativ GmbH

Hamburg, 2022-05-28





Bugs!

- queries trigger errors
- PostgreSQL crashes
- often on complex queries
- SQLsmith – <https://github.com/anse1/sqlsmith>



Big query

```
select case when pg_catalog.lastval() < pg_catalog.pg_stat_get_bgwriter_maxwritten_clean() then case when pg_catalog.circle_sub_pt( cast(cast(null as circle) as circle), cast((select location from public.emp limit 1 offset 13) as point)) ~ cast(nullif(case when cast(null as box) &> (select boxcol from public.brinestest limit 1 offset 2) then (select f1 from public.circle_tbl limit 1 offset 4) else (select f1 from public.circle_tbl limit 1 offset 4) end, case when (select pg_catalog.max(class) from public.f_star) ~~ ref_0.c then cast(null as circle) else cast(null as circle) end ) as circle) then ref_0.a else ref_0.a end else case when pg_catalog.circle_sub_pt( cast(cast(null as circle) as circle), cast((select location from public.emp limit 1 offset 13) as point)) ~ cast(nullif(case when cast(null as box) &> (select boxcol from public.brinestest limit 1 offset 2) then (select f1 from public.circle_tbl limit 1 offset 4) else (select f1 from public.circle_tbl limit 1 offset 4) end, case when (select pg_catalog.max(class) from public.f_star) ~~ ref_0.c then cast(null as circle) else cast(null as circle) end ) as circle) then ref_0.a else ref_0.a end end as c0, case when (select intervalcol from public.brinestest limit 1 offset 1) >= cast(null as "interval") then case when ((select pg_catalog.max(roomno) from public.room) !~~ ref_0.c) and (cast(null as xid) <> 100) then ref_0.b else ref_0.b end else case when ((select pg_catalog.max(roomno) from public.room) !~~ ref_0.c) and (cast(null as xid) <> 100) then ref_0.b else ref_0.b end end as c1, ref_0.a as c2, (select a from public.idxpart1 limit 1 offset 5) as c3, ref_0.b as c4, pg_catalog.stddev( cast((select pg_catalog.sum(float4col) from public.brinestest) as float4)) over (partition by ref_0.a,ref_0.b,ref_0.c order by ref_0.b) as c5, cast(nullif(ref_0.b, ref_0.a) as int4) as c6, ref_0.b as c7, ref_0.c as c8 from public.mlparted3 as ref_0 where true;
```

server closed the connection unexpectedly

- PostgreSQL Git revision 039eb6e92f (April 2018)
- <https://www.postgresql.org/message-id/flat/87woxi24uw.fsf@ansel.ydns.eu>



Now what?

- query makes PostgreSQL seqfault
- which part is to blame?
- is there a smaller query triggering the same error?
- so far manual work



Big query made small manually

```
select
  ref_0.a as c2,
  pg_catalog.stddev(
    cast((select pg_catalog.sum(float4col) from public.brintest)
         as float4))
    over (partition by ref_0.a,ref_0.b,ref_0.c order by ref_0.b)
  as c5
from
  public.mlparted3 as ref_0;
```



Big query made small automatically

```
SELECT pg_catalog.stddev(NULL) OVER () AS c5  
FROM public.mlparted3 AS ref_0
```



How to get there

- SQLreduce
 - <https://github.com/credativ/sqlreduce>
- based on pglast
 - PostgreSQL Languages AST and statements prettifier
 - Python module
 - <https://github.com/lelit/pglast>
- based on libpg_query
 - C library for accessing the PostgreSQL parser outside of the server
 - https://github.com/pganalyze/libpg_query



SQLreduce

def SQLreduce():

- translate query to parse tree
- loop:
 - parse tree -> more simple parse tree
 - translate parse tree back to query, run with PostgreSQL
 - compare error
- return query

- “more simple”: parse tree with less elements
- `sqlreduce -d "connection string" "some nasty query"`

Demo





Details

```
$ psql -c 'select pg_database.reltuples / 1000
           from pg_database, pg_class
           where 0 < pg_database.reltuples / 1000
           order by 1 desc limit 10'
ERROR:  column pg_database.reltuples does not exist
```



Parse tree

```

selectStmt
+-- targetList
|  |-- /
|      +-- pg_database.reltuples
|      |-- 1000
+-- fromClause
|  +-- pg_database
|  |-- pg_class
+-- whereClause
|  |-- <
|      +-- 0
|      |-- /
|          +-- pg_database.reltuples
|          |-- 1000
+-- orderClause
|  |-- 1
|-- limitCount
    |-- 10

```



Regenerated query

Input query: `select pg_database.reltuples / 1000
from pg_database, pg_class
where 0 < pg_database.reltuples / 1000
order by 1 desc limit 10`

Regenerated: `SELECT pg_database.reltuples / 1000
FROM pg_database, pg_class
WHERE 0 < ((pg_database.reltuples / 1000))
ORDER BY 1 DESC LIMIT 10`

Query returns: `ERROR: column pg_database.reltuples does not exist`



LIMIT 10 removed

```
SELECT pg_database.reltuples / 1000  
FROM pg_database, pg_class  
WHERE 0 < ((pg_database.reltuples / 1000))  
ORDER BY 1 DESC
```

Correct ERROR: column pg_database.reltuples does not exist



ORDER BY removed

```
SELECT pg_database.reltuples / 1000  
FROM pg_database, pg_class  
WHERE 0 < ((pg_database.reltuples / 1000))
```

Correct ERROR: column pg_database.reltuples does not exist



Target list removed

```
SELECT  
FROM pg_database, pg_class  
WHERE 0 < ((pg_database.reltuples / 1000))
```

Correct ERROR: column pg_database.reltuples does not exist



FROM list removed

```
SELECT  
WHERE 0 < ((pg_database.reltuples / 1000))
```

Wrong ERROR: missing FROM-clause entry for table "pg_database"



WHERE clause removed

```
SELECT  
FROM pg_database, pg_class
```

Wrong: no error



Parse tree up to now

```
selectStmt
+-- fromClause
|   +-- pg_database
|   +-- pg_class
|-- whereClause
    |-- <
        +-- 0
        |-- /
            +-- pg_database.reltuples
            |-- 1000
```

- descend into tree



Shorten FROM list

```
... FROM pg_database, pg_class
```

```
SELECT  
FROM pg_class  
WHERE 0 < ((pg_database.reltuples / 1000))
```

Wrong ERROR: missing FROM-clause entry for table "pg_database"

```
SELECT  
FROM pg_database  
WHERE 0 < ((pg_database.reltuples / 1000))
```

Correct ERROR: column pg_database.reltuples does not exist



Expression pull-up

```
... WHERE 0 < ((pg_database.reltuples / 1000))
```

```
SELECT FROM pg_database  
WHERE 0
```

Wrong ERROR: argument of WHERE must be type boolean, not type integer

```
SELECT FROM pg_database  
WHERE pg_database.reltuples / 1000
```

Correct ERROR: column pg_database.reltuples does not exist

```
SELECT FROM pg_database  
WHERE pg_database.reltuples
```

Correct ERROR: column pg_database.reltuples does not exist



Minimal query

```
selectStmt
+-- fromClause
|   `-- pg_database
`-- whereClause
    `-- pg_database.reltuples
```

```
SELECT
FROM pg_database
WHERE pg_database.reltuples
```

Correct ERROR: column pg_database.reltuples does not exist



Simplification rules

- remove entire subtrees
- pull-up: replace node with sub-node
- shorten lists
- replace expressions by NULL
- move sub-queries to top-level
- ... descend recursively and apply these rules

- always maintaining the same error message



What SQLreduce doesn't do yet

```
select myaggp05a(a) over (partition by a order by a)
from trigger_parted where pg_trigger_depth() <> a limit 40;
FATAL: server closed the connection unexpectedly
```

SQLreduce:

```
SELECT myaggp05a(NULL) OVER (ORDER BY a)
FROM trigger_parted WHERE pg_trigger_depth() <> a LIMIT 40
```

Tom Lane:

```
select a
from trigger_parted where pg_trigger_depth() <> a order by a limit 40;
```



Tom Lane vs. SQLreduce

Tom Lane:

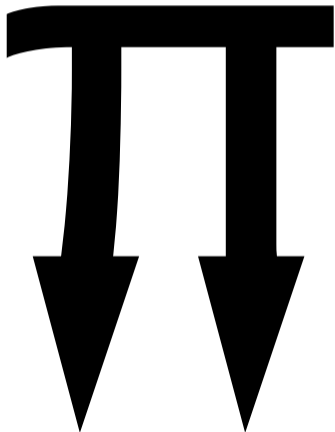
```
select a
from trigger_parted where pg_trigger_depth() <> a order by a limit 40;
```

SQLreduce:

```
SELECT
FROM trigger_parted WHERE pg_trigger_depth() <> a ORDER BY a LIMIT 40
```




`https://github.com/credativ/sqlreduce`





Big query 2

```
set min_parallel_table_scan_size to 0;
```

```
select 66 as c0, ref_1.cid as c1, pg_catalog.min( cast((select timetzcol from public.brintest limit 1 offset 3) as timetz)) over (partition by ref_1.name order by ref_1.name) as c2, ref_0.c as c3 from public.prt1_l as ref_0 right join public.my_property_normal as ref_1 on (ref_0.a <= ref_0.a) where EXISTS ( select ref_2.y as c0, ref_2.y as c1, sample_0.random as c2, ref_1.tel as c3, ref_0.a as c4, sample_0.random as c5, ref_2.y as c6, ref_2.x as c7, case when (true <> (select pg_catalog.bool_and(n) from testxmlschema.test2) ) and (sample_0.seqno = (select int_four from public.test_type_diff2_c3 limit 1 offset 1) ) then ref_2.y else ref_2.y end as c8, sample_0.seqno as c9, ref_1.name as c10, ref_0.a as c11, (select nslots from public.hub limit 1 offset 2) as c12, ref_1.name as c13 from public.hash_name_heap as sample_0 tablesample system (8.2) left join public.tt6 as ref_2 on (((cast(null as tinterval) <= (select f1 from public.tinterval_tbl limit 1 offset 79) ) and (ref_2.y is not NULL)) or (((false) and ((cast(null as tsquery) > (select keyword from public.test_tsquery limit 1 offset 34) ) or (((select pg_catalog.jsonb_agg(sl_name) from public.shoelace_obsolete) <@ cast(null as jsonb)) or (EXISTS ( select 100 as c0, ref_0.a as c1, sample_0.seqno as c2, ref_0.a as c3, sample_0.seqno as c4, ref_0.a as c5, (select a from public.prt3_n limit 1 offset 30) as c6, ref_2.y as c7, ref_1.cid as c8, ref_2.y as c9 from public.num_exp_mul as sample_1 tablesample system (7.1) where true limit 89))) and (cast(null as aclitem) @> cast(null as aclitem)))))) and ((select timecol from public.brintest limit 1 offset 96) > cast(null as "time")))) and (cast(null as timestamptz) < cast(null as "timestamp")))) where ((EXISTS ( select sample_2.int_four as c0, sample_0.seqno as c1, 43 as c2 from public.test_type_diff2_c1 as sample_2 tablesample bernoulli (2.3) where (sample_0.random ~ ref_1.name) and (ref_2.y <> ref_2.y) limit 98)) and (sample_0.random is NULL)) and (cast(null as point) <@ (select b from public.quad_box_tbl limit 1 offset 5) ) limit 61);
```

```
TRAP: FailedAssertion("! (subpath->parallel_safe)", File: "pathnode.c", Line: 1813)
```



Big query 2 reduced

```
SET min_parallel_table_scan_size TO 0;
```

```
SELECT
FROM public.prt1_1 AS ref_0
    RIGHT JOIN public.my_property_normal AS ref_1 ON NULL
WHERE EXISTS (SELECT
    FROM public.hash_name_heap AS sample_0
        LEFT JOIN public.tt6 AS ref_2 ON EXISTS (SELECT ref_1.cid AS c8)
    WHERE EXISTS (SELECT
        WHERE sample_0.random ~~ ref_1.name))
```

```
TRAP: FailedAssertion("!(subpath->parallel_safe)", File: "pathnode.c", L
```