Optimizing Queries on SQL-Level

Thomas Glas

Guided Research

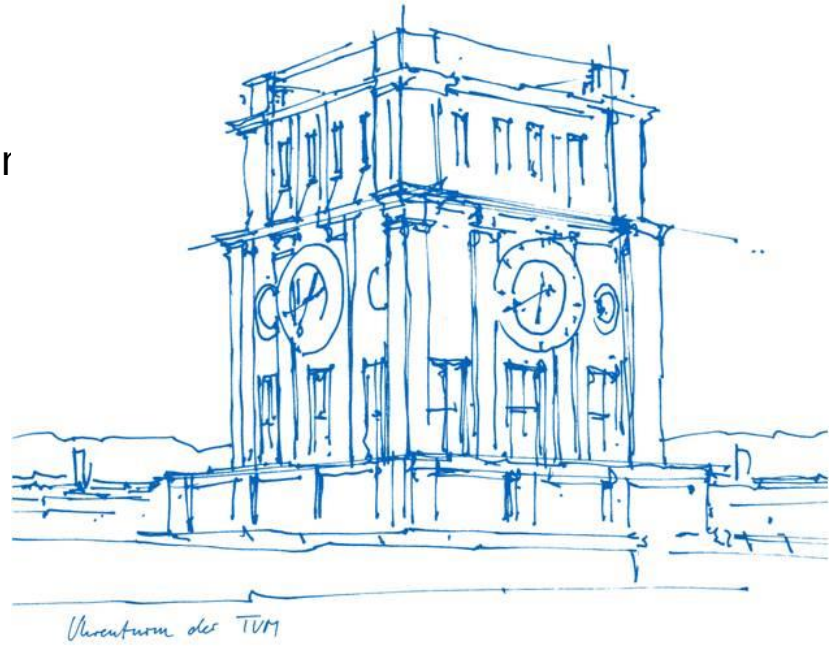
Supervised by Alexander Beischl

Technische Universität München

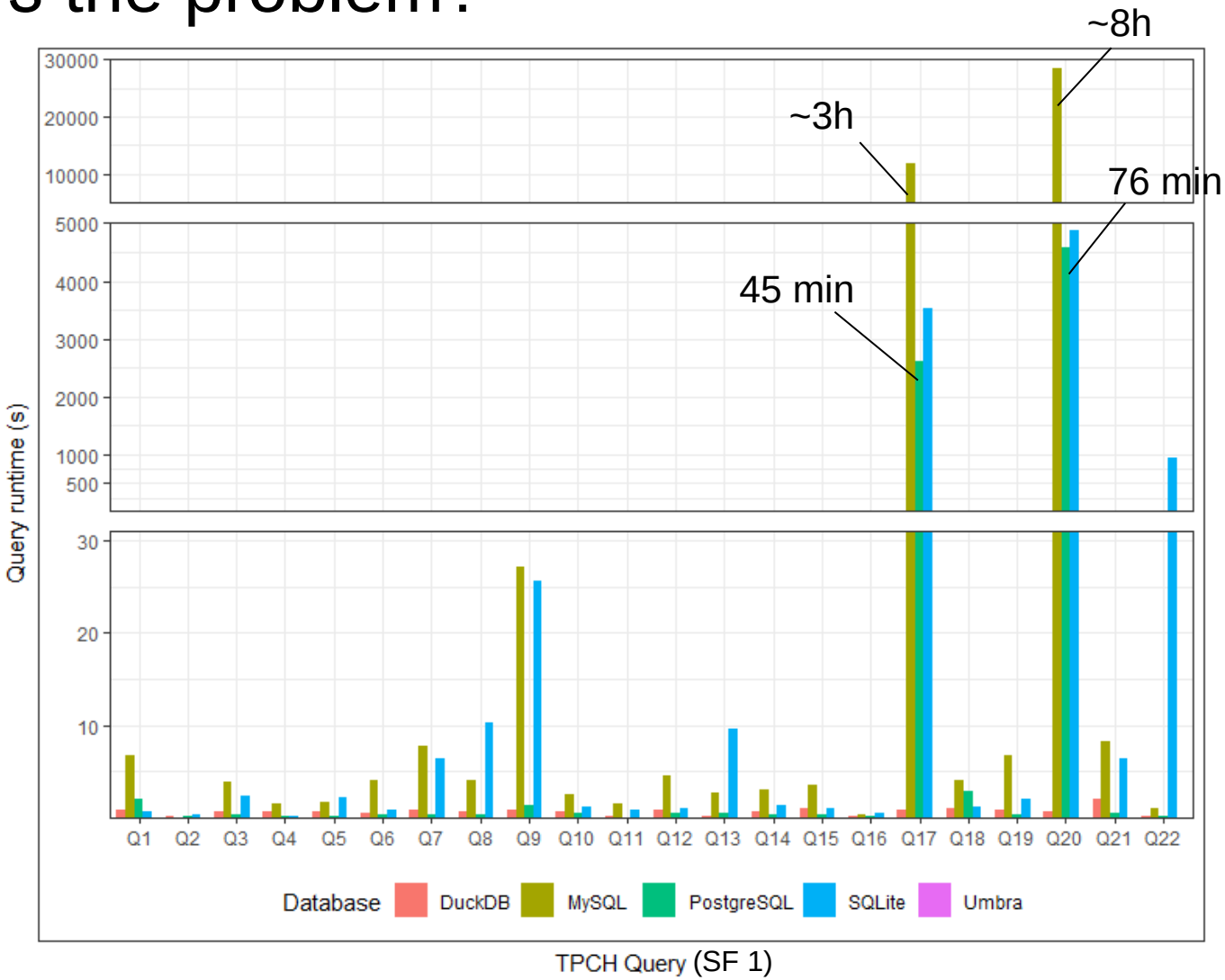
TUM School of Computation, Information and Techr

Lehrstuhl für Datenbankensysteme

München, 04. April 2023



What‘s the problem?



Correlated Subqueries…

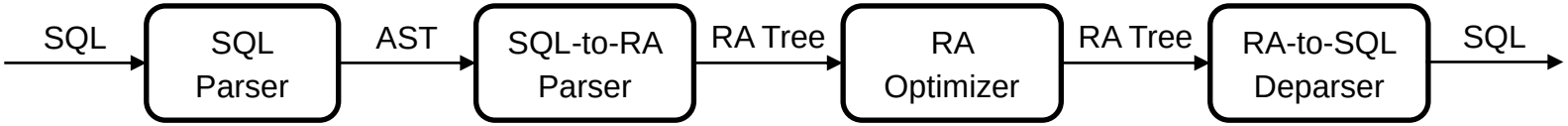
```sql
-- TPC-H Query 17
select
        sum(l_extendedprice) / 7.0 as avg_yearly
from
        lineitem,
        part
where
        p_partkey = l_partkey
        and p_brand = 'Brand#23'
        and p_container = 'MED BOX'
        and l_quantity < (
                select
                        0.2 * avg(l_quantity)
                from
                        lineitem
                where
                        l_partkey = p_partkey
        )
```

We can decorrelate this query in SQL

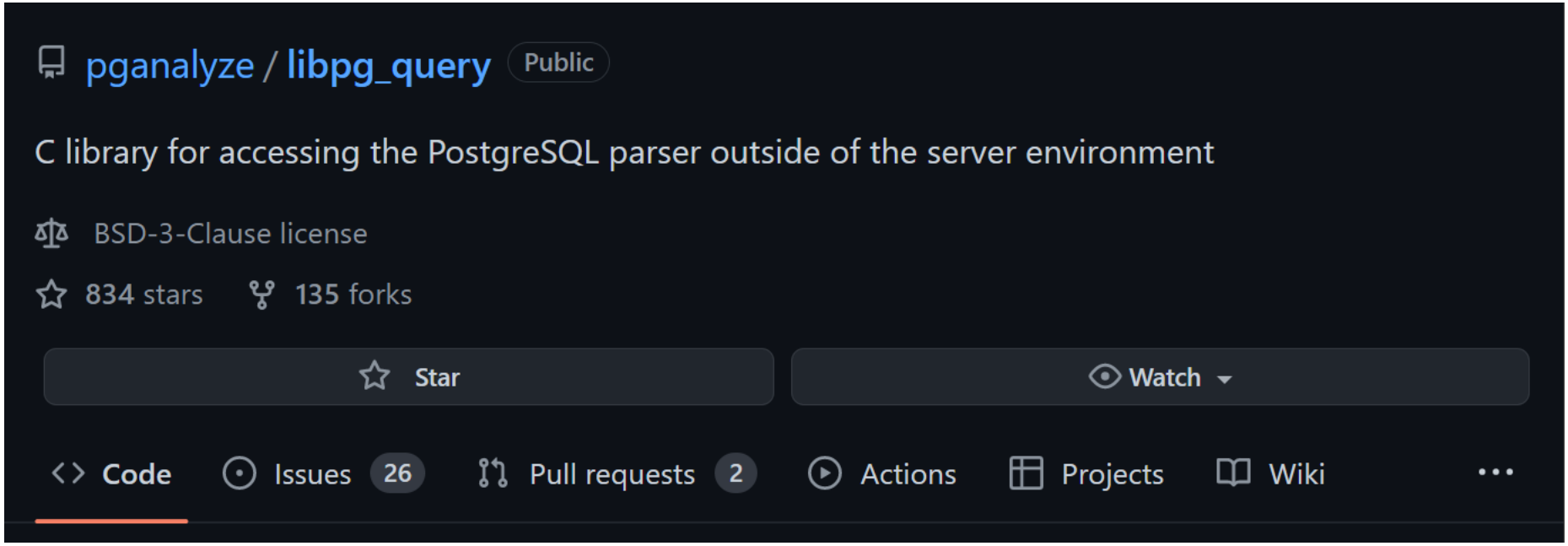
```sql
-- TPC-H Query 17
select
        sum(l_extendedprice) / 7.0 as avg_yearly
from
        lineitem,
        part
where
        p_partkey = l_partkey
        and p_brand = 'Brand#23'
        and p_container = 'MED BOX'
        and l_quantity < (
                select
                        0.2 * avg(l_quantity)
                from
                        lineitem
                where
                        l_partkey = p_partkey
        )
```

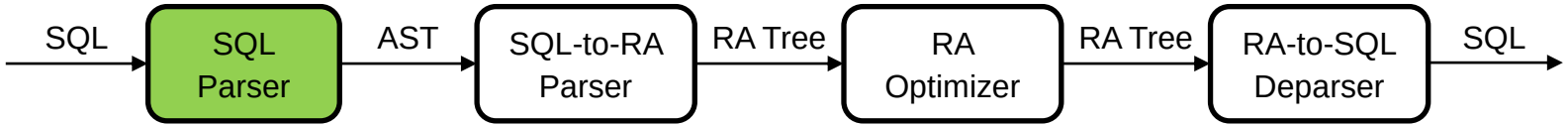
```sql
select
        sum(l_extendedprice)/7.0 as avg_yearly
from
        part,
        lineitem,
        (
                select 0.2*avg(l_quantity),
                        l_partkey
                from
                        lineitem
                group by
                        l_partkey
                ) as t1(m,t1_p_partkey)
where
        l_quantity<t1.m
        and p_partkey=t1.t1_p_partkey
        and p_partkey=l_partkey
        and p_brand='Brand#23'
        and p_container='MED BOX';
```

Architecture



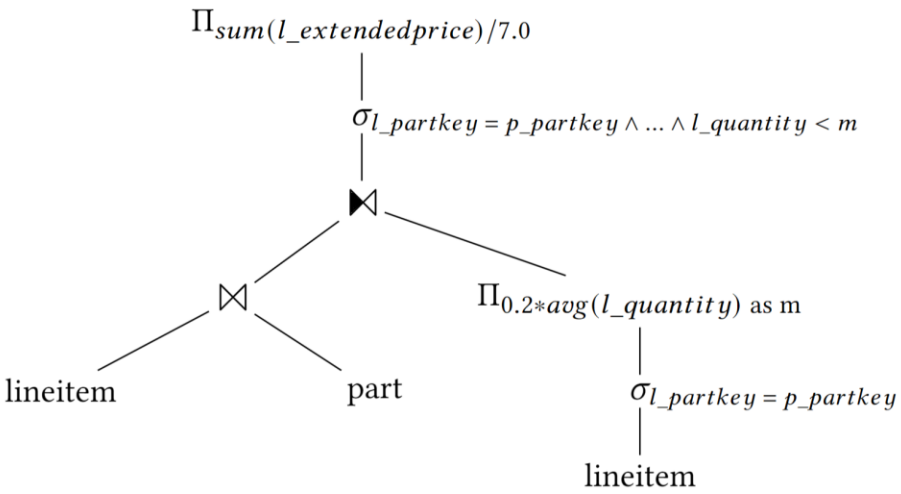
SQL Parser

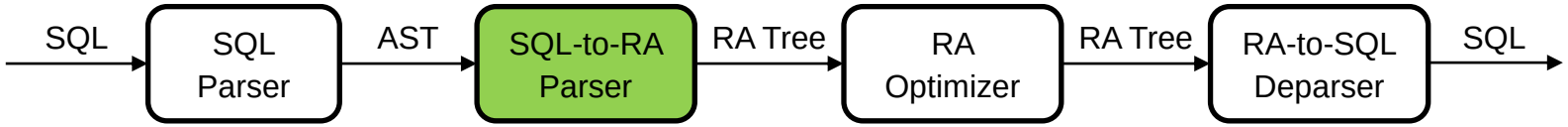

pganalyze / libpg_query
Public
C library for accessing the PostgreSQL parser outside of the server environment
BSD-3-Clause license
834 stars
135 forks
Star
Watch
Code
Issues 26
Pull requests 2
Actions
Projects
Wiki


SQL
SQL Parser
AST
SQL-to-RA Parser
RA Tree
RA Optimizer
RA Tree
RA-to-SQL Deparser
SQL

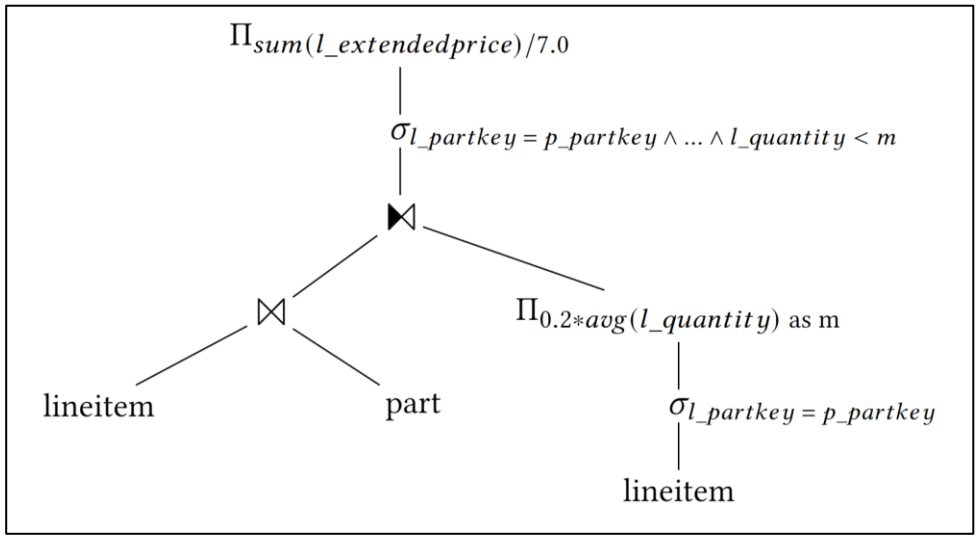
SQL-to-RA Parser

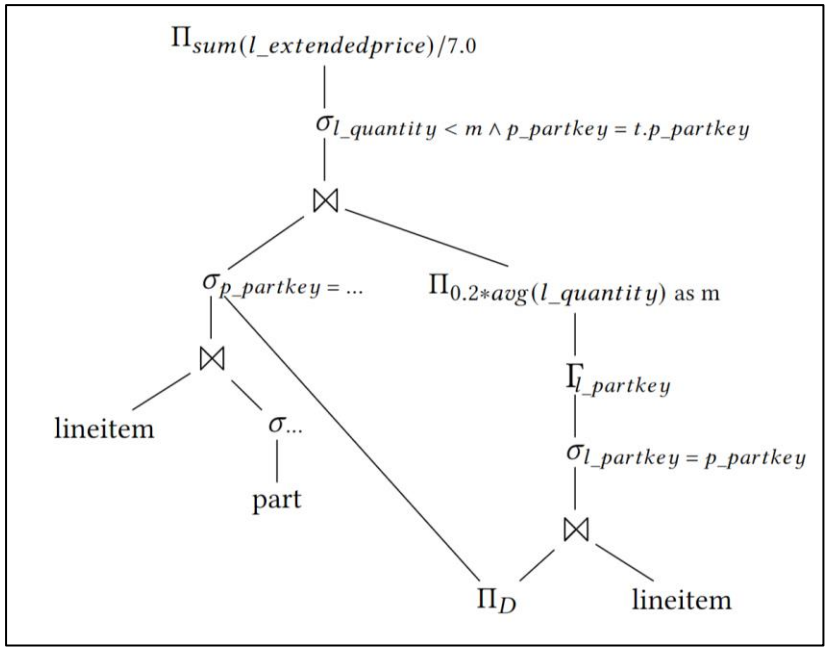
```sql
-- TPC-H Query 17
select
        sum(l_extendedprice) / 7.0 as avg_yearly
from
        lineitem,
        part
where
        p_partkey = l_partkey
        and p_brand = 'Brand#23'
        and p_container = 'MED BOX'
        and l_quantity < (
                select
                        0.2 * avg(l_quantity)
                from
                        lineitem
                where
                        l_partkey = p_partkey
        )
```

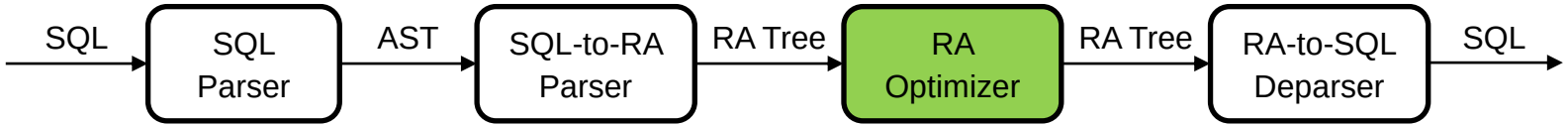




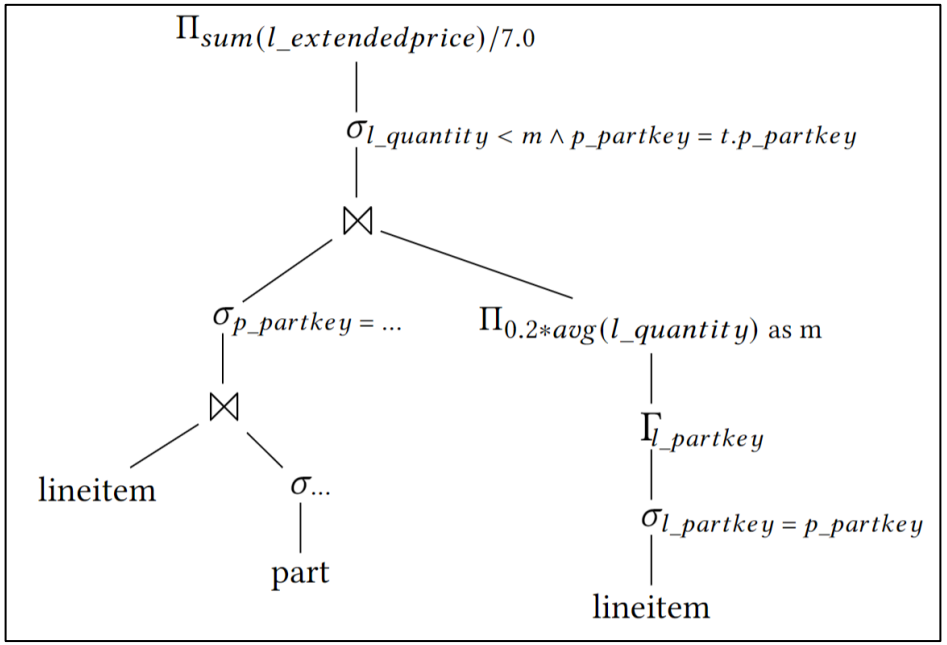
RA Optimizer



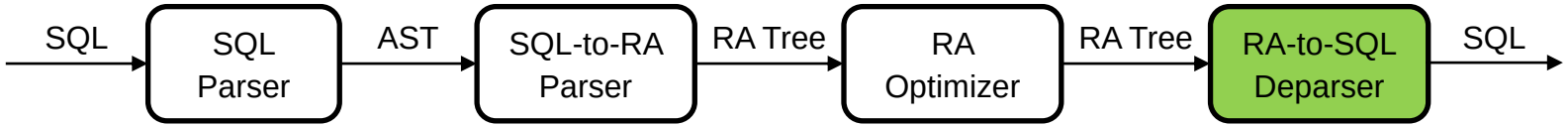




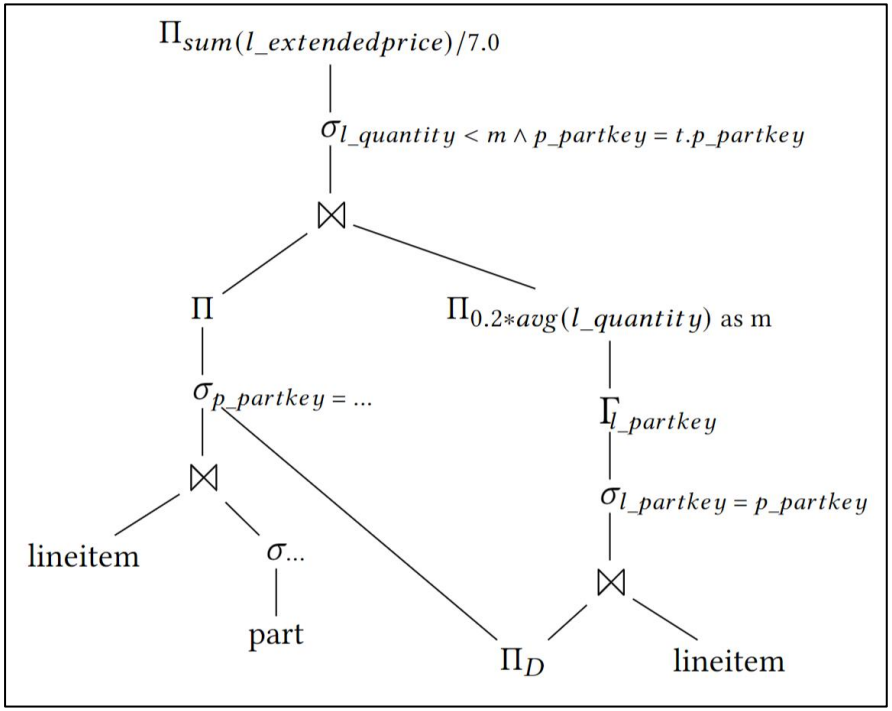
RA-to-SQL Deparser




```
1   select
2           sum(l_extendedprice)/7.0 as avg_yearly
3   from
4           part,
5           lineitem,
6           (
7                   select 0.2*avg(l_quantity),
8                           l_partkey
9                   from
10                          lineitem
11                  group by
12                          l_partkey
13                  ) as t1(m,t1_p_partkey)
14  where
15          l_quantity<t1.m
16          and p_partkey=t1.t1_p_partkey
17          and p_partkey=l_partkey
18          and p_brand='Brand#23'
19          and p_container='MED BOX';
```



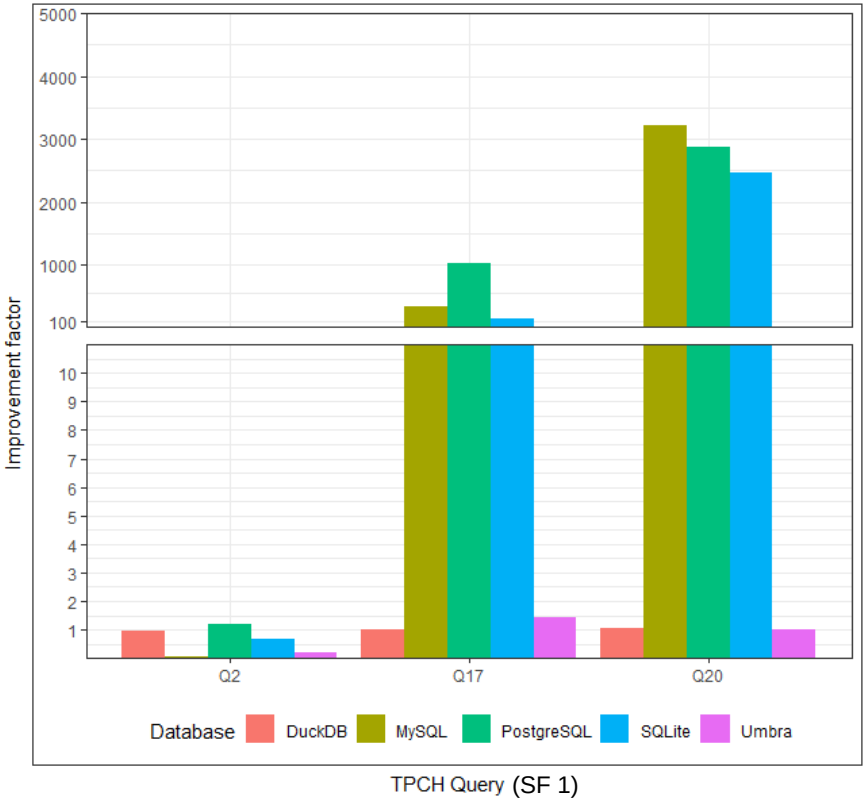
To decouple, or not to decouple



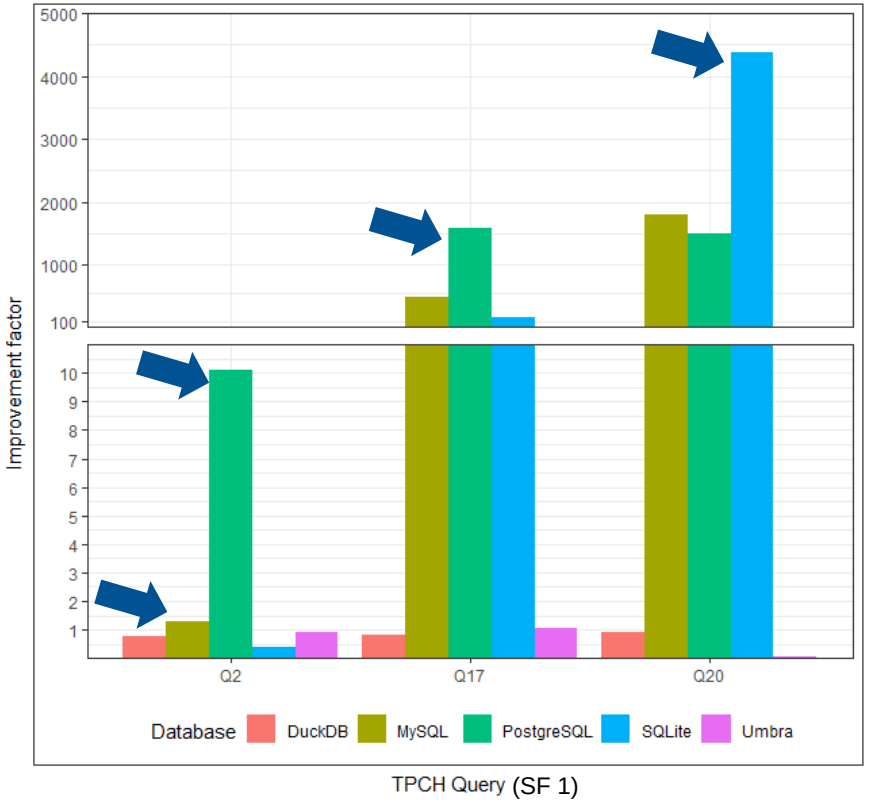

```sql
-- TPC-H Query 17
with cte_2 as (
        select
                l_extendedprice,
                l_quantity,
                p_partkey,
                l_partkey
        from
                part,
                lineitem
        where
                p_partkey=l_partkey
                and (p_brand='Brand#23' and p_container='MED BOX')
)
select
        (sum(l_extendedprice)/7.0) as avg_yearly
from
        cte_2,
        (select
                (0.2*avg(l_quantity)),
                d.p_partkey
        from    (select
                        p_partkey
                from cte_2
                ) as d(p_partkey), lineitem
        where l_partkey=d.p_partkey
        group by d.p_partkey
        ) as t1(m,t1_p_partkey)
where
        l_quantity<t1.m
        and p_partkey=t1.t1_p_partkey;
```

To decouple, or not to decouple

Decoupled



Not decoupled



Correlated *exists* and *in* subqueries

```
select ...
from   lineitem l1 ...
where  exists (select *
               from lineitem l2
               where l2.l_orderkey = l1.l_orderkey)
```

$$l_1 \ltimes (\sigma_{l_1.okey=l_2.okey}(l_2))$$

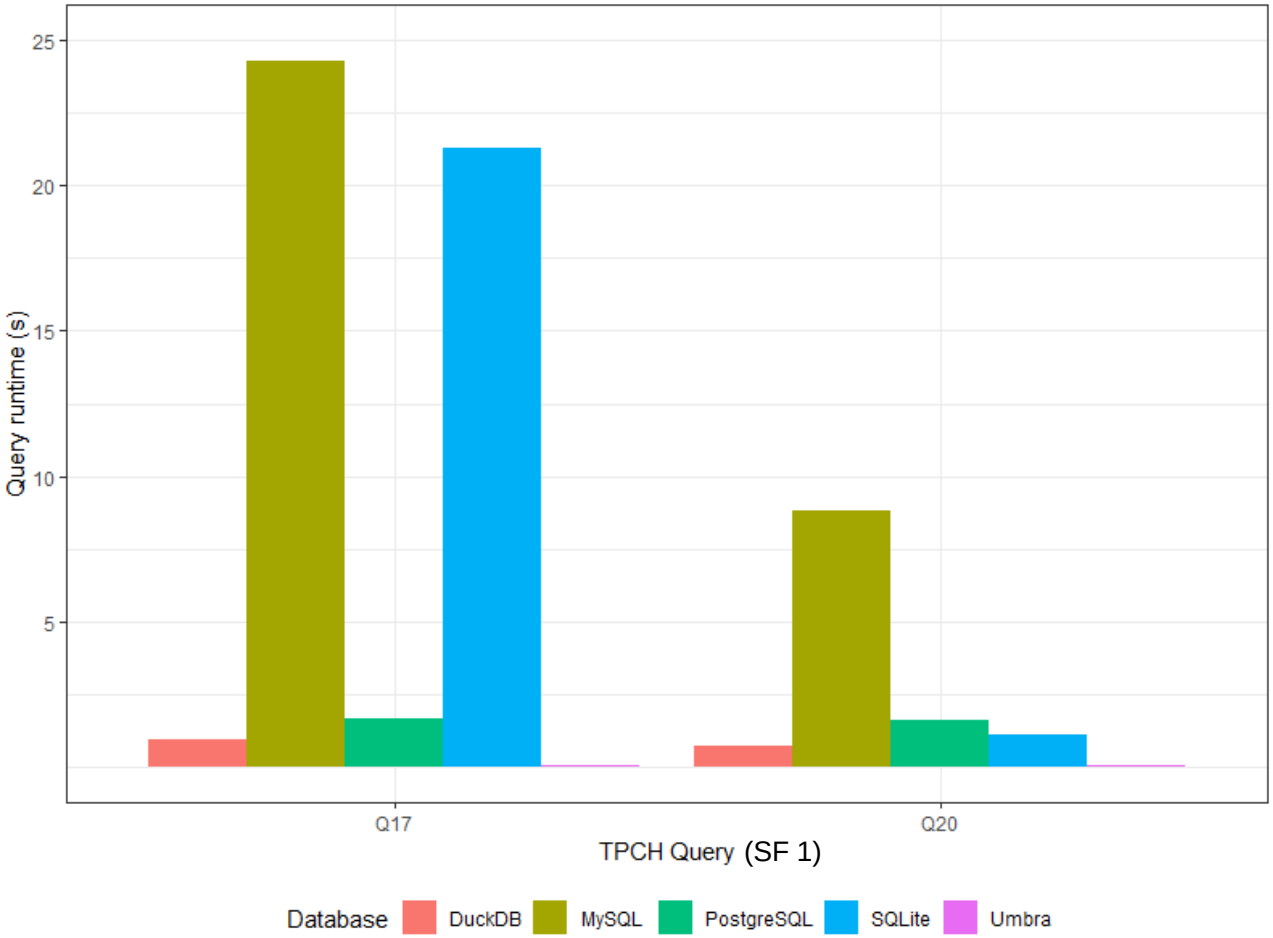
$$l_1 \ltimes_{l_1.okey=l_2.okey} (l_2)$$

Correlated *exists* and *in* subqueries

```sql
-- TPC-H Query 4
select
        o_orderpriority,
        count(*) as order_count
from
        orders
where
        o_orderdate >= date '1993-07-01'
        and o_orderdate < date '1993-07-01' + interval '3' month
        and exists (
                select
                        *
                from
                        lineitem
                where
                        l_orderkey = o_orderkey
                        and l_commitdate < l_receiptdate
        )
group by
        o_orderpriority
order by
        o_orderpriority
```

```sql
with cte_1(l_orderkey) as (
        select l_orderkey
        from lineitem
        where l_commitdate<l_receiptdate
)
select
        o_orderpriority,
        count(*) as order_count
from orders
where exists (
                select *
                from cte_1
                where cte_1.l_orderkey=o_orderkey
        )
        and o_orderdate>=date '1993-07-01'
        and o_orderdate<date '1993-07-01'+interval '3' month
group by o_orderpriority
order by o_orderpriority;
```

Results


25
20
15
10
5
Query runtime (s)
Q17
Q20
TPCH Query (SF 1)
Database
DuckDB
MySQL
PostgreSQL
SQLite
Umbra