Fall B561 Assignment 6
Name: Sricharraan Ramaswamy
Username: sriramas
UID: 2000855651
Date: 11th November, 2021

Collabrated with ATHULYA ANAND, SRI VARSHA CHELLAPILLA, SIDDHANT MESHRAM

Problem 1
Assume the relation schemas P (x), Q(x), R(x, y) and S(x, z). Consider the NOT ALL generalized query
where
(p.x, q.x)—P (p) Q(p) R(p.x) S(p.x)
R(p.x) = r.y — R(r) r.x = p.x
S(q.x) = s.z — S(s) s.x = q.x
In analogy with the analysis for the SOME generalized quantifier, do an analysis for the NOT ALL generalized quantifier.

Solution1 Query 1:

$$\text{set enable}_{m}ergejoin = off;$$
$$\text{set enable}_{h}ashjoin = off; setenable_{m}aterial = off; explainselectp.x, q.x$$
from P p, Q q
where exists (select 1
from r r
where r.x = p.x and exists (select 1
from s s
where s.x ¡¿ q.x or r.y ¡¿ s.z));

Solution

	QUERY PLAN text
1	Nested Loop Semi Join (cost=0.00..525439107.75 rows=1625625 width=8)
2	[...] Join Filter: ((p.x = r.x) AND (SubPlan 1))
3	[...] -> Nested Loop (cost=0.00..155585.50 rows=6502500 width=8)
4	[...] -> Seq Scan on p (cost=0.00..35.50 rows=2550 width=4)
5	[...] -> Seq Scan on q (cost=0.00..35.50 rows=2550 width=4)
6	[...] -> Seq Scan on r (cost=0.00..32.60 rows=2260 width=8)
7	[...] SubPlan 1
8	[...] -> Seq Scan on s (cost=0.00..43.90 rows=2260 width=0)
9	[...] Filter: ((x <> q.x) OR (r.y <> z))

Query 2:

$$\text{set enable}_{m}ergejoin = on; setenable_{h}ashjoin = on;$$

```
explain
select p.x, q.x
from P p, Q q
where exists (select 1
from r r
where r.x = p.x and exists (select 1
from s s
where s.x ¡¿ q.x or r.y ¡¿ s.z));
```

	QUERY PLAN text
1	Hash Semi Join (cost=26.42..523.92 rows=5000 width=8)
2	[...] Hash Cond: (p.x = r.x)
3	[...] Join Filter: (SubPlan 1)
4	[...] -> Nested Loop (cost=0.00..302.00 rows=10000 width=8)
5	[...] -> Seq Scan on p (cost=0.00..2.00 rows=100 width=4)
6	[...] -> Seq Scan on q (cost=0.00..2.00 rows=100 width=4)
7	[...] -> Hash (cost=14.52..14.52 rows=952 width=8)
8	[...] -> Seq Scan on r (cost=0.00..14.52 rows=952 width=8)
9	[...] SubPlan 1
10	[...] -> Seq Scan on s (cost=0.00..19.35 rows=957 width=0)
11	[...] Filter: ((x <> q.x) OR (r.y <> z))

Query 3:

drop index if exists indexed$_s$;
create index indexed$_s$*onsusinghash*(x);
explain
select p.x, q.x
from P p, Q q
where exists (select 1
from r r
where r.x = p.x and exists (select 1
from s s
where s.x ¡¿ q.x or r.y ¡¿ s.z));

	QUERY PLAN text
1	Hash Semi Join (cost=26.42..523.92 rows=5000 width=8)
2	[...] Hash Cond: (p.x = r.x)
3	[...] Join Filter: (SubPlan 1)
4	[...] -> Nested Loop (cost=0.00..302.00 rows=10000 width=8)
5	[...] -> Seq Scan on p (cost=0.00..2.00 rows=100 width=4)
6	[...] -> Seq Scan on q (cost=0.00..2.00 rows=100 width=4)
7	[...] -> Hash (cost=14.52..14.52 rows=952 width=8)
8	[...] -> Seq Scan on r (cost=0.00..14.52 rows=952 width=8)
9	[...] SubPlan 1
10	[...] -> Seq Scan on s (cost=0.00..19.35 rows=957 width=0)
11	[...] Filter: ((x <> q.x) OR (r.y <> z))

Query 4:

```
drop index if exists indexed_s;
explain
select p.x, q.x
from P p, Q q
where exists (select 1
from r r
where r.x = p.x and (q.x not in (select s.x
from s s) or r.y not in(select s.z from s s)));
```

	QUERY PLAN text
1	Hash Semi Join (cost=60.34..528.28 rows=7500 width=8)
2	[...] Hash Cond: (p.x = r.x)
3	[...] Join Filter: ((NOT (hashed SubPlan 1)) OR (NOT (hashed SubPlan 2)))
4	[...] -> Nested Loop (cost=0.00..302.00 rows=10000 width=8)
5	[...] -> Seq Scan on p (cost=0.00..2.00 rows=100 width=4)
6	[...] -> Seq Scan on q (cost=0.00..2.00 rows=100 width=4)
7	[...] -> Hash (cost=14.52..14.52 rows=952 width=8)
8	[...] -> Seq Scan on r (cost=0.00..14.52 rows=952 width=8)
9	[...] SubPlan 1
10	[...] -> Seq Scan on s (cost=0.00..14.57 rows=957 width=4)
11	[...] SubPlan 2
12	[...] -> Seq Scan on s s_1 (cost=0.00..14.57 rows=957 width=4)

Query 5:

```
explain
select p.x, q.x
from P p, Q q, R r
where r.x = p.x and (q.x not in (select s.x
from s s) or r.y not in(select s.z from s s));
```

	QUERY PLAN text
1	Nested Loop (cost=37.17..3396.78 rows=71400 width=8)
2	[...] Join Filter: ((NOT (hashed SubPlan 1)) OR (NOT (hashed SubPlan 2)))
3	[...] -> Hash Join (cost=3.25..30.86 rows=952 width=8)
4	[...] Hash Cond: (r.x = p.x)
5	[...] -> Seq Scan on r (cost=0.00..14.52 rows=952 width=8)
6	[...] -> Hash (cost=2.00..2.00 rows=100 width=4)
7	[...] -> Seq Scan on p (cost=0.00..2.00 rows=100 width=4)
8	[...] -> Seq Scan on q (cost=0.00..2.00 rows=100 width=4)
9	[...] SubPlan 1
10	[...] -> Seq Scan on s (cost=0.00..14.57 rows=957 width=4)
11	[...] SubPlan 2
12	[...] -> Seq Scan on s s_1 (cost=0.00..14.57 rows=957 width=4)

Query 6:

```
explain
select p.x, q.x
from P p, Q q, R r, S s
where r.x = p.x and (q.x ¡¿ s.x or r.y ¡¿ s.z);
```

	QUERY PLAN text
1	Nested Loop (cost=3.25..2756546.86 rows=91097164 width=8)
2	[...] Join Filter: ((q.x <> s.x) OR (r.y <> s.z))
3	[...] -> Nested Loop (cost=3.25..2886.86 rows=95200 width=12)
4	[...] -> Hash Join (cost=3.25..30.86 rows=952 width=8)
5	[...] Hash Cond: (r.x = p.x)
6	[...] -> Seq Scan on r (cost=0.00..14.52 rows=952 width=8)
7	[...] -> Hash (cost=2.00..2.00 rows=100 width=4)
8	[...] -> Seq Scan on p (cost=0.00..2.00 rows=100 width=4)
9	[...] -> Seq Scan on q (cost=0.00..2.00 rows=100 width=4)
10	[...] -> Seq Scan on s (cost=0.00..14.57 rows=957 width=8)

Query 7:

```
explain
select p.x, qs.x
from (select x, y from P natural join R) p
join
(select q.x, s.z from Q inner join S on Q.x ¡¿ S.x) qs on (p.y ¡¿ qs.z);
```

	QUERY PLAN text
1	Nested Loop (cost=3.25..2956230.80 rows=89280972 width=8)
2	[...] Join Filter: (q.x <> s.x)
3	[...] -> Nested Loop (cost=3.25..25289.80 rows=901828 width=8)
4	[...] Join Filter: (r.y <> s.z)
5	[...] -> Hash Join (cost=3.25..30.86 rows=952 width=8)
6	[...] Hash Cond: (r.x = p.x)
7	[...] -> Seq Scan on r (cost=0.00..14.52 rows=952 width=8)
8	[...] -> Hash (cost=2.00..2.00 rows=100 width=4)
9	[...] -> Seq Scan on p (cost=0.00..2.00 rows=100 width=4)
10	[...] -> Seq Scan on s (cost=0.00..14.57 rows=957 width=8)
11	[...] -> Seq Scan on q (cost=0.00..2.00 rows=100 width=4)

.

Problem 2
Consider query Q3

```
select distinct p.a
from P p, R r1, R r2, R r3, S s
where p.a = r1.a and r1.b = r2.a and r2.b = r3.a and r.b = S.b;
```

Problem 2
(a) Translate and optimize this query and call it Q4. Then write Q4 as an RA SQL query just as was done for query Q2 in Example 1.

Solution RA SQL:

```
select distinct p.a
from P as p
inner join R as r1 on p.a = r1.a
inner join R as r2 on r1.b = r2.a
inner join R as r3 on r2.a = r3.a
inner join S as s on r3.b = s.b;
```

.

Problem 2
(b) Compare queries Q3 and Q4 in a similar way as we did for Q1 and Q2 in Example 1. You should experiment with different sizes for R. Incidentally, these relations do not need to use the same parameters as those shown in the above table for Q1 and Q2 in Example 1.

Solution -2cm

R	Q_3 (in ms)	Q_4 (in ms)
10^4	18.137	15.183
10^5	2621.39	2437.42
10^6	22672.48	19124.76

.

Problem 2
(c) What conclusions do you draw from the results of these experiments regarding the effectiveness of query optimization in PostgreSQL and/or by hand?

Solution I would like to conclude that Optimizing manually surely improves the performance of the queries. Although, when we notice the running time for 105, there is not much a significant difference in the running times of Q3 and Q4.

.

Problem 3
Consider the Pure SQL Q5 which is an formulation of a variation of the not subset (not only) set semijoin query
p.a — P(p) R(p.a) S
where
R(p.a) = r.b — R(r) r.a = p.a.
select p.a
from Pp where exists (select 1
from Rr
where r.a = p.a and not exists (select 1 from S s where r.b = s.b));

Problem 3
(a) Translate and optimize this query and call it Q6. Then write Q6 as an RA SQL query just as was done for Q2 in Example 1.

Solution RA SQL:

```
select p.a
from p p
natural join (select r.*
from r
except
select r.*
from r
natural join s) q;
```

.

Problem 3
(b) An alternative way to write a query equivalent with Q5 is as the object-relational query with nested R.
Call this query Q7.
Compare queries Q5, Q6, and Q7 in a similar way as we did in Ex- ample 1. However, now you should experiment with different sizes for P, R and S as well as consider how P and S interact with R.

Solution -2cm

P	S	R	Q_5 (in ms)	Q_6 (in ms)	Q_7 (in ms)
10^2	10^2	10^2	0.120	0.679	1.709
10^2	10^2	10^3	0.274	0.670	0.713
10^2	10^2	10^4	2.811	5.976	1.276
10^2	10^3	10^2	0.152	0.186	0.17
10^2	10^4	10^2	0.172	0.492	0.721
10^3	10^2	10^5	22.295	65.401	28.234
10^4	10^3	10^5	18.67	71.91	51.903
10^5	10^5	10^5	14.676	91.996	153.027

.

Problem 3
(c) What conclusions do you draw from the results of these experiments regarding the effectiveness of query optimization in PostgreSQL and/or by hand?

Solution It can be concluded that the manual optimization of semi/anti semi joins does not improve the run time of queries.

.

Problem 4
Consider the Pure SQL Q8 which is an formulation of a variation of the not superset, (not all) set semijoin query
p.a — —P(p) R(p.a) S
where
R(p.a) = r.b — R(r) r.a = p.a.
select p.a
from Pp
where exists (select 1
from Ss
where not exists (select 1 from R where p.a = r.a and r.b = s.b));

Problem 4
(a) Translate and optimize this query and call it Q9. Then write Q9 as an RA SQL query just as was done for Q2 in Example 1.

Solution RA SQL:

```
select p.a
from p
except
select r.a
from r
natural join s;
```

.

Problem 4
(b) An alternative way to write a query equivalent with Q8 is as the object-relational query.
Call this query Q7
Compare queries Q5, Q6, and Q7 in a similar way as we did in Ex- ample 1. However, now you should experiment with different sizes for P, R and S as well as consider how P and S interact with R.

Solution -2cm

P	S	R	Q_8 (in ms)	Q_9 (in ms)	Q_{10} (in ms)
10^2	10^2	10^2	0.363	0.149	0.679
10^2	10^2	10^3	0.398	0.326	0.476
10^2	10^2	10^4	3.826	2.034	2.238
10^2	10^3	10^4	0.149	0.255	0.302
10^2	10^4	10^2	0.173	0.43	0.357
10^3	10^2	10^5	33.97	19.122	49.886
10^4	10^3	10^5	50.083	53.123	79.81
10^5	10^5	10^5	33.553	89.915	80.003
10^6	10^6	10^6	90437.37	207.008	288.026

.

Problem 4
(c) What conclusions do you draw from the results of these experiments?

Solution I conclude that the query Q9 (optimized) and Q10 (OR) have a similar performances. In this particular case, both Q9 and Q10 shows a better performance than that of Q8 which is unoptimized.
.

Problem 5
Give a brief comparison of your results for Problem 3 and Problem 4. In particular, where the results show significant differences, explain why you think that is the case. And, where the results show similarities, explain why you think that is the case.

Solution From the experiments above, I conclude that the differences in the efficiency of optimized query and unoptimized query are drawn from the relations that are used in the query. For example, when we optimize Q8, the relation S decreases drastically in the query which causes a huge gap in the running times of Q8 and Q9. Similar performances of unoptimized and opti- mized queries can be observed for smaller inputs.

.