Assignment 5

Join Tuning

Database Tuning

New Group 8

Frauenschuh Florian, 12109584

Lindner Peter, 12101607

Weilert Alexander, 12119653

June 10, 2024

Experimental Setup

Describe your experimental setup in a few lines.

For our experiments we used the following hardware and software:

Component	Specs
Processor	i7-13700H 3.7-5.0 GHz
Memory	32 GiB

Table 1: Hardware: Dell XPS 15 9530

Software	Version
OS	Ubuntu 22.04
Postgres	2.3.4
postgresql	42.7.3
MariaDB	10.6.16
mariadb-java-client	3.3.3
Java	18

Table 2: Software

Postgres was hosted on localhost, on which we also executed our experiments.

Join Strategies Proposed by System

Indexes	Join Strategy Q1	Join Strategy Q2
no index	Hash Join	Parallel Hash Join
unique non-clustering on `Publ.pubID`	Hash Join	Nested Loop
clustering on `Publ.pubID` and `Auth.pubID`	Merge Join	Nested Loop

Query Plans **No Index**

```
Hash Join  (cost=68240.32..234990.34 rows=3095201 width=82)
           (actual time=358.940..1408.766 rows=3095201 loops=1)
  Hash Cond: ((auth.pubid)::text = (publ.pubid)::text)
  ->  Seq Scan on auth  (cost=0.00..57762.01 rows=3095201 width=38)
```

```
                       (actual time=0.010..145.113 rows=3095201 loops=1)
  ->  Hash  (cost=34760.14..34760.14 rows=1233214 width=89)
            (actual time=358.310..358.310 rows=1233214 loops=1)
        Buckets: 32768  Batches: 64  Memory Usage: 2566kB
        ->  Seq Scan on publ  (cost=0.00..34760.14 rows=1233214 width=89)
                              (actual time=14.359..115.363 rows=1233214 loops=1)
Planning Time: 0.193 ms
JIT:
  Functions: 11
"  Options: Inlining false, Optimization false, Expressions true, Deforming true"
"  Timing: Generation 0.518 ms, Inlining 0.000 ms, Optimization 0.399 ms,
           Emission 13.983 ms, Total 14.901 ms"
Execution Time: 1465.864 ms

Gather  (cost=43930.96..73426.79 rows=25 width=67)
        (actual time=79.870..129.447 rows=183 loops=1)
  Workers Planned: 2
  Workers Launched: 2
  ->  Parallel Hash Join  (cost=42930.96..72424.29 rows=10 width=67)
                          (actual time=62.836..108.198 rows=61 loops=3)
        Hash Cond: ((publ.pubid)::text = (auth.pubid)::text)
        ->  Parallel Seq Scan on publ  (cost=0.00..27566.39 rows=513839 width=89)
                                       (actual time=0.024..24.990 rows=411071 loops=3)
        ->  Parallel Hash  (cost=42930.84..42930.84 rows=10 width=23)
                           (actual time=56.908..56.909 rows=61 loops=3)
              Buckets: 1024  Batches: 1  Memory Usage: 104kB
              ->  Parallel Seq Scan on auth  (cost=0.00..42930.84 rows=10 width=23)
                                             (actual time=11.146..56.776 rows=61 loops=3)
                    Filter: ((name)::text = 'Divesh Srivastava'::text)
                    Rows Removed by Filter: 1031673
Planning Time: 0.206 ms
Execution Time: 129.479 ms
```

unique non-clustering on `Publ.pubID`

```
Hash Join  (cost=68240.32..200556.26 rows=3095201 width=82)
           (actual time=350.001..1388.464 rows=3095201 loops=1)
  Hash Cond: ((auth.pubid)::text = (publ.pubid)::text)
  ->  Seq Scan on auth  (cost=0.00..57762.01 rows=3095201 width=38)
                        (actual time=0.011..146.583 rows=3095201 loops=1)
  ->  Hash  (cost=34760.14..34760.14 rows=1233214 width=89)
            (actual time=349.340..349.341 rows=1233214 loops=1)
        Buckets: 32768  Batches: 64  Memory Usage: 2566kB
        ->  Seq Scan on publ  (cost=0.00..34760.14 rows=1233214 width=89)
                              (actual time=10.149..109.417 rows=1233214 loops=1)
Planning Time: 0.649 ms
JIT:
  Functions: 11
"  Options: Inlining false, Optimization false, Expressions true, Deforming true"
"  Timing: Generation 0.771 ms, Inlining 0.000 ms, Optimization 0.478 ms,
           Emission 9.687 ms, Total 10.935 ms"
Execution Time: 1446.602 ms

Gather  (cost=1000.43..44017.79 rows=25 width=67)
        (actual time=18.620..73.904 rows=183 loops=1)
  Workers Planned: 2
  Workers Launched: 2
  ->  Nested Loop  (cost=0.43..43015.29 rows=10 width=67)
```

```
                   (actual time=7.765..59.686 rows=61 loops=3)
        ->  Parallel Seq Scan on auth  (cost=0.00..42930.84 rows=10 width=23)
                                       (actual time=7.652..58.637 rows=61 loops=3)
              Filter: ((name)::text = 'Divesh Srivastava'::text)
              Rows Removed by Filter: 1031673
        ->  Index Scan using idx_publ_pubid on publ  (cost=0.43..8.45 rows=1 width=89)
                                                     (actual time=0.016..0.016 rows=1 loops=183)
              Index Cond: ((pubid)::text = (auth.pubid)::text)
Planning Time: 0.347 ms
Execution Time: 73.943 ms
```

clustering on `Publ.pubID` and `Auth.pubID`

```
Merge Join  (cost=0.86..218725.83 rows=3095201 width=82)
            (actual time=6.925..1202.764 rows=3095201 loops=1)
  Merge Cond: ((publ.pubid)::text = (auth.pubid)::text)
  ->  Index Scan using idx_publ_pubid on publ  (cost=0.43..67009.64 rows=1233214 width=89)
                                               (actual time=0.010..129.239 rows=1233208 loops=1)
  ->  Index Scan using idx_auth_pubid on auth  (cost=0.43..110153.45 rows=3095201 width=38)
                                               (actual time=0.017..267.109 rows=3095201 loops=1)
Planning Time: 0.482 ms
JIT:
  Functions: 7
"  Options: Inlining false, Optimization false, Expressions true, Deforming true"
"  Timing: Generation 0.414 ms, Inlining 0.000 ms, Optimization 0.728 ms,
           Emission 6.152 ms, Total 7.294 ms"
Execution Time: 1259.616 ms

Gather  (cost=1000.43..44017.89 rows=25 width=67)
        (actual time=33.730..90.097 rows=183 loops=1)
  Workers Planned: 2
  Workers Launched: 2
  ->  Nested Loop  (cost=0.43..43015.39 rows=10 width=67)
                   (actual time=11.721..64.353 rows=61 loops=3)
        ->  Parallel Seq Scan on auth  (cost=0.00..42930.84 rows=10 width=23)
                                       (actual time=11.682..63.574 rows=61 loops=3)
              Filter: ((name)::text = 'Divesh Srivastava'::text)
              Rows Removed by Filter: 1031673
        ->  Index Scan using idx_publ_pubid on publ  (cost=0.43..8.45 rows=1 width=89)
                                                     (actual time=0.012..0.012 rows=1 loops=183)
              Index Cond: ((pubid)::text = (auth.pubid)::text)
Planning Time: 0.927 ms
Execution Time: 90.142 ms
```

Discussion Discuss your observations. Is the choice of the strategy expected? How does the system come to this choice?

no index With no indexes the usage of a Hash Join was as expected, since hash operations are the fastest while working on the data itself.

We note that in the execution plan of query 2, Postgres makes use of parallelization which can be seen by for example the `Gather` keyword.

unique non-clustering on `Publ.pubID` With a unique non-clustered index on `publ.pubid` we notice that Postgres does not make use of the index for the first query. This might be due to the fact that the `publ` table is not sorted by `pubid` and hence Postgres assumes that using hash operations would still be faster, because there might be the risk of random access onto the table data. The fact that there is no index on `auth.pubid` might enforce the decision of Postgres to avoid using the index. At first this seems to be counterintuitive, but after some consideration the strategy that Postgres uses makes sense.

For the second query Postgres uses a Nested Loop Join. Here it is interesting to see, that the filtering after `auth.name` happens in parallel. For this query Postgres makes use of the created index on `publ.pubid`. This strategy is as expected, except for the usage of parallelization which improves the performance but might not be the first thing coming to mind.

clustering on `Publ.pubID` and `Auth.pubID` For the first query the strategy is as expected, since Postgres is able to make use of both indexes to check for the join condition.

For the second query the execution plan is similar as for the unique non-clustered index on `publ.pubid`. Since Postgres first filters the `auth` table after the `name` condition, we think that Postgres expects the filtered rows to be low in quantity such that a Nested Loop Join is more efficient.

Indexed Nested Loop Join

Response times

Indexes	Response time Q1 [ms]	Response time Q2 [ms]
index on `Publ.pubID`	87	173
index on `Auth.pubID`	88	10534
index on `Publ.pubID` and `Auth.pubID`	77	161

Query plans

Index on `Publ.pubID` (Q1/Q2):

```
Gather  (cost=1000.43..985266.59 rows=3095201 width=82)
        (actual time=51.038..6522.294 rows=3095201 loops=1)
  Workers Planned: 2
  Workers Launched: 2
  ->  Nested Loop  (cost=0.43..674746.49 rows=1289667 width=82)
                   (actual time=63.599..6285.964 rows=1031734 loops=3)
        ->  Parallel Seq Scan on auth  (cost=0.00..39704.67 rows=1289667 width=38)
                                       (actual time=0.028..65.148 rows=1031734 loops=3)
        ->  Index Scan using idx_publ_pubid on publ  (cost=0.43..0.48 rows=1 width=89)
                                                (actual time=0.006..0.006 rows=1 loops=3095201)
              Index Cond: ((pubid)::text = (auth.pubid)::text)
Planning Time: 0.143 ms
JIT:
  Functions: 18
"  Options: Inlining true, Optimization true, Expressions true, Deforming true"
"  Timing: Generation 2.080 ms, Inlining 88.632 ms, Optimization 67.318 ms,
           Emission 34.580 ms, Total 192.611 ms"
Execution Time: 6589.889 ms

Gather  (cost=1000.43..44015.89 rows=25 width=67)
```

```
        (actual time=33.553..77.614 rows=183 loops=1)
  Workers Planned: 2
  Workers Launched: 2
  ->  Nested Loop  (cost=0.43..43013.39 rows=10 width=67)
                   (actual time=43.493..57.137 rows=61 loops=3)
        ->  Parallel Seq Scan on auth  (cost=0.00..42928.84 rows=10 width=23)
                                       (actual time=43.483..56.738 rows=61 loops=3)
              Filter: ((name)::text = 'Divesh Srivastava'::text)
              Rows Removed by Filter: 1031673
        ->  Index Scan using idx_publ_pubid on publ  (cost=0.43..8.45 rows=1 width=89)
                                                     (actual time=0.006..0.006 rows=1 loops=183)
              Index Cond: ((pubid)::text = (auth.pubid)::text)
Planning Time: 0.183 ms
Execution Time: 77.652 ms
```

Index on `Auth.pubID` (Q1/Q2):

```
Gather  (cost=1000.43..675617.53 rows=3095201 width=82)
        (actual time=89.746..4033.355 rows=3095201 loops=1)
  Workers Planned: 2
  Workers Launched: 2
  ->  Nested Loop  (cost=0.43..365097.43 rows=1289667 width=82)
                   (actual time=84.774..3810.102 rows=1031734 loops=3)
        ->  Parallel Seq Scan on publ  (cost=0.00..27564.39 rows=513839 width=89)
                                       (actual time=0.020..35.831 rows=411071 loops=3)
        ->  Index Scan using idx_auth_pubid on auth  (cost=0.43..0.62 rows=4 width=38)
                                                    (actual time=0.007..0.009 rows=3 loops=1233214)
              Index Cond: ((pubid)::text = (publ.pubid)::text)
Planning Time: 0.181 ms
JIT:
  Functions: 18
"  Options: Inlining true, Optimization true, Expressions true, Deforming true"
"  Timing: Generation 1.677 ms, Inlining 107.999 ms, Optimization 103.638 ms,
           Emission 42.278 ms, Total 255.592 ms"
Execution Time: 4099.108 ms

Gather  (cost=1000.43..355823.15 rows=25 width=67)
        (actual time=171.176..3741.105 rows=183 loops=1)
  Workers Planned: 2
  Workers Launched: 2
  ->  Nested Loop  (cost=0.43..354820.65 rows=10 width=67)
                   (actual time=266.050..3707.821 rows=61 loops=3)
        ->  Parallel Seq Scan on publ  (cost=0.00..27564.39 rows=513839 width=89)
                                       (actual time=0.019..36.992 rows=411071 loops=3)
        ->  Index Scan using idx_auth_pubid on auth  (cost=0.43..0.63 rows=1 width=23)
                                                     (actual time=0.009..0.009 rows=0 loops=1233214)
              Index Cond: ((pubid)::text = (publ.pubid)::text)
              Filter: ((name)::text = 'Divesh Srivastava'::text)
              Rows Removed by Filter: 3
Planning Time: 0.167 ms
JIT:
  Functions: 21
"  Options: Inlining false, Optimization false, Expressions true, Deforming true"
"  Timing: Generation 1.190 ms, Inlining 0.000 ms, Optimization 1.247 ms,
           Emission 20.288 ms, Total 22.726 ms"
Execution Time: 3741.807 ms
```

Index on `Auth.pubID` and `Auth.pubID` (Q1/Q2):

```
Gather  (cost=1000.43..661486.96 rows=3095201 width=82)
        (actual time=71.871..4027.451 rows=3095201 loops=1)
  Workers Planned: 2
  Workers Launched: 2
  ->  Nested Loop  (cost=0.43..350966.86 rows=1289667 width=82)
                   (actual time=79.942..3810.147 rows=1031734 loops=3)
        ->  Parallel Seq Scan on publ  (cost=0.00..27564.39 rows=513839 width=89)
                                       (actual time=0.026..36.205 rows=411071 loops=3)
        ->  Index Scan using idx_auth_pubid on auth  (cost=0.43..0.60 rows=3 width=38)
                                                     (actual time=0.007..0.009 rows=3 loops=1233214)
              Index Cond: ((pubid)::text = (publ.pubid)::text)
Planning Time: 0.248 ms
JIT:
  Functions: 18
"  Options: Inlining true, Optimization true, Expressions true, Deforming true"
"  Timing: Generation 1.525 ms, Inlining 116.783 ms, Optimization 81.539 ms,
          Emission 41.062 ms, Total 240.909 ms"
Execution Time: 4092.914 ms
```

```
Gather  (cost=1000.43..44015.79 rows=24 width=67)
        (actual time=41.790..83.937 rows=183 loops=1)
  Workers Planned: 2
  Workers Launched: 2
  ->  Nested Loop  (cost=0.43..43013.39 rows=10 width=67)
                   (actual time=43.673..56.285 rows=61 loops=3)
        ->  Parallel Seq Scan on auth  (cost=0.00..42928.84 rows=10 width=23)
                                      (actual time=43.662..55.809 rows=61 loops=3)
              Filter: ((name)::text = 'Divesh Srivastava'::text)
              Rows Removed by Filter: 1031673
        ->  Index Scan using idx_publ_pubid on publ  (cost=0.43..8.45 rows=1 width=89)
                                                     (actual time=0.007..0.007 rows=1 loops=183)
              Index Cond: ((pubid)::text = (auth.pubid)::text)
Planning Time: 0.264 ms
Execution Time: 83.982 ms
```

Discussion Discuss your observations. Are the response times expected? Why (not)?

For the first query the response times are as expected. As one can see the execution plans are the same for index on `publ.pubid` and index on `publ.pubid and auth.pubid`. It only differs for the case where we only have an index on `auth.pubid`, but this does not effect the response time as the tables are not too different in size.

For the second query we have the same pattern. In the case with the index only on `auth.pubid`, we can only make use of an index when the `name` condition is satisfied. Hence, the response time is much higher than in the other cases.

Sort-Merge Join

Response times

Indexes	Response time Q1 [ms]	Response time Q2 [ms]
no index	11657	4486
two non-clustering indexes	68	4455
two clustering indexes	54	1279

Query plans

No index (Q1/Q2):

```
Gather  (cost=528105.98..866474.71 rows=3095201 width=82)
        (actual time=5802.444..8326.897 rows=3095201 loops=1)
  Workers Planned: 2
  Workers Launched: 2
  ->  Merge Join  (cost=527105.98..555954.61 rows=1289667 width=82)
                  (actual time=5781.599..7971.966 rows=1031734 loops=3)
        Merge Cond: ((auth.pubid)::text = (publ.pubid)::text)
        ->  Sort  (cost=241128.63..244352.80 rows=1289667 width=38)
                  (actual time=2398.987..3050.968 rows=1031734 loops=3)
              Sort Key: auth.pubid
              Sort Method: external merge  Disk: 49888kB
              Worker 0:  Sort Method: external merge  Disk: 48504kB
              Worker 1:  Sort Method: external merge  Disk: 50040kB
              ->  Parallel Seq Scan on auth  (cost=0.00..39704.67 rows=1289667 width=38)
                                         (actual time=0.015..53.500 rows=1031734 loops=3)
        ->  Materialize  (cost=285977.35..292143.42 rows=1233214 width=89)
                         (actual time=3320.389..4326.188 rows=1549380 loops=3)
              ->  Sort  (cost=285977.35..289060.38 rows=1233214 width=89)
                        (actual time=3320.384..4237.252 rows=1233207 loops=3)
                    Sort Key: publ.pubid
                    Sort Method: external merge  Disk: 121600kB
                    Worker 0:  Sort Method: external merge  Disk: 121608kB
                    Worker 1:  Sort Method: external merge  Disk: 121592kB
                    ->  Seq Scan on publ  (cost=0.00..34758.14 rows=1233214 width=89)
                                      (actual time=0.023..90.531 rows=1233214 loops=3)
Planning Time: 0.325 ms
JIT:
  Functions: 27
"  Options: Inlining true, Optimization true, Expressions true, Deforming true"
"  Timing: Generation 1.241 ms, Inlining 56.289 ms, Optimization 82.660 ms,
           Emission 47.668 ms, Total 187.859 ms"
Execution Time: 8402.264 ms

Gather  (cost=169147.54..171711.65 rows=24 width=67)
        (actual time=1061.816..1427.720 rows=183 loops=1)
  Workers Planned: 2
  Workers Launched: 2
  ->  Merge Join  (cost=168147.54..170709.25 rows=10 width=67)
                  (actual time=1040.071..1396.399 rows=61 loops=3)
        Merge Cond: ((publ.pubid)::text = (auth.pubid)::text)
        ->  Sort  (cost=102648.98..103933.58 rows=513839 width=89)
                  (actual time=872.941..1205.738 rows=409987 loops=3)
              Sort Key: publ.pubid
              Sort Method: external merge  Disk: 41264kB
              Worker 0:  Sort Method: external merge  Disk: 40296kB
              Worker 1:  Sort Method: external merge  Disk: 40144kB
              ->  Parallel Seq Scan on publ  (cost=0.00..27564.39 rows=513839 width=89)
                                         (actual time=5.574..39.415 rows=411071 loops=3)
        ->  Sort  (cost=65498.56..65498.62 rows=24 width=23)
                  (actual time=139.274..139.288 rows=183 loops=3)
              Sort Key: auth.pubid
              Sort Method: quicksort  Memory: 39kB
              Worker 0:  Sort Method: quicksort  Memory: 39kB
              Worker 1:  Sort Method: quicksort  Memory: 39kB
              ->  Seq Scan on auth  (cost=0.00..65498.01 rows=24 width=23)
```

```
                                    (actual time=18.086..139.219 rows=183 loops=3)
                    Filter: ((name)::text = 'Divesh Srivastava'::text)
                    Rows Removed by Filter: 3095018
Planning Time: 0.335 ms
JIT:
  Functions: 36
"  Options: Inlining false, Optimization false, Expressions true, Deforming true"
"  Timing: Generation 1.486 ms, Inlining 0.000 ms, Optimization 0.809 ms,
           Emission 16.000 ms, Total 18.295 ms"
Execution Time: 1433.280 ms
```

Two non-clustering indexes (Q1/Q2):

```
Merge Join  (cost=0.86..366183.44 rows=3095201 width=82)
            (actual time=2.982..5811.830 rows=3095201 loops=1)
  Merge Cond: ((publ.pubid)::text = (auth.pubid)::text)
  ->  Index Scan using idx_publ_pubid on publ  (cost=0.43..134259.26 rows=1233214 width=89)
                                            (actual time=0.008..1722.422 rows=1233208 loops=1)
  ->  Index Scan using idx_auth_pubid on auth  (cost=0.43..190563.20 rows=3095201 width=38)
                                            (actual time=0.017..3063.996 rows=3095201 loops=1)
Planning Time: 0.337 ms
JIT:
  Functions: 7
"  Options: Inlining false, Optimization false, Expressions true, Deforming true"
"  Timing: Generation 0.355 ms, Inlining 0.000 ms, Optimization 0.254 ms,
           Emission 2.685 ms, Total 3.293 ms"
Execution Time: 5874.441 ms

Gather  (cost=169147.54..171711.65 rows=24 width=67)
        (actual time=1098.197..1465.949 rows=183 loops=1)
  Workers Planned: 2
  Workers Launched: 2
  ->  Merge Join  (cost=168147.54..170709.25 rows=10 width=67)
                  (actual time=1066.146..1422.252 rows=61 loops=3)
        Merge Cond: ((publ.pubid)::text = (auth.pubid)::text)
        ->  Sort  (cost=102648.98..103933.58 rows=513839 width=89)
                  (actual time=891.047..1223.767 rows=409987 loops=3)
              Sort Key: publ.pubid
              Sort Method: external merge  Disk: 41960kB
              Worker 0:  Sort Method: external merge  Disk: 39536kB
              Worker 1:  Sort Method: external merge  Disk: 40248kB
              ->  Parallel Seq Scan on publ  (cost=0.00..27564.39 rows=513839 width=89)
                                          (actual time=14.731..51.140 rows=411071 loops=3)
        ->  Sort  (cost=65498.56..65498.62 rows=24 width=23)
                  (actual time=146.933..146.948 rows=183 loops=3)
              Sort Key: auth.pubid
              Sort Method: quicksort  Memory: 39kB
              Worker 0:  Sort Method: quicksort  Memory: 39kB
              Worker 1:  Sort Method: quicksort  Memory: 39kB
              ->  Seq Scan on auth  (cost=0.00..65498.01 rows=24 width=23)
                                    (actual time=16.108..146.881 rows=183 loops=3)
                    Filter: ((name)::text = 'Divesh Srivastava'::text)
                    Rows Removed by Filter: 3095018
Planning Time: 0.592 ms
JIT:
  Functions: 36
"  Options: Inlining false, Optimization false, Expressions true, Deforming true"
"  Timing: Generation 2.071 ms, Inlining 0.000 ms, Optimization 1.194 ms,
          Emission 43.042 ms, Total 46.307 ms"
Execution Time: 1471.310 ms
```

Two clustering indexes (Q1/Q2):

```
Merge Join  (cost=0.86..366621.28 rows=3095201 width=82)
            (actual time=5.269..1221.581 rows=3095201 loops=1)
  Merge Cond: ((publ.pubid)::text = (auth.pubid)::text)
  ->  Index Scan using idx_publ_pubid on publ  (cost=0.43..134272.15 rows=1233214 width=89)
                                               (actual time=0.006..134.270 rows=1233208 loops=1)
  ->  Index Scan using idx_auth_pubid on auth  (cost=0.43..190576.08 rows=3095201 width=38)
                                               (actual time=0.019..272.160 rows=3095201 loops=1)
Planning Time: 0.836 ms
JIT:
  Functions: 7
"  Options: Inlining false, Optimization false, Expressions true, Deforming true"
"  Timing: Generation 0.549 ms, Inlining 0.000 ms, Optimization 0.284 ms,
          Emission 4.946 ms, Total 5.779 ms"
Execution Time: 1279.286 ms


Gather  (cost=169151.54..171723.36 rows=24 width=67)
        (actual time=407.404..543.130 rows=183 loops=1)
  Workers Planned: 2
  Workers Launched: 2
  ->  Merge Join  (cost=168151.54..170720.96 rows=10 width=67)
                  (actual time=384.208..516.172 rows=61 loops=3)
        Merge Cond: ((publ.pubid)::text = (auth.pubid)::text)
        ->  Sort  (cost=102650.98..103935.58 rows=513839 width=89)
                  (actual time=216.376..311.272 rows=409986 loops=3)
              Sort Key: publ.pubid
              Sort Method: external merge  Disk: 42008kB
              Worker 0:  Sort Method: external merge  Disk: 39664kB
              Worker 1:  Sort Method: external merge  Disk: 40032kB
              ->  Parallel Seq Scan on publ  (cost=0.00..27566.39 rows=513839 width=89)
                                             (actual time=9.398..45.287 rows=411071 loops=3)
        ->  Sort  (cost=65500.56..65500.62 rows=24 width=23)
                  (actual time=154.120..154.131 rows=179 loops=3)
              Sort Key: auth.pubid
              Sort Method: quicksort  Memory: 39kB
              Worker 0:  Sort Method: quicksort  Memory: 39kB
              Worker 1:  Sort Method: quicksort  Memory: 39kB
              ->  Seq Scan on auth  (cost=0.00..65500.01 rows=24 width=23)
                                    (actual time=10.622..154.050 rows=183 loops=3)
                    Filter: ((name)::text = 'Divesh Srivastava'::text)
                    Rows Removed by Filter: 3095018
Planning Time: 0.546 ms
JIT:
  Functions: 36
"  Options: Inlining false, Optimization false, Expressions true, Deforming true"
"  Timing: Generation 2.573 ms, Inlining 0.000 ms, Optimization 1.625 ms,
          Emission 26.610 ms, Total 30.808 ms"
Execution Time: 548.867 ms
```

Discussion Discuss your observations. Are the response times expected? Why (not)?

For the first query the results are as expected. We note that with no indexes the DBMS first has to sort both tables in order to execute a Merge Join. Hence, the response time is pretty high. As soon as there are indexes on `pubid`, Postgres can use these to perform the Join more efficiently.

For the second query Postgres still has to sort both tables, but now the `auth` table first gets filtered by the `name` condition. Hence, sorting the `auth` table is much faster now. We note that for the case of the clustered indexes on `auth.pubid` and `publ.pubid`, the sorting is already done. Thus, this results in the fastest response times.

Hash Join

Response times

Indexes	Response time Q1 [ms]	Response time [ms] Q2
no index	341	274

Query plans

No Index (Q1/Q2):

```
Hash Join  (cost=68238.32..234986.34 rows=3095201 width=82)
           (actual time=363.000..1493.759 rows=3095201 loops=1)
  Hash Cond: ((auth.pubid)::text = (publ.pubid)::text)
  ->  Seq Scan on auth  (cost=0.00..57760.01 rows=3095201 width=38)
                        (actual time=0.024..150.751 rows=3095201 loops=1)
  ->  Hash  (cost=34758.14..34758.14 rows=1233214 width=89)
            (actual time=362.613..362.613 rows=1233214 loops=1)
        Buckets: 32768  Batches: 64  Memory Usage: 2566kB
        ->  Seq Scan on publ  (cost=0.00..34758.14 rows=1233214 width=89)
                              (actual time=9.259..110.827 rows=1233214 loops=1)
Planning Time: 0.129 ms
JIT:
  Functions: 11
"  Options: Inlining false, Optimization false, Expressions true, Deforming true"
"  Timing: Generation 0.512 ms, Inlining 0.000 ms, Optimization 0.289 ms,
           Emission 8.984 ms, Total 9.785 ms"
Execution Time: 1550.477 ms

Gather  (cost=43928.96..73422.79 rows=25 width=67)
        (actual time=68.279..120.483 rows=183 loops=1)
  Workers Planned: 2
  Workers Launched: 2
  ->  Parallel Hash Join  (cost=42928.96..72420.29 rows=10 width=67)
                          (actual time=54.799..102.273 rows=61 loops=3)
        Hash Cond: ((publ.pubid)::text = (auth.pubid)::text)
        ->  Parallel Seq Scan on publ  (cost=0.00..27564.39 rows=513839 width=89)
                                       (actual time=0.016..25.055 rows=411071 loops=3)
        ->  Parallel Hash  (cost=42928.84..42928.84 rows=10 width=23)
                           (actual time=51.264..51.264 rows=61 loops=3)
              Buckets: 1024  Batches: 1  Memory Usage: 40kB
              ->  Parallel Seq Scan on auth  (cost=0.00..42928.84 rows=10 width=23)
                                             (actual time=38.296..51.217 rows=61 loops=3)
                    Filter: ((name)::text = 'Divesh Srivastava'::text)
                    Rows Removed by Filter: 1031673
Planning Time: 0.077 ms
Execution Time: 120.510 ms
```

Discussion What do you think about the response time of the hash join vs. the response times of sort-merge and index nested loop join for each of the queries? Explain.

While in general hash operations are pretty fast, in some cases where the other join methods are able to use indexes, the usage of Hash Joins can be slower. This is due to the fact, that we still have to hash the whole table (query 1), or a filtered version of the table (query 2) to be able to perform the join operations.

Time Spent on this Assignment

Time in hours per person:

- Florian Frauenschuh: **4.5**
- Peter Lindner: **5.5**
- Alexander Weilert: **4.5**