Query 7

```
select s.sname
from sailors s
where
s.sid in( select r.sid

from reserves r where r.bid = 103 );
```

1) Without index

"QUERY PLAN"

1) "Hash Semi Join (cost=20000000833.72..20000001396.76 rows=498 width=21) (actual time=9.077..15.592 rows=511 loops=1)"
2) " Hash Cond: (s.sid = r.sid)"
3) " -> Seq Scan on sailors s (cost=10000000000.00..10000000507.00 rows=19000 width=25) (actual time=0.182..2.843 rows=19000 loops=1)"
4) " -> Hash (cost=10000000827.50..10000000827.50 rows=498 width=4) (actual time=8.841..8.843 rows=511 loops=1)"
5) "   Buckets: 1024 Batches: 1 Memory Usage: 26kB"
6) "   -> Seq Scan on reserves r (cost=10000000000.00..10000000827.50 rows=498 width=4) (actual time=0.040..8.673 rows=511 loops=1)"
7) "   Filter: (bid = 103)"
8) "   Rows Removed by Filter: 34489"
9) "Planning Time: 0.284 ms"
10) "Execution Time: 15.688 ms"

Flags:
-- SET enable_hashjoin=ON; -- SET enable_material = ON; -- SET enable_mergejoin=ON;
-- SET enable_nestloop=ON; -- SET enable_seqscan=OFF; -- SET enable_indexscan =ON;
-- SET enable_indexonlyscan = ON; -- SET enable_bitmapscan = OFF; -- SET enable_sort=ON;
-- SET enable_async_append =ON; -- SET enable_indexscan=ON;

---

2) With B+tree

```
Create index Btree_reserves_bid on reserves using Btree(bid);
Create index Btree_reserves_sid on reserves using Btree(sid);
Create index Btree_sailors_sid on sailors using Btree(sid);
CREATE INDEX Btree_boat_bid on boat using Btree(bid);
```

"QUERY PLAN"

1) "Hash Semi Join (cost=1235.12..2112.15 rows=498 width=21) (actual time=0.405..14.680 rows=511 loops=1)"
2) " Hash Cond: (s.sid = r.sid)"
3) " -> Index Scan using btree_sailors_sid on sailors s (cost=0.29..821.29 rows=19000 width=25) (actual time=0.010..8.166 rows=19000 loops=1)"
4) " -> Hash (cost=1228.60..1228.60 rows=498 width=4) (actual time=0.338..0.339 rows=511 loops=1)"
5) "   Buckets: 1024 Batches: 1 Memory Usage: 26kB"
6) "   -> Index Scan using btree_reserves_bid on reserves r (cost=0.29..1228.60 rows=498 width=4) (actual time=0.010..0.184 rows=511 loops=1)"
7) "   Index Cond: (bid = 103)"
8) "Planning Time: 0.696 ms"
9) "Execution Time: 14.770 ms"

Flags:
-- SET enable_hashjoin=ON; -- SET enable_material = ON; -- SET enable_mergejoin=ON;
-- SET enable_nestloop=ON; -- SET enable_seqscan=OFF; -- SET enable_indexscan =ON;
-- SET enable_indexonlyscan = ON; -- SET enable_bitmapscan = OFF; -- SET enable_sort=ON;
-- SET enable_async_append =ON; -- SET enable_indexscan=ON;

NOTE :

- other indicies was used in neted loop join
- Cost reduced and excution time

---

3) With Hash index

```
CREATE INDEX HASH_reserves_bid on reserves using hash(bid);
CREATE INDEX HASH_reserves_sid on reserves using hash(sid);
CREATE INDEX HASH_sailors_sid on sailors using hash(sid);
CREATE INDEX HASH_boat_bid on boat using hash(bid);
```

"QUERY PLAN"

1) "Nested Loop (cost=1233.97..2627.62 rows=498 width=21) (actual time=0.908..5.154 rows=511 loops=1)"
2) " -> HashAggregate (cost=1233.96..1238.91 rows=495 width=4) (actual time=0.880..1.191 rows=511 loops=1)"
3) "   Group Key: r.sid"
4) "   Batches: 1 Memory Usage: 73kB"
5) "   -> Index Scan using hash_reserves_bid on reserves r (cost=0.00..1232.71 rows=498 width=4) (actual time=0.026..0.427 rows=511 loops=1)"
6) "   Index Cond: (bid = 103)"
7) " -> Memoize (cost=0.01..2.81 rows=1 width=25) (actual time=0.005..0.006 rows=1 loops=511)"
8) "   Cache Key: r.sid"
9) "   Cache Mode: logical"
10) "   Hits: 0 Misses: 511 Evictions: 0 Overflows: 0 Memory Usage: 64kB"
11) "   -> Index Scan using hash_sailors_sid on sailors s (cost=0.00..2.80 rows=1 width=25) (actual time=0.003..0.004 rows=1 loops=511)"
12) "   Index Cond: (sid = r.sid)"
13) "Planning Time: 0.965 ms"
14) "Execution Time: 5.358 ms"

Flags:
-- SET enable_hashjoin=ON; -- SET enable_material = ON; -- SET enable_mergejoin=ON;
-- SET enable_nestloop=ON; -- SET enable_seqscan=OFF; -- SET enable_indexscan =ON;
-- SET enable_indexonlyscan = ON; -- SET enable_bitmapscan = OFF; -- SET enable_sort=ON;
-- SET enable_async_append =ON; -- SET enable_indexscan=ON;

NOTE :

- other indicies was used in neted loop join
- Cost reduced and excution time
- Cost is greater than BTREE but excution time less

---

4) With BRIN Index

-- CREATE INDEX BRIN_reserves_bid on reserves using brin(bid);
-- CREATE INDEX BRIN_sailors_sid on sailors using brin(sid);

"QUERY PLAN"

1) "Nested Loop (cost=6781.04..2964415.27 rows=498 width=21) (actual time=14.206..276.790 rows=511 loops=1)"
2) " -> HashAggregate (cost=841.00..845.95 rows=495 width=4) (actual time=13.175..13.367 rows=511 loops=1)"
3) "   Group Key: r.sid"
4) "   Batches: 1 Memory Usage: 73kB"
5) "   -> Bitmap Heap Scan on reserves r (cost=12.25..839.75 rows=498 width=4) (actual time=0.117..12.842 rows=511 loops=1)"
6) "   Recheck Cond: (bid = 103)"
7) "   Rows Removed by Index Recheck: 34489"
8) "   Heap Blocks: lossy=191"
9) "   -> Bitmap Index Scan on brin_reserves_bid (cost=0.00..12.13 rows=35000 width=0) (actual time=0.057..0.057 rows=3900 loops=1)"
10) "   Index Cond: (bid = 103)"
11) " -> Memoize (cost=5940.04..6023.20 rows=1 width=25) (actual time=0.287..0.515 rows=1 loops=511)"
12) "   Cache Key: r.sid"
13) "   Cache Mode: logical"
14) "   Hits: 0 Misses: 511 Evictions: 0 Overflows: 0 Memory Usage: 64kB"
15) "   -> Bitmap Heap Scan on sailors s (cost=5940.03..6023.19 rows=1 width=25) (actual time=0.285..0.513 rows=1 loops=511)"
16) "   Recheck Cond: (sid = r.sid)"
17) "   Rows Removed by Index Recheck: 7349"
18) "   Heap Blocks: lossy=31467"
19) "   -> Bitmap Index Scan on brin_sailors_sid (cost=0.00..5940.03 rows=6333 width=0) (actual time=0.003..0.003 rows=620 loops=511)"
20) "   Index Cond: (sid = r.sid)"
21) "Planning Time: 0.374 ms"
    "Execution Time: 276.978 ms"

Flags:
-- SET enable_hashjoin=ON; -- SET enable_material = ON; -- SET enable_mergejoin=ON;
-- SET enable_nestloop=ON; -- SET enable_seqscan=OFF; -- SET enable_indexscan =ON;
-- SET enable_indexonlyscan = ON; -- SET enable_bitmapscan = ON; -- SET enable_sort=ON;
-- SET enable_async_append =ON; -- SET enable_indexscan=ON;

NOTE :

- Cost reduced and excution time
- Cost is greater than BTREE and hash index
- Excution time is better than both

---

5) With Hash and Btree

CREATE INDEX hash_reserves_bid on reserves using hash(bid);
CREATE INDEX btree_sailors_sid on sailors using BTREE(sid);

"QUERY PLAN"

1) " "QUERY PLAN"
2) "Hash Semi Join (cost=435.41..1312.44 rows=498 width=21) (actual time=0.167..3.768 rows=511 loops=1)"
3) " Hash Cond: (s.sid = r.sid)"
4) " -> Index Scan using btree_sailors_sid on sailors s (cost=0.29..821.29 rows=19000 width=25) (actual time=0.004..1.865 rows=19000 loops=1)"
5) " -> Hash (cost=428.89..428.89 rows=498 width=4) (actual time=0.140..0.140 rows=511 loops=1)"
6) "   Buckets: 1024 Batches: 1 Memory Usage: 26kB"
7) "   -> Bitmap Heap Scan on reserves r (cost=11.86..428.89 rows=498 width=4) (actual time=0.022..0.080 rows=511 loops=1)"
8) "   Recheck Cond: (bid = 103)"
9) "   Heap Blocks: exact=15"
10) "   -> Bitmap Index Scan on hash_reserves_bid (cost=0.00..11.73 rows=498 width=0) (actual time=0.017..0.017 rows=511 loops=1)"
11) "   Index Cond: (bid = 103)"
12) "Planning Time: 0.197 ms"
13) "Execution Time: 3.811 ms"

Flags:
-- SET enable_hashjoin=ON; -- SET enable_material = ON; -- SET enable_mergejoin=ON;
-- SET enable_nestloop=ON; -- SET enable_seqscan=OFF; -- SET enable_indexscan =ON;
-- SET enable_indexonlyscan = ON; -- SET enable_bitmapscan = OFF; -- SET enable_sort=ON;
-- SET enable_async_append =ON; -- SET enable_indexscan=ON;

NOTE :

- Cost reduced and excution time
- Cost is better than BTREE ,hash index and BRIN
- Excution time is better than all

IN conclusion:

- MIX between hash and BTREE is the best to use

ALTERNATIVE QUERY :

```
CREATE MATERIALIZED VIEW boatview AS select r.sid from
reserves r where r.bid = 103;
explain analyze
select s.sname
from sailors s
where
s.sid in (select b.sid from boatview b );
```

reason : the most optimized way is to use materialized view and it is better than CTE (With caluse) which gives the same cost

6) Without index

1) "QUERY PLAN"
2) "Hash Semi Join (cost=20000000014.50..20000000577.06 rows=511 width=21) (actual time=0.393..15.702 rows=511 loops=1)"
3) " Hash Cond: (s.sid = b.sid)"
4) " -> Seq Scan on sailors s (cost=10000000000.00..10000000507.00 rows=19000 width=25) (actual time=0.126..5.047 rows=19000 loops=1)"
5) " -> Hash (cost=10000000008.11..10000000008.11 rows=511 width=4) (actual time=0.217..0.219 rows=511 loops=1)"
6) " Buckets: 1024 Batches: 1 Memory Usage: 26kB"
7) " -> Seq Scan on boatview b (cost=10000000000.00..10000000008.11 rows=511 width=4) (actual time=0.012..0.085 rows=511 loops=1)"
8) "Planning Time: 0.839 ms"
9) "Execution Time: 15.791 ms"

Flags:
-- SET enable_hashjoin=ON; -- SET enable_material = ON; -- SET enable_mergejoin=ON;
-- SET enable_nestloop=ON; -- SET enable_seqscan=OFF; -- SET enable_indexscan =ON;
-- SET enable_indexonlyscan = ON; -- SET enable_bitmapscan = OFF; -- SET enable_sort=ON;
-- SET enable_async_append =ON; -- SET enable_indexscan=ON;

NOTE :

- Cost is better than original query

---

7) With B+tree

```
CREATE INDEX BTREE_reserves_bid on reserves using btree(bid);
CREATE INDEX BTREE_reserves_sid on reserves using btree(sid);
CREATE INDEX BTREE_sailors_sid on sailors using btree(sid);
```

"QUERY PLAN"

1) "Merge Join (cost=10000000051.81..10000000923.31 rows=2550 width=21) (actual time=1.556..12.480 rows=511 loops=1)"
2) " Merge Cond: (s.sid = b.sid)"
3) " -> Index Scan using btree_sailors_sid on sailors s (cost=0.29..821.29 rows=19000 width=25) (actual time=0.023..6.525 rows=18098 loops=1)"
4) " -> Sort (cost=10000000051.52..10000000052.02 rows=200 width=4) (actual time=1.430..1.506 rows=511 loops=1)"
5) "   Sort Key: b.sid"
6) "   Sort Method: quicksort Memory: 48kB"
7) "   -> HashAggregate (cost=10000000041.88..10000000043.88 rows=200 width=4) (actual time=0.743..1.016 rows=511 loops=1)"
8) "   Group Key: b.sid"
9) "   Batches: 1 Memory Usage: 89kB"
10) "   -> Seq Scan on boatview b (cost=10000000000.00..10000000035.50 rows=2550 width=4) (actual time=0.045..0.228 rows=511 loops=1)"
11) "Planning Time: 0.473 ms"
12) "Execution Time: 12.602 ms"

Flags:

```
-- SET enable_hashjoin=ON; -- SET enable_material = ON; -- SET enable_mergejoin=ON;
-- SET enable_nestloop=ON; -- SET enable_seqscan=OFF; -- SET enable_indexscan =ON;
-- SET enable_indexonlyscan = ON; -- SET enable_bitmapscan = OFF; -- SET enable_sort=ON;
 -- SET enable_async_append =ON; -- SET enable_indexscan=ON;
```

NOTE :

- some indicies was created inside view for optimization
- Cost reduced and execution time

---

8) With hash

```
-- CREATE INDEX Hash_reserves_bid on reserves using hash(bid);
-- CREATE INDEX Hash_reserves_sid on reserves using hash(sid);
-- CREATE INDEX Hash_sailors_sid on sailors using hash(sid);
```

"QUERY PLAN"

1) "Nested Loop (cost=10000000009.39..10000001424.55 rows=511 width=21) (actual time=0.136..0.821 rows=511 loops=1)"
2) " -> HashAggregate (cost=10000000009.39..10000000014.50 rows=511 width=4) (actual time=0.127..0.247 rows=511 loops=1)"
3) "   Group Key: b.sid"
4) "   Batches: 1 Memory Usage: 73kB"
5) "   -> Seq Scan on boatview b (cost=10000000000.00..10000000008.11 rows=511 width=4) (actual time=0.008..0.036 rows=511 loops=1)"

6) " -> Index Scan using hash_sailors_sid on sailors s (cost=0.00..2.75 rows=1 width=25) (actual time=0.001..0.001 rows=1 loops=511)"
7) "   Index Cond: (sid = b.sid)"
8) "Planning Time: 1.031 ms"
9) "Execution Time: 0.865 ms"

Flags:
-- SET enable_hashjoin=ON; -- SET enable_material = ON; -- SET enable_mergejoin=ON;
-- SET enable_nestloop=ON; -- SET enable_seqscan=OFF; -- SET enable_indexscan =ON;
-- SET enable_indexonlyscan = ON; -- SET enable_bitmapscan = OFF; -- SET enable_sort=ON;
-- SET enable_async_append =ON; -- SET enable_indexscan=ON;

NOTE :

- some indicies was created inside view for optimization
- Cost reduced and execution time
- Btree is better than hash index

---

9) WITH BRIN

-- CREATE INDEX BRIN_reserves_bid on reserves using BRIN(bid);
-- CREATE INDEX BRIN_sailors_sid on sailors using BRIN(sid);
-- CREATE INDEX BRIN_reserves_sid on reserves using BRIN(sid);

"QUERY PLAN"

1) "Nested Loop (cost=10000006141.42..10003175984.12 rows=511 width=21) (actual time=2.411..273.747 rows=511 loops=1)"
2) " -> HashAggregate (cost=10000000009.39..10000000014.50 rows=511 width=4) (actual time=0.647..0.833 rows=511 loops=1)"
3) "   Group Key: b.sid"
4) "   Batches: 1 Memory Usage: 73kB"
5) "   -> Seq Scan on boatview b (cost=10000000000.00..10000000008.11 rows=511 width=4) (actual time=0.025..0.184 rows=511 loops=1)"
6) " -> Bitmap Heap Scan on sailors s (cost=6132.03..6215.19 rows=1 width=25) (actual time=0.296..0.532 rows=1 loops=511)"
7) "   Recheck Cond: (sid = b.sid)"
8) "   Rows Removed by Index Recheck: 7349"
9) "   Heap Blocks: lossy=31467"
10) "   -> Bitmap Index Scan on brin_sailors_sid (cost=0.00..6132.03 rows=6333 width=0) (actual time=0.004..0.004 rows=620 loops=511)"
11) "   Index Cond: (sid = b.sid)"
12) "Planning Time: 0.325 ms"
13) "Execution Time: 273.915 ms"

NOTE : BRIN increases execution time decreases cost

Flags:
-- SET enable_hashjoin=ON; -- SET enable_material = ON; -- SET enable_mergejoin=ON;

-- SET enable_nestloop=ON; -- SET enable_seqscan=OFF; -- SET enable_indexscan =ON;
-- SET enable_indexonlyscan = ON; -- SET enable_bitmapscan = ON; -- SET enable_sort=ON;
-- SET enable_async_append =ON; -- SET enable_indexscan=ON;

NOTE :

- some indicies was created inside view for optimization
- Cost reduced and execution time
- BTREE and hash are better than BRIN

---

10) With Hash and B+tree

CREATE INDEX hash_reserves_bid on reserves using hash(bid);
CREATE INDEX btree_sailors_sid on sailors using BTREE(sid);

"QUERY PLAN"

1) "Merge Semi Join (cost=10000000036.05..10000000866.27 rows=511 width=21) (actual time=0.664..15.573 rows=511 loops=1)"
2) " Merge Cond: (s.sid = b.sid)"
3) " -> Index Scan using btree_sailors_sid on sailors s (cost=0.29..821.29 rows=19000 width=25) (actual time=0.020..8.780 rows=18098 loops=1)"
4) " -> Sort (cost=10000000031.10..10000000032.38 rows=511 width=4) (actual time=0.541..0.641 rows=511 loops=1)"
5) "   Sort Key: b.sid"
6) "   Sort Method: quicksort Memory: 48kB"
7) "   -> Seq Scan on boatview b (cost=10000000000.00..10000000008.11 rows=511 width=4) (actual time=0.026..0.181 rows=511 loops=1)"
8) "Planning Time: 0.767 ms"
9) "Execution Time: 15.691 ms"

Flags:
-- SET enable_hashjoin=ON; -- SET enable_material = ON; -- SET enable_mergejoin=ON;
-- SET enable_nestloop=ON; -- SET enable_seqscan=OFF; -- SET enable_indexscan =ON;
-- SET enable_indexonlyscan = ON; -- SET enable_bitmapscan = OFF; -- SET enable_sort=ON;
-- SET enable_async_append =ON; -- SET enable_indexscan=ON;

NOTE :

- some indicies was created inside view for optimization
- Cost reduced and execution time
- The mix between hash and btree is the best to use

IN conclusion:

- MIX between hash and BTREE is the best to use

Query 8

1) Without index

"QUERY PLAN"

1) "Hash Semi Join (cost=30000000936.37..30000001500.90 rows=688 width=21) (actual time=67.033..74.533 rows=697 loops=1)"
2) " Hash Cond: (s.sid = r.sid)"
3) " -> Seq Scan on sailors s (cost=10000000000.00..10000000507.00 rows=19000 width=25) (actual time=0.180..3.019 rows=19000 loops=1)"
4) " -> Hash (cost=20000000927.77..20000000927.77 rows=688 width=4) (actual time=66.838..66.841 rows=697 loops=1)"
5) " Buckets: 1024 Batches: 1 Memory Usage: 33kB"
6) " -> Hash Semi Join (cost=20000000088.24..20000000927.77 rows=688 width=4) (actual time=1.290..66.164 rows=697 loops=1)"
7) " Hash Cond: (r.bid = b.bid)"
8) " -> Seq Scan on reserves r (cost=10000000000.00..10000000740.00 rows=35000 width=8) (actual time=0.009..43.327 rows=35000 loops=1)"
9) " -> Hash (cost=10000000087.50..10000000087.50 rows=59 width=4) (actual time=1.270..1.272 rows=59 loops=1)"
10) " Buckets: 1024 Batches: 1 Memory Usage: 11kB"
11) " -> Seq Scan on boat b (cost=10000000000.00..10000000087.50 rows=59 width=4) (actual time=0.040..1.227 rows=59 loops=1)"
12) " Filter: (color = 'red'::bpchar)"
13) " Rows Removed by Filter: 2941"
14) "Planning Time: 0.554 ms"
15) "Execution Time: 74.662 ms"

```
Flags :  SET enable_hashjoin=ON;
-- SET enable_material = ON;            -- SET enable_mergejoin=ON;      -- SET enable_sort=ON;
-- SET enable_nestloop=ON;                SET enable_seqscan=OFF;         -- SET enable_indexscan=ON;
-- SET enable_indexscan =ON;            -- SET enable_indexonlyscan = ON;
-- SET enable_bitmapscan = OFF;         -- SET enable_async_append =ON;
```

---

2) WITH BTREE

```
CREATE INDEX BTREE_reserves_bid on reserves using Btree(bid);

CREATE INDEX BTREE_reserves_sid on reserves using Btree(sid);

CREATE INDEX BTREE_sailors_sid on sailors using Btree(sid);

CREATE INDEX BTREE_sailors_sname on sailors using Btree(sname);

CREATE INDEX BTREE_boat_bid on boat using Btree(bid);

CREATE INDEX BTREE_boat_color on boat using Btree(color);
```

"QUERY PLAN"

1) "Nested Loop (cost=1733.31..2001.72 rows=688 width=21) (actual time=1.922..6.249 rows=697 loops=1)"

2) " -> HashAggregate (cost=1733.02..1739.90 rows=688 width=4) (actual time=1.910..2.314 rows=697 loops=1)"
3) " Group Key: r.sid"
4) " Batches: 1 Memory Usage: 105kB"
5) " -> Nested Loop (cost=0.57..1731.30 rows=688 width=4) (actual time=0.028..1.354 rows=697 loops=1)"
6) " -> Index Scan using btree_boat_color on boat b (cost=0.28..71.47 rows=59 width=4) (actual time=0.015..0.049 rows=59 loops=1)"
7) " Index Cond: (color = 'red'::bpchar)"
8) " -> Index Scan using btree_reserves_bid on reserves r (cost=0.29..28.01 rows=12 width=8) (actual time=0.004..0.016 rows=12 loops=59)"
9) " Index Cond: (bid = b.bid)"
10) " -> Index Scan using btree_sailors_sid on sailors s (cost=0.29..0.38 rows=1 width=25) (actual time=0.004..0.004 rows=1 loops=697)"
11) " Index Cond: (sid = r.sid)"
12) "Planning Time: 0.781 ms"
13) "Execution Time: 6.454 ms"

```
Flags :  SET enable_hashjoin=ON;
-- SET enable_material = ON;            -- SET enable_mergejoin=ON;      -- SET enable_sort=ON;
-- SET enable_nestloop=ON;                SET enable_seqscan=OFF;         -- SET enable_indexscan=ON;
-- SET enable_indexscan =ON;            -- SET enable_indexonlyscan = ON;
-- SET enable_bitmapscan = OFF;         -- SET enable_async_append =ON;
```

NOTE :

- some indicies was used in joins
- Cost reduced and execution time

---

3) With hash index

```
CREATE INDEX HASH_reserves_bid on reserves using hash(bid);

-- CREATE INDEX HASH_reserves_sid on reserves using hash(sid);

-- CREATE INDEX HASH_sailors_sid on sailors using hash(sid);

-- CREATE INDEX HASH_sailors_sname on sailors using hash(sname);

-- CREATE INDEX HASH_boat_bid on boat using hash(bid);

-- CREATE INDEX HASH_boat_color on boat using hash(color);
```

"QUERY PLAN"

1) "Nested Loop (cost=1862.96..1944.71 rows=688 width=21) (actual time=2.455..4.769 rows=697 loops=1)"
2) " -> HashAggregate (cost=1862.96..1869.84 rows=688 width=4) (actual time=2.444..2.687 rows=697 loops=1)"
3) " Group Key: r.sid"
4) " Batches: 1 Memory Usage: 105kB"
5) " -> Nested Loop (cost=157.18..1861.24 rows=688 width=4) (actual time=0.155..1.810 rows=697 loops=1)"
6) " -> HashAggregate (cost=157.18..157.77 rows=59 width=4) (actual time=0.140..0.179 rows=59 loops=1)"

7) " Group Key: b.bid"
8) " Batches: 1 Memory Usage: 24kB"
9) " -> Index Scan using hash_boat_color on boat b (cost=0.00..157.03 rows=59 width=4) (actual time=0.028..0.080 rows=59 loops=1)"
10) " Index Cond: (color = 'red'::bpchar)"
11) " -> Index Scan using hash_reserves_bid on reserves r (cost=0.00..28.75 rows=12 width=8) (actual time=0.005..0.022 rows=12 loops=59)"
12) " Index Cond: (bid = b.bid)"
13) " -> Index Scan using hash_sailors_sid on sailors s (cost=0.00..0.10 rows=1 width=25) (actual time=0.002..0.002 rows=1 loops=697)"
14) " Index Cond: (sid = r.sid)"
15) "Planning Time: 0.561 ms"
16) "Execution Time: 4.932 ms"

```
Flags :  SET enable_hashjoin=ON;
-- SET enable_material = ON;            -- SET enable_mergejoin=ON;      -- SET enable_sort=ON;
-- SET enable_nestloop=ON;                SET enable_seqscan=OFF;         -- SET enable_indexscan=ON;
-- SET enable_indexscan =ON;            -- SET enable_indexonlyscan = ON;
-- SET enable_bitmapscan = OFF;         -- SET enable_async_append =ON;
```

NOTE :

- some indicies were used in joins
- Cost reduced and execution time
- BTREE is better than hash

---

4) With BRIN index

```
CREATE INDEX BRIN_reserves_bid on reserves using brin(bid);
-- CREATE INDEX BRIN_reserves_sid on reserves using brin(sid);
-- CREATE INDEX BRIN_sailors_sid on sailors using brin(sid);
-- CREATE INDEX BRIN_sailors_sname on sailors using brin(sname);
-- CREATE INDEX BRIN_boat_bid on boat using brin(bid);
-- CREATE INDEX BRIN_boat_color on boat using brin(color);
```

"QUERY PLAN"

1) "Nested Loop (cost=295243.35..155652967.07 rows=688 width=21) (actual time=165.529..621.739 rows=697 loops=1)"
2) " -> HashAggregate (cost=69187.32..69194.20 rows=688 width=4) (actual time=165.362..165.577 rows=697 loops=1)"
3) " Group Key: r.sid"
4) " Batches: 1 Memory Usage: 105kB"
5) " -> Nested Loop (cost=807.83..69185.60 rows=688 width=4) (actual time=1.762..165.005 rows=697 loops=1)"
6) " -> HashAggregate (cost=99.69..100.28 rows=59 width=4) (actual time=1.673..1.710 rows=59 loops=1)"
7) " Group Key: b.bid"
8) " Batches: 1 Memory Usage: 24kB"
9) " -> Bitmap Heap Scan on boat b (cost=12.05..99.55 rows=59 width=4) (actual time=0.088..1.616 rows=59 loops=1)"
10) " Recheck Cond: (color = 'red'::bpchar)"
11) " Rows Removed by Index Recheck: 2941"
12) " Heap Blocks: lossy=25"

```
13) "                  ->  Bitmap Index Scan on brin_boat_color  (cost=0.00..12.03 rows=3000 width=0) (actual
    time=0.029..0.030 rows=500 loops=1)"
14) "                        Index Cond: (color = 'red'::bpchar)"
15) "          ->  Bitmap Heap Scan on reserves r  (cost=708.13..1170.82 rows=12 width=8) (actual
    time=0.016..2.762 rows=12 loops=59)"
16) "                Recheck Cond: (bid = b.bid)"
17) "                Rows Removed by Index Recheck: 34438"
18) "                Heap Blocks: lossy=11093"
19) "                ->  Bitmap Index Scan on brin_reserves_bid  (cost=0.00..708.13 rows=35000 width=0) (actual
    time=0.009..0.009 rows=3857 loops=59)"
20) "                      Index Cond: (bid = b.bid)"
21) "  ->  Bitmap Heap Scan on sailors s  (cost=226056.03..226139.19 rows=1 width=25) (actual time=0.303..0.653
    rows=1 loops=697)"
22) "       Recheck Cond: (sid = r.sid)"
23) "       Rows Removed by Index Recheck: 10241"
24) "       Heap Blocks: lossy=59722"
25) "       ->  Bitmap Index Scan on brin_sailors_sid  (cost=0.00..226056.03 rows=6333 width=0) (actual
    time=0.004..0.004 rows=1057 loops=697)"
26) "             Index Cond: (sid = r.sid)"
27) "Planning Time: 0.591 ms"
28) "Execution Time: 621.966 ms"
```

NOTE : BRIN increases excution time but decreases cost

```
Flags :  SET enable_hashjoin=ON;
-- SET enable_material = ON;              -- SET enable_mergejoin=ON;      -- SET enable_sort=ON;
-- SET enable_nestloop=ON;                  SET enable_seqscan=OFF;         -- SET enable_indexscan=ON;
-- SET enable_indexscan =ON;             -- SET enable_indexonlyscan = ON;
-- SET enable_bitmapscan = ON;          -- SET enable_async_append =ON;
```

NOTE :

- some indicies were used in joins
- Cost reduced
- Excution time incresed
- BTREE and hash are better than brin

---

5) With BRIN and hash

```
-- CREATE INDEX BTREE_reserves_bid on reserves using BTree(bid);

-- CREATE INDEX BTREE_reserves_sid on reserves using btree(sid);

-- CREATE INDEX BTREE_sailors_sid on sailors using btree(sid);

-- CREATE INDEX BTREE_sailors_sname on sailors using btree(sname);

-- CREATE INDEX hash_boat_bid on boat using hash(bid);

-- CREATE INDEX hash_boat_color on boat using hash(color);
```

"QUERY PLAN"

```
1)  "Nested Loop  (cost=1819.61..2097.12 rows=688 width=21) (actual time=2.306..7.056 rows=697 loops=1)"
2)  "  ->  HashAggregate  (cost=1819.32..1826.20 rows=688 width=4) (actual time=2.294..2.704 rows=697 loops=1)"
3)  "        Group Key: r.sid"
4)  "        Batches: 1  Memory Usage: 105kB"
5)  "        ->  Nested Loop  (cost=157.47..1817.60 rows=688 width=4) (actual time=0.151..1.666 rows=697 loops=1)"
6)  "              ->  HashAggregate  (cost=157.18..157.77 rows=59 width=4) (actual time=0.136..0.176 rows=59 loops=1)"
7)  "                    Group Key: b.bid"
8)  "                    Batches: 1  Memory Usage: 24kB"
```

```
9)  "                ->  Index Scan using hash_boat_color on boat b  (cost=0.00..157.03 rows=59 width=4) (actual time=0.023..0.076 rows=59
    loops=1)"
10) "                      Index Cond: (color = 'red'::bpchar)"
11) "           ->  Index Scan using btree_reserves_bid on reserves r  (cost=0.29..28.01 rows=12 width=8) (actual time=0.004..0.019 rows=12
    loops=59)"
12) "                 Index Cond: (bid = b.bid)"
13) "  ->  Index Scan using btree_sailors_sid on sailors s  (cost=0.29..0.38 rows=1 width=25) (actual time=0.004..0.005 rows=1 loops=697)"
14) "        Index Cond: (sid = r.sid)"
15) "Planning Time: 0.814 ms"
16) "Execution Time: 7.284 ms"
```

```
Flags :  SET enable_hashjoin=ON;
-- SET enable_material = ON;              -- SET enable_mergejoin=ON;      -- SET enable_sort=ON;
-- SET enable_nestloop=ON;                  SET enable_seqscan=OFF;         -- SET enable_indexscan=ON;
-- SET enable_indexscan =ON;              -- SET enable_indexonlyscan = ON;
-- SET enable_bitmapscan = OFF;           -- SET enable_async_append =ON;
```

NOTE :

- some indicies were used in joins
- Cost reduced and execution time
- Hash and btree are better than this mix

IN Conclusion: hash is the best index to use in this query

OPTIMIZED QUERY : **QUERY 8**

```
-- CREATE MATERIALIZED VIEW mymatview AS select b.bid from boat b where b.color like 'red%';
-- explain analyze
-- select s.sname
-- from sailors s inner join reserves r on s.sid =r.sid
-- where r.bid IN (select bid from mymatview)
```

NOTE :--

- CREATED MATERIALIZED VIEW on boat replaced by sub query and use inner join instead of subquery

We used a view after done all possible optimization on query

- With clause reduced the cost but using view is the best optimized way

6) Without index

```
"QUERY PLAN"
1)  "Hash Join  (cost=30000000850.48..30000001435.62 rows=689 width=21) (actual time=17.754..24.690 rows=697 loops=1)"
2)  "  Hash Cond: (s.sid = r.sid)"
3)  "  ->  Seq Scan on sailors s  (cost=10000000000.00..10000000507.00 rows=19000 width=25) (actual time=0.182..2.476
    rows=19000 loops=1)"
4)  "  ->  Hash  (cost=20000000841.87..20000000841.87 rows=689 width=4) (actual time=17.557..17.560 rows=697 loops=1)"
5)  "        Buckets: 1024  Batches: 1  Memory Usage: 33kB"
6)  "        ->  Hash Semi Join  (cost=20000000002.33..20000000841.87 rows=689 width=4) (actual time=0.088..17.253 rows=697
    loops=1)"
7)  "              Hash Cond: (r.bid = mymatview.bid)"
8)  "              ->  Seq Scan on reserves r  (cost=10000000000.00..10000000740.00 rows=35000 width=8) (actual time=0.009..7.220
    rows=35000 loops=1)"
9)  "              ->  Hash  (cost=10000000001.59..10000000001.59 rows=59 width=4) (actual time=0.065..0.067 rows=59 loops=1)"
10) "                    Buckets: 1024  Batches: 1  Memory Usage: 11kB"
11) "                    ->  Seq Scan on mymatview  (cost=10000000000.00..10000000001.59 rows=59 width=4) (actual time=0.010..0.026
    rows=59 loops=1)"
```

12) "Planning Time: 0.394 ms"
13) "Execution Time: 24.807 ms"

Flags : SET enable_hashjoin=ON;
-- SET enable_material = ON; -- SET enable_mergejoin=ON; -- SET enable_sort=ON;
-- SET enable_nestloop=ON; SET enable_seqscan=OFF; -- SET enable_indexscan=ON;
-- SET enable_indexscan =ON; -- SET enable_indexonlyscan = ON;
-- SET enable_bitmapscan = OFF; -- SET enable_async_append =ON;

NOTE :

- Cost reduced and execution time

---

7) With BTREE

```
-- CREATE INDEX BTREE_reserves_bid on reserves using BTree(bid);

-- CREATE INDEX BTREE_reserves_sid on reserves using btree(sid);

-- CREATE INDEX BTREE_sailors_sid on sailors using btree(sid);

-- CREATE INDEX BTREE_sailors_sname on sailors using btree(sname);

-- CREATE INDEX BTREE_boat_bid on boat using btree(bid);

-- CREATE INDEX BTREE_boat_color on boat using btree(color);
```

"QUERY PLAN"
1) "Hash Join (cost=10000000065.68..10000000964.82 rows=689 width=21) (actual time=0.974..10.479 rows=697 loops=1)"
2) " Hash Cond: (s.sid = r.sid)"
3) " -> Index Scan using btree_sailors_sid on sailors s (cost=0.29..821.29 rows=19000 width=25) (actual time=0.012..5.365 rows=19000 loops=1)"
4) " -> Hash (cost=10000000056.78..10000000056.78 rows=689 width=4) (actual time=0.952..0.955 rows=697 loops=1)"
5) " Buckets: 1024 Batches: 1 Memory Usage: 33kB"
6) " -> Merge Semi Join (cost=10000000003.62..10000000056.78 rows=689 width=4) (actual time=0.046..0.741 rows=697 loops=1)"
7) " Merge Cond: (r.bid = mymatview.bid)"
8) " -> Index Scan using btree_reserves_bid on reserves r (cost=0.29..2244.90 rows=35000 width=8) (actual time=0.007..0.436 rows=698 loops=1)"
9) " -> Sort (cost=10000000003.33..10000000003.47 rows=59 width=4) (actual time=0.033..0.042 rows=59 loops=1)"
10) " Sort Key: mymatview.bid"
11) " Sort Method: quicksort Memory: 27kB"
12) " -> Seq Scan on mymatview (cost=10000000000.00..10000000001.59 rows=59 width=4) (actual time=0.009..0.017 rows=59 loops=1)"
13) "Planning Time: 0.528 ms"
14) "Execution Time: 10.592 ms"

Flags : SET enable_hashjoin=ON;
-- SET enable_material = ON; -- SET enable_mergejoin=ON; -- SET enable_sort=ON;
-- SET enable_nestloop=ON; SET enable_seqscan=OFF; -- SET enable_indexscan=ON;
-- SET enable_indexscan =ON; -- SET enable_indexonlyscan = ON;
-- SET enable_bitmapscan = OFF; -- SET enable_async_append =ON;

NOTE :

- Some indicies used in joins
- Cost reduced and execution time

---

8) With hash index

```
-- CREATE INDEX HASH_reserves_bid on reserves using hash(bid);
-- CREATE INDEX HASH_reserves_sid on reserves using hash(sid);
```

```
-- CREATE INDEX HASH_sailors_sid on sailors using hash(sid);
-- CREATE INDEX HASH_sailors_sname on sailors using hash(sname);
-- CREATE INDEX HASH_boat_bid on boat using hash(bid);
-- CREATE INDEX HASH_boat_color on boat using hash(color);
NOTE : we use indicies on primary keys to reudce cost of join
"QUERY PLAN"
    1) "QUERY PLAN"
    2) "Nested Loop  (cost=1.74..1754.90 rows=689 width=21) (actual time=0.135..3.912 rows=697 loops=1)"
    3) "  ->  Nested Loop  (cost=1.74..1705.80 rows=689 width=4) (actual time=0.124..1.351 rows=697 loops=1)"
    4) "        ->  HashAggregate  (cost=1.74..2.33 rows=59 width=4) (actual time=0.102..0.132 rows=59 loops=1)"
    5) "              Group Key: mymatview.bid"
    6) "              Batches: 1  Memory Usage: 24kB"
    7) "              ->  Seq Scan on mymatview  (cost=0.00..1.59 rows=59 width=4) (actual time=0.021..0.038 rows=59 loops=1)"
    8) "        ->  Index Scan using hash_reserves_bid on reserves r  (cost=0.00..28.75 rows=12 width=8) (actual time=0.004..0.016
       rows=12 loops=59)"
    9) "              Index Cond: (bid = mymatview.bid)"
    10) "  ->  Index Scan using hash_sailors_sid on sailors s  (cost=0.00..0.06 rows=1 width=25) (actual time=0.002..0.003 rows=1
       loops=697)"
    11) "        Index Cond: (sid = r.sid)"
    12) "Planning Time: 0.505 ms"
    13) "Execution Time: 4.073 ms"
Flags :  SET enable_hashjoin=OFF;
-- SET enable_material = ON;            -- SET enable_mergejoin=ON;      -- SET enable_sort=ON;
-- SET enable_nestloop=ON;                SET enable_seqscan=OFF;         -- SET enable_indexscan=ON;
-- SET enable_indexscan =ON;            -- SET enable_indexonlyscan = ON;
-- SET enable_bitmapscan = OFF;         -- SET enable_async_append =ON;
```

NOTE :

- Some indicies used in joins
- Cost reduced and execution time
- Hash index is better than BTREE

---

9) With BRIN index

```
-- CREATE INDEX BRIN_reserves_bid on reserves using brin(bid);
-- CREATE INDEX BRIN_reserves_sid on reserves using brin(sid);
-- CREATE INDEX BRIN_sailors_sid on sailors using brin(sid);
-- CREATE INDEX BRIN_sailors_sname on sailors using brin(sname);
-- CREATE INDEX BRIN_boat_bid on boat using brin(bid);
-- CREATE INDEX BRIN_boat_color on boat using brin(color);
```

"QUERY PLAN"

```
1)  "Nested Loop  (cost=10000420709.90..10289506415.72 rows=689 width=21) (actual time=0.296..624.886 rows=697 loops=1)"
2)  "  ->  Nested Loop  (cost=10000000709.87..10000069087.64 rows=689 width=4) (actual time=0.200..141.544 rows=697 loops=1)"
3)  "        ->  HashAggregate  (cost=10000000001.74..10000000002.33 rows=59 width=4) (actual time=0.102..0.201 rows=59 loops=1)"
4)  "              Group Key: mymatview.bid"
5)  "              Batches: 1  Memory Usage: 24kB"
6)  "              ->  Seq Scan on mymatview  (cost=10000000000.00..10000000001.59 rows=59 width=4) (actual time=0.014..0.032 rows=59
    loops=1)"
7)  "        ->  Bitmap Heap Scan on reserves r  (cost=708.13..1170.82 rows=12 width=8) (actual time=0.014..2.391 rows=12 loops=59)"
8)  "              Recheck Cond: (bid = mymatview.bid)"
9)  "              Rows Removed by Index Recheck: 34438"
10) "              Heap Blocks: lossy=11093"
11) "              ->  Bitmap Index Scan on brin_reserves_bid  (cost=0.00..708.13 rows=35000 width=0) (actual time=0.008..0.008 rows=3857
    loops=59)"
12) "                    Index Cond: (bid = mymatview.bid)"
13) "  ->  Bitmap Heap Scan on sailors s  (cost=420000.03..420083.19 rows=1 width=25) (actual time=0.320..0.691 rows=1 loops=697)"
14) "        Recheck Cond: (sid = r.sid)"
15) "        Rows Removed by Index Recheck: 10241"
16) "        Heap Blocks: lossy=59722"
```

17) " -> Bitmap Index Scan on brin_sailors_sid (cost=0.00..420000.03 rows=6333 width=0) (actual time=0.004..0.004 rows=1057 loops=697)"
18) " Index Cond: (sid = r.sid)"
19) "Planning Time: 0.523 ms"
20) "Execution Time: 625.161 ms"

Flags : SET enable_hashjoin=OFF;
-- SET enable_material = ON; -- SET enable_mergejoin=ON; -- SET enable_sort=ON;
-- SET enable_nestloop=ON; SET enable_seqscan=OFF; -- SET enable_indexscan=ON;
-- SET enable_indexscan =ON; -- SET enable_indexonlyscan = ON;
-- SET enable_bitmapscan = ON; -- SET enable_async_append =ON;

NOTE :

- Some indicies used in joins
- Cost reduced and execution time increased
- Hash and BTREE are better than BRIN

---

10) WITH BTREE and hash

```
-- CREATE INDEX BTREE_reserves_sid on reserves using BTREE(sid);
CREATE INDEX HASH_sailors_sid on sailors using BTREE(sid);
-- CREATE INDEX HASH_boat_color on boat using hash(color);
```

"QUERY PLAN"

1) "Hash Semi Join (cost=10000000002.95..10000003683.28 rows=689 width=21) (actual time=0.073..29.164 rows=697 loops=1)"
2) " Hash Cond: (r.bid = mymatview.bid)"
3) " -> Merge Join (cost=0.62..3581.41 rows=35000 width=25) (actual time=0.027..22.630 rows=35000 loops=1)"
4) " Merge Cond: (s.sid = r.sid)"
5) " -> Index Scan using hash_sailors_sid on sailors s (cost=0.29..821.29 rows=19000 width=25) (actual time=0.011..3.168 rows=19000 loops=1)"
6) " -> Index Scan using btree_reserves_sid on reserves r (cost=0.29..2275.12 rows=35000 width=8) (actual time=0.007..9.404 rows=35000 loops=1)"
7) " -> Hash (cost=10000000001.59..10000000001.59 rows=59 width=4) (actual time=0.036..0.037 rows=59 loops=1)"
8) " Buckets: 1024 Batches: 1 Memory Usage: 11kB"
9) " -> Seq Scan on mymatview (cost=10000000000.00..10000000001.59 rows=59 width=4) (actual time=0.009..0.017 rows=59 loops=1)"
10) "Planning Time: 0.864 ms"
11) "Execution Time: 29.249 ms"

-- SET enable_material = ON; -- SET enable_mergejoin=ON; -- SET enable_sort=ON;
-- SET enable_nestloop=ON; SET enable_seqscan=OFF; -- SET enable_indexscan=ON;
-- SET enable_indexscan =ON; -- SET enable_indexonlyscan = ON;
-- SET enable_bitmapscan = ON; -- SET enable_async_append =ON;

NOTE :

- Cost reduced and execution time
- Hash is better than this

IN CONCLUSION : hash index is the best to use

Query 9

1) Without index

"QUERY PLAN"

1) "Hash Join (cost=60000004078.37..60000004657.03 rows=41 width=21) (actual time=33.023..36.645 rows=348 loops=1)"
2) " Hash Cond: (s.sid = r.sid)"
3) " -> Seq Scan on sailors s (cost=10000000000.00..10000000507.00 rows=19000 width=25) (actual time=0.116..1.414 rows=19000 loops=1)"
4) " -> Hash (cost=50000004077.86..50000004077.86 rows=41 width=12) (actual time=32.898..32.905 rows=348 loops=1)"
5) " Buckets: 1024 Batches: 1 Memory Usage: 23kB"
6) " -> Merge Join (cost=50000004068.35..50000004077.86 rows=41 width=12) (actual time=32.556..32.844 rows=348 loops=1)"
7) " Merge Cond: (r.sid = s2.sid)"
8) " -> Sort (cost=20000000998.79..20000001000.51 rows=688 width=4) (actual time=14.510..14.546 rows=697 loops=1)"
9) " Sort Key: r.sid"
10) " Sort Method: quicksort Memory: 57kB"
11) " -> Hash Join (cost=20000000088.24..20000000966.37 rows=688 width=4) (actual time=0.615..14.233 rows=697 loops=1)"
12) " Hash Cond: (r.bid = b.bid)"
13) " -> Seq Scan on reserves r (cost=10000000000.00..10000000740.00 rows=35000 width=8) (actual time=0.005..5.653 rows=35000 loops=1)"
14) " -> Hash (cost=10000000087.50..10000000087.50 rows=59 width=4) (actual time=0.604..0.605 rows=59 loops=1)"
15) " Buckets: 1024 Batches: 1 Memory Usage: 11kB"
16) " -> Seq Scan on boat b (cost=10000000000.00..10000000087.50 rows=59 width=4) (actual time=0.024..0.580 rows=59 loops=1)"
17) " Filter: (color = 'red'::bpchar)"
18) " Rows Removed by Filter: 2941"
19) " -> Sort (cost=30000003069.55..30000003072.38 rows=1132 width=8) (actual time=18.039..18.097 rows=1126 loops=1)"
20) " Sort Key: s2.sid"
21) " Sort Method: quicksort Memory: 103kB"
22) " -> HashAggregate (cost=30000003000.81..30000003012.13 rows=1132 width=8) (actual time=17.648..17.773 rows=1164 loops=1)"
23) " Group Key: s2.sid"
24) " Batches: 1 Memory Usage: 129kB"
25) " -> Merge Join (cost=30000002886.00..30000002997.98 rows=1132 width=8) (actual time=14.943..17.413 rows=1164 loops=1)"
26) " Merge Cond: (r2.sid = s2.sid)"
27) " -> Sort (cost=20000001028.70..20000001031.53 rows=1132 width=4) (actual time=11.864..11.923 rows=1164 loops=1)"
28) " Sort Key: r2.sid"
29) " Sort Method: quicksort Memory: 103kB"
30) " -> Hash Join (cost=20000000088.71..20000000971.28 rows=1132 width=4) (actual time=0.649..11.533 rows=1164 loops=1)"
31) " Hash Cond: (r2.bid = b2.bid)"
32) " -> Seq Scan on reserves r2 (cost=10000000000.00..10000000740.00 rows=35000 width=8) (actual time=0.015..4.549 rows=35000 loops=1)"
33) " -> Hash (cost=10000000087.50..10000000087.50 rows=97 width=4) (actual time=0.587..0.588 rows=97 loops=1)"
34) " Buckets: 1024 Batches: 1 Memory Usage: 12kB"
35) " -> Seq Scan on boat b2 (cost=10000000000.00..10000000087.50 rows=97 width=4) (actual time=0.035..0.553 rows=97 loops=1)"
36) " Filter: (color = 'green'::bpchar)"
37) " Rows Removed by Filter: 2903"
38) " -> Sort (cost=10000001857.30..10000001904.80 rows=19000 width=4) (actual time=3.072..3.912 rows=18094 loops=1)"
39) " Sort Key: s2.sid"
40) " Sort Method: quicksort Memory: 1659kB"
41) " -> Seq Scan on sailors s2 (cost=10000000000.00..10000000507.00 rows=19000 width=4) (actual time=0.063..1.509 rows=19000 loops=1)"
42) "Planning Time: 0.466 ms"
43) "Execution Time: 36.759 ms"

FLAGS:
SET enable_hashjoin=ON; SET enable_material = ON; SET enable_mergejoin=ON; SET enable_nestloop=ON; SET enable_seqscan=OFF;
SET enable_indexscan =ON; SET enable_indexonlyscan = ON; SET enable_bitmapscan = OFF; SET enable_async_append =ON;
SET enable_sort=ON; SET enable_indexscan=ON;

2) WITH BTREE

```
-- CREATE INDEX BTREE_reserves_bid on reserves using btree(bid);

-- CREATE INDEX BTREE_reserves_sid on reserves using BTREE(sid);

-- CREATE INDEX BTREE_sailors_sid on sailors using BTREE(sid);

-- CREATE INDEX BTREE_sailor_sname on sailors using BTREE(sname);

-- CREATE INDEX BTREE_boat_bid on boat using btree(bid);

-- CREATE INDEX BTREE_boat_color on boat using btree(color);
```

"QUERY PLAN"

1) "Nested Loop (cost=1.72..2698.67 rows=41 width=21) (actual time=0.131..18.079 rows=348 loops=1)"
2) " -> Nested Loop (cost=0.57..1731.62 rows=688 width=4) (actual time=0.034..1.207 rows=697 loops=1)"
3) " -> Index Scan using btree_boat_color on boat b (cost=0.28..71.47 rows=59 width=4) (actual time=0.017..0.050 rows=59 loops=1)"
4) " Index Cond: (color = 'red'::bpchar)"
5) " -> Index Scan using btree_reserves_bid on reserves r (cost=0.29..28.02 rows=12 width=8) (actual time=0.003..0.014 rows=12 loops=59)"
6) " Index Cond: (bid = b.bid)"
7) " -> Nested Loop Semi Join (cost=1.15..1.40 rows=1 width=33) (actual time=0.023..0.023 rows=0 loops=697)"
8) " -> Index Scan using btree_sailors_sid on sailors s (cost=0.29..0.35 rows=1 width=25) (actual time=0.003..0.004 rows=1 loops=697)"
9) " Index Cond: (sid = r.sid)"
10) " -> Nested Loop (cost=0.86..1.04 rows=1 width=8) (actual time=0.018..0.018 rows=0 loops=697)"
11) " -> Nested Loop (cost=0.58..0.72 rows=1 width=12) (actual time=0.008..0.009 rows=2 loops=697)"
12) " Join Filter: (s2.sid = r2.sid)"
13) " -> Index Only Scan using btree_sailors_sid on sailors s2 (cost=0.29..0.32 rows=1 width=4) (actual time=0.003..0.003 rows=1 loops=697)"
14) " Index Cond: (sid = s.sid)"
15) " Heap Fetches: 0"
16) " -> Index Scan using btree_reserves_sid on reserves r2 (cost=0.29..0.38 rows=2 width=8) (actual time=0.004..0.005 rows=2 loops=697)"
17) " Index Cond: (sid = r.sid)"
18) " -> Index Scan using btree_boat_bid on boat b2 (cost=0.28..0.31 rows=1 width=4) (actual time=0.004..0.004 rows=0 loops=1335)"
19) " Index Cond: (bid = r2.bid)"
20) " Filter: (color = 'green'::bpchar)"
21) " Rows Removed by Filter: 1"
22) "Planning Time: 2.840 ms"
23) "Execution Time: 18.220 ms"

FLAGS:
SET enable_hashjoin=ON; SET enable_material = ON; SET enable_mergejoin=ON; SET enable_nestloop=ON; SET enable_seqscan=OFF;
SET enable_indexscan =ON; SET enable_indexonlyscan = ON; SET enable_bitmapscan = OFF; SET enable_async_append =ON;
SET enable_sort=ON; SET enable_indexscan=ON;

NOTE :

- Some indicies used in joins
- Cost reduced and execution time

---

3) With hash index

```
-- CREATE INDEX HASH_reserves_bid on reserves using hash(bid);

-- CREATE INDEX HASH_reserves_sid on reserves using hash(sid);

-- CREATE INDEX HASH_sailors_sid on sailors using hash(sid);

-- CREATE INDEX HASH_sailors_sname on sailors using hash(sname);

-- CREATE INDEX HASH_boat_bid on boat using hash(bid);

-- CREATE INDEX HASH_boat_color on boat using hash(color);
```

"QUERY PLAN"

1) "Nested Loop Semi Join (cost=0.00..2085.83 rows=41 width=21) (actual time=0.349..11.936 rows=348 loops=1)"
2) " Join Filter: (s.sid = s2.sid)"
3) " -> Nested Loop (cost=0.00..1909.54 rows=688 width=29) (actual time=0.073..4.167 rows=697 loops=1)"
4) " -> Nested Loop (cost=0.00..1860.50 rows=688 width=4) (actual time=0.058..1.488 rows=697 loops=1)"
5) " -> Index Scan using hash_boat_color on boat b (cost=0.00..157.03 rows=59 width=4) (actual time=0.038..0.084 rows=59 loops=1)"
6) " Index Cond: (color = 'red'::bpchar)"
7) " -> Index Scan using hash_reserves_bid on reserves r (cost=0.00..28.75 rows=12 width=8) (actual time=0.004..0.019 rows=12 loops=59)"
8) " Index Cond: (bid = b.bid)"
9) " -> Index Scan using hash_sailors_sid on sailors s (cost=0.00..0.06 rows=1 width=25) (actual time=0.002..0.003 rows=1 loops=697)"
10) " Index Cond: (sid = r.sid)"
11) " -> Nested Loop (cost=0.00..0.24 rows=1 width=8) (actual time=0.010..0.010 rows=0 loops=697)"
12) " Join Filter: (r2.sid = s2.sid)"
13) " -> Nested Loop (cost=0.00..0.17 rows=1 width=4) (actual time=0.009..0.009 rows=0 loops=697)"
14) " -> Index Scan using hash_reserves_sid on reserves r2 (cost=0.00..0.09 rows=2 width=8) (actual time=0.003..0.004 rows=1 loops=697)"
15) " Index Cond: (sid = r.sid)"
16) " -> Index Scan using hash_boat_bid on boat b2 (cost=0.00..0.03 rows=1 width=4) (actual time=0.002..0.002 rows=0 loops=987)"
17) " Index Cond: (bid = r2.bid)"
18) " Filter: (color = 'green'::bpchar)"
19) " Rows Removed by Filter: 1"
20) " -> Index Scan using hash_sailors_sid on sailors s2 (cost=0.00..0.06 rows=1 width=4) (actual time=0.001..0.001 rows=1 loops=348)"
21) " Index Cond: (sid = r.sid)"
22) "Planning Time: 3.039 ms"
23) "Execution Time: 12.081 ms"

FLAGS:
SET enable_hashjoin=ON; SET enable_material = ON; SET enable_mergejoin=ON; SET enable_nestloop=ON; SET enable_seqscan=OFF;
SET enable_indexscan =ON; SET enable_indexonlyscan = ON; SET enable_bitmapscan = OFF; SET enable_async_append =ON;
SET enable_sort=ON; SET enable_indexscan=ON;

NOTE :

- Some indicies used in joins
- Cost reduced and execution time reduced
- Hash is better than Btree

---

4) With BRIN index

```
-- CREATE INDEX BRIN_reserves_bid on reserves using brin(bid);
-- CREATE INDEX BRIN_reserves_sid on reserves using brin(sid);
-- CREATE INDEX BRIN_sailors_sid on sailors using brin(sid);
-- CREATE INDEX BRIN_boat_bid on boat using brin(bid);
```

```
"QUERY PLAN"
"Nested Loop Semi Join  (cost=10001296708.32..10916827941.24 rows=41 width=21) (actual time=12.392..1822.508 rows=348 loops=1)"
"  ->  Nested Loop  (cost=10000420708.16..10289086417.68 rows=688 width=29) (actual time=0.212..618.834 rows=697 loops=1)"
"        ->  Nested Loop  (cost=10000000708.13..10000069172.82 rows=688 width=4) (actual time=0.115..136.967 rows=697 loops=1)"
"              ->  Seq Scan on boat b  (cost=10000000000.00..10000000087.50 rows=59 width=4) (actual time=0.051..0.352 rows=59 loops=1)"
"                    Filter: (color = 'red'::bpchar)"
"                    Rows Removed by Filter: 2941"
"              ->  Bitmap Heap Scan on reserves r  (cost=708.13..1170.82 rows=12 width=8) (actual time=0.017..2.309 rows=12 loops=59)"
"                    Recheck Cond: (bid = b.bid)"
"                    Rows Removed by Index Recheck: 34438"
"                    Heap Blocks: lossy=11093"
"                    ->  Bitmap Index Scan on brin_reserves_bid  (cost=0.00..708.13 rows=35000 width=0) (actual time=0.010..0.010 rows=3857 loops=59)"
"                          Index Cond: (bid = b.bid)"
"        ->  Bitmap Heap Scan on sailors s  (cost=420000.03..420083.19 rows=1 width=25) (actual time=0.315..0.688 rows=1 loops=697)"
"              Recheck Cond: (sid = r.sid)"
"              Rows Removed by Index Recheck: 10241"
"              Heap Blocks: lossy=59722"
"              ->  Bitmap Index Scan on brin_sailors_sid  (cost=0.00..420000.03 rows=6333 width=0) (actual time=0.005..0.005 rows=1057 loops=697)"
"                    Index Cond: (sid = r.sid)"
"  ->  Nested Loop  (cost=876000.16..912415.00 rows=1 width=8) (actual time=1.726..1.726 rows=0 loops=697)"
"        Join Filter: (r2.sid = s2.sid)"
"        ->  Nested Loop  (cost=648000.13..684331.79 rows=1 width=4) (actual time=1.604..1.604 rows=0 loops=697)"
"              ->  Bitmap Heap Scan on reserves r2  (cost=228000.08..228290.51 rows=2 width=8) (actual time=0.242..1.520 rows=2 loops=697)"
"                    Recheck Cond: (sid = s.sid)"
"                    Rows Removed by Index Recheck: 23866"
"                    Heap Blocks: lossy=90475"
"                    ->  Bitmap Index Scan on brin_reserves_sid  (cost=0.00..228000.08 rows=22914 width=0) (actual time=0.005..0.005 rows=2299 loops=697)"
"                          Index Cond: (sid = s.sid)"
"              ->  Memoize  (cost=420000.04..420049.04 rows=1 width=4) (actual time=0.042..0.042 rows=0 loops=1335)"
"                    Cache Key: r2.bid"
"                    Cache Mode: logical"
"                    Hits: 870  Misses: 465  Evictions: 0  Overflows: 0  Memory Usage: 14kB"
"                    ->  Bitmap Heap Scan on boat b2  (cost=420000.03..420049.03 rows=1 width=4) (actual time=0.118..0.118 rows=1 loops=465)"
"                          Recheck Cond: (bid = r2.bid)"
"                          Rows Removed by Index Recheck: 1529"
"                          Filter: (color = 'green'::bpchar)"
"                          Rows Removed by Filter: 0"
"                          Heap Blocks: lossy=6057"
"                          ->  Bitmap Index Scan on brin_boat_bid  (cost=0.00..420000.03 rows=3000 width=0) (actual time=0.004..0.004 rows=500 loops=465)"
"                                Index Cond: (bid = r2.bid)"
"        ->  Bitmap Heap Scan on sailors s2  (cost=228000.03..228083.19 rows=1 width=4) (actual time=0.243..0.243 rows=1 loops=348)"
"              Recheck Cond: (sid = s.sid)"
"              Rows Removed by Index Recheck: 3620"
"              Heap Blocks: lossy=10730"
"              ->  Bitmap Index Scan on brin_sailors_sid  (cost=0.00..228000.03 rows=6333 width=0) (actual time=0.004..0.004 rows=1057 loops=348)"
"                    Index Cond: (sid = s.sid)"
"Planning Time: 1.604 ms"
"Execution Time: 1823.217 ms"
```

NOTE : BRIN index reduce cost but increase execytion time

FLAGS:
SET enable_hashjoin=ON; SET enable_material = ON; SET enable_mergejoin=ON; SET enable_nestloop=ON; SET enable_seqscan=OFF;
SET enable_indexscan =ON; SET enable_indexonlyscan = ON; SET enable_bitmapscan = ON; SET enable_async_append =ON;
SET enable_sort=ON; SET enable_indexscan=ON;

NOTE :

- Some indicies used in joins
- Cost reduced and execution time increased
- Hash and BTREE are better than BRIN

---

5) With hash and bitmap

```
-- CREATE INDEX HASH_reserves_bid on reserves using hash(bid);
-- CREATE INDEX HASH_reserves_sid on reserves using hash(sid);
-- CREATE INDEX Btree_sailors_sid on sailors using btree(sid);
-- CREATE INDEX BTree_boat_bid on boat using btree(bid);
```

"QUERY PLAN"

1) "Nested Loop (cost=1.14..2623.93 rows=41 width=21) (actual time=0.481..14.993 rows=348 loops=1)"
2) " -> Nested Loop (cost=0.28..1853.25 rows=688 width=4) (actual time=0.095..2.072 rows=697 loops=1)"
3) "   -> Index Scan using btree_boat_bid on boat b (cost=0.28..149.78 rows=59 width=4) (actual time=0.052..1.047 rows=59 loops=1)"
4) "   Filter: (color = 'red'::bpchar)"
5) "   Rows Removed by Filter: 2941"
6) "   -> Index Scan using hash_reserves_bid on reserves r (cost=0.00..28.75 rows=12 width=8) (actual time=0.004..0.014 rows=12 loops=59)"
7) "   Index Cond: (bid = b.bid)"
8) " -> Nested Loop Semi Join (cost=0.85..1.11 rows=1 width=33) (actual time=0.018..0.018 rows=0 loops=697)"
9) "   -> Index Scan using btree_sailors_sid on sailors s (cost=0.29..0.35 rows=1 width=25) (actual time=0.003..0.003 rows=1 loops=697)"
10) "   Index Cond: (sid = r.sid)"
11) "   -> Nested Loop (cost=0.57..0.75 rows=1 width=8) (actual time=0.013..0.013 rows=0 loops=697)"
12) "   -> Nested Loop (cost=0.29..0.44 rows=1 width=12) (actual time=0.005..0.006 rows=1 loops=697)"
13) "   Join Filter: (s2.sid = r2.sid)"
14) "   -> Index Only Scan using btree_sailors_sid on sailors s2 (cost=0.29..0.32 rows=1 width=4) (actual time=0.002..0.002 rows=1 loops=697)"
15) "   Index Cond: (sid = s.sid)"
16) "   Heap Fetches: 0"
17) "   -> Index Scan using hash_reserves_sid on reserves r2 (cost=0.00..0.09 rows=2 width=8) (actual time=0.002..0.003 rows=1 loops=697)"
18) "   Index Cond: (sid = r.sid)"
19) "   -> Index Scan using btree_boat_bid on boat b2 (cost=0.28..0.31 rows=1 width=4) (actual time=0.004..0.004 rows=0 loops=987)"
20) "   Index Cond: (bid = r2.bid)"
21) "   Filter: (color = 'green'::bpchar)"
22) "   Rows Removed by Filter: 1"
23) "Planning Time: 15.209 ms"
24) "Execution Time: 15.120 ms"

FLAGS:
SET enable_hashjoin=ON; SET enable_material = ON; SET enable_mergejoin=ON; SET enable_nestloop=ON; SET enable_seqscan=OFF;
SET enable_indexscan =ON; SET enable_indexonlyscan = ON; SET enable_bitmapscan = OFF; SET enable_async_append =ON;
SET enable_sort=ON; SET enable_indexscan=ON

NOTE :

- Some indicies used in joins
- Cost reduced and execution time
- This mix is better than BTREE but hash is th best

IN Conlusion : hash index is the best to use

Optimized query :

```
select s.sname
from sailors s
where s.sid in (
   select r1.sid
   from reserves r1,reserves r2
   where r1.sid = r2.sid and r1.bid != r2.bid
   and r1.bid in(
      select b.bid
      from boat b
      where b.color = 'red'
   )
   and r2.bid in(
      select b.bid
      from boat b
      where b.color = 'green'
)
)
```

REASON : we replaced the join with in clause

1) Without index

"QUERY PLAN"

```
1)  "Hash Semi Join  (cost=50000001960.92..50000002518.25 rows=41 width=21) (actual time=27.242..30.309 rows=348 loops=1)"
2)  "  Hash Cond: (s.sid = r1.sid)"
3)  "  ->  Seq Scan on sailors s  (cost=10000000000.00..10000000507.00 rows=19000 width=25) (actual time=0.178..1.338 rows=19000
    loops=1)"
4)  "  ->  Hash  (cost=40000001960.41..40000001960.41 rows=41 width=8) (actual time=27.053..27.057 rows=348 loops=1)"
5)  "        Buckets: 1024  Batches: 1  Memory Usage: 22kB"
6)  "        ->  Merge Join  (cost=40000001950.79..40000001960.41 rows=41 width=8) (actual time=26.771..27.019 rows=348 loops=1)"
7)  "              Merge Cond: (r1.sid = r2.sid)"
8)  "              Join Filter: (r1.bid <> r2.bid)"
9)  "              ->  Sort  (cost=20000000960.19..20000000961.91 rows=688 width=8) (actual time=17.963..17.993 rows=697 loops=1)"
10) "                    Sort Key: r1.sid"
11) "                    Sort Method: quicksort  Memory: 57kB"
12) "                    ->  Hash Semi Join  (cost=20000000088.24..20000000927.77 rows=688 width=8) (actual time=1.275..17.629 rows=697
    loops=1)"
13) "                          Hash Cond: (r1.bid = b.bid)"
14) "                          ->  Seq Scan on reserves r1  (cost=10000000000.00..10000000740.00 rows=35000 width=8) (actual time=0.009..6.904
    rows=35000 loops=1)"
15) "                          ->  Hash  (cost=10000000087.50..10000000087.50 rows=59 width=4) (actual time=1.254..1.255 rows=59 loops=1)"
16) "                                Buckets: 1024  Batches: 1  Memory Usage: 11kB"
17) "                                ->  Seq Scan on boat b  (cost=10000000000.00..10000000087.50 rows=59 width=4) (actual time=0.039..1.211 rows=59
    loops=1)"
18) "                                      Filter: (color = 'red'::bpchar)"
```

19) "   Rows Removed by Filter: 2941"
20) "   -> Sort (cost=20000000990.60..20000000993.43 rows=1132 width=8) (actual time=8.799..8.847 rows=1126 loops=1)"
21) "   Sort Key: r2.sid"
22) "   Sort Method: quicksort Memory: 103kB"
23) "   -> Hash Semi Join (cost=20000000088.71..20000000933.18 rows=1132 width=8) (actual time=0.647..8.532 rows=1164 loops=1)"
24) "   Hash Cond: (r2.bid = b_1.bid)"
25) "   -> Seq Scan on reserves r2 (cost=10000000000.00..10000000740.00 rows=35000 width=8) (actual time=0.017..3.137 rows=35000 loops=1)"
26) "   -> Hash (cost=10000000087.50..10000000087.50 rows=97 width=4) (actual time=0.600..0.601 rows=97 loops=1)"
27) "   Buckets: 1024 Batches: 1 Memory Usage: 12kB"
28) "   -> Seq Scan on boat b_1 (cost=10000000000.00..10000000087.50 rows=97 width=4) (actual time=0.037..0.566 rows=97 loops=1)"
29) "   Filter: (color = 'green'::bpchar)"
30) "   Rows Removed by Filter: 2903"
31) "Planning Time: 0.745 ms"
32) "Execution Time: 30.410 ms"

FLAGS:
SET enable_hashjoin=ON; SET enable_material = ON; SET enable_mergejoin=ON; SET enable_nestloop=ON; SET enable_seqscan=OFF;
SET enable_indexscan =ON; SET enable_indexonlyscan = ON; SET enable_bitmapscan = OFF; SET enable_async_append =ON;
SET enable_sort=ON; SET enable_indexscan=ON;

NOTE :

- Cost reduced and execution time

---

2) With B+ REE

```
-- CREATE INDEX BTREE_reserves_bid on reserves using btree(bid);
-- CREATE INDEX BTREE_reserves_sid on reserves using btree(sid);
-- CREATE INDEX Btree_sailors_sid on sailors using btree(sid);
-- CREATE INDEX BTree_boat_bid on boat using btree(bid);
```

"QUERY PLAN"

"QUERY PLAN"

1) "Nested Loop (cost=2244.40..2260.65 rows=41 width=21) (actual time=4.161..5.105 rows=348 loops=1)"
2) " -> HashAggregate (cost=2244.11..2244.52 rows=41 width=8) (actual time=4.155..4.234 rows=348 loops=1)"
3) "   Group Key: r1.sid"
4) "   Batches: 1 Memory Usage: 85kB"
5) "   -> Hash Semi Join (cost=301.50..2244.01 rows=41 width=8) (actual time=0.787..3.973 rows=348 loops=1)"
6) "   Hash Cond: (r2.bid = b_1.bid)"
7) "   -> Nested Loop (cost=150.51..2089.22 rows=1273 width=12) (actual time=0.390..3.407 rows=638 loops=1)"
8) "   -> Nested Loop (cost=150.22..1810.67 rows=688 width=8) (actual time=0.386..1.023 rows=697 loops=1)"
9) "   -> HashAggregate (cost=149.93..150.52 rows=59 width=4) (actual time=0.382..0.395 rows=59 loops=1)"
10) "   Group Key: b.bid"
11) "   Batches: 1 Memory Usage: 24kB"
12) "   -> Index Scan using btree_boat_bid on boat b (cost=0.28..149.78 rows=59 width=4) (actual time=0.005..0.368 rows=59 loops=1)"
13) "   Filter: (color = 'red'::bpchar)"
14) "   Rows Removed by Filter: 2941"

15) " -> Index Scan using btree_reserves_bid on reserves r1 (cost=0.29..28.02 rows=12 width=8) (actual time=0.002..0.008 rows=12 loops=59)"
16) " Index Cond: (bid = b.bid)"
17) " -> Index Scan using btree_reserves_sid on reserves r2 (cost=0.29..0.38 rows=2 width=8) (actual time=0.002..0.003 rows=1 loops=697)"
18) " Index Cond: (sid = r1.sid)"
19) " Filter: (r1.bid <> bid)"
20) " Rows Removed by Filter: 1"
21) " -> Hash (cost=149.78..149.78 rows=97 width=4) (actual time=0.395..0.395 rows=97 loops=1)"
22) " Buckets: 1024 Batches: 1 Memory Usage: 12kB"
23) " -> Index Scan using btree_boat_bid on boat b_1 (cost=0.28..149.78 rows=97 width=4) (actual time=0.015..0.379 rows=97 loops=1)"
24) " Filter: (color = 'green'::bpchar)"
25) " Rows Removed by Filter: 2903"
26) " -> Index Scan using btree_sailors_sid on sailors s (cost=0.29..0.38 rows=1 width=25) (actual time=0.002..0.002 rows=1 loops=348)"
27) " Index Cond: (sid = r1.sid)"
28) "Planning Time: 0.683 ms"
29) "Execution Time: 5.173 ms"

FLAGS:
SET enable_hashjoin=ON; SET enable_material = ON; SET enable_mergejoin=ON; SET enable_nestloop=ON; SET enable_seqscan=OFF;
SET enable_indexscan =ON; SET enable_indexonlyscan = ON; SET enable_bitmapscan = OFF; SET enable_async_append =ON;
SET enable_sort=ON; SET enable_indexscan=ON;

NOTE :

- Some indicies used in joins
- Cost reduced and execution time

---

3) With hash index

"QUERY PLAN"
1) "Nested Loop (cost=10000001836.91..10000001841.77 rows=41 width=21) (actual time=46.738..47.191 rows=348 loops=1)"
2) " -> HashAggregate (cost=10000001836.91..10000001837.32 rows=41 width=8) (actual time=46.732..46.773 rows=348 loops=1)"
3) " Group Key: r1.sid"
4) " Batches: 1 Memory Usage: 85kB"
5) " -> Nested Loop Semi Join (cost=10000000000.00..10000001836.81 rows=41 width=8) (actual time=0.089..46.549 rows=348 loops=1)"
6) " -> Nested Loop (cost=10000000000.00..10000001801.05 rows=1273 width=12) (actual time=0.078..45.679 rows=638 loops=1)"
7) " -> Nested Loop Semi Join (cost=10000000000.00..10000001718.88 rows=688 width=8) (actual time=0.049..43.788 rows=697 loops=1)"
8) " -> Seq Scan on reserves r1 (cost=10000000000.00..10000000740.00 rows=35000 width=8) (actual time=0.019..3.432 rows=35000 loops=1)"
9) " -> Index Scan using hash_boat_bid on boat b (cost=0.00..0.03 rows=1 width=4) (actual time=0.001..0.001 rows=0 loops=35000)"
10) " Index Cond: (bid = r1.bid)"
11) " Filter: (color = 'red'::bpchar)"
12) " Rows Removed by Filter: 1"
13) " -> Index Scan using hash_reserves_sid on reserves r2 (cost=0.00..0.10 rows=2 width=8) (actual time=0.002..0.002 rows=1 loops=697)"
14) " Index Cond: (sid = r1.sid)"

15) "   Filter: (r1.bid <> bid)"
16) "   Rows Removed by Filter: 1"
17) "   -> Index Scan using hash_boat_bid on boat b_1 (cost=0.00..0.03 rows=1 width=4) (actual time=0.001..0.001 rows=1 loops=638)"
18) "   Index Cond: (bid = r2.bid)"
19) "   Filter: (color = 'green'::bpchar)"
20) "   Rows Removed by Filter: 0"
21) " -> Index Scan using hash_sailors_sid on sailors s (cost=0.00..0.10 rows=1 width=25) (actual time=0.001..0.001 rows=1 loops=348)"
22) "   Index Cond: (sid = r1.sid)"
23) "Planning Time: 1.206 ms"
24) "Execution Time: 47.301 ms"

NOTE

FLAGS:
SET enable_hashjoin=ON; SET enable_material = ON; SET enable_mergejoin=ON; SET enable_nestloop=ON; SET enable_seqscan=OFF;
SET enable_indexscan =ON; SET enable_indexonlyscan = ON; SET enable_bitmapscan = OFF; SET enable_async_append =ON;
SET enable_sort=ON; SET enable_indexscan=ON;

NOTE :

- Some indicies used in joins
- Hash index increased execution time and reduced cost
- BTREE is better than hash

---

4) With BRIN index

-- CREATE INDEX BRIN_reserves_bid on reserves using brin(bid);
-- CREATE INDEX BRIN_reserves_sid on reserves using brin(sid);
-- CREATE INDEX BRIN_sailors_sid on sailors using brin(sid);
-- CREATE INDEX BRIN_boat_bid on boat using brin(bid);

"QUERY PLAN"

1) "Nested Loop (cost=10824217820.39..10833298512.16 rows=41 width=21) (actual time=1470.756..1694.019 rows=348 loops=1)"
2) " -> HashAggregate (cost=10823990888.36..10823990888.77 rows=41 width=8) (actual time=1470.530..1470.647 rows=348 loops=1)"
3) "   Group Key: r1.sid"
4) "   Batches: 1 Memory Usage: 85kB"
5) "   -> Nested Loop Semi Join (cost=10000840795.90..10823990888.26 rows=41 width=8) (actual time=4.231..1470.150 rows=348 loops=1)"
6) "   -> Nested Loop (cost=10000420795.86..10289268469.79 rows=1273 width=12) (actual time=3.718..1382.355 rows=638 loops=1)"
7) "   -> Nested Loop (cost=10000000795.78..10000069173.55 rows=688 width=8) (actual time=1.006..133.280 rows=697 loops=1)"
8) "   -> HashAggregate (cost=10000000087.65..10000000088.24 rows=59 width=4) (actual time=0.929..1.060 rows=59 loops=1)"
9) "   Group Key: b.bid"
10) "   Batches: 1 Memory Usage: 24kB"
11) "   -> Seq Scan on boat b (cost=10000000000.00..10000000087.50 rows=59 width=4) (actual time=0.040..0.883 rows=59 loops=1)"
12) "   Filter: (color = 'red'::bpchar)"
13) "   Rows Removed by Filter: 2941"

14) " -> Bitmap Heap Scan on reserves r1 (cost=708.13..1170.82 rows=12 width=8) (actual time=0.014..2.235 rows=12 loops=59)"
15) " Recheck Cond: (bid = b.bid)"
16) " Rows Removed by Index Recheck: 34438"
17) " Heap Blocks: lossy=11093"
18) " -> Bitmap Index Scan on brin_reserves_bid (cost=0.00..708.13 rows=35000 width=0) (actual time=0.009..0.009 rows=3857 loops=59)"
19) " Index Cond: (bid = b.bid)"
20) " -> Bitmap Heap Scan on reserves r2 (cost=420000.08..420347.79 rows=2 width=8) (actual time=0.875..1.789 rows=1 loops=697)"
21) " Recheck Cond: (sid = r1.sid)"
22) " Rows Removed by Index Recheck: 28096"
23) " Filter: (r1.bid <> bid)"
24) " Rows Removed by Filter: 1"
25) " Heap Blocks: lossy=106271"
26) " -> Bitmap Index Scan on brin_reserves_sid (cost=0.00..420000.08 rows=22914 width=0) (actual time=0.006..0.006 rows=2299 loops=697)"
27) " Index Cond: (sid = r1.sid)"
28) " -> Bitmap Heap Scan on boat b_1 (cost=420000.03..420049.03 rows=1 width=4) (actual time=0.135..0.135 rows=1 loops=638)"
29) " Recheck Cond: (bid = r2.bid)"
30) " Rows Removed by Index Recheck: 1927"
31) " Filter: (color = 'green'::bpchar)"
32) " Rows Removed by Filter: 0"
33) " Heap Blocks: lossy=10382"
34) " -> Bitmap Index Scan on brin_boat_bid (cost=0.00..420000.03 rows=3000 width=0) (actual time=0.004..0.004 rows=500 loops=638)"
35) " Index Cond: (bid = r2.bid)"
36) " -> Bitmap Heap Scan on sailors s (cost=226932.03..227015.19 rows=1 width=25) (actual time=0.231..0.640 rows=1 loops=348)"
37) " Recheck Cond: (sid = r1.sid)"
38) " Rows Removed by Index Recheck: 10239"
39) " Heap Blocks: lossy=29812"
40) " -> Bitmap Index Scan on brin_sailors_sid (cost=0.00..226932.03 rows=6333 width=0) (actual time=0.004..0.004 rows=1057 loops=348)"
41) " Index Cond: (sid = r1.sid)"
42) "Planning Time: 1.011 ms"
43) "Execution Time: 1694.390 ms"

FLAGS:
SET enable_hashjoin=ON; SET enable_material = ON; SET enable_mergejoin=ON; SET enable_nestloop=ON; SET enable_seqscan=OFF;
SET enable_indexscan =ON; SET enable_indexonlyscan = ON; SET enable_bitmapscan = ON; SET enable_async_append =ON;
SET enable_sort=ON; SET enable_indexscan=ON;

NOTE :

- Some indicies used in joins
- Cost reduced and execution time increased
- BTREE and hash are better than BRIN

---

5) With BTREE and hash

-- CREATE INDEX BTREE_reserves_bid on reserves using btree(bid);
-- CREATE INDEX BTREE_reserves_sid on reserves using btree(sid);
CREATE INDEX HASH_sailors_sid on sailors using hash(sid);
CREATE INDEX HASH_boat_bid on boat using hash(bid);

"QUERY PLAN"
1) "Nested Loop (cost=3568.41..3573.27 rows=41 width=21) (actual time=45.350..45.862 rows=348 loops=1)"
2) " -> HashAggregate (cost=3568.41..3568.82 rows=41 width=8) (actual time=45.344..45.390 rows=348 loops=1)"

```
3)  "       Group Key: r1.sid"
4)  "       Batches: 1  Memory Usage: 85kB"
5)  "       ->  Nested Loop Semi Join  (cost=0.58..3568.31 rows=41 width=8) (actual time=0.050..45.221 rows=348 loops=1)"
6)  "             ->  Nested Loop  (cost=0.58..3532.55 rows=1273 width=12) (actual time=0.045..44.529 rows=638 loops=1)"
7)  "                   ->  Nested Loop Semi Join  (cost=0.29..3254.00 rows=688 width=8) (actual time=0.027..43.121 rows=697 loops=1)"
8)  "                         ->  Index Scan using btree_reserves_sid on reserves r1  (cost=0.29..2275.12 rows=35000 width=8) (actual
    time=0.011..9.423 rows=35000 loops=1)"
9)  "                         ->  Index Scan using hash_boat_bid on boat b  (cost=0.00..0.03 rows=1 width=4) (actual time=0.001..0.001
    rows=0 loops=35000)"
10) "                               Index Cond: (bid = r1.bid)"
11) "                               Filter: (color = 'red'::bpchar)"
12) "                               Rows Removed by Filter: 1"
13) "                   ->  Index Scan using btree_reserves_sid on reserves r2  (cost=0.29..0.38 rows=2 width=8) (actual time=0.001..0.002
    rows=1 loops=697)"
14) "                         Index Cond: (sid = r1.sid)"
15) "                         Filter: (r1.bid <> bid)"
16) "                         Rows Removed by Filter: 1"
17) "             ->  Index Scan using hash_boat_bid on boat b_1  (cost=0.00..0.03 rows=1 width=4) (actual time=0.001..0.001 rows=1
    loops=638)"
18) "                   Index Cond: (bid = r2.bid)"
19) "                   Filter: (color = 'green'::bpchar)"
20) "                   Rows Removed by Filter: 0"
21) "  ->  Index Scan using hash_sailors_sid on sailors s  (cost=0.00..0.10 rows=1 width=25) (actual time=0.001..0.001 rows=1
    loops=348)"
22) "       Index Cond: (sid = r1.sid)"
23) "Planning Time: 1.275 ms"
24) "Execution Time: 45.940 ms"
```

FLAGS:
SET enable_hashjoin=ON; SET enable_material = ON; SET enable_mergejoin=ON; SET enable_nestloop=ON; SET enable_seqscan=OFF;
SET enable_indexscan =ON; SET enable_indexonlyscan = ON; SET enable_bitmapscan = ON; SET enable_async_append =ON;
SET enable_sort=ON; SET enable_indexscan=OFF;

NOTE :

- Some indicies used in joins
- Cost reduced and execution time
- This mix is better than hash
- Btree is better than this mix

IN Conclusion : BTREE is the best to use for optimized query