

## Query 7

- Find the names of sailors who have reserved boat 103.

### Note !

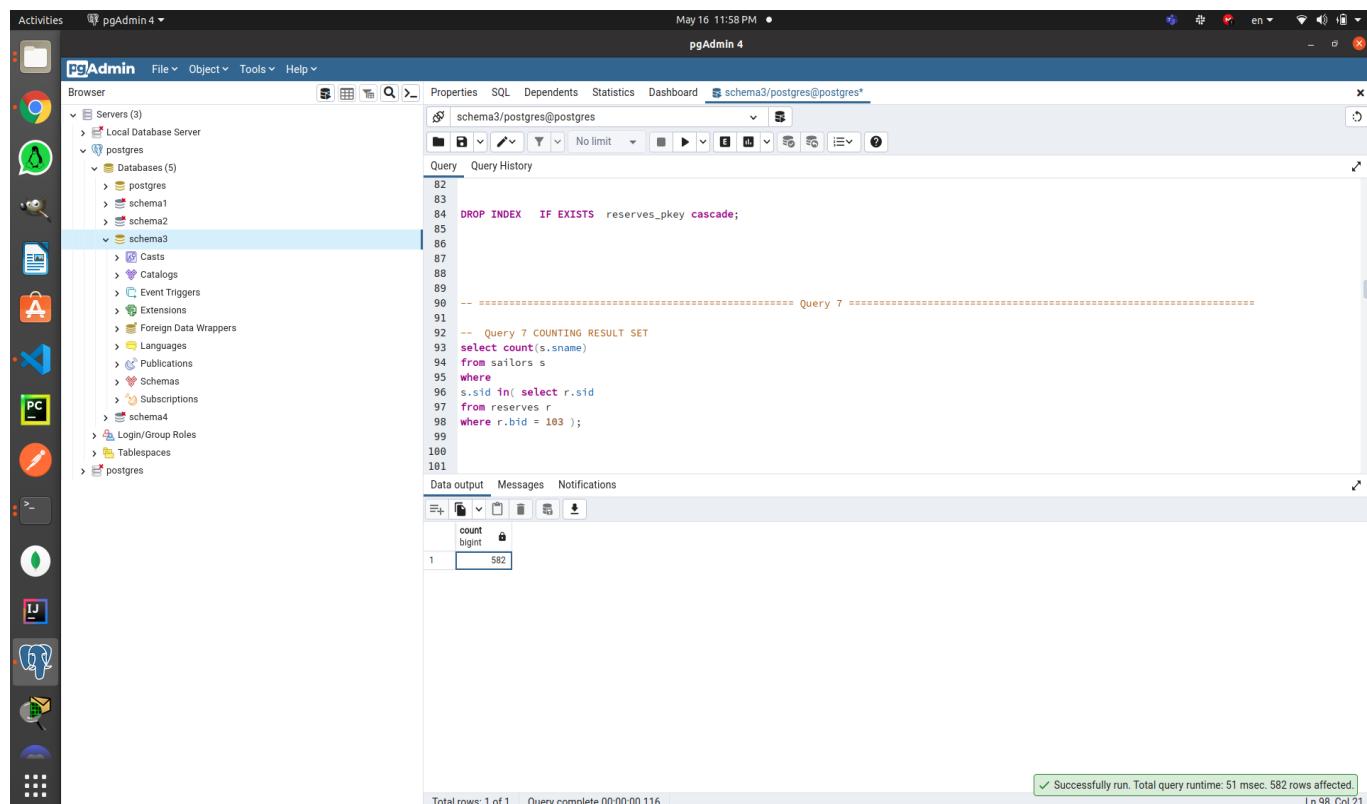
- Flags Hashjoin and HashAgg here were disabled for future after many trials and errors , I've discovered the best way to show the difference in terms of the cost and to beat the Postgres Query Optimizer Algorithm to be able to show indices effect and cost differences .
- The Execution time was changing by 10 Ms in each Execution which is considered high and I can't take it as a measurable Metric because it was Linux (Ubuntu) Operating System performance and I took permission from Prof. Wael as do not take it as my main objective I take the Overall Cost and Compare it .

### Original Query

```
select s.sname
from sailors s
where
s.sid in( select r.sid
from reserves r
where r.bid = 103 );
```

### Result Set

- 582 Rows



The screenshot shows the pgAdmin 4 interface with the following details:

- Toolbar:** Activities, pgAdmin 4, File, Object, Tools, Help.
- Servers:** Local Database Server, postgres (selected), schema3/postgres@postgres\*
- Properties:** Properties tab selected.
- Query Editor:** Shows the SQL query and its execution history. The query is:

```
DROP INDEX IF EXISTS reserves_pkey cascade;
-- ====== Query 7 ======
-- Query 7 COUNTING RESULT SET
select count(s.sname)
from sailors s
where
s.sid in( select r.sid
from reserves r
where r.bid = 103 );
```
- Data Output:** A table showing the result of the query:

count	bigint
1	582
- Status Bar:** Total rows: 1 of 1, Query complete 00:00:00.116, Successfully run. Total query runtime: 51 msec. 582 rows affected, Ln 98, Col 21.

## Report

### 1. given query without an index :

pgAdmin 4 - May 17 12:08 AM

```

15
16 select count(sid)
17 from sailors;
18
19 select count(*)
20 from boat;
21
22 select count(*)
23 from reserves;
24
25
26 delete from Reserves;
27
28 delete from sailors;
29
30 delete from Boat;
31
32
33 --- Sailors
34 -- see what indexes are created for that table
35 select *
36 from pg_indexes
37 where tablename = 'sailors' or tablename='reserves' or tablename='boat';
38
39 -- see constraint names
40 SELECT con.*
41   FROM pg_catalog.pg_constraint con
42     INNER JOIN pg_catalog.pg_class rel

```

Total rows: 0 of 0    Query complete 00:00:00.117    Successfully run. Total query runtime: 51 msec. 582 rows affected. Ln 35, Col 1

pgAdmin 4 - Jun 9 2:11 PM

```

124 -- Query 7 (STATISTICS)
125 set enable_hashagg = off;
126 set enable_hashjoin = off;
127
128 explain analyze select s.sname
129   from sailors s
130   where
131     s.sid in( select r.sid
132       from reserves r
133       where r.bid = 103 );
134
135
136 -- Query 7 optimized (STATISTICS)
137
138 -- view of Query 7
139 set enable_hashagg = off;
140 set enable_hashjoin = off;
141
142 create MATERIALIZED VIEW query_7
143 as
144   select r.sid
145   from reserves r
146   where r.bid =103 ;
147
148
149 explain analyze select s.sname
150   from sailors s
151   where exists (select R.sid
152     from query_7 R
153     where s.sid =R.sid)
154
155
156
157
158
159 -- ====== Query 8 ======
160
161 -- Find the names of sailors 'who ha've reserved a red boat.
162 -- Query 8 (COUNTING RESULT SET) , Number Of Rows = 673
163
164 explain analyze select count(s.sname)

```

Total rows: 1 of 1    Query complete 00:00:00.243    Ln 128, Col 16

```

graph LR
    sailors[sailors] --> Sort1[Sort]
    reserves[reserves] --> Sort2[Sort]
    Sort1 --> MSJ[Merge Semi Join]
    Sort2 --> MSJ

```

The screenshot shows the pgAdmin 4 interface. On the left, the 'Servers' tree view shows a connection to 'schema3/postgres'. In the center, a query editor window displays a series of SQL commands, including `explain analyze` statements for Query 7 and Query 8. On the right, a 'Data output' pane shows the 'QUERY PLAN' for the executed queries, detailing the execution plan with various steps like 'Merge Semi Join', 'Sort', and 'Seq Scan'. The bottom status bar indicates 'Total rows: 14 of 14' and 'Query complete 0:00:00.146'.

```

-- Query 7 (STATISTICS)
set enable_hashagg = off;
set enable_hashjoin = off;

explain analyze select s.sname
from sailors s
where
  s.sid in( select r.sid
  from reserves r
  where r.bid = 103 );

-- Query 7 optimized (STATISTICS)
set enable_hashagg = off;
set enable_hashjoin = off;

create MATERIALIZED VIEW query_7
as
select r.sid
from reserves r
where r.bid =103 ;

explain analyze select s.sname
from sailors s
where exists (select R.sid
               from query_7 R
               where s.sid =R.sid);

-- ====== Query 8 ======
-- Find the names of sailors who ha'ue reserved a red boat.
-- Query 8 (COUNTING RESULT SET) , Number Of Rows = 673
explain analyze select count(s.sname);

```

### Explanation :

- Metrics :

**Execution Time : 15.430 ms    Total Expected Cost : 2457.26**

---

- given query with B+ trees indices only :

Activities pgAdmin 4 Jun 9 3:12 PM ● pgAdmin 4

Servers Local | PostgreSQL Data Query History Execute/Refresh (F5)

```

-- ====== Query 7 ======
-- Find the names of sailors who have reserved boat 103.
-- Query 7 COUNTING RESULT SET , Number Of Rows = 582

CREATE INDEX b_sailorsSID ON sailors USING btree(sid );
CREATE INDEX b_reservesSID ON reserves USING btree(sid );
CREATE INDEX b_reservesBID ON reserves USING btree(bid );
CREATE INDEX R_reservesBID ON query_7 USING btree(sid );

select *
from pg_indexes
where tablename = 'sailors' or tablename='reserves' or tablename='boat' or tablename='query_7';

select count(s.sname)
from sailors s
where
s.sid in( select r.sid
from reserves r
where r.bid = 103 );

-- Query 7 optimized (COUNTING RESULT SET) , Number Of Rows = 582

select count(s.sname)
from sailors s
where exists (select R.sid
from query_7 R
where s.sid =R.sid);

Total rows: 4 of 4   Query complete 00:00:00.141

```

Ln 95, Col 1

Activities pgAdmin 4 Jun 9 2:57 PM ● pgAdmin 4

Servers Local | PostgreSQL Data Query History Explain Analyze (Shift+F7)

```

-- Query 7 (STATISTICS)
set enable_hashagg = off;
set enable_hashjoin = off;

explain analyze select s.sname
from sailors s
where
s.sid in( select r.sid
from reserves r
where r.bid = 103 );

-- Query 7 optimized (STATISTICS)
-- view of Query 7
set enable_hashagg = off;
set enable_hashjoin = off;

-- Query 7 view table of reserves with bid =103
create MATERIALIZED VIEW query_7
as
select r.sid
from reserves r
where r.bid =103 ;

explain analyze select s.sname
from sailors s
where exists (select R.sid
from query_7 R
where s.sid =R.sid);

-- ====== Query 8 ======
-- Find the names of sailors 'who ha'tue reserved a red boat.

Total rows: 1 of 1   Query complete 00:00:00.055

```

Successfully run. Total query runtime: 55 msec. 1 rows aff  
Ln 132, Col 17

```

-- Query 7 (STATISTICS)
125
126
127
128 -- Query 7 optimized (STATISTICS)
129 set enable_hashagg = off;
130 set enable_hashjoin = off;
131
132 explain analyze select s.sname
133   from sailors s
134   where
135     s.sid in( select r.sid
136       from reserves r
137      where r.bid = 103 );
138
139
140
141
142 -- view of Query 7
143 set enable_hashagg = off;
144 set enable_hashjoin = off;
145
146 -- Query 7 view table of reserves with bid =103
147 create MATERIALIZED VIEW query_7
148 as
149 select r.sid
150   from reserves r
151  where r.bid =103 ;
152
153
154 explain analyze select s.sname
155   from sailors s
156   where exists (select R.sid
157     from query_7 R
158    where s.sid =R.sid);
159
160
161
162
163 -- ====== Query 8 ======
164
165 -- Find the names of sailors 'who ha'ue reserved a red boat.
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
888
889
889
890
891
892
893
894
895
896
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
919
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
948
949
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1189
1190
1191
1192
1193
1194
1195
1196
1197
1197
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1289
1290
1291
1292
1293
1294
1295
1296
1297
1297
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1397
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1489
1490
1491
1492
1493
1494
1495
1496
1497
1497
1498
1499
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1589
1590
1591
1592
1593
1594
1595
1596
1597
1597
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1689
1690
1691
1692
1693
1694
1695
1696
1697
1697
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1789
1790
1791
1792
1793
1794
1795
1796
1797
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1888
1889
1889
1890
1891
1892
1893
1894
1895
1896
1897
1897
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2098
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2198
2199
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
```

Activities pgAdmin 4 Jun 9 3:32 PM ● pgAdmin 4

Servers Local PostgreSQL Data Query History

```

DROP INDEX IF EXISTS reserves_pkey cascade;
-- ====== Query 7 ======
-- Find the names of sailors who have reserved boat 103.
-- Query 7 COUNTING RESULT SET , Number Of Rows = 582
CREATE INDEX b_sailorsSID ON sailors USING hash(sid);
CREATE INDEX b_reservesSID ON reserves USING hash(reserve_id);
CREATE INDEX b_reservesBID ON reserves USING hash(bid);
CREATE INDEX R_reservesBID ON query_7 USING hash(sid);

select *
from pg_indexes
where tablename = 'sailors' or tablename='reserves' or tablename='boat' or tabl
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
Total rows: 4 of 4 Query complete 00:00:00.207

```

Data output Messages Explain Notifications

schemaname	tablename	indexname	tablespace	indexdef
public	sailors	b_sailorsSID	[null]	CREATE INDEX b_sailorsSID ON public.sailors USING hash (sid)
public	reserves	b_reservesSID	[null]	CREATE INDEX b_reservesSID ON public.reserves USING hash (reserve_id)
public	reserves	b_reservesBID	[null]	CREATE INDEX b_reservesBID ON public.reserves USING hash (bid)
public	query_7	R_reservesBID	[null]	CREATE INDEX R_reservesBID ON public.query_7 USING hash (sid)

Activities pgAdmin 4 Jun 9 3:36 PM ● pgAdmin 4

Servers Local PostgreSQL Data Query History Explain Analyze

```

-- Query 7 optimized (COUNTING RESULT SET ), Number Of Rows = 582
select count(s.sname)
from sailors s
where exists (select R.sid
              from query_7 R
              where s.sid=R.sid)

-- Query 7 (STATISTICS)
set enable_hashagg = off;
set enable_hashjoin = off;

explain analyze select s.sname
from sailors s
where
s.sid in( select r.sid
            from reserves r
            where r.bid = 103 );

-- Query 7 optimized (STATISTICS)
-- view of Query 7
set enable_hashagg = off;
set enable_hashjoin = off;

-- Query 7 view table of reserves with bid =103
create MATERIALIZED VIEW query_7
as
select r.sid
from reserves r
where r.bid =103 ;
explain analyze select s.sname

```

Data output Messages Explain Notifications

Graphical Analysis Statistics

```

graph LR
    A[b_reservesbid] --> B[reserves]
    B --> C[Sort]
    C --> D[Unique]
    D --> E[Nested Loop Inner Join]
    E --> F[b_sailorssid]

```

Total rows: 1 of 1 Query complete 00:00:00.051

```

-- Query 7 optimized (COUNTING)
-- Number Of Rows = 582
121 select count(s.sname)
from sailors s
where exists (select R.sid
               from query_7 R
              where s.sid = R.sid)
122
123
124
125
126
127
128
129
130 -- Query 7 (STATISTICS)
131 set enable_hashagg = off;
132 set enable_hashjoin = off;
133
134 explain analyze select s.sname
from sailors s
where
135   s.sid in( select r.sid
136     from reserves r
137     where r.bid = 103 );
138
139
140
141
142 -- Query 7 optimized (STATISTICS)
143
144 -- view of Query 7
145 set enable_hashagg = off;
146 set enable_hashjoin = off;
147
148 -- Query 7 view table of reserves with bid =103
149
150 create MATERIALIZED VIEW query_7
151 as
152 select r.sid
153 from reserves r
154 where r.bid =103 ;
155
156
157
158 explain analyze select s.sname
Total rows: 14 of 14   Query complete 00:00:00.089

```

Successfully run. Total query runtime: 89 msec. 14 rows affected  
Ln 131, Col 1

### Explanation :

- Metrics :

**Execution Time : 0.993 ms    Total Expected Cost : 1159.32**

- The Hash helped in the performance it decreased the Execution Time and Expected Cost .
- The Query Planner used the Hash index because Hash is O(1) performance with Exact Values and considered the best for exact values queries .
- Here it showed the improvement due to for condition of Joining in the Query the Nested Loop Semi Join using index scan using Hash based algorithm on the condition (sid=s.sid) which approximatly maded to be O(n) performance .
- Here it showed the improvement due to for condition of (bid = 103) with performance of O(1) .

### 4. given query with BRIN indices only :

Activities pgAdmin 4 Jun 9 4:32 PM ● pgAdmin 4

Dat Query History Execute/Refresh

```

> p 84 DROP INDEX IF EXISTS reserves;
> s 85
> s 86
> s 87
> s 88
> s 89
> s 90 -- -----
> s 91 -- Find the names of sailors who have reserved boat 103.
> s 92 -- Query 7 COUNTING RESULT SET , Number Of Rows = 582
> s 93
> s 94
> s 95
> s 96 CREATE INDEX b_sailorsSID ON sailors USING BRIN(sid);
> s 97 CREATE INDEX b_reservesSID ON reserves USING BRIN(sid);
> s 98 CREATE INDEX b_reservesBID ON reserves USING BRIN(bid);
> s 99 CREATE INDEX R_reservesBID ON query_7 USING BRIN(sid);

> s 100
> Log 101
> Tab 102
> postgr 103 select *
from pg_indexes
where tablename = 'sailors' or tablename='reserves' or tablename='boat' or tablename='query_7';

104
105
106
107
108 DROP INDEX IF EXISTS b_sailorsSID cascade;
109 DROP INDEX IF EXISTS b_reservesSID cascade;
110 DROP INDEX IF EXISTS b_reservesBID cascade;
111 DROP INDEX IF EXISTS R_reservesBID cascade;
112
113
114
115
116 select count(s.sname)
from sailors s
117 where
118 s.sid in( select r.sid
119 from reserves r
120 where r.bid = 103 );
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159

```

Total rows: 4 of 4 Query complete 00:00:00.070 Ln 96, Col 1

Activities pgAdmin 4 Jun 9 5:07 PM ● pgAdmin 4

Dat Query History Explain Analyze Shift [F7]

```

> p 119 s.sid in( select r.sid
> s 120 from reserves r
> s 121 where r.bid = 103 );
> s 122
> s 123
> s 124
> s 125 -- Query 7 optimized (COUNTING RESULT SET) , Number Of Rows = 582
> s 126
> s 127
> s 128 select count(s.sname)
from sailors s
129 where exists (select R.sid
130 from query_7 R
131 where s.sid =R.sid);
132
133
134
135
136
137 -- Query 7 (STATISTICS)
138 set enable_hashagg = off;
139 set enable_hashjoin = off;
140 set enable_seqscan = off;
141
142
143 explain analyze select s.sname
from sailors s
144 where
145 s.sid in( select r.sid
146 from reserves r
147 where r.bid = 103 );
148
149
150
151 -- Query 7 optimized (STATISTICS)
152
153 -- view of Query 7
154 set enable_hashagg = off;
155 set enable_hashjoin = off;
156
157 -- Query 7 view table of reserves with bid =103
158
159 create MATERIALIZED VIEW query_7

```

Total rows: 1 of 1 Query complete 00:00:00.786 Ln 149, Col 1

Data output Messages Explain Notifications

Graphical Analysis Statistics

Successfully run. Total query runtime: 786 msec. 1 rows affected.

Successfully run. Total query runtime: 137 msec. 4 rows affected.

```

119 s.sid in( select r.sid
120   from reserves r
121   where r.bid = 103 );
122
123
124
125 -- Query 7 optimized (COUNTING RESULT SET) , Number Of Rows = 582
126
127
128 select count(s.sname)
129   from sailors s
130   where exists (select R.sid
131      from query_7 R
132      where s.sid =R.sid)
133
134
135
136 -- Query 7 (STATISTICS)
137 set enable_hashagg = off;
138 set enable_hashjoin = off;
139 set enable_seqscan = off;
140
141
142
143 explain analyze select s.sname
144   from sailors s
145   where
146     s.sid in( select r.sid
147       from reserves r
148       where r.bid = 103 );
149
150
151 -- Query 7 optimized (STATISTICS)
152
153 -- view of Query 7
154 set enable_hashagg = off;
155 set enable_hashjoin = off;
156
157 -- Query 7 view table of reserves with bid =103
158
159 create MATERIALIZED VIEW query_7
Total rows: 23 of 23   Query complete 00:00:00.865

```

Successfully run. Total query runtime: 137 msec. 4 rows affected  
Ln 149, Col 1

### Explanation :

- Metrics :

**Execution Time : 851.120 ms    Total Expected Cost : 4080651.27**

- Here the BRIN was not used in the original Query Plan settings (Hashjoin and HashAgg are off) so I've made seqscan=off too.
- The Execution Time and Expected Cost became the Worst of all .
- This happened because the Query Optimizer didnt used it from the first place due to BRIN Usage here was not suitable so we have used it to simulate seqscan behaviour only we traversed it all and followed all its pointers so it is worst index to use in this case.

- given query with mixed indices (any mix of your choice) :

pgAdmin 4

Jun 9 4:56 PM • pgAdmin 4

Servers Local | PostgreSQL Query History Data output Messages Explain Notifications

```

84 DROP INDEX IF EXISTS reserve;
85
86
87
88
89
90 -- ====== Query 7 ======
91
92 -- Find the names of sailors who have reserved boat 103.
93 -- Query 7 COUNTING RESULT SET , Number Of Rows = 582
94
95
96 CREATE INDEX b_sailorsSID ON sailors USING btree(sid );
97 CREATE INDEX b_reservesSID ON reserves USING hash(sid );
98 CREATE INDEX b_reservesBID ON reserves USING hash(bid );
99 CREATE INDEX R_reservesBID ON query_7 USING btree(sid );
100
101
102 select *
103 from pg_indexes
104 where tablename = 'sailors' or tablename='reserves' or tablename='boat' or tablename='query_7';
105
106
107
108 DROP INDEX IF EXISTS b_sailorsSID cascade;
109 DROP INDEX IF EXISTS b_reservesSID cascade;
110 DROP INDEX IF EXISTS b_reservesBID cascade;
111 DROP INDEX IF EXISTS R_reservesBID cascade;
112
113
114
115
116 select count(s.sname)
117 from sailors s
118 where
119 s.sid in( select r.sid
120 from reserves r
121 where r.bid = 103 );
122
123
124
-- Total rows: 4 of 4 Query complete 00:00:00.056

```

Successfully run. Total query runtime: 137 msec. 4 rows affected.

Ln 95, Col 1

pgAdmin 4

Jun 9 4:57 PM • pgAdmin 4

Servers Local | PostgreSQL Query History Explain Analyze Graphical Analysis Statistics

```

115
116 select count(s.sname)
117 from sailors s
118 where
119 s.sid in( select r.sid
120 from reserves r
121 where r.bid = 103 );
122
123
124
125
126
127
128 select count(s.sname)
129 from sailors s
130 where exists (select R.sid
131 from query_7 R
132 where s.sid = R.sid);
133
134
135
136
137 -- Query 7 (STATISTICS)
138 set enable_hashagg = off;
139 set enable_hashjoin = off;
140
141 explain analyze select s.sname
142 from sailors s
143 where
144 s.sid in( select r.sid
145 from reserves r
146 where r.bid = 103 );
147
148
149 -- Query 7 optimized (STATISTICS)
150
151 -- view of Query 7
152 set enable_hashagg = off;
153 set enable_hashjoin = off;
154
-- Query 7 view table of reserves with bid =103
Total rows: 1 of 1 Query complete 00:00:00.148

```

Successfully run. Total query runtime: 137 msec. 4 rows affected.

Ln 141, Col 17

```

-- Query 7 optimized (COUNTING RESULT SET) , Number Of Rows = 582
select count(s.sname)
from sailors s
where
s.sid in( select r.sid
from reserves r
where r.bid = 103 );

-- Query 7 optimized (STATISTICS)
set enable_hashagg = off;
set enable_hashjoin = off;
explain analyze select s.sname
from sailors s
where
s.sid in( select r.sid
from reserves r
where r.bid = 103 );
-- Query 7 optimized (STATISTICS)
-- view of Query 7
set enable_hashagg = off;
set enable_hashjoin = off;
-- Query 7 view table of reserves with bid =103
Total rows: 13 of 13   Query complete 00:00:00.174

```

Successfully run. Total query runtime: 137 msec. 4 rows affected  
Ln 138, Col 1

### Explanation :

- Metrics :

**Execution Time : 0.845 ms    Total Expected Cost : 960.03**

- The Query Planner used the Merge semi join by using both the ZigZag join and the Hash based algorithm together on the condition (s.sid =r.sid) which improved the Execution time and the expected cost way more better which made the join in  $O(n \log n)$ .
- And It used the Hash indexed scan on bid =103

### Optimized Query

```

-- Query 7 view table of reserves with bid = 103

create MATERIALIZED VIEW query_7
as
select r.sid
from reserves r
where r.bid =103  ;

select s.sname
from sailors s
where exists (select R.sid
              from query_7 R
              where s.sid =R.sid);

```

## Result Set

- 582 Rows

## Report

1. given query without an index :

pgAdmin 4 - Jun 9 2:22 PM • pgAdmin 4

```

explain analyze select s.sname
  from sailors s
 where
   s.sid in( select r.sid
  from reserves r
   where r.bid = 103 );

-- Query 7 optimized (STATISTICS)
-- view of Query 7
set enable_hashagg = off;
set enable_hashjoin = off;

-- Query 7 view table of reserves with bid =103
create MATERIALIZED VIEW query_7
as
select r.sid
  from reserves r
 where r.bid =103 ;

explain analyze select s.sname
  from sailors s
 where exists (select R.sid
                  from query_7 R
                 where s.sid=R.sid);

-- ====== Query 8 ======
-- Find the names of sailors 'who ha'ue reserved a red boat.
-- Query 8 (COUNTING RESULT SET) , Number Of Rows = 673

```

Total rows: 1 of 1    Query complete 00:00:00.103

Successfully run. Total query runtime: 103 msec. 1 rows affected.

Ln 150, Col 17

pgAdmin 4 - Jun 9 2:22 PM • pgAdmin 4

```

explain analyze select s.sname
  from sailors s
 where
   s.sid in( select r.sid
  from reserves r
   where r.bid = 103 );

-- Query 7 optimized (STATISTICS)
-- view of Query 7
set enable_hashagg = off;
set enable_hashjoin = off;

-- Query 7 view table of reserves with bid =103
create MATERIALIZED VIEW query_7
as
select r.sid
  from reserves r
 where r.bid =103 ;

explain analyze select s.sname
  from sailors s
 where exists (select R.sid
                  from query_7 R
                 where s.sid=R.sid);

-- ====== Query 8 ======
-- Find the names of sailors 'who ha'ue reserved a red boat.
-- Query 8 (COUNTING RESULT SET) , Number Of Rows = 673

```

Total rows: 12 of 12    Query complete 00:00:00.068

Ln 149, Col 1

**Explanation :**

- Metrics :

**Execution Time : 8.309 ms    Total Expected Cost : 1746.50**

---

\* Reason :

- This Query Improved in the Execution time and Expected Cost than the Original Query from 2475.26 to 1746.50 .

- Because I used Materialized Views which already made an Intermediate Ready Table with smaller Size that Optimized Query used it which decreased the number of steps needed(Filtration of the table over r.bid =103 ) for the Query to get Executed .

- Because the loops ends faster and exits due to I used the Exist Operator instead of the In Operator so it helped in the intermediate results.

2. given query with B+ trees indices only :

Activities pgAdmin 4 Jun 9 3:12 PM ● pgAdmin 4

Servers Local | PostgreSQL Data Query History Execute/Refresh (F5)

```

-- ====== Query 7 ======
-- Find the names of sailors who have reserved boat 103.
-- Query 7 COUNTING RESULT SET , Number Of Rows = 582

CREATE INDEX b_sailorsSID ON sailors USING btree(sid );
CREATE INDEX b_reservesSID ON reserves USING btree(sid );
CREATE INDEX b_reservesBID ON reserves USING btree(bid );
CREATE INDEX R_reservesBID ON query_7 USING btree(sid );

select *
from pg_indexes
where tablename = 'sailors' or tablename='reserves' or tablename='boat' or tablename='query_7';

-- Query 7 optimized (COUNTING RESULT SET) , Number Of Rows = 582

select count(s.sname)
from sailors s
where
s.sid in( select r.sid
from reserves r
where r.bid = 103 );

-- Query 7 optimized (COUNTING RESULT SET) , Number Of Rows = 582

select count(s.sname)
from sailors s
where exists (select R.sid
from query_7 R
where s.sid =R.sid)

Total rows: 4 of 4 Query complete 00:00:00.141

```

Ln 95, Col 1

Activities pgAdmin 4 Jun 9 3:13 PM ● pgAdmin 4

Servers Local | PostgreSQL Data Query History Explain Analyze (Shift+F7)

```

-- Query 7 (STATISTICS)
set enable_hashagg = off;
set enable_hashjoin = off;

explain analyze select s.sname
from sailors s
where
s.sid in( select r.sid
from reserves r
where r.bid = 103 );

-- Query 7 optimized (STATISTICS)

-- view of Query 7
set enable_hashagg = off;
set enable_hashjoin = off;

create MATERIALIZED VIEW query_7
as
select r.sid
from reserves r
where r.bid =103 ;

explain analyze select s.sname
from sailors s
where exists (select R.sid
from query_7 R
where s.sid =R.sid);

-- ====== Query 8 ======
-- Find the names of sailors 'who ha'ue reserved a red boat.
-- Query 8 (COUNTING RESULT SET) , Number Of Rows = 673

Total rows: 1 of 1 Query complete 00:00:00.088

```

Ln 158, Col 17

```

graph TD
    A[b_sailorsSID] --> B[Merge Semi Join]
    C[r_reservesBID] --> B

```

The screenshot shows the pgAdmin 4 interface with the following details:

- Servers:** Local [schema3/postgres@postgres]
- Query History:**

```

> 130 -- Query 7 (STATISTICS)
> 131 set enable_hashagg = off;
> 132 set enable_hashjoin = off;
> 133
> 134 explain analyze select s.sname
> 135 from sailors s
> 136 where
> 137 s.sid in( select r.sid
> 138 from reserves r
> 139 where r.bid = 103 );
> 140
> 141
> 142 -- Query 7 optimized (STATISTICS)
> 143
> 144 -- view of Query 7
> 145 set enable_hashagg = off;
> 146 set enable_hashjoin = off;
> 147
> 148 -- Query 7 view table of reserves with bid =103
> 149
> 150 create MATERIALIZED VIEW query_7
> 151 as
> 152 select r.sid
> 153 from reserves r
> 154 where r.bid =103 ;
> 155
> 156
> 157
> 158 explain analyze select s.sname
> 159 from sailors s
> 160 where exists (select R.sid
> 161     from query_7 R
> 162     where s.sid =R.sid);
> 163
> 164
> 165
> 166
> 167 -- ====== Query 8 ======
> 168
> 169 -- Find the names of sailors 'who ha'ue reserved a red boat.
> 170 -- Query 8 (COUNTING RESULT SET) , Number Of Rows = 673
Total rows: 7 of 7   Query complete 00:00:00.104

```
- Execute/Refresh (FS) button:** Located at the top of the query history panel.
- Query Plan:** A table showing the execution plan steps:

	QUERY PLAN
1	Merge Semi Join (cost=0.60..60.37 rows=582 width=21) (actual time=0.019..0.354 rows=582 loops=1)
2	Merge Cond: (s.sid = r.sid)
3	-> Index Scan using b_sailorssid on sailors s (cost=0.29..663.29 rows=19000 width=25) (actual time=0.005..0.093 rows=584 ..)
4	-> Index Only Scan using r_reservesbid on query_7 r (cost=0.28..31.00 rows=582 width=4) (actual time=0.011..0.129 rows=5..)
5	Heap Fetches: 582
6	Planning Time: 0.154 ms
7	Execution Time: 0.386 ms
- Text Editor:** Shows the SQL code for creating a materialized view and performing an explain analyze on it.
- Status Bar:** Shows "Ln 158, Col 1".

### Explanation :

- Metrics :

**Execution Time : 0.386 ms    Total Expected Cost : 60.37**

- The B-tree helped in the performance it decreased the Execution Time and Expected Cost .
- The Query Planner used the B-tree index because B-Tree is O(Log n) performance with Exact Values .
- Here it showed the improvement due to for (s.sid=R.sid (it used the index that was built on query\_7 view)) condition of Joining in the Query the Merge Semi Join used index scan using ZigZag and algorithm and these columns where built on it an b-tree index.

- given query with hash indices only :

Activities pgAdmin 4 Jun 9 3:32 PM ● pgAdmin 4

Servers Local PostgreSQL Data Query History

```

83 DROP INDEX IF EXISTS reserves_pkey cascade;
84
85
86
87
88
89
90 -- ====== Query 7 ======
91
92 -- Find the names of sailors who have reserved boat 103.
93 -- Query 7 COUNTING RESULT SET , Number Of Rows = 582
94
95
96 CREATE INDEX b_sailorsSID ON sailors USING hash(sid);
97 CREATE INDEX b_reservesSID ON reserves USING hash(reserveid);
98 CREATE INDEX b_reservesBID ON reserves USING hash(bid);
99 CREATE INDEX R_reservesBID ON query_7 USING hash(reserveid);

100 select *
101 from pg_indexes
102 where tablename = 'sailors' or tablename='reserves' or tablename='boat' or tab
103
104
105
106
107
108
109 select count(s.sname)
110 from sailors s
111 where
112 s.sid in (select r.sid
113 from reserves r
114 where r.bid = 103 );
115
116
117
118 -- Query 7 optimized (COUNTING RESULT SET) , Number Of Rows = 582
119
120
121 select count(s.sname)
122 from sailors s
123 where exists (select R.sid
124
Total rows: 4 of 4 Query complete 00:00:00.207

```

Data output Messages Explain Notifications

schemaname	tablename	indexname	tablespace	indexdef
public	sailors	b_sailorsSID	[null]	CREATE INDEX b_sailorsSID ON public.sailors USING hash (sid)
public	reserves	b_reservesSID	[null]	CREATE INDEX b_reservesSID ON public.reserves USING hash (reserveid)
public	reserves	b_reservesBID	[null]	CREATE INDEX b_reservesBID ON public.reserves USING hash (bid)
public	query_7	R_reservesBID	[null]	CREATE INDEX R_reservesBID ON public.query_7 USING hash (reserveid)

Activities pgAdmin 4 Jun 9 4:01 PM ● pgAdmin 4

Servers Local PostgreSQL Data Query History

```

150 create MATERIALIZED VIEW query_7
151 as
152 select r.sid
153 from reserves r
154 where r.bid=103 ;
155
156
157
158 explain analyze select s.sname
159 from sailors s
160 where exists (select R.sid
161 from query_7 R
162 where s.sid=R.sid);
163
164
165
166
167 -- ====== Query 8 ======
168
169 -- Find the names of sailors 'who ha've reserved a r
170 -- Query 8 (COUNTING RESULT SET) , Number Of Rows =
171
172 explain analyze select count(s.sname)
173 from sailors s
174 where s.sid in ( select r.sid
175 from reserves r
176 where r.bid in (select b.bid
177 from boat b
178 where b.color = 'red'));
179
180 -- Query 8 optimized (COUNTING RESULT SET) , Number
181
182
183 select count(s.sname)
184 from sailors s where exists (select * from query_8 r1 where s.sid=r1.sid);
185
186
187
188
189
190
Total rows: 1 of 1 Query complete 00:00:00.068

```

Explain Analyze (Shift+F7) Data output Messages Explain Notifications

Graphical Analysis Statistics

```

graph TD
    sailors[sailors] -->|Nested Loop Semi Join| r_reservesbid[r_reservesbid]

```

Successfully run. Total query runtime: 68 msec. 1 rows affected.

```

create MATERIALIZED VIEW query_7
as
select r.sid
from reserves r
where r.bid =103 ;
explain analyze select s.sname
from sailors s
where exists (select R.sid
from query_7 R
where s.sid =R.sid);
-- Find the names of sailors 'who ha'ue reserved a
-- Query 8 (COUNTING RESULT SET) , Number Of Rows =
explain analyze select count(s.sname)
from sailors s
where s.sid in ( select r.sid
from reserves r
where r.bid in (select b.bid
from boat b
where b.color = 'red'));
-- Query 8 optimized (COUNTING RESULT SET) , Number
-- select count(s.sname)
from sailors s where exists (select * from query_8 r1 where s.sid=r1.sid);

```

Total rows: 6 of 6    Query complete 00:00:00.049    Ln 157, Col 1

### Explanation :

- Metrics :

**Execution Time : 11.828 ms    Total Expected Cost : 715.32**

- The Hash helped in the performance it decreased the Execution Time (but in the execution time processor was overwhelmed) and Expected Cost .
- The Query Planner used the Hash index because Hash is O(1) performance with Exact Values and considered the best for exact values queries.
- Here it showed the improvement due to for every condition of Joining in the Query the Nested Loop Semi Join using index scan using Hash based algorithm on the condition (sid(from query\_7 view)=s.sid) which approximatly maded to be O(n) performance.

- given query with BRIN indices only :

Activities pgAdmin 4 Jun 9 4:32 PM ● pgAdmin 4

Servers schema3/postgres Local [ Data Query History Execute/Refresh ]

```

> p 84 DROP INDEX IF EXISTS reserve_7;
> s 85
> s 86
> s 87
> s 88
> s 89
> s 90 -- ====== Query 7 ======
> s 91 -- Find the names of sailors who have reserved boat 103.
> s 92 -- Query 7 COUNTING RESULT SET , Number Of Rows = 582
> s 93
> s 94
> s 95
> s 96 CREATE INDEX b_sailorsSID ON sailors USING BRIN(sid);
> s 97 CREATE INDEX b_reservesSID ON reserves USING BRIN(sid);
> s 98 CREATE INDEX b_reservesBID ON reserves USING BRIN(bid);
> s 99 CREATE INDEX R_reservesBID ON query_7 USING BRIN(bid);

> Log 100
> Tab 102
> pgsql 103 select *
from pg_indexes
where tablename = 'sailors' or tablename='reserves' or tablename='boat' or tablename='query_7';

104
105
106
107
108 DROP INDEX IF EXISTS b_sailorsSID cascade;
109 DROP INDEX IF EXISTS b_reservesSID cascade;
110 DROP INDEX IF EXISTS b_reservesBID cascade;
111 DROP INDEX IF EXISTS R_reservesBID cascade;
112
113
114
115
116 select count(s.sname)
from sailors s
117 where
118 s.sid in( select r.sid
119 from reserves r
120 where r.bid = 103 );
121
122
123
124

```

Total rows: 4 of 4 Query complete 00:00:00.070 Ln 96, Col 1

Activities pgAdmin 4 Jun 9 5:09 PM ● pgAdmin 4

Servers schema3/postgres Local [ Data Query History Explain analyze Shift (F7) ]

```

> p 143 explain analyze select s.sname
from sailors s
144 where
145 s.sid in( select r.sid
146 from reserves r
147 where r.bid = 103 );
148
149
150
151
152
153 -- Query 7 optimized (STATISTICS)
154 -- view of Query 7
155 set enable_hashagg = off;
156 set enable_hashjoin = off;
157
158 -- Query 7 view table of reserves with bid =103
159 create MATERIALIZED VIEW query_7
as
160
161 select r.sid
from reserves r
162 where r.bid =103 ;
163
164
165
166
167 explain analyze select s.sname
from sailors s
168 where exists (select R.sid
169
170
171
172
173
174
175
176 -- ====== Query 8 ======
177
178 -- Find the names of sailors 'who ha've reserved a red boat.
179 -- Query 8 (COUNTING RESULT SET) , Number Of Rows = 673
180
181 explain analyze select count(s.sname)
from sailors s
182 where s.sid in ( select r.sid
183
184

```

Graphical Analysis Statistics

```

graph LR
    A[query_7] --> B[Sort]
    B --> C[Unique]
    C --> D[Nested Loop Inner Join]
    E[b_sailorsSID] --> F[sailors]
    D --> F

```

Successfully run. Total query runtime: 769 msec. 1 rows affected. Ln 167, Col 16

```

explain analyze select s.sname
from sailors s
where
  s.sid in( select r.sid
    from reserves r
    where r.bid = 103 );
-- Query 7 optimized (STATISTICS)
set enable_hashagg = off;
set enable_hashjoin = off;
-- Query 7 view table of reserves with bid =103
create MATERIALIZED VIEW query_7
as
select r.sid
from reserves r
where r.bid =103 ;
explain analyze select s.sname
from sailors s
where exists( select R.sid
              from query_7 R
              where s.sid =R.sid);
-- ====== Query 8 ======
-- Find the names of sailors 'who ha'ue reserved a red boat.
-- Query 8 (COUNTING RESULT SET) , Number Of Rows = 673
explain analyze select count(s.sname)
from sailors s
where s.sid in ( select r.sid
    from reserves r
    Total rows: 18 of 18 Query complete 00:00:00.834

```

Successfully run. Total query runtime: 137 msec. 4 rows affected.  
Ln 167, Col 1

### Explanation :

- Metrics :

**Execution Time : 4679 ms    Total Expected Cost : 10004136191.55**

- Here the BRIN was not used in the original Query Plan settings (Hashjoin and HashAgg are off) so I've made seqscan=off too.
- The Execution Time and Expected Cost became the Worst of all .
- This happened because the Query Optimizer didnt used it from the first place due to BRIN Usage here was not suitable so we have used it to simulate seqscan behaviour only we traversed it all and followed all its pointers so it is worst index to use in this case.

- given query with mixed indices (any mix of your choice) :

pgAdmin 4

Jun 9 4:56 PM • pgAdmin 4

Activities

File Object Tools Help

Servers Local PostgreSQL

Data Query History

```

> p 84 DROP INDEX IF EXISTS reserve;
> s 85
> t 86
> r 87
> q 88
> e 89
> v 90
-- ====== Query 7 ======
> q 91
> q 92
-- Find the names of sailors who have reserved boat 103.
-- Query 7 COUNTING RESULT SET , Number Of Rows = 582
> q 93
> q 94
> q 95
> q 96 CREATE INDEX b_sailorsSID ON sailors USING btree(sid );
> q 97 CREATE INDEX b_reservesID ON reserves USING hash(sid );
> q 98 CREATE INDEX b_reservesBID ON reserves USING hash(bid );
> q 99 CREATE INDEX R_reservesBID ON query_7 USING btree(sid );
> s 100
> Log 101
> Tab 102
> pgsql 103 select * from pg_indexes
where tablename = 'sailors' or tablename='reserves' or tablename='boat' or tablename='query_7';
104
105
106
107
108 DROP INDEX IF EXISTS b_sailorsSID cascade;
109 DROP INDEX IF EXISTS b_reservesID cascade;
110 DROP INDEX IF EXISTS b_reservesBID cascade;
111 DROP INDEX IF EXISTS R_reservesBID cascade;
112
113
114
115
116 select count(s.sname)
from sailors s
where
s.sid in( select r.sid
from reserves r
where r.bid = 103 );
117
118
119
120
121
122
123
124
--
```

Total rows: 4 of 4    Query complete 00:00:00.056

Data output Messages Explain Notifications

schemaName	tableName	indexName	tableSpace	indexDef
public	sailors	b_sailorsSID	[null]	CREATE INDEX b_sailorsSID ON public.sailors USING btree (sid )
public	reserves	b_reserves...	[null]	CREATE INDEX b_reservesID ON public.reserves USING hash (sid )
public	reserves	b_reserves...	[null]	CREATE INDEX b_reservesBID ON public.reserves USING hash (bid )
public	query_7	R_reservesBID	[null]	CREATE INDEX R_reservesBID ON public.query_7 USING btree (sid )

Successfully run. Total query runtime: 137 msec. 4 rows affected.

Ln 95, Col 1

pgAdmin 4

Jun 9 4:59 PM • pgAdmin 4

Activities

File Object Tools Help

Servers Local PostgreSQL

Data Query History

```

> p 133
> s 134
> t 135
> r 136
> q 137
-- Query 7 (STATISTICS)
> q 138 set enable_hashagg = off;
> q 139 set enable_hashjoin = off;
> q 140
> q 141 explain analyze select s.sname
from sailors s
where
s.sid in( select r.sid
from reserves r
where r.bid = 103 );
> s 142
> Log 143
> Tab 144
> pgsql 145
-- Query 7 optimized (STATISTICS)
> q 146 set enable_hashagg = off;
> q 147 set enable_hashjoin = off;
> s 148
> Log 149
-- view of Query 7
> Tab 150
> pgsql 151
-- view of Query 7
set enable_hashagg = off;
set enable_hashjoin = off;
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
create MATERIALIZED VIEW query_7
as
select r.sid
from reserves r
where r.bid =103 ;
explain analyze select s.sname
from sailors s
where exists (select R.sid
from query_7 R
where s.sid =R.sid);
--
```

Total rows: 1 of 1    Query complete 00:00:00.067

Data output Messages Explain Statistics

Graphical Analysis Statistics

```

graph TD
    A[b_sailorssid] --> B[Merge Semi Join]
    C[r_reservesbid] --> B

```

Successfully run. Total query runtime: 67 msec. 1 rows affected.

Successfully run. Total query runtime: 137 msec. 4 rows affected.

Ln 165, Col 16

```

-- Query 7 (STATISTICS)
set enable_hashagg = off;
set enable_hashjoin = off;

explain analyze select s.sname
from sailors s
where
s.sid in( select r.sid
from reserves r
where r.bid = 103 );

-- Query 7 optimized (STATISTICS)
set enable_hashagg = off;
set enable_hashjoin = off;

-- view of Query 7
create MATERIALIZED VIEW query_7
as
select r.sid
from reserves r
where r.bid =103 ;

explain analyze select s.sname
from sailors s
where exists (select R.sid
              from query_7 R
              where s.sid =R.sid);

```

Total rows: 7 of 7    Query complete 00:00:00.047

Successfully run. Total query runtime: 0.392 ms. 4 rows affected.  
Ln 165, Col 1

### Explanation :

- Metrics :

**Execution Time : 0.392 ms    Total Expected Cost : 60.37**

- The Query Planner used the Merge join by using both the ZigZag join and the Hash based algorithm together which improved the Execution time and the expected cost way more better which made the join in O( $n \log n$ ).