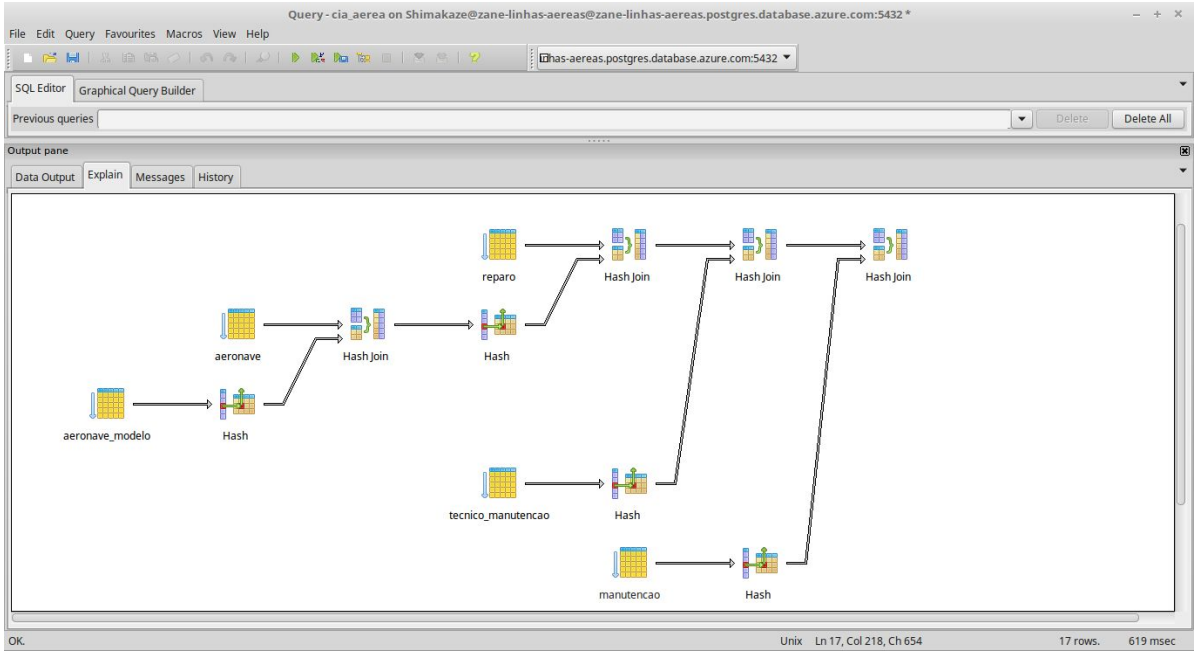
Turma 04

Marcos Henrique Monteiro da Silva	8802392
Marcos Vinicius Oliveira Lunardelli	9019302
Peterson Medeiros Santos	9004550

Reescrevendo uma consulta

ORIGINAL

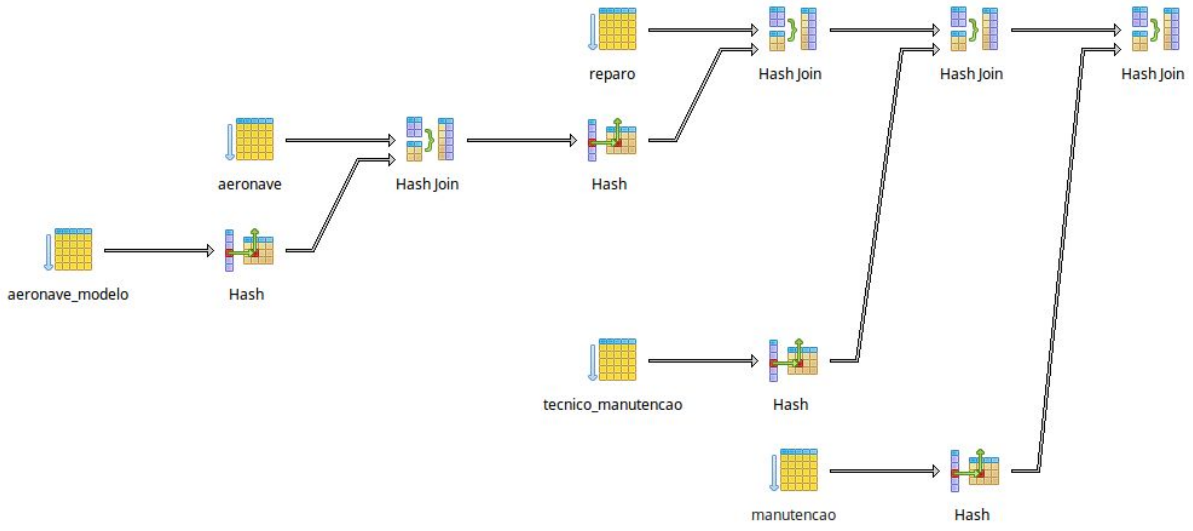
```
SELECT aeronave.avi_serial_number,
       aeronave.avi_matricula,
       aeronave.avi_mod_modelo,
       aeronave.avi_categoria,
       aeronave_modelo.avi_mod_capacidade,
       tecnico_manutencao.pes_cpf,
       tecnico_manutencao.tec_anac,
       tecnico_manutencao.tec_tipo_contrato,
       manutencao.man_nome,
       manutencao.man_desricao,
       reparo.rep_orcamento
  FROM aeronave,
       aeronave_modelo,
       reparo,
       tecnico_manutencao,
       manutencao
 WHERE aeronave_modelo.avi_mod_modelo = aeronave.avi_mod_modelo AND
reparo.avi_serial_number = aeronave.avi_serial_number AND
tecnico_manutencao.pes_cpf = reparo.tec_cpf AND manutencao.man_codigo =
reparo.man_codigo;
```



	QUERY PLAN text
1	Hash Join (cost=5.24..7.58 rows=44 width=1004) (actual time=0.163..0.199 rows=45 loops=1)
2	Hash Cond: (reparo.man_codigo = manutencao.man_codigo)
3	-> Hash Join (cost=4.15..6.28 rows=44 width=360) (actual time=0.101..0.127 rows=45 loops=1)
4	Hash Cond: (reparo.tec_cpf = tecnico_manutencao.pes_cpf)
5	-> Hash Join (cost=3.02..4.92 rows=44 width=192) (actual time=0.076..0.091 rows=45 loops=1)
6	Hash Cond: (reparo.avi_serial_number = aeronave.avi_serial_number)
7	-> Seq Scan on reparo (cost=0.00..1.44 rows=44 width=16) (actual time=0.007..0.008 rows=45 loops=1)
8	-> Hash (cost=2.64..2.64 rows=30 width=180) (actual time=0.057..0.057 rows=30 loops=1)
9	Buckets: 1024 Batches: 1 Memory Usage: 11kB
10	-> Hash Join (cost=1.16..2.64 rows=30 width=180) (actual time=0.037..0.049 rows=30 loops=1)
11	Hash Cond: (aeronave.avi_mod_modelo = aeronave_modelo.avi_mod_modelo)
12	-> Seq Scan on aeronave (cost=0.00..1.30 rows=30 width=176) (actual time=0.007..0.009 rows=30 loops=1)
13	-> Hash (cost=1.07..1.07 rows=7 width=108) (actual time=0.012..0.012 rows=7 loops=1)
14	Buckets: 1024 Batches: 1 Memory Usage: 9kB
15	-> Seq Scan on aeronave_modelo (cost=0.00..1.07 rows=7 width=108) (actual time=0.006..0.009 rows=7 loops=1)
16	-> Hash (cost=1.06..1.06 rows=6 width=172) (actual time=0.014..0.014 rows=6 loops=1)
17	Buckets: 1024 Batches: 1 Memory Usage: 9kB
18	-> Seq Scan on tecnico_manutencao (cost=0.00..1.06 rows=6 width=172) (actual time=0.009..0.010 rows=6 loops=1)
19	-> Hash (cost=1.04..1.04 rows=4 width=652) (actual time=0.050..0.050 rows=4 loops=1)
20	Buckets: 1024 Batches: 1 Memory Usage: 9kB
21	-> Seq Scan on manutencao (cost=0.00..1.04 rows=4 width=652) (actual time=0.020..0.021 rows=4 loops=1)
22	Planning time: 0.479 ms
23	Execution time: 0.309 ms

NOVA

```
SELECT * FROM
    (SELECT aeronave.*
     FROM aeronave, aeronave_modelo
     WHERE aeronave_modelo.avi_mod_modelo = aeronave.avi_mod_modelo) AS aero
INNER JOIN
    (SELECT tecnico_manutencao.*, manutencao.man_nome, manutencao.man_desricao,
reparo.rep_orcamento, reparo.avi_serial_number
     FROM tecnico_manutencao, manutencao, reparo
     WHERE tecnico_manutencao.pes_cpf = reparo.tec_cpf AND
manutencao.man_codigo = reparo.man_codigo)
     AS manu ON manu.avi_serial_number = aero.avi_serial_number;
```



	QUERY PLAN text
1	Hash Join (cost=5.24..7.58 rows=44 width=1004) (actual time=0.151..0.189 rows=45 loops=1)
2	Hash Cond: (reparo.man_codigo = manutencao.man_codigo)
3	-> Hash Join (cost=4.15..6.28 rows=44 width=360) (actual time=0.110..0.136 rows=45 loops=1)
4	Hash Cond: (reparo.tec_cpf = tecnico_manutencao.pes_cpf)
5	-> Hash Join (cost=3.02..4.92 rows=44 width=192) (actual time=0.069..0.085 rows=45 loops=1)
6	Hash Cond: (reparo.avi_serial_number = aeronave.avi_serial_number)
7	-> Seq Scan on reparo (cost=0.00..1.44 rows=44 width=16) (actual time=0.007..0.009 rows=45 loops=1)
8	-> Hash (cost=2.64..2.64 rows=30 width=176) (actual time=0.050..0.050 rows=30 loops=1)
9	Buckets: 1024 Batches: 1 Memory Usage: 11kB
10	-> Hash Join (cost=1.16..2.64 rows=30 width=176) (actual time=0.031..0.042 rows=30 loops=1)
11	Hash Cond: (aeronave.avi_mod_modelo = aeronave_modelo.avi_mod_modelo)
12	-> Seq Scan on aeronave (cost=0.00..1.30 rows=30 width=176) (actual time=0.006..0.009 rows=30 loops=1)
13	-> Hash (cost=1.07..1.07 rows=7 width=104) (actual time=0.011..0.011 rows=7 loops=1)
14	Buckets: 1024 Batches: 1 Memory Usage: 9kB
15	-> Seq Scan on aeronave_modelo (cost=0.00..1.07 rows=7 width=104) (actual time=0.006..0.007 rows=7 loops=1)
16	-> Hash (cost=1.06..1.06 rows=6 width=172) (actual time=0.031..0.031 rows=6 loops=1)
17	Buckets: 1024 Batches: 1 Memory Usage: 9kB
18	-> Seq Scan on tecnico_manutencao (cost=0.00..1.06 rows=6 width=172) (actual time=0.007..0.008 rows=6 loops=1)
19	-> Hash (cost=1.04..1.04 rows=4 width=652) (actual time=0.026..0.026 rows=4 loops=1)
20	Buckets: 1024 Batches: 1 Memory Usage: 9kB
21	-> Seq Scan on manutencao (cost=0.00..1.04 rows=4 width=652) (actual time=0.020..0.021 rows=4 loops=1)
22	Planning time: 0.514 ms
23	Execution time: 0.297 ms

Nota-se que não houveram diferenças entre as duas execuções. O código novo foi alterado no sentido de tentar diminuir o tamanho das tabelas a serem analisadas antes de cada join (através do SELECT), mas o próprio SGBD busca o melhor caminho de execução em ambos os casos, tenho o mesmo desempenho e usando as mesmas ferramentas que o código anterior. Vale notar que um volume de dados maior poderia alterar o caminho que o SGBD busca para otimizar o processo, mas que esse caminho seria o mesmo no caso dos dois algoritmos.