## QUERY 1 – INTERVENÇÃO NO BANCO DE DADOS

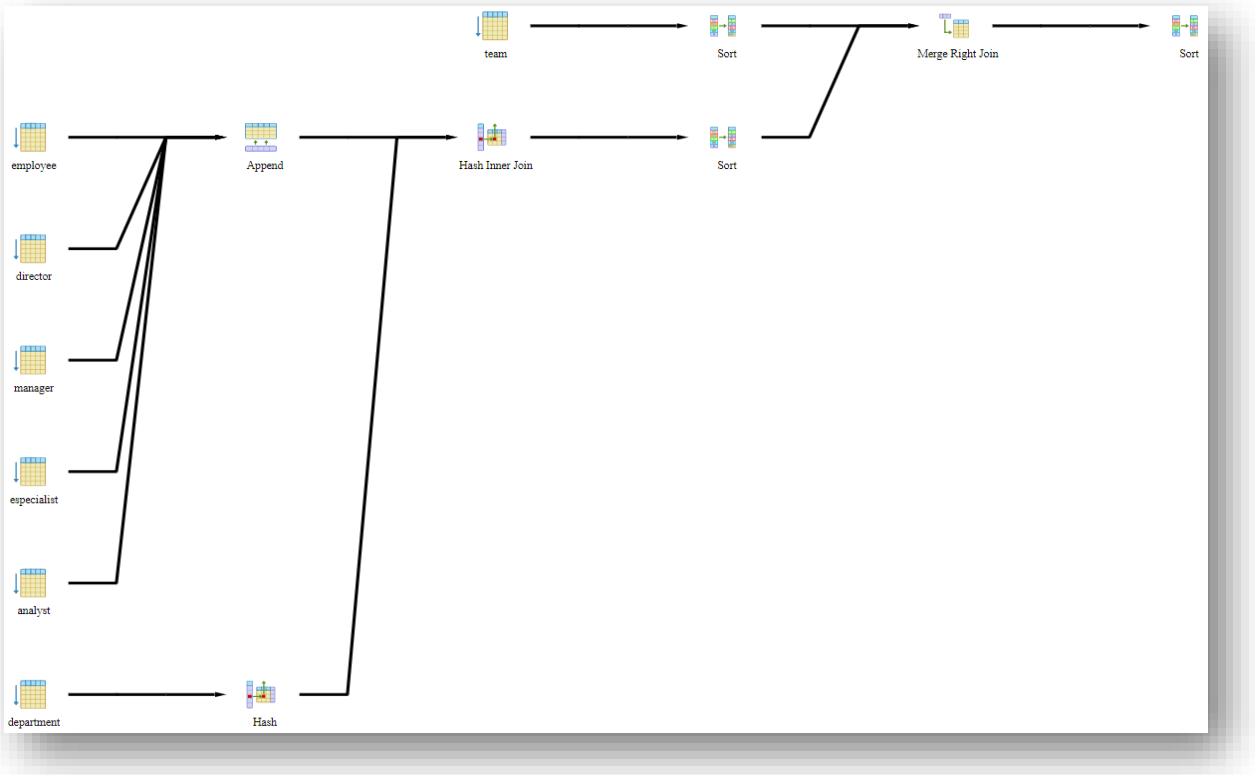
```
/* Query 1 correspondente apos modificacoes realizadas no modelo e apos o artefato A
*   Pelo fato de que agora apenas um analista ou especialista possuem nivel de conhecimento,
* tais atributos e tabelas foram tirados da consulta a fim de evitar muitos registros 'nulos'.
*/

explain SELECT a.idt_employee, a.name_employee, t.idt_team,  t.name_team
FROM(
    SELECT e.name_employee, e.idt_employee, d.idt_project
    FROM Employee as e
    INNER JOIN Department as d ON d.idt_department  = e.idt_department
) as a
LEFT JOIN Team as t ON t.idt_project = a.idt_project
ORDER BY t.name_team;
```

Consulta

Plano de Consulta:

| | QUERY PLAN text |
|---|---|
| 1 | Sort  (cost=914.30..934.71 rows=8166 width=72) |
| 2 | Sort Key: t.name_team |
| 3 | -> Merge Right Join  (cost=255.20..383.69 rows=8166 width=72) |
| 4 | Merge Cond: (t.idt_project = d.idt_project) |
| 5 | -> Sort  (cost=83.37..86.37 rows=1200 width=40) |
| 6 | Sort Key: t.idt_project |
| 7 | -> Seq Scan on team t  (cost=0.00..22.00 rows=1200 width=40) |
| 8 | -> Sort  (cost=171.83..175.23 rows=1361 width=40) |
| 9 | Sort Key: d.idt_project |
| 10 | -> Hash Join  (cost=37.00..100.99 rows=1361 width=40) |
| 11 | Hash Cond: (e.idt_department = d.idt_department) |
| 12 | -> Append  (cost=0.00..60.41 rows=1361 width=40) |
| 13 | -> Seq Scan on employee e  (cost=0.00..0.00 rows=1 width=40) |
| 14 | -> Seq Scan on director e_1  (cost=0.00..13.40 rows=340 width=40) |
| 15 | -> Seq Scan on manager e_2  (cost=0.00..13.40 rows=340 width=40) |
| 16 | -> Seq Scan on especialist e_3  (cost=0.00..13.40 rows=340 width=40) |
| 17 | -> Seq Scan on analyst e_4  (cost=0.00..13.40 rows=340 width=40) |
| 18 | -> Hash  (cost=22.00..22.00 rows=1200 width=8) |
| 19 | -> Seq Scan on department d  (cost=0.00..22.00 rows=1200 width=8) |

Árvore de Consulta:



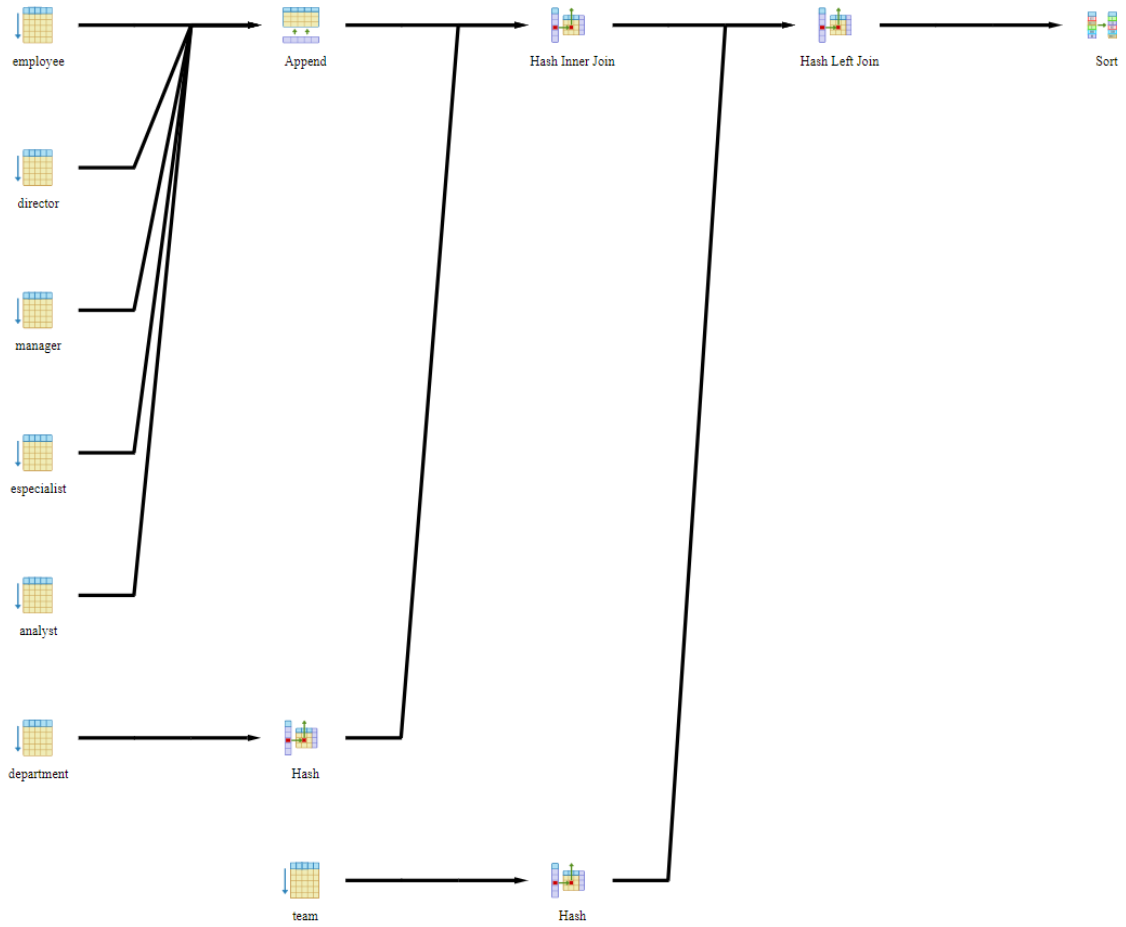

## QUERY 1: CRIAÇÃO DE ÍNDICES

```
CREATE INDEX indicePrimeiraQuery
 ON Employee (idt_employee, name_employee);

 CREATE INDEX indicePrimeiraQueryTeam
 ON Team (idt_team, name_team);
```

Relatório do plano de consulta a partir do comando explain do Postgress:

```
1  Sort  (cost=178.39..181.79 rows=1361 width=72)
2    Sort Key: t.name_team
3    ->  Hash Left Join  (cost=38.11..107.55 rows=1361 width=72)
4          Hash Cond: (d.idt_project = t.idt_project)
5          ->  Hash Join  (cost=37.00..100.99 rows=1361 width=40)
6                Hash Cond: (e.idt_department = d.idt_department)
7                ->  Append  (cost=0.00..60.41 rows=1361 width=40)
8                      ->  Seq Scan on employee e  (cost=0.00..0.00 rows=1 width=40)
9                      ->  Seq Scan on director e_1  (cost=0.00..13.40 rows=340 width=40)
10                     ->  Seq Scan on manager e_2  (cost=0.00..13.40 rows=340 width=40)
11                     ->  Seq Scan on especialist e_3  (cost=0.00..13.40 rows=340 width=40)
12                     ->  Seq Scan on analyst e_4  (cost=0.00..13.40 rows=340 width=40)
13               ->  Hash  (cost=22.00..22.00 rows=1200 width=8)
14                     ->  Seq Scan on department d  (cost=0.00..22.00 rows=1200 width=8)
15         ->  Hash  (cost=1.05..1.05 rows=5 width=40)
16               ->  Seq Scan on team t  (cost=0.00..1.05 rows=5 width=40)
```

Árvore do plano de consulta a partir do comando explain do Postgress:




## Comparação antes e depois da alteração

A consulta realizada na primeira parte do trabalho foi modificada, dando origem ao que está sendo analisado, tal query seleciona o ID do funcionário, seu nome, ID do time e o nome do time. Analisando o relatório antes da alteração – inclusão do index – vemos através do comando explain a progressão da consulta, primeiramente com muitos sorts, tendo assim um custo muito acima do desejável para tal iteração, percebe-se que há inclusive apenas duas buscas sequenciais, sendo assim seria melhor uma saída com mais destas, pensando numa possível busca com muitos valores já que nos estado atual seria insustentável devido ao extremo custo imposto.

Portanto, houve a criação de dois índices, “indicePrimeiraQuery” e “indicePrimeiraQueryTeam”, “indexando” os atributos “idt_employee e “name_employee”, assim como “idt_team” e “name_team”, respectivamente, o que leva ao segundo resultado exposto com o comando explain, é possível perceber rapidamente a diferença, seja na diferença dos passos como na diferença notável no custo. O número de sorts caiu pela metade, assim como seu custo, como visto na primeira linha de ambos, uma melhoria de mais de 60% apenas nesse primeiro momento, não só mas como também é possível ver que o número de buscas sequenciais triplicou, seguindo também o plano inicial para diminuir o custo, mostrando que os dois índices revelaram-se essenciais.

Por fim, pode-se concluir que após a inclusão dos dois index que consultas em cargas maiores de dados não se mostraram tão custosas quanto ao modelo inicial, sendo assim, pode-se incluir muito mais funcionários e times ao banco de dados sem maiores problemas no futuro, ao menos comparado com os resultados iniciais, provando também a necessidade da análise dos dados para que sejam criados índices corretos que realmente façam diferença, como é o caso.