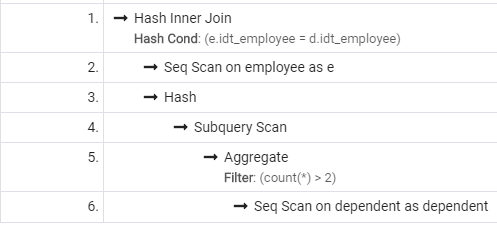
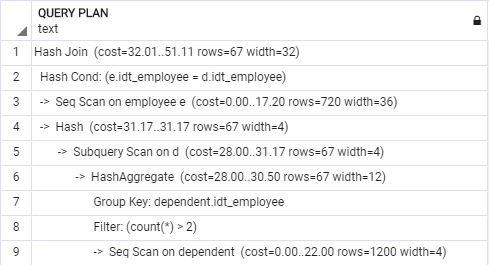
**Comando EXPLAIN sem alteração no modelo e sem melhoria QUERY 4**

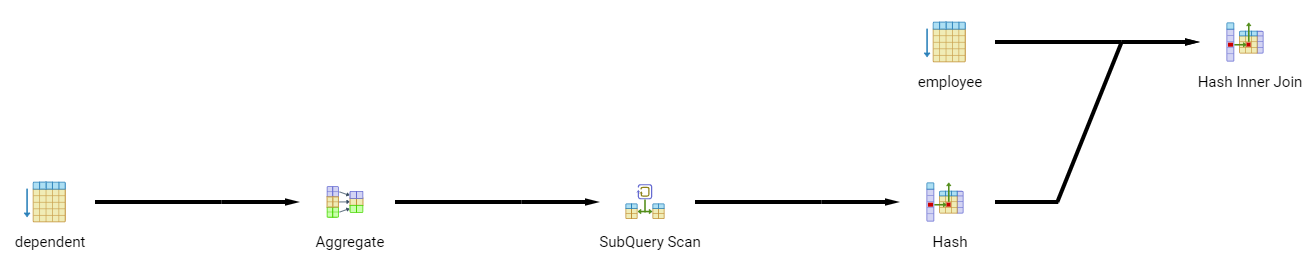
**Uma imagem contendo mesa

Descrição gerada automaticamente**

**Consulta executada com o comando EXPLAIN**

****

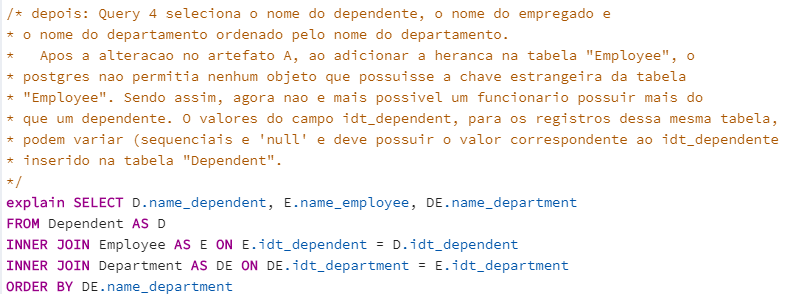
**Plano de consulta executada com o comando EXPLAIN**

****

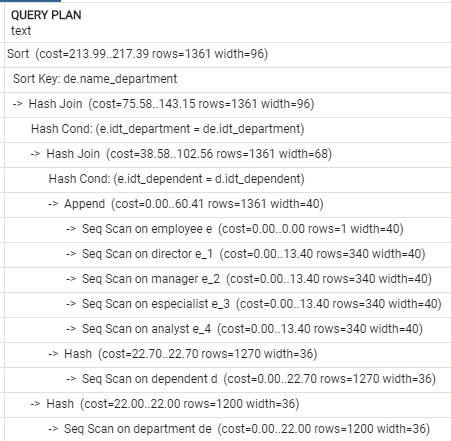
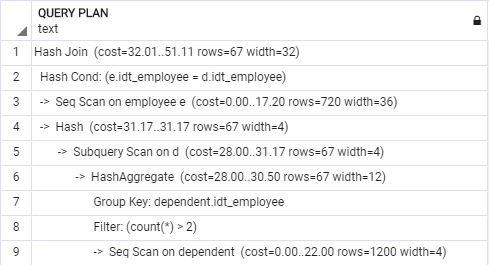
**Árvore de consulta executada com o comando EXPLAIN**

O plano de execução acima é feito na consulta que seleciona o empregado que possui mais de dois dependentes. Apresenta a busca sequencial na tabela principal “Employee”, e possui uma função de agregação na “SubQuery” através da cláusula “WHERE”. Se a tabela tiver um volume maior de dados a busca sequencial não seria uma vantagem, aqui ela conta com a ajuda da cláusula “WHERE” o que diminui o custo dessa busca, mas ainda assim ,no SGBD PostgreSQL, com base nas estatísticas que o comando “EXPLAIN” nos retorna, torna-se difícil validar se realmente esse custo irá diminuir, e, como comentado acima, se essas tabelas tivessem um volume de dados muito grande o processo seria ainda seria muito custoso e demoraria muito tempo para ser executado.

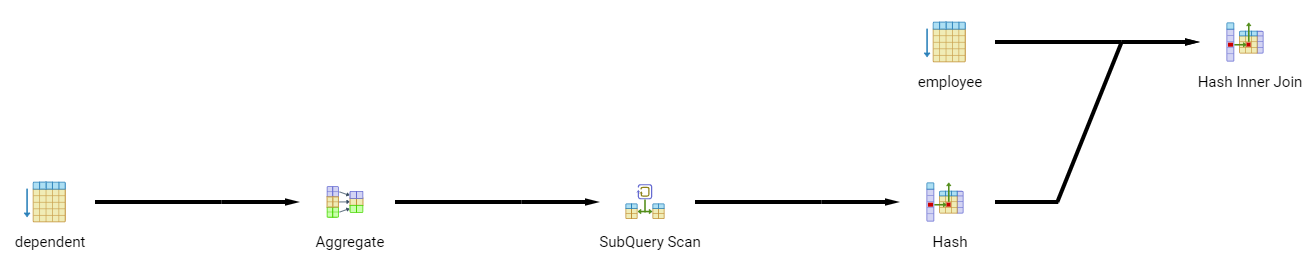
**Comando EXPLAIN com alteração no modelo e sem melhoria na QUERY 4**

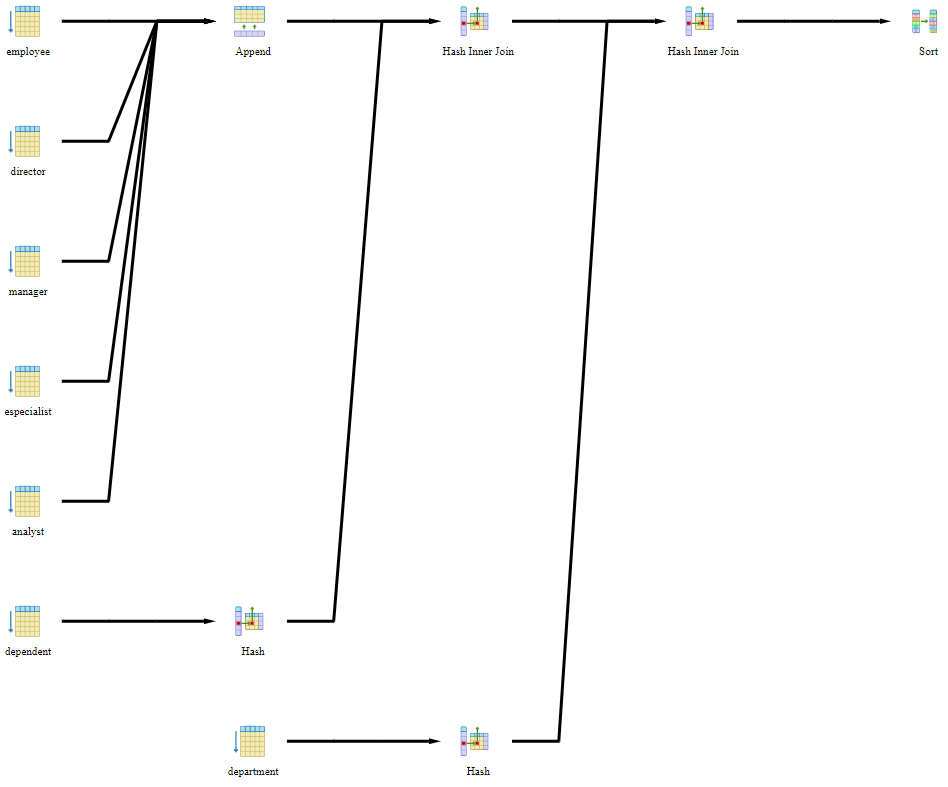
****

**Consulta alterada e executada com o comando EXPLAIN**

****

**Comparação entre os planos de consultas antes x depois da alteração do modelo**

****



**Comparação entre as árvores de consultas antes x depois da alteração do modelo**

O plano de execução para a consulta realizada na “Parte 1” do trabalho seleciona o empregado que possui mais de dois dependentes. Apresenta a busca sequencial na tabela principal “Employee”, e possui uma função de agregação na “SubQuery” (comentado anteriormente no relatório da “Query 3”) através da cláusula “WHERE”. Se a tabela tiver um volume maior de dados a busca sequencial não seria uma vantagem, aqui ela conta com a ajuda da cláusula “WHERE” o que diminui o custo dessa busca, mas ainda assim ,no SGBD PostgreSQL, com base nas estatísticas que o comando “EXPLAIN” nos retorna, torna-se difícil validar se realmente esse custo irá diminuir, e, como comentado acima, se essas tabelas tivessem um volume de dados muito grande o processo seria ainda seria muito custoso e demoraria muito tempo para ser executado.

Após a alteração do artefato A, e da inclusão da herança, o PostgreSQL não permitia a adição da chave estrangeira derivada do “Pai” de uma herança. Ou seja, “Dependent” já não poderia mais satisfazer uma das solicitações da “Parte 1” do trabalho, de possuir ao menos uma entidade-fraca. Nesse caso, realizamos ajustes para que a tabela “Team” continue cumprindo com o requisito.

Visto as alterações realizadas em consultas anteriores, tentamos alterar coisas já observadas anteriormente. Como na primeira consulta é realizado um filtro, derivado de uma “SUBQUERY”, também comentada anteriormente, optamos por modificações que tornassem a consulta simples e que tirasse características já observadas.

Feita as alterações, a segunda consulta retorna o nome do dependente, o nome do empregado e o nome do departamento (e ordenado por este último) por meio de junções (“joins”). Desse ela realiza uma busca sequencial (“seq scan”) por cada tabela que herda a tabela “Employee”, utilizando o campo “idt\_employee”. Por mais que tenhamos tirado subconsultas e filtros, o custo diminuía conforme execução e é possível notar que o maior “impacto”, na segunda consulta em comparativo com a primeira consulta, foi o número de linhas afetadas. A janela também teve um aumento, provavelmente não tão significativo, visto que a quantidade de linhas afetadas mais que “triplicou”.