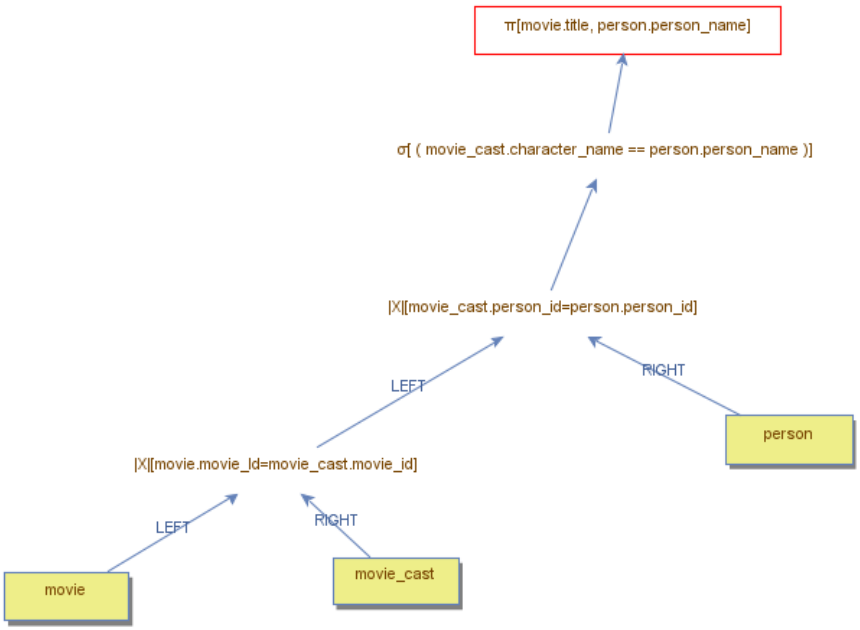
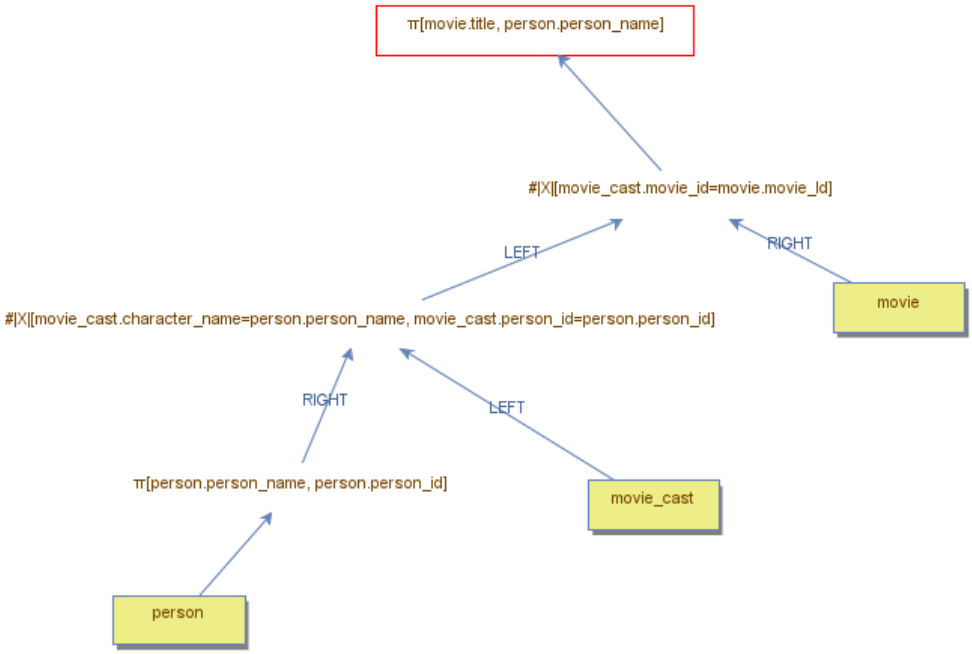
Aluno: Diego Rockenbach

Consulta com *Nested Loop Join:*



Consulta com *Hash Join:*



Comparação de desempenho (à esquerda *Nested Loop Join*, à direita *Hash Join*):

Total tuples loaded: 52		
	π [movie.title, person.p...	π [movie.title, person.p..
Tuples loaded	26	26
Accessed blocks	7857	697
Loaded blocks	0	0
Saved blocks	0	0
Filter comparations	3179	0
Memory Used	0	346840
Next Calls	13008	13497
Primary key searches	3419	0
Records Read	10015	8419
Sorted tuples	0	0

Como podemos observar, o desempenho usando *Hash Join* é muito superior se comparado ao desempenho utilizando *Nested Loop Join*, e isto se dá por na segunda operação aplicarmos os filtros de *"movie_cast.character_name == person.person_name"* muito mais cedo, reduzindo drasticamente o número de tuplas intermediárias. O *Hash Join* elimina muitas leituras redundantes e repetidas e torna a execução muito mais rápida, mesmo sem usar índices sobre as colunas envolvidas. Isto leva com que precisemos acessar menos blocos de memória, reduzindo o IO em disco. Apesar disso, vale citar que sua performance vem a um custo, visto que seu uso de memória é muito maior, de modo que o *Nested Loop Join* possa ser mais vantajoso em situações de baixo volume de dados ou sistemas com menos memória disponível.