Trabalho Prático Implementação Cuckoo-Hashing

Autor

Rubens Zandomenighi Laszlo GRR20206147

Objetivo

- Implementação algoritmo Cuckoo-Hashing conforme especificado no arquivo *especificacao.md*.

Hipótese

O armazenamento em Hash garante operações de junção e where de forma otimizada.

Pergunta de pesquisa

- Como realizar otimização em operações de busca ?

Contribuição/Experimentos

-

Detalhes implementação

- Para o formato do Result_Set utilizei um formato personalido de Matriz, utilizando UNION para a definição do tipo do elemento, para este projeto considerei válidos inicialmente os formatos de INTEGER e VARCHAR, sendo um elemento de forma genérica, possibilitando utilização dentre os formatos especificados.
- Utilizei um formato de armazenamento colunar na matriz de resultados, sendo que cada linha da matriz contém os valores referentes a uma mesma coluna.
- Utilizei a função QSORT da glibc para ordenação da matriz.

Conclusão

- Conforme os requisitos especificados:

    1. Não temos duplicação de valores.
    2. Em caso de colisão, armazenar em outra tabela.

- Sendo assim, dado que no trabalho temos apenas 2 tabelas de armazenamento, sem colisão, é possível verificar que nas funções de busca por valores, é necessário apenas no máximo 2

passos.

- Porém vale notar que estamos abstraindo complexidades no custo de armazenamento e manutenção das tabelas, trabalhando apenas com 2 tabelas/2 funções Hashes e ignorando casos de valores duplicados. Conforme necessidade particionamento dos dados em mais tabelas, pode gera um alto custo do armazenamento dessas tabelas Hashes, aumentando o custo de escrita, porém diminuindo o custo de leitura utilizando este formato de armazenamento.
- Considero essa estratégia extremamente válida no contexto de otimização por possíveis índices de tabelas, utilizando uma estratégia de armazenamento mista, para campos que não necessariamente são utilizados nas operações citadas como otimizadas nesse formato.