Índice Hash Estático

‘main.py’:

Importações

import tkinter as tk: Importa a biblioteca tkinter, que é uma interface padrão do Python para o toolkit Tk GUI. O as tk é um alias para simplificar a chamada de métodos da biblioteca.
from gui import setup_gui: Importa a função setup_gui do módulo gui. Presume-se que gui.py é um arquivo separado que contém esta função, responsável por configurar os elementos da GUI.

Função main

A função main é o ponto de entrada para o programa e executa as seguintes etapas para criar e mostrar a GUI:

root = tk.Tk(): Cria o widget principal da tkinter, que serve como a janela raiz da aplicação GUI. Este widget é geralmente o ponto de partida para a criação de uma aplicação tkinter.
setup_gui(root): Chama a função setup_gui e passa o widget root como argumento. Esta função é definida no módulo gui e é responsável por adicionar todos os elementos da GUI necessários (como botões, campos de texto, etc.) ao widget root. A função setup_gui é onde a maior parte da configuração da interface é feita.
root.mainloop(): Inicia o loop principal da GUI. Este loop é o que mantém a janela aberta, escutando eventos (como cliques de botão e entradas de teclado). É um loop de espera que mantém a aplicação em execução até que o usuário feche a janela da GUI.

Bloco if __name__ == "__main__":

Este bloco condicional verifica se o script está sendo executado diretamente (ou seja, não está sendo importado como um módulo em outro script). Se for verdade, a função main é chamada. Este padrão é uma prática comum em Python para adicionar código que só deve ser executado quando o script é o ponto de entrada do programa, o que ajuda a organizar o código e facilita a reutilização em outros scripts como um módulo.

‘gui.py’:

Classe HashIndexGUI

Construtor (__init__): Inicializa a GUI. root é o widget principal da tkinter, que atua como a janela da aplicação. Uma instância de Tabela é criada com um tamanho de página padrão de 100. O método setup_gui é chamado para configurar os elementos da GUI.
setup_gui: Define o título e o tamanho da janela principal e chama outros métodos para configurar diferentes partes da interface, como área de carregamento de dados, definição do tamanho da página, busca, table scan e exibição de estatísticas.

Métodos de Configuração da GUI

Cada um desses métodos configura uma parte específica da interface:

setup_load_frame: Configura a área para carregar dados, incluindo um botão que, quando clicado, chama load_data.
setup_page_size_frame: Permite ao usuário definir o tamanho da página por meio de um campo de entrada.
setup_search_frame: Contém um campo de entrada e um botão para realizar buscas no índice hash.
setup_table_scan_frame: Configura a área para realizar um table scan, permitindo ao usuário especificar um limite para o número de registros a serem exibidos.
setup_stats_display: Configura a área para exibir estatísticas sobre o índice hash, como o total de entradas e a taxa de colisões.

Métodos de Ação
load_data: Abre uma caixa de diálogo para o usuário selecionar um arquivo de texto. O arquivo é lido, e cada linha é adicionada como uma tupla na tabela. O índice hash é construído com base nesses dados.
set_page_size: Atualiza o tamanho da página com base na entrada do usuário e exibe uma mensagem informativa.
search: Realiza uma busca no índice hash usando a chave fornecida pelo usuário e exibe os resultados.
table_scan: Realiza um table scan limitado, exibindo as tuplas até o limite especificado pelo usuário.
show_statistics: Calcula e exibe estatísticas sobre o índice hash, como o total de entradas e a taxa de colisões.

Função setup_gui
Fora da classe, a função setup_gui é definida para instanciar a HashIndexGUI e configurá-la no widget root.

Bloco if __name__ == "__main__":
Quando o script é executado diretamente (não importado como um módulo), uma nova janela tkinter é criada e a GUI é configurada e exibida chamando setup_gui.

‘data_sctructures.py’:

Classe Tupla
Propósito: Representa um único registro ou linha de dados, contendo uma chave de busca e os dados associados.

Atributos:
chave: Identificador único para a tupla, usado na função hash para alocar a tupla em um bucket específico.
dados: Informações associadas à chave, que podem ser de qualquer tipo ou estrutura.

Classe Pagina
Propósito: Simula uma página de memória ou de armazenamento em disco, limitando o número de tuplas que podem ser armazenadas juntas, o que é uma prática comum em sistemas de banco de dados para otimizar o acesso e a leitura dos dados.

Atributos:

tamanho_max: O número máximo de tuplas que a página pode conter.
tuplas: Uma lista das tuplas armazenadas na página.

Métodos:
adicionar_tupla(tupla): Tenta adicionar uma tupla à página. Se a página já estiver cheia (atingiu tamanho_max), a tupla não será adicionada e o método retorna False.

Classe Bucket
Propósito: Representa um bucket no índice hash, que pode armazenar múltiplas tuplas que têm a mesma chave hash (quando colisões ocorrem).

Atributos:
entradas: Uma lista das tuplas armazenadas no bucket.

Métodos:
adicionar_entrada(tupla): Adiciona uma tupla ao bucket sem verificar o tamanho ou possíveis colisões, pois a resolução de colisões não é explicitamente tratada neste método.

Classe Tabela
Propósito: Agrega todas as tuplas e organiza-as em páginas e buckets, formando a estrutura completa do índice hash.

Atributos:
tamanho_pagina: Define o número máximo de tuplas por página.
paginas: Uma lista das páginas que contêm as tuplas.
buckets: Uma lista dos buckets usados no índice hash.
num_buckets: O número total de buckets na tabela.

Métodos:
adicionar_tupla(tupla): Adiciona uma tupla à tabela, criando uma nova página se necessário.
inicializar_buckets(num_buckets): Inicializa a lista de buckets com base em um número especificado.
funcao_hash(chave): Uma função hash simples que determina o índice do bucket para uma chave dada.
construir_indice(): Constrói o índice hash mapeando cada tupla para um bucket com base na sua chave.
buscar(chave): Busca tuplas com a chave fornecida e retorna uma lista de tuplas e suas referências de página.
table_scan(limite): Realiza um table scan limitado, retornando as primeiras limite tuplas.
calcular_estatisticas(): Calcula estatísticas sobre o índice, como total de entradas e taxa de colisões.
calcular_num_buckets(): Determina o número ideal de buckets com base no número total de tuplas e um fator de carga desejado.
encontrar_pagina_ref(tupla): Encontra a referência da página para uma dada tupla.