Nama : Restu Wibisono
NPM : 2340506061

TABEL HASH

I. Pengertian

Tabel hash adalah struktur data yang digunakan dalam pemrograman komputer untuk menyimpan dan mengakses data dengan efisien. Tabel hash terdiri dari array atau daftar elemen yang disebut “slot” atau “bucket”. Setiap slot memiliki indeks numerik yang unik yang disebut “hash code”.

Proses penyimpanan dan pencarian data dalam tabel hash melibatkan dua langkah utama. Pertama, data diubah menjadi hash code menggunakan fungsi hash. Fungsi hash ini mengonversi data menjadi indeks numerik yang digunakan untuk menentukan lokasi penyimpanan dalam tabel hash. Kedua, data disimpan dalam slot yang sesuai dengan hash code yang dihasilkan.

Keuntungan utama dari tabel hash adalah kemampuannya untuk mencari dan mengakses data dengan cepat. Dalam kasus yang ideal, pencarian dan penyimpanan data dalam tabel hash memiliki kompleksitas waktu konstan O(1), yang berarti waktu yang dibutuhkan tidak tergantung pada jumlah data yang ada dalam tabel.

Namun, dalam beberapa kasus, terjadi “tabrakan hash” atau “collision” di mana dua atau lebih data memiliki hash code yang sama. Untuk menangani tabrakan hash, teknik seperti “chaining” atau “open addressing” digunakan. Dalam chaining, setiap slot tabel hash berfungsi sebagai daftar terkait yang menyimpan semua data dengan hash code yang sama. Dalam open addressing, jika terjadi tabrakan hash, data disimpan dalam slot lain yang tersedia dalam tabel.

II. Istilah-istilah

Dalam tabel Hash, terdapat beberapa istilah yang umum digunakan, antara lain:

1. Slot: Setiap elemen dalam tabel hash disebut slot atau bucket. Setiap slot memiliki indeks numerik yang unik yang digunakan untuk mengakses dan menyimpan data.
2. Hash code: Hash code adalah nilai numerik yang dihasilkan dari fungsi hash. Hash code digunakan untuk menentukan lokasi penyimpanan data dalam tabel hash.
3. Fungsi hash: Fungsi hash adalah fungsi yang mengonversi data menjadi nilai hash code. Fungsi ini harus menghasilkan hash code yang unik untuk setiap data yang berbeda, sehingga meminimalkan kemungkinan terjadinya tabrakan hash.
4. Tabrakan hash (collision): Tabrakan hash terjadi ketika dua atau lebih data memiliki hash code yang sama. Ini dapat terjadi karena fungsi hash yang tidak sempurna atau karena ukuran tabel hash yang terbatas.
5. Chaining: Chaining adalah teknik penanganan tabrakan hash di mana setiap slot tabel hash berfungsi sebagai daftar terkait (linked list) yang menyimpan semua data dengan hash code yang sama. Ketika terjadi tabrakan hash, data ditambahkan ke daftar terkait yang sesuai.
6. Open addressing: Open addressing adalah teknik penanganan tabrakan hash di mana jika terjadi tabrakan hash, data disimpan dalam slot lain yang tersedia dalam tabel. Beberapa metode open addressing termasuk linear probing, quadratic probing, dan double hashing.
7. Load factor: Load factor adalah rasio antara jumlah data yang disimpan dalam tabel hash dengan kapasitas total tabel. Load factor digunakan untuk mengukur seberapa penuh tabel hash, dan dapat mempengaruhi kinerja operasi pencarian dan penyimpanan.
8. Rehashing: Rehashing adalah proses memperbesar ukuran tabel hash dan memindahkan data yang ada ke tabel yang lebih besar. Ini dilakukan ketika load factor mencapai ambang batas tertentu untuk menjaga efisiensi operasi tabel hash.
9. Kunci (key): Kunci adalah nilai yang digunakan untuk mengidentifikasi dan mengakses data dalam tabel hash. Kunci biasanya digunakan sebagai input untuk fungsi hash.

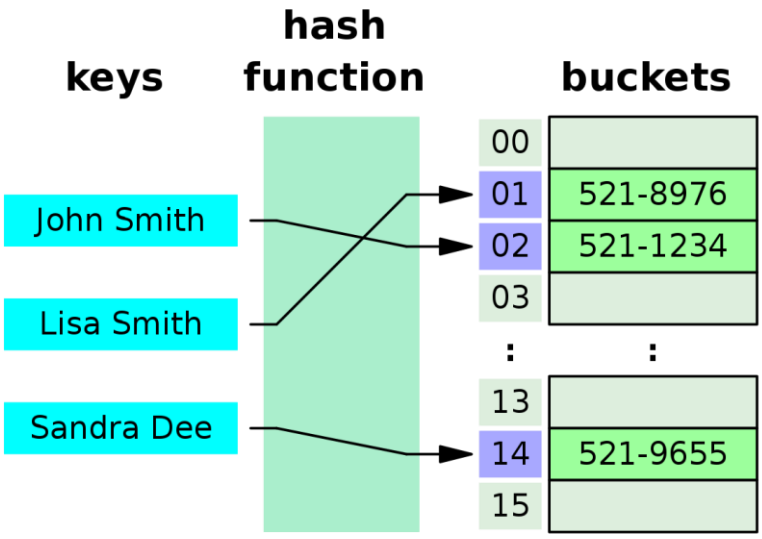
10. Nilai (value): Nilai adalah data yang disimpan dalam tabel hash dan terkait dengan kunci tertentu. Nilai dapat diakses dengan menggunakan kunci yang sesuai.

III. Struktur

Struktur tabel hash adalah struktur data yang digunakan untuk menyimpan dan mengakses data dengan efisien. Tabel hash terdiri dari array atau daftar elemen yang disebut “slot” atau “bucket”. Setiap slot memiliki indeks numerik yang unik yang disebut “hash code”.

Proses penyimpanan dan pencarian data dalam tabel hash melibatkan dua langkah utama. Pertama, data diubah menjadi hash code menggunakan fungsi hash. Fungsi hash ini mengonversi data menjadi indeks numerik yang digunakan untuk menentukan lokasi penyimpanan dalam tabel hash. Kedua, data disimpan dalam slot yang sesuai dengan hash code yang dihasilkan.

Keuntungan utama dari struktur tabel hash adalah kemampuannya untuk mencari dan mengakses data dengan cepat. Dalam kasus yang ideal, pencarian dan penyimpanan data dalam tabel hash memiliki kompleksitas waktu konstan O(1), yang berarti waktu yang dibutuhkan tidak tergantung pada jumlah data yang ada dalam tabel.



I. Karakteristik

1. Efisiensi pencarian: Tabel hash dirancang untuk memberikan pencarian data yang efisien. Dalam kasus yang ideal, pencarian dalam tabel hash memiliki kompleksitas waktu konstan O(1), yang berarti waktu yang dibutuhkan tidak tergantung pada jumlah data yang ada dalam tabel. Ini membuat tabel hash sangat efisien dalam mencari data.
2. Fungsi hash: Tabel hash menggunakan fungsi hash untuk mengonversi data menjadi hash code. Fungsi hash harus menghasilkan hash code yang unik untuk setiap data yang berbeda. Fungsi hash yang baik akan meminimalkan kemungkinan terjadinya tabrakan hash, yaitu dua atau lebih data memiliki hash code yang sama.
3. Penanganan tabrakan hash: Tabrakan hash terjadi ketika dua atau lebih data memiliki hash code yang sama. Tabel hash menggunakan teknik seperti chaining atau open addressing untuk menangani tabrakan hash. Dalam chaining, setiap slot tabel hash berfungsi sebagai daftar terkait yang menyimpan semua data dengan hash code yang sama. Dalam open addressing, jika terjadi tabrakan hash, data disimpan dalam slot lain yang tersedia dalam tabel.
4. Ukuran tabel: Ukuran tabel hash harus dipilih dengan hati-hati. Jika ukuran tabel terlalu kecil, maka akan ada lebih banyak tabrakan hash. Namun, jika ukuran tabel terlalu besar, maka akan membuang-buang ruang. Pemilihan ukuran tabel yang tepat sangat penting untuk menjaga kinerja tabel hash.
5. Rehashing: Rehashing adalah proses memperbesar ukuran tabel hash dan memindahkan data yang ada ke tabel yang lebih besar. Ini dilakukan ketika load factor (rasio antara jumlah data dan kapasitas tabel) mencapai ambang batas tertentu. Rehashing penting untuk menjaga efisiensi operasi tabel hash saat jumlah data meningkat.
6. Kunci dan nilai: Tabel hash menyimpan data dalam bentuk pasangan kunci-nilai. Kunci digunakan untuk mengidentifikasi dan mengakses data dalam tabel, sedangkan nilai adalah data yang sebenarnya yang

disimpan. Kunci harus unik dalam tabel hash untuk memastikan pengaksesan yang benar.