LOFIAN: Jurna**l** Tekn**o**logi In**f**orma**s**i d**a**n Komu**n**ikasi
https://ejournal.umbp.ac.id/index.php/lofian/
Vol. 2, No. 1, September 2022, Halaman 16-23

e-ISSN: 2798-9593
p-ISSN: 2798-9836

Pemanfaatan Link List Untuk Mengatasi database Tidak Normal

Jaidup Banjarnahor

Universitas Mandiri Bina Prestasi
Jl. Letjend. Djamin Ginting No. 285-287, Padang Bulan, Medan Baru, Medan, Sumatera Utara, Indonesia - 20155

marbun2005@gmail.com

DOI: xx.xxxx/j.ccs.xxxx.xx.xxx

Abstrak

Database Relasional yang merupakan database modern yang dapat meminimalkan terdapatnya tabel tabel yang tidak normal serta memudahkan pemisahan objek objek, namun demikian penentuan relasi dan penggunaan query akan semakin kompleks untuk mendapatkan informasi sesuai dengan kebutuhan. Namun untuk kebutuhan aplikasi yang membutuhkan sumberdaya memori yang tetap bertambah, penggunaan pointer dapat menjadi alternatif, dimana penggunaan memori dapat di pakai dan dilepas pada saat aplikasi sedang berjalan.

Kata Kunci: *Pointer, Link List, Database Relasional*.

1. Pendahuluan

1.1. Latar Belakang

Data merupakan bagian penting dari suatu bisnis atau proyek apa pun dalam berbagai bentuk, representasi data abtrak menjadi data digital menjadi sangat penting dalam mendukung kinerja sistem yang berdampak pada kecepatan dalam pengambilan informasi, kemudahan dalam pengelolaan, keceapatan dalam pencarian, efisiensi dalam penyimpanan dan juga ketersediaan informasi.

Database non relasional dan database relasional merupakan cara yang tepatuntuk merepresantasikan data dengan kekurangan dan kelebihan masing masing. Dalam database non relasional merupakan database yang mudah di implementasikan dimana setiap objek dapat direprentasikan kedalam array 2 dimensi yaitu baris dan kolom, hal yang sama juga dilakukan dengan pada database relasional semua objek direprentasikan kedalam array 2 dimensi dengan meminimalkan adannya kolom yang tidak kosong. Untuk meminimalkan kolom kolom yang kosong, maka dibutuhkan relasi dari 1 tabel dengan tabel yang lain sehingga akan menuliskan nilai kolom yang sama pada tabel yang berbeda.

Penggunaan pointer yang merupakan bagian link list dapat digunakan sebagai alternatif penggunaan array 2 dimensi untuk meminimalkan kolom kolom yang kosong dalam database relasional. Maka dalam penelitian ini akan menerapkan struktur data pointer untuk meminimalkan kolom kolom yang kosong dalam database erelasional.

1.2. Rumusan Masalah

Adapun rumusan masalah dalam penelitian ini adalah bagaimana menggunakan struktur data pointer untuk meminimalkan kolom kolom yang kosong dalam rancangan database relasional.

1.3. Batasan Penelitian

Dalam penelitian ini membahas tentang model penggunaan struktur data pointer dan gambaran logika penyimpanan dan metode pengaksesan data.

1.4. Target Luaran

Luaran dari penelitian ini adalah penggunaan pointer untuk meminimalkan kolom yang kosong yang dipublikasikan di jurnal.

2. Tinjauan Pustaka

2.1. Konsep Database

Pada awal tahun 1960, Charles Bachman di perusahaan General Electric mendesain generasi pertama DBMS yang disebut Penyimpanan Data Terintegrasi (Integrated Data Store). Dasar untuk model data jaringan terbentuk lalu distandarisasi oleh Conference on Data System Languages (CODASYL). Kemudian, Bachman menerima CM Turing Award

e-ISSN: 2798-9593
p-ISSN: 2798-9836

(penghargaan semacam nobel pada ilmu komputer) tahun 1973.

Di akhir 1960-an, Sistem Informasi Manajemen dikembangkan oleh Perusahaan Penerbangan Amerika yang bekerjasa sama dengan IBM. Dalam sitem informasi yang dikembangkan memberi kemampuan untuk beberapa user dapat menggunakan atau pung mengakses data yang sama dalam jaringan komputer, dan pada tahun 1970, model data Relasional diusulkan oleh Edgar Codd yang merupakan hasil penelitian di laboratorium San Jose.

Database management System yang merupakan kumpulan data, dimana data data yang ada pada kumpulan data itu saling tehubung. Kumpulan data disebut dengan basisdata yang mana basis data ini berisi fakta, nilai ataupun informasi yang tersipan dalam komputer secara sistematik.

Dengan adanya hubungan antar data, maka proses pencarian informasi akan lebih mudah dan lebih cepat, meminimalkan penyimpanan data yang berulang yang berdampak pada efisiensi ruang peyimpanan, meningkatkatnya keakuratan dan ketersediaan data serta akan lebih mudah untuk menjaga keamanan data juga kemampuan untuk menyediakan layanan secara bersama.

2.2. Database Non-Relasional

Database non-relasional adalah database yang tidak menggunakan skema tabular baris dan kolom yang ditemukan di sebagian besar sistem database tradisional. Penggunaan database non relasional banyak digunakan untuk penyimpanan data seperti grafik, penyimpanan data dari hasil proses dimana data data yang disimpan biasanya adalah berskala besar. Kelebihan database non relasional ini terletak pada penekanan kerumitan pada metode pembacaan data dengan menggunakan query yang kompleks serta kecepatan dalam membaca data karena membaca data yang dibutuhkan.

1. Penyimpanan data dokumen
   Penyimpanan data dokumen mengelola sekumpulan bidang string bernama dan nilai data objek dalam entitas yang disebut sebagai dokumen. Penyimpanan data ini biasanya menyimpan data dalam bentuk dokumen JSON. Setiap nilai bidang bisa menjadi item skalar, seperti angka, atau elemen majemuk, seperti daftar atau koleksi induk-turunan. Data dalam bentuk dokumen dapat disimbolkan dengan berbagai seperti penggunaan XML, YAML, JSON, BSON atau tersimpan sebagai teks. Untuk data seperti ini biasana akan berisi semua entitas dari data.

Key	Document
1001	{ "CustomerID": 99, "OrderItems": [{ "ProductID": 2010, "Quantity": 2, "Cost": 520 }, { "ProductID": 4365, "Quantity": 1, "Cost": 18 }], "OrderDate": "04/01/2017" }
1002	{ "CustomerID": 220, "OrderItems": [{ "ProductID": 1285, "Quantity": 1, "Cost": 120 }], "OrderDate": "05/08/2017" }

Gambar 1. Peyimpanan data dokumen

Aplikasi dapat membaca data dengan menggunakan kunci data. Kunci ini adalah pengenal yang spesifik yang digunakan untuk membedakan antar data.

2. Penyimpanan data kolom
   Penyimpanan data yang terdiri dari kolom ataupun data yang dikelompokkan dalam bentuk kolom, yang kemudian diatur menjadi data yang terdiri dari baris dan kolom, pengelompokan data ini sperti terlihat menjadai database relasional jika ditinjau dari bentuk konseptualnya. Penggunaan model data ini adalah merupakan pendekatan tabel yang tidak normal atau yang dikenal dengan denormalsasi terhadap data data yang jarang.
   Dalam penyimpanan data kolom adalah menyimpan data dalam bentuk tabulasi yaitu dengan baris dan kolom, dimana kolom terdiri dari beberapa kelompok yang disebut menjadi kelompok kolom. Untuk setiap kelompok kolom merupakan satu set kolom yang terkait secara logis yang diambil diambil ataupun dimanipulasi menjadi satu unit. Sementara untuk data data yang lain dapat dimanipulasi secara terpisah.
   Gambar 2 merupakan gamabaran untuk dua kelompok kolom yitu kolom untuk identitas dan kolom untuk info kontak. Untuk satu entitas terdapat satu data yang merupakan kunci baris, dimana disetiap bari terdapata data yang sama untuk setiap kelompok kolom.

CustomerID	Column Family: Identity
001	**First name:** Mu Bae **Last name:** Min
002	**First name:** Francisco **Last name:** Vila Nova **Suffix:** Jr.
003	**First name:** Lena **Last name:** Adamcyz **Title:** Dr.

CustomerID	Column Family: Contact Info
001	**Phone number:** 555-0100 **Email:** someone@example.com
002	**Email:** vilanova@contoso.com
003	**Phone number:** 555-0120

Gambar 2. Penyimpanan Dokumen dengan skema

e-ISSN: 2798-9593
p-ISSN: 2798-9836

Tidak seperti penyimpanan kunci/nilai atau database dokumen, sebagian besar database kelompok kolom secara fisik menyimpan data dalam urutan kunci, bukan dengan menghitung hash. Kunci baris dianggap sebagai indeks utama dan memungkinkan akses berbasis kunci melalui kunci tertentu atau berbagai tombol. Beberapa penerapan memungkinkan Anda membuat indeks sekunder di atas kolom tertentu dalam kelompok kolom. Indeks sekunder memungkinkan Anda mengambil data berdasarkan nilai kolom, bukan kunci baris.

Pada disk, semua kolom dalam kelompok kolom disimpan bersama dalam file yang sama, dengan jumlah baris tertentu di setiap file. Dengan himpunan data yang besar, pendekatan ini menciptakan keuntungan performa dengan mengurangi jumlah data yang perlu dibaca dari disk saat hanya beberapa kolom yang dikueri bersama pada satu waktu.

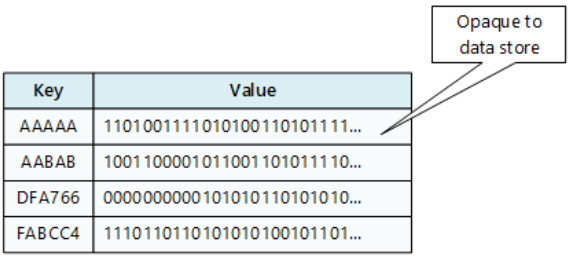
Operasi baca dan tulis untuk baris biasanya atomik dalam satu kelompok kolom, meskipun beberapa penerapan memberikan atomitas di seluruh baris, mencakup beberapa kelompok kolom.

3. Penyimpanan data kunci/nilai

Penyimpanan kunci/nilai pada dasarnya adalah tabel hash besar. Anda mengaitkan setiap nilai data dengan kunci unik, dan penyimpanan kunci/nilai menggunakan kunci ini untuk menyimpan data dengan menggunakan algoritme hash yang sesuai. Algoritme hash dipilih untuk menyediakan distribusi kunci hash yang merata di seluruh penyimpanan data.

Sebagian besar penyimpanan kunci/nilai hanya mendukung kueri, sisip, dan hapus operasi yang sederhana. Untuk memodifikasi nilai (baik sebagian atau seluruhnya), aplikasi harus menimpa data yang ada untuk seluruh nilai. Dalam kebanyakan penerapan, membaca atau menulis nilai tunggal adalah operasi atomik. Jika nilainya besar, penulisan dapat memakan waktu.

Aplikasi dapat menyimpan data arbitrer sebagai sekumpulan nilai, meskipun beberapa penyimpanan kunci/nilai memberlakukan batasan pada ukuran nilai maksimum. Nilai yang disimpan tidak terlalu terlihat oleh perangkat lunak sistem penyimpanan. Setiap informasi skema harus disediakan dan ditafsirkan oleh aplikasi. Pada dasarnya, nilai adalah blob dan penyimpanan kunci/nilai hanya mengambil atau menyimpan nilai berdasarkan kunci.



Gambar 3. Peyimpanan dokumen Kunci

Penyimpanan kunci/nilai sangat dioptimalkan untuk aplikasi yang melakukan pencarian sederhana menggunakan nilai kunci, atau dengan berbagai kunci, tetapi kurang cocok untuk sistem yang perlu mengkueri data di berbagai tabel kunci/nilai, seperti menggabungkan data di beberapa tabel.

Penyimpanan kunci/nilai juga tidak dioptimalkan untuk skenario saat proses kueri atau pemfilteran berdasarkan nilai non-kunci penting, dan tidak melakukan pencarian hanya berdasarkan kunci. Misalnya, dengan database relasional, Anda dapat menemukan rekaman dengan menggunakan klausa WHERE untuk memfilter kolom non-kunci, tetapi penyimpanan kunci/nilai biasanya tidak memiliki jenis kemampuan pencarian untuk nilai, atau jika memiliki kemampuan untuk melakukannya, diperlukan pemindaian lambat dari semua nilai.

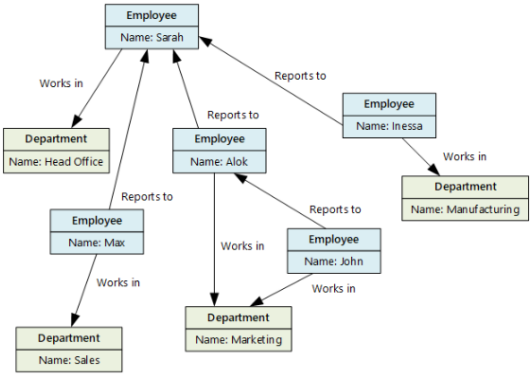
Satu penyimpanan kunci/nilai dapat sangat terukur, karena penyimpanan data dapat dengan mudah mendistribusikan data di beberapa node pada mesin yang berbeda.

4. Penyimpanan data grafik

Penyimpanan data grafik mengelola dua jenis informasi, node, dan tepi. Node mewakili entitas, dan tepi menentukan hubungan antara entitas ini. Baik node dan tepi dapat memiliki properti yang memberikan informasi tentang node atau tepi, serupa dengan kolom dalam tabel. Tepi juga dapat memiliki arah yang menunjukkan sifat hubungan.

Tujuan dari penyimpanan data grafik adalah untuk memungkinkan aplikasi mengkueri yang melintasi jaringan node dan tepi secara efisien, dan untuk menganalisis hubungan antara entitas. Diagram berikut menunjukkan data personalia organisasi yang terstruktur sebagai grafik. Entitas adalah karyawan dan departemen, dan tepi akan menunjukkan hubungan pelaporan dan departemen tempat karyawan bekerja. Dalam grafik ini, arah panah pada tepi menunjukkan arah hubungan.

e-ISSN: 2798-9593
p-ISSN: 2798-9836



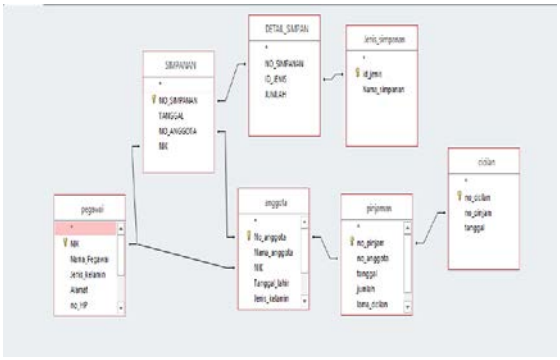
Gambar 4. Peyimpanan dengan grafik

Struktur ini membuatnya mudah untuk melakukan kueri seperti "Temukan semua karyawan yang melapor secara langsung atau tidak langsung kepada Sarah" atau "Siapa yang bekerja di departemen yang sama dengan John?" Untuk grafik besar dengan banyak entitas dan hubungan, Anda dapat melakukan analisis yang kompleks dengan cepat. Banyak database grafik menyediakan bahasa kueri yang dapat Anda gunakan untuk melintasi jaringan hubungan secara efisien.

2.3. Database Relasional

Database relasional merupakan jenis Database Management System (DBMS) yang terbaru, yang memberikan gambaran atau bagam skema yang menjelaskan tentang hubungan antar tabel bisa dilakuan di dalam sebuah database. Model database ini digagas oleh seorang pakar database bernama EF codd.

Jenis database relasional ini merupakan jenis database yang paling sederhana disamping jenis database pendahulunya yaitu database Hierarki (Hierarchical Database Model), dan database Jaringan (Network Database Model). Jenis database relasional menggunakan strukrur database 2D (dimensi). Perlu diketahui bahwa kedua model pendahulu relasional database yaitu database hirarki dan database jaringan untuk saat ini sudah tidak banyak digunakan, hal ini karena adanya berbagai kelemahan dan fungsionalitas yang ada dari kedua jenis database tersebut yang sudah memenuhi spesifikasj atau kebutuhan aplikasi modern saat ini, yang menuntut sistem database yang lebih kompleks dan terstruktur untuk memenuhi berbagai kebutuhan komputasi skala besar saat ini, baik dalam skala personal maupun Enterprise.



Gambar 5. Database relasional

2.4. Struktur Data

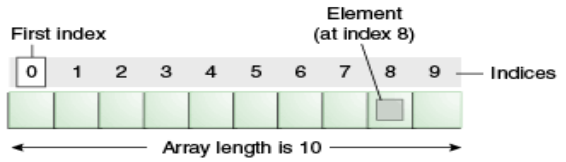
Dalam perkembangan teknologi informasi saat ini, data menjadi kompnen yang sangat berharga tidak dapat ditinggalkan dalam pemakaian komputer, daimana data dapat diperoleh dari berbagai sumber, misalkan hasil pengukuran laboratorium, hasil survei, hasil angket dan lain sebagainya.Dalam istilah ilmu komputer, struktur data adalah cara penyimpanan, pengorganisasian , dan pengaturan data di dalam media penyimpanan komputer sehingga data tersebut dapat digunakan secara efisien.Dalam teknik pemrograman, struktur data berarti tata letak data yang berisi kolom-kolom data,baik itu kolom yang tampak oleh pengguna (user) ataupun kolom yang hanya digunakan untuk keperluan pemrograman yang tidak tampak oleh pengguna. Setiap baris dari kumpulan kolom-kolom tersebut dinamakan catatan (record). Lebar kolom untuk data dapat berubah dan bervariasi. Ada kolom yang lebarnya berubah secara dinamis sesuai masukan dari pengguna dan juga ada kolom yang lebarnya tetap. Dengan sifatnya ini, sebuah struktur data dapat diterapkan untuk pengolahan database, misalnya untuk keperluan data keuangan, atau untuk pengolah kata (word processor) yang kolomnya berubah secara dinamis. Contoh struktur data dapat dilihat pada file-file spreadsheet, database, pengolahan kata, gambar yang dikompres, dan pemampatan file (kompres) dengan teknik tertentu yang memanfaatkan struktur data.

2.5. Struktur data Array

Array merupakan struktur data terstruktur yang banyak digunakan dalam pemrogaman, dimana struktur data ini memiliki kekuatan pada metode pengaksesan dimana untuk membaca data tidak lah harus berurutan dari data awal ke data yang terakhir namun dapat dilakukan secara random. Struktur data array sangat banyak digunakan dalam aplikasi seperti array satu dimensi yang menggambarkan 1 baris dengan n Kolom atau pun sebaliknya n baris 1 kolom

e-ISSN: 2798-9593
p-ISSN: 2798-9836

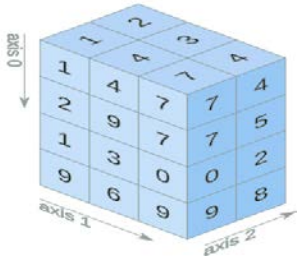
yang disimulasikan dalam gambar 6a. array dua dimensi yang menggambarkan antara baris dan kolom yang artinya data disusun dalam bentuk n baris dan m kolom, dan data seperti ini juga dinamakan sebagai array dua dimensi seperti gambar 6b. dalam ilmu-ilmu sains, penggunaan array banak digunakan seperti arra untuk 3 dimensi yang disimulasikan dengan sumbu x,y,z.



Gambar 6a. data 1 dimensi

Tabel 1. Data dua dimensi

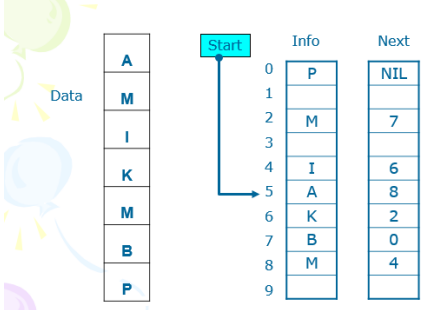
No	Nama	Tahun Ijazah			TAHUN JAFUNG		
		S1	S2	S3	AA	LEKTOR	LK
1	Dosen 1	thn1	thn2	--	thn3	thn4	--
2	Dosen 2	thn1	thn2	--	--	--	--
3	Dosen 3	thn1	--	--	--	--	--
4	Dosen 4	thn1	thn2	thn3	thn4	thn5	thn
	Dst						



Gambar 6c. Data 3 dimensi

2.6. Struktur data Pointer

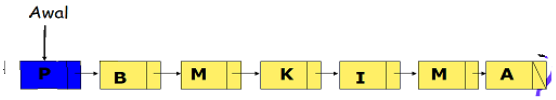
Pemakaian larik tidak selalu tepat untuk program-program terapan yang datanya selalu bertambah selama eksekusi program tersebut. Untuk mengatasi hal tersebut dibutuhkan sebuah tipe data yang sifatnya adalah dinamis yaitu penggunaan memori dilakukan sesuai dengan kebutuhan dan akan dibebaskan jika sudah selesai walapun pada saat itu program masih sedang berjalan. Gambaran logika dari peyimpanan data dengan array dan pointer seperti pada gambar 7.



Gambar 7. Peyimpanan data dengan menggunakan array dan pointer

2.7. Link List

Salah satu bentuk struktur data yang berisi kumpulan data yang tersusun secara sekuensial, saling bersambungan, dinamis adalah senarai berkait (linked list). Suatu senarai berkait (linked list) adalah suatu simpul (node) yang dikaitkan dengan simpul yang lain dalam suatu urutan tertentu. Suatu simpul dapat berbentuk suatu struktur atau class. Simpul harus mempunyai satu atau lebih elemen struktur atau class yang berisi data.



Gambar 8. Data yang tersusun dengan Link List

Bagian dari list yang dikenal dengan istilah simpul terdiri dari 2 bagian yaitu bagian yang menyimpan informasi dan penunjuk ke simpul yang lainnya. Gambar 8 terdiri dari 7 simpul yaitu simpul yang berisi informasi P, Simul yang berisi Informasi B dan seterusnya.

Pada gambar di atas, pointer awal menunjuk ke simpul pertama dari senerai tersebut. Medan penyambung (pointer) dari suatu simpul yang tidak menunjuk simpul lain disebut pointer kosong, yang nilainya dinyatakan sebagai null (null adalah kata baku yang berarti bahwa pointer 0 atau bilangan negatif). Jadi kita bisa melihat bahwa dengan hanya sebuah pointer Awal saja maka kita bisa membaca semua informasi yang tersimpan dalam senerai. Secara teori, linked list adalah sejumlah node yang dihubungkan secara linier dengan bantuan pointer. Dikatakan single linked apabila hanya ada satu pointer yangmenghubungkan setiap node. single artinya field pointernya hanya satu buah saja dan satu arah.

Senarai berkait adalah struktur data yang paling dasar. Senarai berkait terdiri atas sejumlah unsur-unsur dikelompokkan, atau terhubung, bersama-sama di suatu deret yang spesifik. Senarai berkait bermanfaat di dalam memelihara koleksi-koleksi data, yang serupa dengan array/larik yang sering digunakan. Bagaimanapun juga, senarai berkait memberikan keuntungan-keuntungan penting yang melebihi array/larik dalam banyak hal. Secara rinci, senarai berkait lebih efisien di dalam melaksanakan penyisipan-penyisipan dan penghapusan-penghapusan.

Senarai berkait juga menggunakan alokasi penyimpanan secara dinamis, yang merupakan penyimpanan yang dialokasikan pada runtime. Karena di dalam banyak aplikasi, ukuran dari data itu tidak diketahui pada saat kompile, hal ini bisa merupakan suatu atribut yang baik juga. Setiap node akan

e-ISSN: 2798-9593
p-ISSN: 2798-9836

berbentuk struct dan memiliki satu buah field bertipe struct yang sama, yang berfungsi sebagai pointer. Dalam menghubungkan setiap node, kita dapat menggunakan cara first-createfirst-access ataupun first-create-last-access. Yang berbeda dengan deklarasi struct sebelumnya adalah satu field bernama next, yang bertipe struct node. Hal ini sekilas dapat membingungkan. Namun, satu hal yang jelas, variabel next ini akan menghubungkan kita dengan node di sebelah kita, yang juga bertipe struct node. Hal inilah yang menyebabkan next harus bertipe struct node.

3. Metode Penelitian

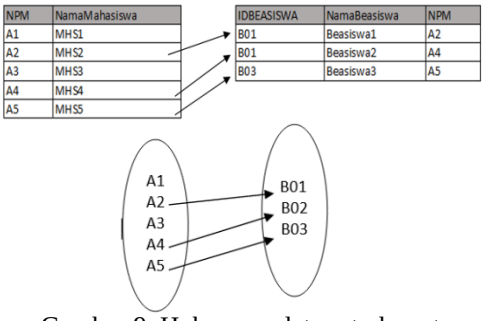
Metode penelitian yang digunakan adalah metode penelitian kualitatif, yaitu merupakan kajian terhadap penggunaan basis data tradisional, database modern, database non relasional maupun database relasional serta penggunaan struktur data array 2 dimensi dan struktur data pointer.

4. Pembahasan

4.1. Database Relasional

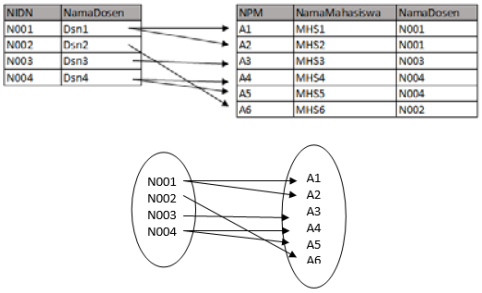
Database relasional adalah bentuk database modern dimana data yang kompleks dapat dipecah menjadi beberapa bagian. Setiap bagian akan disimpan menjadi satu tabel dan antar tabel akan terhubung dengan relasi yang menggunakan primary key dan foreign key. Dalam database relasioan terdapat tiga bentuk relasi yaitu satu ke satu, Satu ke banyak dan banyak ke banyak.

1. Satu ke Satu
   Hubugan satu ke satu adalah yang menggambarkan hubungan antara objek 1 dengan objek 2 terhubung secara satu ke satu, misalkan terdapat objek yang terdiri dari beberapa mahasiswa dan juga objek beasiswa yang terdiri dari beberapa jenis beasiswa, dimana aturannya adalah setiap 1 beasiswa hanya untuk 1 mahasiswa dan juga 1 mahasiswa hanya boleh mendapatkan 1 beasiswa, Aturan ini seperti pada gambar 9.



Gambar 9. Hubungan data satu ke satu

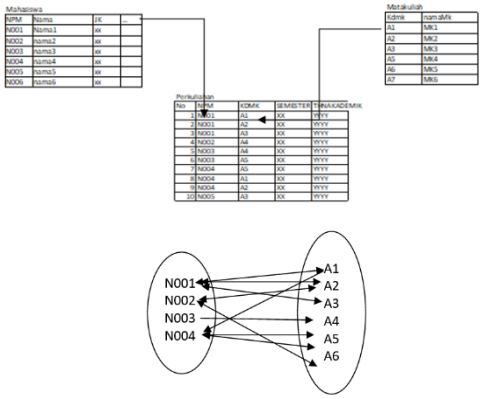
2. Satu ke Banyak
   Hubungan satu kebanyak menggambarkan hubungan antara objek1 dengan objek2 berhubungan dengan satu kebanyak, sebagai contoh jika dalam pendidikan tinggi yang menerapkan perwalian, maka setiap satu orang dosen boleh memiliki perwalian lebih dari satu mahasiswa dan sebaliknya setiap mahasasiswa hanya memiliki satu dosen wali. Gambaran seperti ini dapat digambarkan pada gambar 10.



Gambar 10. Hubungan data satu ke Banyak

3. Banyak ke Banyak
   Hubugan banyak kebanyak juga merupakan cara meminimalkan tabel yang tidak normal, dimana jika ada dua objek yaitu objek1 dan objek2 dan satu elemen dari objek1 terhubung dengan banyak elemen di objek2 dan begitu juga sebaliknya, satu elemen dari objek2 terhubung dengan banyak elemen di objek1.
   Jika dalam institusi pendidikian menerapkan pembelajaran dengan sistem semester, maka setiap satu mahasiswa dapat membawa lebih dari satu matakuliah dalam satu semester atau juga setiap satu matakuliah dapat diambil oleh banyak mahasiswa disemester yang sama. Hubungan seperti ini seperti pada gambar 11

e-ISSN: 2798-9593
p-ISSN: 2798-9836



Gambar 11. Gambaran logika hubungan banyak ke banyak

4.2. Penggunaan Pointer

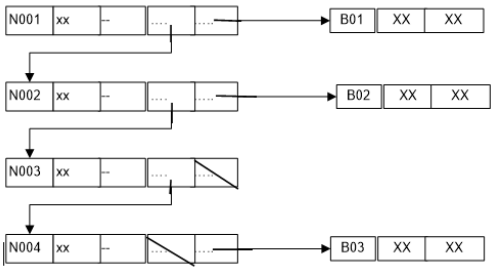
Struktur data pointer yang merupakan struktur data dinamis dimana struktur data ini tepat digunakan untuk jumlah data yang berubah dan juga untuk keterbatasan sumber daya, khusunya main memori. Kondisi ini dapat di implementasikan dengan mengosongkan memori pada saat aplikasi dalam keadaan running serta dapat meminta layanan memori ketika ada kebutuhan untuk penyimpanan data.

1. List satu arah
   Defenisi Sturuktur data

```
Beasiswa = ^databeasiswa
Databeasiswa = Record
   IdBeasiswa        : Tipedata
   NamaBeasiswa      : Tipedata
   ……..
   ……...
End

Mahasiswa = ^datamahasiswa
Datamahasiswa = Record
   NPM               : tipeadata
   Namamahasiswa     : tipedata
   ……..
   ……..
   Beasiswa          :Beasiswa
   Nextmhs           :mahasiswa
  End
```

Gambar 12 mengambarkan secara logika untuk list satu arah, dimana pembacaan dapat dilakukan dari mahasiswa.

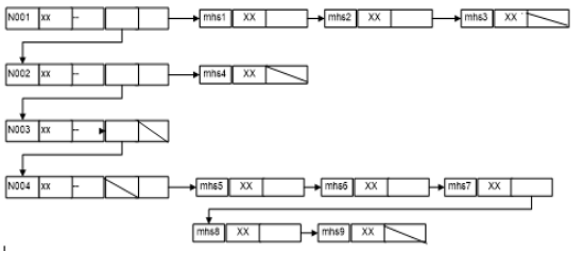


Gambar 12. Gambaran Logika dari list satu arah

2. Multi List
   Multilist dapat digunakan untuk objek yang berhubungan dengan satu ke banyak, misalkan dosen pembimbing akademik akan membimbing beberapa mahasiswa, sementara satu mahasiswa hanya memiliki 1 dosen pembibing akademiknya, Contoh ini jika kita defenisikan dengan menggunakan pointer sebagai berikut

```
Mhspa = ^mahasiswa
Mahasiswa = record
   Npm              : string
   Namamhs          : string
   ……
   ……
   Mhsnext          : mhspa
End

Dosenpa = ^dosen
Dosen = record
   Nidn     : string
   Nama     : string
   ……..
   ……..
   Dsn      : dosenpa
   Mhs      :mhspa
End
```



Gambar 13 Hubungan satu kebanyak dengan menggunkan pointer

e-ISSN: 2798-9593
p-ISSN: 2798-9836

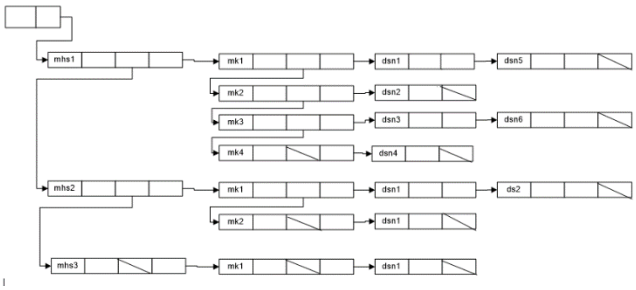
3. Multilist untuk Banyak ke Banyak

Defenisi struktur data

```
Dosen = ^dsn
Dsn = record
   Nidn          : string
   ……..
   ……..
   Dsnnext       : dosen
End

Matakuliah = ^mkul
Mkul =record
   Kdmk          : string
   ……..
   ……..
   Mkulnet       : matakuliah
   Dsnmkul       : dosen
End

Mahasiswa = ^mhs
Mhs = record
   Npm           : string
   ……..
   ……..
   Mhsnext       : mahasiswa
   Mtkul         : matakuliah
End
```



Gambar 14 Hubungan Multilist untuk Banyak ke Banyak

5. Kesimpulan

Pointer sebagai tipe data dinamis dapat digunakan untuk merepresentasikan hubungan antar objek, baik itu untuk yang berhubungan dengan satu ke satu, satu kebanyak dan banyak ke banyak. Model

ini dapat digunakan untuk data yang dinamis dimana penambahan data terjadi pada saat aplikasi dalam keadaan running dan dimana kebutuhan memori cukup besar.

Referensi

[1] Abdul Kadir, Teori dan Aplikasi Stuktur data Menggunakan C++, Penerbit Andi, Yogyakarta, 2013
[2] Adi Nugroho,Perancanngan dan Implementasi Basis Data, Penerbit Andi Yoyakarta, 2011
[3] Erwin Daniel Sitanggang et al 2019 J. Phys.: Conf. Ser. 1235 012061
[4] Irawan, B., Kurnia, R. A., Sitanggang, E. D., & Sembiring, M. (2021). The College Academic Service Decision Support System Uses Service Quality and Importance-Performance Analysis Methods. INFOKUM, 10(1), 74-85. Retrieved from http://infor.seaninstitute.org/index.php/infokum/article/view/219
[5] Muhammad Fikry, BASIS DATA, Unimal Press, 2019
[6] Raghuvanshi, Durgesh. (2018). Data Structure: Theoretical Approach. International Journal of Trend in Scientific Research and Development. Volume-3. 268-273. 10.31142/ijtsrd18977
[7] Sembiring, M., & Simbolon, F. H. (2021). Perancangan Perangkat Lunak Pembelajaran Algoritma Hamming Code dalam Mencari Bit Error pada Komunikasi Data. LOFIAN: Jurnal Teknologi Informasi Dan Komunikasi, 1(1), 24–28. Retrieved from https://ejournal.umbp.ac.id/index.php/lofian/article/view/161
[8] https://docs.microsoft.com/id-id/azure/ architecture/data-guide/big-data/non-relational-data