Dasar-Dasar
Sistem Basis Data
Teori dan Praktik

Basis data adalah kumpulan data yang disimpan dan diorganisasi secara sistematis sehingga mudah untuk dicari, diambil, dan digunakan. Basis data menyediakan mekanisme untuk menyimpan, mengelola, dan mengambil data dengan efisien dan mudah. Ada beberapa jenis basis data yang digunakan, termasuk database relasional (SQL) dan database NoSQL.

Database relasional (SQL) menggunakan tabel untuk menyimpan data, dengan setiap baris mewakili sebuah catatan dan setiap kolom mewakili sebuah atribut. Database relasional memiliki beberapa kelebihan seperti fleksibilitas dalam mengolah data dan integritas data yang tinggi.

Database NoSQL tidak mengikuti model tabel seperti database relasional, tetapi memiliki beberapa jenis model data unik seperti document-oriented, key-value, columnar, dan graph databases. NoSQL lebih cocok untuk aplikasi yang membutuhkan skalabilitas tinggi dan kecepatan akses data yang cepat.Manajemen basis data (DBMS) adalah software yang digunakan untuk mengelola dan mengontrol akses ke basis data. Ada beberapa jenis DBMS populer seperti MySQL, PostgreSQL, MongoDB, Cassandra, dan Microsoft SQL Server.

DEWA PUBLISHING
Anggota IKAPI
Ds. Kalianyar RT. 003/ RW. 002, Kec. Ngronggot, Kab. Nganjuk Jatim
www.dewapublishing.com
publishingdewa@gmail.com
dewapublishing
0877-7141-5004

ISBN 978-623-8491-69-8
9 786238 491698

Dasar-Dasar Sistem Basis Data Teori dan Praktik

Dr. Hendi Sama, S.Kom., M.M.e-Business
Tukino, S.Kom.,M. SI.

Dasar-Dasar
Sistem Basis Data
Teori dan Praktik

Dr. Hendi Sama, S.Kom., M.M.e-Business
Tukino, S.Kom.,M. SI.

Editor: Algifanri Maulana, M.MSI.

DEWA PUBLISHING

DASAR-DASAR SISTEM BASIS DATA:
TEORI DAN PRAKTIK

Sanksi Pelanggaran Pasal 113
Undang-Undang No. 28 Tahun 2014 Tentang Hak Cipta

1. Setiap Orang yang dengan tanpa hak melakukan pelanggaran hak ekonomi sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf i untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 1 (satu) tahun dan/atau pidana denda paling banyak Rp 100.000.000 (seratus juta rupiah).
2. Setiap Orang yang dengan tanpa hak dan/atau tanpa izin Pencipta atau pemegang Hak Cipta melakukan pelanggaran hak ekonomi Pencipta sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf c, huruf d, huruf f, dan/atau huruf h untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 3 (tiga) tahun dan/atau pidana denda paling banyak Rp 500.000.000,00 (lima ratus juta rupiah).
3. Setiap Orang yang dengan tanpa hak dan/atau tanpa izin Pencipta atau pemegang Hak Cipta melakukan pelanggaran hak ekonomi Pencipta sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf a, huruf b, huruf e, dan/atau huruf g untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 4 (empat) tahun dan/atau pidana denda paling banyak Rp 1.000.000.000,00 (satu miliar rupiah).
4. Setiap Orang yang memenuhi unsur sebagaimana dimaksud pada ayat (3) yang dilakukan dalam bentuk pembajakan, dipidana dengan pidana penjara paling lama 10 (sepuluh) tahun dan/atau pidana denda paling banyak Rp 4.000.000.000,00 (empat miliar rupiah).

DASAR-DASAR SISTEM BASIS DATA: TEORI DAN PRAKTIK

Dr. Hendi Sama, S.Kom., M.M.e-Business
Tukino, S.Kom.,M. SI.

DEWA PUBLISHING

2024

Dasar-Dasar Sistem Basis Data:
Teori dan Praktik

Dr. Hendi Sama, S.Kom., M.M.e-Business
Tukino, S.Kom.,M. SI.

Editor Naskah : Algifanri Maulana, M.MSI
Perancang Sampul : Tim Dewa Publishing
Penata Letak : Tim Dewa Publishing

Diterbitkan oleh:

DEWA PUBLISHING

Redaksi:

CV. Dewa Publishing
Desa Kalianyar RT 003/RW 002, Kec. Ngronggot
Kab. Nganjuk, Jawa Timur

Email : publishingdewa@gmail.com
Website : www.dewapublishing.com
Phone : 0877-7141-5004

Cetakan Pertama, Februari 2024
i-xi+221 hlm, 15.5 cm x 23 cm

ISBN 978-623-8491-69-8

KATA PENGANTAR

Puji serta syukur kami panjatkan Kehadirat Tuhan Yang Maha Esa yang telah memberikan karunia dan rahmat-Nya, sehingga kami mampu menyelesaikan buku yang membahas konsep dasar Sistem Basis Data yang dapat dipakai di lingkungan perguruan tinggi khususnya program studi sistem informasi.

Sistem Basis Data memainkan peran sentral dalam dunia teknologi informasi, menjadi fondasi bagi penyimpanan, pengelolaan, dan akses informasi yang efisien. Buku ini, "Dasar-Dasar Sistem Basis Data: Teori dan Praktik," merupakan panduan yang komprehensif bagi pembaca yang ingin memahami esensi sistem basis data, baik dari segi teori maupun implementasi praktisnya.

Dengan menggabungkan konsep dasar teori dan aplikasi praktis, buku ini dirancang untuk memberikan pemahaman mendalam tentang prinsip-prinsip desain basis data, manajemen transaksi, serta optimisasi kinerja query. Pembaca akan dibimbing melalui perjalanan yang membahas aspek-aspek kunci, termasuk model data relasional, normalisasi, dan strategi pengindeksan yang efektif.

Pentingnya keamanan data dan manajemen integritas juga menjadi fokus utama, mengingat peran kritis sistem basis data dalam menjaga ketersediaan, keutuhan, dan kerahasiaan informasi. Tak hanya itu, buku ini juga membahas tren terkini dalam dunia basis data, seperti sistem NoSQL dan teknologi basis data terdistribusi.

Penekanan pada pendekatan praktis disertai dengan contoh kasus nyata dan panduan implementasi, memungkinkan pembaca untuk mengaplikasikan pengetahuan yang diperoleh dalam proyek-proyek dunia nyata. Buku ini diharapkan tidak hanya menjadi referensi berharga bagi mahasiswa dan profesional IT, tetapi juga menjadi panduan yang mendalam untuk siapa saja yang ingin menjelajahi dunia kompleks dan dinamis sistem basis data.

Penulis berharap buku ini memberikan wawasan yang berharga dan memperkaya pemahaman Anda tentang sistem basis data, serta membantu Anda menghadapi tantangan dan kesempatan yang muncul di dunia teknologi informasi yang terus berkembang.

Batam, Januari, 2024

Dr. Hendi Sama, S.Kom., M.M.e-Business

DAFTAR ISI

BAB 1
PENDAHULUAN

1.1. Pengantar

Basis data adalah kumpulan data yang disimpan dan diorganisasi secara sistematis sehingga mudah untuk dicari, diambil, dan digunakan. Basis data menyediakan mekanisme untuk menyimpan, mengelola, dan mengambil data dengan efisien dan mudah. Ada beberapa jenis basis data yang digunakan, termasuk database relasional (SQL) dan database NoSQL.

Database relasional (SQL) menggunakan tabel untuk menyimpan data, dengan setiap baris mewakili sebuah catatan dan setiap kolom mewakili sebuah atribut. Database relasional memiliki beberapa kelebihan seperti fleksibilitas dalam mengolah data dan integritas data yang tinggi.

Database NoSQL tidak mengikuti model tabel seperti database relasional, tetapi memiliki beberapa jenis model data unik seperti document-oriented, key-value, columnar, dan graph databases. NoSQL lebih cocok untuk aplikasi yang membutuhkan skalabilitas tinggi dan kecepatan akses data yang cepat.

Manajemen basis data (DBMS) adalah software yang digunakan untuk mengelola dan mengontrol akses ke basis data. Ada beberapa jenis DBMS populer seperti MySQL, PostgreSQL, MongoDB, Cassandra, dan Microsoft SQL Server.

Dalam pengembangan aplikasi, basis data sering digunakan untuk menyimpan informasi seperti informasi pengguna, transaksi, dan data produk. Penggunaan basis data membuat aplikasi lebih robust dan mudah dikembangkan dan diperbaiki.

Berikut adalah gambaran umum tentang perkembangan basis data:

1. Basis Data Terpusat: Pada awalnya, basis data terpusat digunakan untuk menyimpan data organisasi dalam satu lokasi sentral. Model ini cukup sederhana dan mudah digunakan, namun rentan terhadap kegagalan sistem yang dapat menyebabkan hilangnya data yang disimpan.
2. Basis Data Terdistribusi: Model basis data terdistribusi memungkinkan data untuk disimpan di beberapa lokasi terpisah dan diakses oleh pengguna dari jarak jauh. Hal ini membawa keuntungan dalam hal penyebaran data dan meningkatkan ketersediaan, namun juga menimbulkan tantangan dalam mengelola keseluruhan jaringan dan memastikan konsistensi data.
3. Basis Data Berorientasi Objek: Model basis data ini memperkenalkan konsep pengelolaan data dengan cara yang berbeda, yaitu dengan memperlakukan data

sebagai objek. Pendekatan ini memungkinkan pengembangan aplikasi yang lebih fleksibel dan mudah diatur, namun mengharuskan pengguna untuk memahami konsep objek-objek tersebut.

4. Basis Data Terdistribusi Berorientasi Objek: Model ini menggabungkan pendekatan terdistribusi dan berorientasi objek untuk memungkinkan pengguna untuk mengakses dan memanipulasi data secara terdistribusi. Model ini memungkinkan pengguna untuk mengelola data yang sangat besar secara efisien, namun juga menimbulkan tantangan dalam hal pengaturan dan pemeliharaan sistem secara keseluruhan.
5. Basis Data Berbasis Cloud: Basis data berbasis cloud memungkinkan pengguna untuk menyimpan dan mengelola data dalam lingkungan cloud. Pendekatan ini memungkinkan pengguna untuk mengelola data dengan biaya yang lebih rendah dan meningkatkan ketersediaan data, namun juga menimbulkan tantangan dalam hal keamanan dan privasi data.
6. Basis Data Berbasis Graf: Basis data berbasis graf memanfaatkan konsep graf untuk mengelola dan menganalisis data. Pendekatan ini memungkinkan pengguna untuk mengelola hubungan antar data dengan lebih efektif, namun memerlukan pemahaman yang lebih dalam tentang struktur graf dan cara kerjanya.

7. Basis Data Berbasis Kecerdasan Buatan (AI): Basis data berbasis AI memanfaatkan kecerdasan buatan untuk menganalisis dan memproses data dengan cara yang lebih efektif dan efisien. Pendekatan ini memungkinkan pengguna untuk mengambil keputusan berdasarkan data secara lebih cepat dan akurat, namun memerlukan pemahaman yang lebih dalam tentang teknologi AI.

Misalnya diambil sebuah contoh sebuah perusahaan makanan ingin mengembangkan sistem manajemen inventaris mereka. Untuk mencapai tujuan ini, mereka memutuskan untuk menggunakan basis data. Beberapa proses dalam konsep basis data yang dapat digunakan dalam kasus ini adalah:

1. Pengumpulan data: Perusahaan makanan harus mengumpulkan data dari semua bahan mentah, produk jadi, dan bahan pengemas yang mereka miliki. Data ini harus mencakup detail seperti jumlah, harga, tanggal kadaluwarsa, dan lain sebagainya.
2. Desain database: Setelah data terkumpul, perusahaan makanan perlu merancang basis data untuk mengelola informasi tersebut. Mereka perlu memutuskan tabel mana yang perlu dibuat, hubungan antara tabel, dan tipe data yang diperlukan untuk masing-masing kolom.
3. Pengolahan data: Setelah basis data dibuat, perusahaan makanan harus memasukkan data ke dalam basis data mereka dan memastikan bahwa data tersebut akurat

dan terkini. Mereka juga perlu mengambil langkah-langkah untuk melindungi basis data mereka dari kerusakan atau kehilangan data.

4. Analisis data: Perusahaan makanan dapat menggunakan basis data mereka untuk menganalisis data dan membuat laporan. Mereka dapat melihat stok bahan mentah dan produk jadi, memeriksa pengeluaran dan pendapatan, dan membuat ramalan inventaris untuk jangka waktu tertentu.

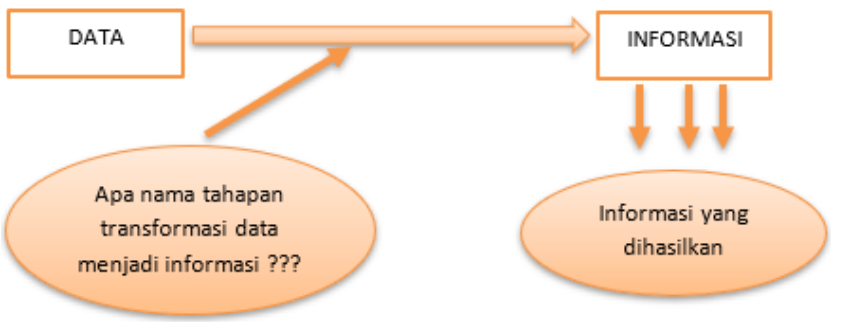
Dengan menggunakan basis data, perusahaan makanan dapat meningkatkan efisiensi dan efektivitas manajemen inventaris mereka. Mereka dapat memperoleh wawasan yang lebih dalam tentang inventaris mereka dan membuat keputusan yang lebih baik dan cepat tentang pengelolaan stok.

1.2. Perkembangan dan Konsep Basis Data

Basis data dapat berupa aplikasi yang bisa digunakan untuk mengelolaan organisasi baik itu organisasi yang bersifat informal maupun formal, namun dalam implementasinya pengelolaan basis data belum dimanfaatkan secera optimal, hal ini dikarenakan pandangan terhadap basis data sebagai dasar pengambilan keputusan belum banyak berperan padahal dalam dunia bisnis, data sangat penting sekali terutama dalam mendapatkan informasi yang bisa dipakai untuk mengambil keputusan, Disinilah diperlukan pemahaman dari siklus hidup informasi yang diawali dari data

yang menjadi fakta dan di olah menjadi informasi sehingga menjadi bahan pertimbangan dalam pengambilan keputusan.

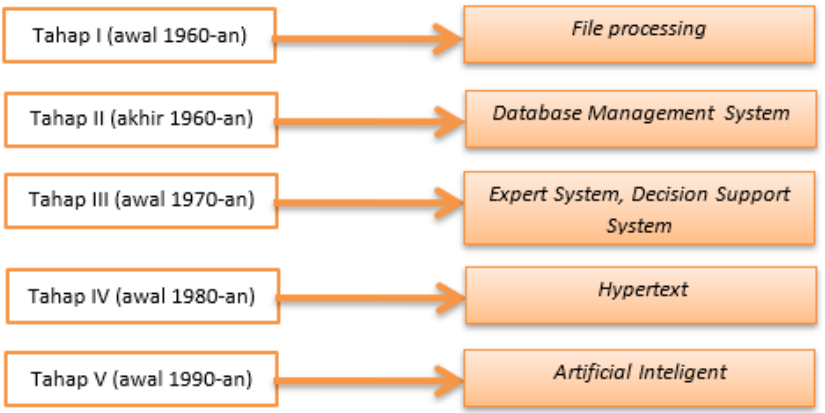
Berikut ini akan digambarkan perubahan data menjadi informasi dan bagaimana proses data traformasi menjadi sebuah tersebut yang didasari oleh proses input data, proses data, dan output data sehingga informasi dapat digunakan oleh pemakai sebagai bahan pertimangan keputusan untuk kegiatan kegiata yang ada didalam sebuah organisa atau perusahan seperti yang terdapat pada gambar dibawah :



Gambar 1.1 Konsep perubahan data menjadi informasi

Dari gambar di atas konsep data berubah menjadi informasi dalam paradigma input proses dan *output* sehingga data menjadi variabel input, tanda panah proses menjadi tahapan transformasi data menjadi informasi dan *output* informasi adalah hasil dari pengolahan data

Konsep tersebut diatas telah menjadi dasar bagian dalam mendukung perkembangan basis data yang dapat dilihat tahapan-tahapannya menjadi beberapa bagian di bawah ini Perkembangan konsep basis data dibedakan dalam 5 (lima) tahapan, yaitu :



Gambar 1.2 Perkembangan Basis Data

Dari gambar 1.2 perkembangan basis data dari tahap awal sampai akhir menunjukkan ini kejadian dengan penjelasan sebagai berikut :

1. *File processing* : Pengelolaan data untuk pemrosesan file yang hanya dapat di lakukan oleh satu program aplkasi untuk persoalan dan system tertentu saja sehingga kemungkinan terjadi kerangkapan data sangat tinggi
2. *Database Management System* : Pengelolaan datan untuk pemrosesan file yang dapat di lakukan dari berbagai aplikasi dengan manajemen file yang baik yang berelasi sehingga tidak memungkinkan terjadi kerangkapan data
3. Expert System, Decision Support System : Merupakan pengembangan dari DBMS yang menjadi paket perangkat lunak yang dapat melakukan pengolahan data menjadi informasi yang mampu memberikan interpretasi, prediksi, diagnosis, perenaan,

penyimpanan dan perbaikan kepada pengguna atau pihak yang berkepentingan.

4. Hypertext : Merupakan perkembangan basis data dengan data teks yang dapat berelasi / link dengan informasi lainya, sehingga struktur datanya lebih efisien.
5. Artificial Inteligent : Merupakan Kecerdasan buatan dari penggunaan basis data sebagai sumber data dan informasi secara otomatis dalam operasional system kecerdasan buatan.

Dalam implementasinya perkembangan basis data juga dapat dilihat dari permodelan basis data yang berkembang, dimulai dari zaman model hirarkis, zaman model jaringan, dan zaman model relasional. Sejalan dengan itu konsep basis data terus berkembang dan sampai saat ini basis data relasional masih menjadi standar industri. Namun, darib beberapa model arsitektur basis data baru, seperti NoSQL dan NewSQL, telah muncul untuk mengatasi keterbatasan basis data relasional.

Pengembangan basis data telah melihat banyak perubahan dan perkembangan sejak awal. Awalnya, sistem database hanya digunakan untuk menyimpan informasi statis, tetapi menjadi lebih dinamis dan fleksibel dari waktu ke waktu. Konsep basis data saat ini mencakup konsep normalisasi, integritas data, dan sistem manajemen basis data (DBMS) untuk mengelola dan mengontrol akses ke data. Dalam perkembangan terakhir, teknologi basis data NoSQL dan Big Data membantu memproses data dalam jumlah besar dan

kompleks, dan komputasi awan memungkinkan data disimpan dan diakses dari berbagai lokasi.

Dari gambaran diatas dapat dinyatakan konsep basis data pada dasarnya adalah menyimpan, mengelola, dan mengakses data dengan efisien. Hal ini dilakukan melalui pemodelan data yang sesuai dengan dunia nyata dan menggunakan sistem manajemen basis data (DBMS) untuk mengelolanya. Tujuan utama dari sebuah basis data adalah menjamin integritas, keandalan, dan aksesibilitas data yang dapat digunakan oleh aplikasi-aplikasi lain.

1.3. Aplikasi Sistem Basis Data

Aplikasi sistem basis data dapat mendukung proses pengelolaan data menjadi informasi yang dapat di gunakan untuk pengambilan keputusan dan membantu dalam kemajuan perusahaan sebagai pengguna. Sistem basis data membantu mengelola dan menggunakan data secara efektif, memastikan integritas dan keamanan data, dan memungkinkan analisis data untuk membuat keputusan yang lebih baik. Aplikasi basis data dari segi penggunaan basis data dalam kehidupan manusia, seperti :

1. Industri manufaktur
2. Manajemen rumah sakit
3. Manajemen perpustakaan
4. Perhotelan
5. Perbankan

6. Perguruan Tinggi
7. Penerbangan
8. Penjualan
9. Personalia

Dari semua bidang diatas dapat diimplementasikan dengan menggunakan beberapa aplikasi sistem basis data meliputi :

1. MySQL

Merupakan aplikasi basis data popular yang bersifat *open source* yang bisa digunakan dalam sistem operasi *Linux, Windows, Mac OS X* dan aplikasi berisifat *network* sehingga bisa bersifat *multiuser*

2. MariaDB

Merupakan aplikasi basis data yang dikembangkan Bersama aplikasi *MYSQL* yang bertujuan utnuk mempertahannan dan menjaga kompatibilitas tinggi terhadap *MYSQL*.

3. Microsoft SQL Server

Merupakan aplikasi basis data yang berasal dari *Microsoft* yang menggunakan bahasa *Transact-SQL* yang pada awalnya digunakan untuk *database* skala kecil, namun sekarang sudah banyak digunakan untuk skala besar dan diakses oleh banyak *platform.*

4. Oracle Database

Aplikasi ini merupakan aplikasi *database* yang paling baik untuk pengolahan data dalam skala besar, bahkan mampu mengolah data sampai ukuran tera*byte*.

5. PostgreSQL

Merupakan aplikasi basis data bersifat relasional sehingga dapat digunakan menyimpan dan

mengembalikan data dengan aman. Aplikasi ini lebih ringan untuk data skala besar untuk pengguna dengan sistem operasi *MacOS Server* yang terinstall bawaan dari sistem operasi tersebut.

6. SQLite

   Aplikasi ini sangat baik bagi data yang terstruktur sebagai *caching* dari data *cloud*.

7. DBeaver

   Merupakan aplikasi mode graphical atau GUI dengan fitur menarik dalam dukungan banyak *platform* serta kemampuan menulis berbagai file ekstensi / *plugins*

8. MongoDB

   Aplikasi basis datan ini mempunyai cara kerja open source serta cross platform yang di kategorikan dalam basis data *NoSQL* / Not Only *SQL* dengan prinsip javascrip JSON

9. Apache Cassandra

Aplikasi ini berisifat *cluster* memiliki bendmark yanbg lebih baik jika dibandingkan dengan *NoSQL* lainya. Aplikasi ini juga dilengkapi dengan apliksi keperluan *enterprise*

1.4. Data dan Informasi

Data merupakan bahan keterangan tentang kejadian-kejadian nyata atau fakta-fakta yang dirumuskan dalam sekelompok lambang tertentu yang tidak acak, yang menunjukkan jumlah, tindakan, atau hal. Contoh nya nama, umur, status perkawinan

Data dapat diklasifikasikan sebagai data terstruktur seperti angka, teks, dan tanggal, atau sebagai data tidak terstruktur seperti audio, video, atau gambar. Data dapat digambarkan sebagai deskriptif seperti rincian produk atau informasi kontak, atau analitik seperti hasil analisis data atau laporan keuangan.

Dari data dapat diolah menghasilkan sebuah informasi maka Informasi merupakan hasil pengolahan data sehingga menjadi bentuk yang penting bagi penerimanya dan mempunyai kegunaan sebagai dasar dalam pengambilan

keputusan. Contohnya laporan data karyawan, laporan laba/rugi

Data dan informasi sangat penting untuk banyak bisnis dan organisasi, dan sistem basis data memainkan peran penting dalam mengelola, menggunakan, dan memproses informasi tersebut untuk membuat keputusan yang lebih baik.

Informasi yang diperoleh dari pengolahan data dapat dinilai berdasarkan sifatnya. Sifat informasi yang menentukan nilai informasi adalah :

1. Kemudahan dalam perolehannya
2. Sifat luas dan kelengkapannya
3. Ketelitian (*accuracy*)
4. Kecocokan dengan pengguna (*relevancy)*
5. Ketepatan waktu
6. Kejelasan
7. Fleksibilitas/keluwesannya
8. Tidak adanya prasangka
9. Dapat diukur

Informasi diperlukan oleh para pemakai (manajemen) pada seluruh level manajemen. Informasi tersebut mempunyai manfaat, beberapa fungsi informasi adalah antara lain :

1. Menambah pengetahuan

2. Mengurangi ketidakpastian
3. Mengurangi resiko kegagalan
4. Mengurangi keanekaragaman/variasi yang tidak diperlukan
5. Memberi standar, aturan, ukuran dan keputusan yang menentukan pencapaian sasaran dan tujuan

Disamping itu aspek kebutuhan informasi tersebut sangat tergantung pada tiga faktor, yaitu :

1. Fungsi operasional
2. Kegiatan manajemen
3. Pembuat keputusan

Jadi, data adalah bahan baku untuk membuat informasi, dan informasi adalah hasil dari pengolahan data. Dalam sistem basis data, data disimpan dan dikelola sebagai basis data, dan informasi dapat diperoleh melalui proses pengambilan data dari basis data.

1.5. Sistem Informasi dan Basis Data

Konsep sebuah sistem informasi dapat dinyatakan sebagai sekumpulan subsistem yang saling berhubungan, berkumpul bersama-sama dan membentuk satu kesatuan, saling berinteraksi dan bekerja sama antara satu bagian dengan yang lainnya dengan cara-cara tertentu untuk melakukan fungsi pengolahan data, menerima masukan (*input*), berupa data, kemudian mengolahnya (*processing*),

dan menghasilkan keluaran (*output*) berupa informasi sebagai dasar pengambilan keputusan. Contohnya Sistem Informasi Penjualan, Sistem Informasi Persediaan.

Sistem informasi adalah sekumpulan teknologi informasi yang digunakan untuk memproses, menyimpan, mengakses, dan menganalisis data dan informasi untuk meningkatkan efisiensi dan efektivitas bisnis atau organisasi. Basis data adalah bagian dari sistem informasi yang bertanggung jawab untuk menyimpan dan mengelola data dan informasi.

Dalam implementasi sistem informasi, komponen sistem informasi secara umum dapat terdiri dari :

1. Perangkat keras (*hardware*)
2. Perangkat lunak (*software*)
3. Berkas basis data (*file*)
4. Prosedur (*procedure*)
5. Manusia (*brainware*)

Basis data merupakan bagian penting dalam sebuah sistem informasi. Peranan basis data dalam sistem informasi adalah sebagai berikut :

1. Basis data sebagai komponen penyusun sistem informasi
2. Basis data sebagai infrastruktur
3. Basis data sebagai sumber informasi bagi sistem informasi

4. Basis data sebagai sarana mencapai efisiensi sistem informasi
5. Basis data sebagai sarana mencapai efektifitas sistem informasi

Sistem informasi basis data membantu dalam memproses dan mengelola data dan informasi secara efisien, memastikan integritas dan keamanan data, dan memungkinkan analisis data untuk membantu pengambilan keputusan yang lebih baik.

1.6. Pengelolaan Sistem Informasi

Pengelola sebagai spesialis informasi *(information specialist)* menggambarkan pegawai perusahaan yang bertanggung jawab penuh untuk mengembangkan dan memelihara sistem informasi berbasis komputer *(Computer Based Information Systems/CBIS)*, dapat digolongkan menjadi 5 (lima) macam :

1. Analis Sistem
2. Pengelola basis data *(Database Administrator/DBA)*
3. Spesialis Jaringan
4. Pemrogram *(Programmer)*
5. Operator

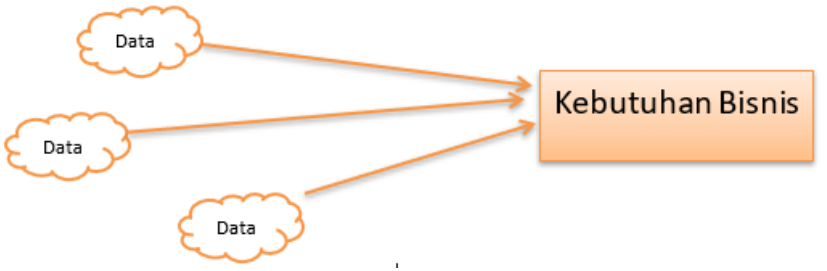
Pengelolaan sistem informasi melibatkan beberapa aktifitas dan tugas yang apat dijabarkan sebagia barikut :

1. Analisis kebutuhan: identifikasi kebutuhan bisnis atau organisasi terkait dengan sistem informasi dan basis data.
2. Desain sistem: menentukan arsitektur dan komponen sistem informasi yang sesuai dengan kebutuhan bisnis atau organisasi.
3. Implementasi: pemasangan dan pengaturan sistem informasi dan basis data.
4. Pengoperasian: memastikan sistem informasi dan basis data berfungsi dengan baik dan memenuhi kebutuhan bisnis atau organisasi.
5. Maintenance: melakukan perawatan dan perbaikan sistem informasi dan basis data sesuai dengan kebutuhan.
6. Analisis dan evaluasi: melakukan analisis dan evaluasi terhadap sistem informasi dan basis data untuk memastikan kinerjanya sesuai dengan standar dan memenuhi kebutuhan bisnis atau organisasi.

Pengelolaan sistem informasi dan basis data yang efektif dan efisien sangat penting untuk memastikan sistem informasi dan basis data dapat memenuhi kebutuhan bisnis atau organisasi dan membantu dalam pengambilan keputusan yang lebih baik.

1.7. Pentingnya Pemahaman Konsep Basis Data

Pemahaaman kosep basis data wajib di ketahui bagi penggunan dan pengelola data dalam organisasi merupakan sumber daya dasar yang penting.



Gambar 1.3 Data sebagai kebutuhan bisnis

Basis data yang lengkap, akurat, mudah digunakan, serta efisien akan meningkatkan kualitas sistem informasi. Penyusunan basis data yang baik dan benar adalah penting agar mampu memenuhi semua/sebagian besar kebutuhan informasi bagi para pemakai dan pengambil keputusan. Pemahaman tentang konsep basis data ini memerlukan dukungan pemahaman tentang beberapa hal, antara lain :

1. Aplikasi basis data
2. Bahasa *SQL (Structure Query Language)*
3. Manajemen basis data
4. Manajemen dan Bisnis
5. *Software* basis data
6. *Hardware* / teknologi informasi

Pemahaman konsep basis data sangat penting karena memiliki beberapa manfaat, seperti:

1. Mengelola data dan informasi secara efisien: konsep basis data membantu dalam mengelola data dan informasi secara terstruktur, memastikan integritas dan keamanan data, dan mempercepat akses dan analisis data.
2. Mempermudah pengambilan keputusan: konsep basis data memungkinkan analisis data yang lebih baik, membantu dalam membuat laporan dan memantau kinerja, sehingga mempermudah pengambilan keputusan.
3. Meningkatkan efisiensi bisnis atau organisasi: konsep basis data membantu dalam mempercepat proses bisnis dan mengurangi redundansi, sehingga meningkatkan efisiensi bisnis atau organisasi.
4. Mendukung keputusan data-driven: konsep basis data memungkinkan akses dan analisis data yang lebih baik, sehingga membantu dalam membuat keputusan berdasarkan data dan informasi yang akurat.
5. Meningkatkan kualitas data dan informasi: konsep basis data memastikan integritas dan keamanan data, membantu dalam mengelola data dan informasi secara terstruktur, dan mempercepat proses validasi data, sehingga meningkatkan kualitas data dan informasi.

Pemahaman konsep basis data sangat penting bagi para profesional teknologi informasi dan bisnis, karena membantu dalam memastikan sistem informasi dan basis data berfungsi dengan baik dan memenuhi kebutuhan bisnis atau organisasi. Misalnya melihan pentingnya basis data adalah dalam industri perbankan. Di industri perbankan, basis data sangat penting untuk mengelola informasi pelanggan, transaksi, dan keuangan. Berikut pembahasan pentingnya basis data dalam industri perbankan:

1. Pengelolaan informasi pelanggan: Basis data dapat digunakan untuk menyimpan informasi pelanggan, seperti nama, alamat, nomor telepon, dan informasi rekening. Ini memungkinkan bank untuk mengakses informasi pelanggan dengan mudah dan cepat, serta memberikan layanan pelanggan yang lebih baik.
2. Pengelolaan transaksi: Basis data juga digunakan untuk menyimpan informasi transaksi, seperti penarikan uang, setoran uang, dan transfer uang. Ini memungkinkan bank untuk melacak aktivitas keuangan pelanggan, memeriksa transaksi yang mencurigakan, dan mengidentifikasi kecurangan.
3. Pengelolaan keuangan: Basis data digunakan untuk menyimpan informasi keuangan bank, seperti laporan keuangan dan neraca. Ini memungkinkan bank untuk melacak kesehatan keuangan mereka dan membuat keputusan bisnis yang cerdas.

4. Analisis data: Basis data digunakan untuk menghasilkan informasi bisnis yang berguna, seperti analisis risiko kredit dan penilaian kelayakan kredit. Ini memungkinkan bank untuk membuat keputusan yang lebih baik dan mengurangi risiko kehilangan uang.

Dalam industri perbankan, pentingnya basis data tidak bisa diragukan lagi. Basis data memungkinkan bank untuk menyimpan dan mengakses informasi pelanggan, transaksi, dan keuangan dengan mudah, serta memberikan layanan pelanggan yang lebih baik dan mengambil keputusan bisnis yang lebih baik. Tanpa basis data, bank akan kesulitan dalam mengelola bisnis mereka secara efisien dan efektif.

1.8. Rangkuman

Data merupakan hal atau fakta yang terjadi setelah proses tindakan terhadap sebuah keputusan. Dalam pengelolaannya untuk mendukung proses pengolahan data maka perkembangan *database* dari zaman ke zaman sangat cepat terutama mampu membantu proses bagaimana hubungan antara data dan informasi dan bagaimana sistem informasi mampu mengolah data menjadi sebuah informasi hal ini perlu didalami dalam konsep dan memahami sistem basis data. Perkembangan basis data dari tahap awal sampai akhir menunjukkan ini kejadian dengan penjelasan sebagai berikut :

1. *File processing* : Pengelolaan data untuk pemrosesan file yang hanya dapat di lakukan oleh satu program aplkasi

untuk persoalan dan system tertentu saja sehingga kemungkinan terjadi kerangkapan data sangat tinggi

2. *Database Management System* : Pengelolaan datan untuk pemrosesan file yang dapat di lakukan dari berbagai aplikasi dengan manajemen file yang baik yang berelasi sehingga tidak memungkinkan terjadi kerangkapan data
3. Expert System, Decision Support System : Merupakan pengembangan dari DBMS yang menjadi paket perangkat lunak yang dapat melakukan pengolahan data menjadi informasi yang mampu memberikan interpretasi, prediksi, diagnosis, perenaan, penyimpanan dan perbaikan kepada pengguna atau pihak yang berkepentingan.
4. Hypertext : Merupakan perkembangan basis data dengan data teks yang dapat berelasi / link dengan informasi lainya, sehingga struktur datanya lebih efisien.

Artificial Inteligent : Merupakan Kecerdasan buatan dari penggunaan basis data sebagai sumber data dan informasi secara otomatis dalam operasional system kecerdasan buatan.

Pentingnya pemahaman tentang basis data agar mampu menjawab kebutuhan bisnis untuk proses pengambilan keputusan sehingga tidak terjadi pemaparan atau pengambilan tindakan tanpa didasari oleh sebuah data

Ada beberapa hal yang perlu dipersiapkan dalam membangun sebuah sistem basis data diantaranya perangkat

keras perangkat lunak data sistem basis data dan ketersediaan sumber daya untuk mengelola sistem basis data tersebut

Pemahaman konsep basis data sangat penting karena memiliki beberapa manfaat, seperti:

1. Mengelola data dan informasi secara efisien: konsep basis data membantu dalam mengelola data dan informasi secara terstruktur, memastikan integritas dan keamanan data, dan mempercepat akses dan analisis data.
2. Mempermudah pengambilan keputusan: konsep basis data memungkinkan analisis data yang lebih baik, membantu dalam membuat laporan dan memantau kinerja, sehingga mempermudah pengambilan keputusan.
3. Meningkatkan efisiensi bisnis atau organisasi: konsep basis data membantu dalam mempercepat proses bisnis dan mengurangi redundansi, sehingga meningkatkan efisiensi bisnis atau organisasi.
4. Mendukung keputusan data-driven: konsep basis data memungkinkan akses dan analisis data yang lebih baik, sehingga membantu dalam membuat keputusan berdasarkan data dan informasi yang akurat.

Meningkatkan kualitas data dan informasi: konsep basis data memastikan integritas dan keamanan data, membantu dalam mengelola data dan informasi secara terstruktur, dan

mempercepat proses validasi data, sehingga meningkatkan kualitas data dan informasi

1.9. Bahan Diskusi

Berikut gambaran sistem informasi yang menggunakan basis data:

1. Sistem Manajemen Inventori Perusahaan: Sebuah perusahaan membutuhkan sistem untuk mengelola inventori barang yang mereka miliki. Basis data dapat digunakan untuk mencatat informasi tentang setiap item, seperti nama, jumlah, harga, dan tanggal masuk. Sistem ini juga dapat digunakan untuk melacak pembelian dan penjualan barang, sehingga perusahaan dapat memantau tingkat stok mereka dan membuat keputusan bisnis yang informatif.
2. Sistem Informasi Karyawan: Sebuah perusahaan membutuhkan sistem untuk mengelola informasi tentang karyawan, seperti nama, alamat, tanggal lahir, dan informasi kontak. Basis data dapat digunakan untuk mencatat informasi tentang setiap karyawan dan membuat laporan yang membantu perusahaan memahami struktur organisasi mereka dan mengelola sumber daya manusia dengan lebih efektif.
3. Sistem Informasi Akademik: Sebuah universitas membutuhkan sistem untuk mengelola informasi tentang mahasiswa, seperti nama, nomor identitas, tanggal lahir, dan informasi akademik. Basis data dapat

digunakan untuk mencatat informasi tentang setiap mahasiswa, seperti nilai, absensi, dan riwayat akademik. Sistem ini juga dapat digunakan untuk membuat laporan dan mengelola kegiatan akademik secara efektif.

4. Jelaskan pentingnya basis data dalam sistem informasi tersebu, dan bagaimana data yang dan informasi apa saja yang di dapatkan dari masing masing sistem informasi tersebut.

BAB II
BASIS DATA DAN SISTEM BASIS DATA

2.1. Pengantar

Sistem basis data adalah kombinasi dari hardware, software, dan prosedur yang digunakan untuk mengelola dan memanipulasi data secara efisien. Sistem basis data terdiri dari beberapa komponen utama, termasuk:

1. Basis data: Kumpulan data yang disimpan dan diorganisasi dalam format tertentu seperti tabel atau dokumen.
2. Manajemen basis data (DBMS): Software yang digunakan untuk mengelola basis data dan membuat data mudah diakses dan diambil.
3. Hardware: Perangkat keras seperti komputer, disk, dan memori yang digunakan untuk menyimpan dan mengakses data.
4. Prosedur: Prosedur yang digunakan untuk memasukkan, mengambil, dan memanipulasi data.
5. Pengguna: Orang atau aplikasi yang menggunakan data dalam sistem basis data.

Sistem basis data memiliki beberapa kelebihan, termasuk efisiensi dalam mengakses dan memanipulasi data, integritas data yang tinggi, dan skalabilitas yang baik. Sistem basis data juga membantu untuk memastikan keamanan dan privasi data dengan mengatur hak akses dan membatasi akses ke data yang sensitif.

Dalam pengembangan aplikasi, sistem basis data sering digunakan sebagai bagian dari arsitektur aplikasi untuk menyimpan informasi seperti informasi pengguna, transaksi, dan data produk. Penggunaan sistem basis data membuat aplikasi lebih robust dan mudah dikembangkan dan diperbaiki.

2.2. Definisi Basis Data

Secara gramatikal Basis Data tersusun atas dua suku kata yaitu BASIS + DATA. BASIS dapat dikatakan sebagai dasar, tempat mankal misalnya "Basis Massa PDIP", "Basis Massa PKB:, "Basis tukang Ojek". Sedangkan DATA dapat diakatakan sebagai fakta atau kenyataan misalnya SPP per bulan menjadi Rp. 2.000.000

Dari ungkapan diatas dapat dinyatakan bahwa basis data dapat disebut sebagai tempat berkumpulnya beberapa data yang saling berinteraksi, dapat dimanfaatkan oleh pengguna dalam mengambil sebuah keputusan. Untuk itu perlu kita pahami Apa yang menjadi sebuah data pernyataan pada informasi. Perhatikan pernyataan berikut ini :

1. Gaji Pokok Karyawan akan naik

2. Budi dalam sebulan tidak hadir sebanyak 4 hari
3. Penjualan perusahaan meningkat sebesar 25%
4. Pak Rusdi adalah seorang pegawai negeri sipil
5. Harga Blackberry Dakota turun

Manakah yang menjadi data dari pernyataan diatas ?

Pengertian basis data suatu kumpulan data terhubung (*interrelated data*) yang disimpan secara bersama-sama pada suatu media, tanpa menghadap satu sama lain atau tidak perlu suatu kerangkapan data (kalaupun ada maka kerangkapan data tersebut harus seminimal mungkin dan terkontrol *(controlled redudancy),* data disimpan dengan cara-cara tertentu. Tujuan Basis Data adalah :

1. Dapat menampilkan data Kembali dengan mudah
2. Dapat digunakan oleh satu atau lebih program aplikasi secara optimal
3. Mempermudah proses penambahan, pengambilan dan modifikasi data dengan cara yang terkontrol

Kriteria penting yang harus dipenuhi basis data (Martin, 1975, dalam buku Basis Data dalam Tinjauan Konseptual, Edhy Sutanta, 2011, 30) :

1. Berorientasi pada Data
2. Data dalam basis data dapat berkembang dengan mudah, baik volume maupun strukturnya

3. Data yang ada dapat memenuhi kebutuhan sistem-sistem baru secara mudah
4. Data dapat digunakan dengan cara yang berbeda-beda
5. Kerangkapan data (*data redudancy*) minimal

Pengelolaan data dalam file tradisional (*file processing*), (Martin, 1975, dalam buku Basis Data dalam Tinjauan Konseptual, Edhy Sutanta, 2011, 31), yaitu :

1. Hanya dapat digunakan oleh satu program aplikasi
2. Berhubungan dengan suatu persoalan tertentu untuk sistem yang direncanakan
3. Perkembangan data hanya mungkin terjadi pada volume data saja
4. Hanya dapat digunakan dengan satu cara tertentu saja
5. Kerangkapan data terlalu sering muncul
6. Kriteria-kriteria yang telah dijelaskan sebelumnya telah membedakan antara pengorganisasian data secara basis data *(database processing)* dengan pengelolaan data dalam file tradisional *(file processing)*.

Berikut definisi-definisi basis data *(database)* menurut para pakar :

1. C.J Date (1995) : beberapa kumpulan data yang akan tetap tersimpan, digunakan oleh sistem-sistem aplikasi yang diberikan oleh organisasi.

2. J.L. Whitten & L.D Bentley (1998) : sekumpulan data dalam file yang saling terhubung *(interrelated file)*, record dalam file harus mengizinkan adanya kerelasian ke *record-record* lain dalam file yang lain.
3. Raghu Ramakrishnan (1998) sekumpulan data yang saling berhubungan yang menjelaskan aktivitas-aktivitas pada sebuah organisasi.
4. Abraham Silberschatz, Henry F. Korth, dan S. Sudarshan (2001) : sekumpulan data yang saling berhubungan dan menjadi bagian dari DBMS.
5. Raymond McLeod dan George Schell (2001) : keseluruhan data yang disimpan dalam sistem komputer yang menjadi sumber daya organisasi

Dari gambaran diatas dinyatakan basis data adalah kumpulan data yang terstruktur dan terorganisir, yang digunakan untuk menyimpan, memanipulasi, dan memanfaatkan data dan informasi. Basis data memiliki beberapa fitur, seperti:

1. Struktur: data dan informasi disimpan dalam format yang terstruktur, seperti tabel, baris, dan kolom, yang mempermudah akses dan analisis.
2. Integritas: basis data memastikan integritas data dan informasi, seperti memastikan tidak adanya duplikasi data atau kesalahan data.

3. Keamanan: basis data memastikan keamanan data dan informasi, seperti melindungi data dan informasi dari akses yang tidak sah atau modifikasi yang tidak sah.
4. Scalability: basis data memastikan skalabilitas data dan informasi, seperti memastikan sistem dapat beradaptasi dengan pertumbuhan data dan informasi.
5. Ekstensibilitas: basis data memastikan ekstensibilitas data dan informasi, seperti memastikan sistem dapat menambahkan data dan informasi baru secara mudah dan efisien.

Basis data digunakan dalam berbagai aplikasi dan sistem, seperti sistem informasi manajemen, aplikasi bisnis, dan aplikasi web, dan membantu dalam mengelola data dan informasi secara efisien dan mempermudah akses dan analisis data.

2.3. Definisi Sistem Basis Data

Sistem Basis Data merupakan sekumpulan basis data dalam suatu sistem yang mungkin tidak ada hubungan satu sama lain, tetapi secara keseluruhan mempunyai hubungan sebagai suatu sistem dengan didukung oleh komponen lainnya. Menurut Martin (1975) pengertian sistem basis data merupakan Sekumpulan subsistem yang terdiri atas basis data dengan para pemakai yang menggunakan basis data secara bersama-sama, personal-personal yang merancang dan mengelola basis data, serta sistem komputer untuk mendukungnya.

Sistem Basis Data (DBMS) adalah software atau aplikasi yang digunakan untuk memanajemen dan mengelola basis data. DBMS memiliki beberapa fungsi, seperti:

1. Mengelola data dan informasi: DBMS membantu dalam mengelola data dan informasi secara terstruktur, memastikan integritas dan keamanan data, dan mempercepat akses dan analisis data.
2. Mempermudah pengambilan keputusan: DBMS memungkinkan analisis data yang lebih baik, membantu dalam membuat laporan dan memantau kinerja, sehingga mempermudah pengambilan keputusan.
3. Meningkatkan efisiensi bisnis atau organisasi: DBMS membantu dalam mempercepat proses bisnis dan mengurangi redundansi, sehingga meningkatkan efisiensi bisnis atau organisasi.
4. Mendukung keputusan data-driven: DBMS memungkinkan akses dan analisis data yang lebih baik, sehingga membantu dalam membuat keputusan berdasarkan data dan informasi yang akurat.
5. Menjaga integritas dan keamanan data: DBMS memastikan integritas dan keamanan data, seperti memastikan tidak adanya duplikasi data atau kesalahan data, dan melindungi data dan informasi dari akses yang tidak sah atau modifikasi yang tidak sah.

DBMS merupakan bagian penting dari sistem informasi dan membantu dalam memastikan sistem basis data berfungsi

dengan baik dan memenuhi kebutuhan bisnis atau organisasi. Beberapa contoh DBMS yang populer adalah Microsoft SQL Server, Oracle, MySQL, dan PostgreSQL.

Elemen penting dalam sistem basis data adalah sebagai berikut (Sutanta, 2004) :

1. Basis Data sebagai inti dari sistem basis data
2. Perangkat lunak *(software)* untuk perancangan dan pengelolaan basis data
3. Perangkat keras *(hardware)* sebagai pendukung operasi pengolahan data
4. Manusia *(brainware)* yang mempunyai perang penting dalam sistem tersebut, yaitu sebagai pemakai atau para spesialis informasi yang mempunyai fungsi sebagai perancang atau pengelola

DBMS (Database Management System) adalah perangkat lunak yang dirancang untuk memfasilitasi pengelolaan basis data. Berikut adalah beberapa contoh perangkat lunak DBMS:

1. MySQL: MySQL adalah salah satu DBMS open-source yang paling populer dan banyak digunakan di seluruh dunia. MySQL mendukung banyak sistem operasi, termasuk Windows, Linux, dan Mac OS, dan dapat digunakan untuk membangun aplikasi web, aplikasi desktop, dan banyak lagi.

2. Oracle Database: Oracle Database adalah salah satu DBMS yang paling canggih dan kompleks yang tersedia. Oracle Database digunakan oleh banyak organisasi besar dan pemerintah, dan mendukung banyak fitur canggih seperti manajemen transaksi, penyimpanan data terdistribusi, dan skalabilitas horizontal.
3. Microsoft SQL Server: Microsoft SQL Server adalah DBMS yang populer dan banyak digunakan di lingkungan bisnis dan perusahaan. SQL Server mendukung banyak fitur canggih seperti manajemen transaksi, pengolahan paralel, dan pemulihan bencana.
4. PostgreSQL: PostgreSQL adalah DBMS open-source yang canggih dan handal. PostgreSQL mendukung banyak fitur canggih seperti manajemen transaksi, pengolahan paralel, dan penyimpanan data terdistribusi.
5. MongoDB: MongoDB adalah DBMS yang populer dan banyak digunakan untuk aplikasi web dan mobile. MongoDB menggunakan model data dokumen, yang memungkinkan pengguna untuk menyimpan data dalam format yang lebih fleksibel dan mudah dikelola.
6. SQLite: SQLite adalah DBMS open-source yang ringan dan sederhana. SQLite digunakan oleh banyak aplikasi desktop dan mobile karena kecepatannya, ukuran file yang kecil, dan kemampuannya untuk menyimpan basis data dalam satu file.

7. IBM DB2: IBM DB2 adalah DBMS yang canggih dan kompleks yang banyak digunakan di lingkungan bisnis dan perusahaan. DB2 mendukung banyak fitur canggih seperti manajemen transaksi, pemulihan bencana, dan penyimpanan data terdistribusi.

Perangkat lunak dapat dikategorikan dalam tiga bagian yaitu (Sutanta, 2005) :

1. Perangkat lunak sistem operasi *(Operating System/OS)*, digunakan untuk mengendalikan dan mengkoordinasikan kegiatan dari perangkat keras sistem komputer.
2. Perangkat lunak bahasa *(language software)*, digunakan untuk menterjemahkan instruksi-instruksi yang ditulis dalam bahasa pemrograman ke dalam bahasa mesin supaya dapat dimengerti komputer.
3. Perangkat lunak aplikasi *(application software)*, program yang ditulis dan diterjemahkan oleh language *software* untuk menyelesaikan aplikasi tertentu.

Perangkat keras pendukung operasi pengolahan data adalah sebagai berikut :

1. Perangkat keras unit masukan
    a. Alat input langsung
    b. Alat input tidak langsung
2. Perangkat keras unit keluaran

3. Perangkat keras unit pengolah *Central Processing Unit (CPU)*
4. Perangkat keras memori sekunder

2.4. Hirarki Data

Hierarki data adalah model pengoranisasian data dengan mengunakan tingkatan atau lapisan dalam mengelompokan dan informasi dimana setiap tingkatan atau lapisan memiliki karakteristik dan spesifikasi yang berbeda atau juga bisa dikatakan dalam bentuk representasi visual yang memperlihatkan struktur dan relasi antar data dan informasi yang berbeda dalam suatu sistem basis data yang mempunyai tingkat abstraksi dan keterkaitan.

Konsep hierarki data sangat penting dalam basis data dan membantu dalam mengelola data dan informasi secara efisien. Berikut ini digambarkan hierarki data menurut Martin 1975 : Tujuan utama dari hierarki data adalah untuk memastikan bahwa data dapat disimpan, dikelola, dan diakses dengan efisien dan memastikan bahwa data yang lebih penting dapat diakses dengan cepat dan mudah.

Beberapa tujuan dari hierarki data adalah:

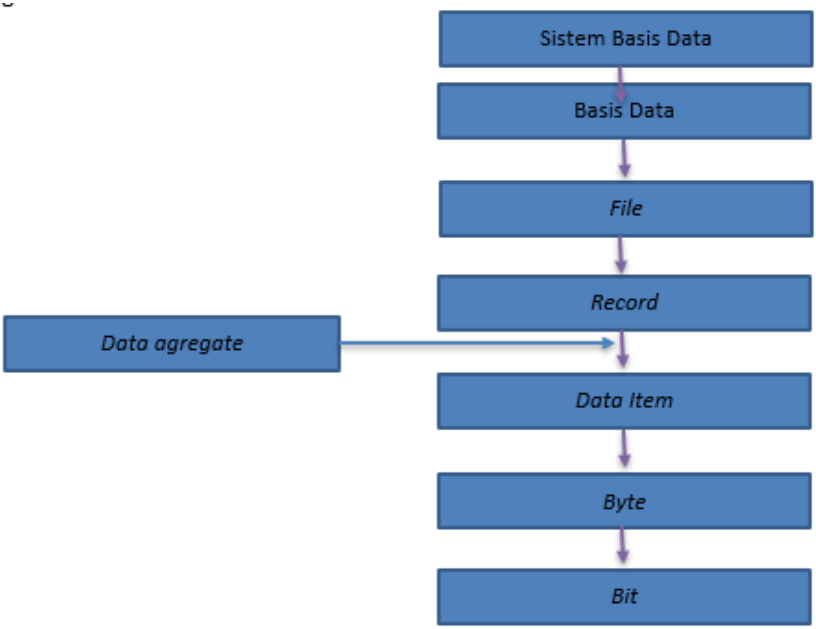
1. Peningkatan efisiensi akses data: Hierarki data membantu untuk memastikan bahwa data yang sering diakses dapat diakses dengan cepat dan mudah.
2. Peningkatan integritas data: Hierarki data memastikan bahwa data yang penting dapat ditempatkan pada

tingkat yang lebih tinggi dalam hierarki untuk memastikan integritas data.

3. Pemeliharaan data: Hierarki data memastikan bahwa data dapat dikelola dan diteruskan dengan baik dari tingkat satu ke tingkat lain.
4. Peningkatan keamanan data: Hierarki data membantu untuk memastikan bahwa data yang sensitif dapat ditempatkan pada tingkat yang lebih tinggi dalam hierarki untuk memastikan keamanan data.
5. Peningkatan scalabilitas sistem: Hierarki data membantu untuk memastikan bahwa sistem basis data dapat dikembangkan dan ditingkatkan dengan mudah untuk memenuhi kebutuhan bisnis yang berubah.

Dengan demikian, hierarki data memainkan peran penting dalam memastikan bahwa sistem basis data dapat berfungsi dengan efisien, memastikan integritas dan keamanan data, dan memastikan bahwa sistem basis data dapat dikembangkan dan ditingkatkan seiring waktu untuk memenuhi kebutuhan bisnis yang berubah.

Berikut struktur hierarki data menurut martin dapat digambarkan dalam urutan sebagai berikut :



Gambar 2.1 Hierarki Data

Berikut keterangan dari Hierarki Data (Martin, 1975) :

1. Sistem Basis Data, sekumpulan susbsistem yang terdiri atas basis data dengan para pemakai yang menggunakan basis data secara bersama-sama, personal-personal yang merancang dan mengelola basis data, teknik-teknik untuk merancang dan mengelola basis data, serta sistem komputer untuk mendukungnya.
2. Basis Data, sekumpulan dari bermacam-macam tipe *record* yang memiliki hubungan antar-*record* dan rincian data terhadap objek tertentu.

3. File, merupakan sekumpulan *record* sejenis secara relasi yang tersimpan dalam media penyimpanan sekunder
4. *Record*, merupakan sekumpulan field/atribut/data item yang saling berhubungan terhadap objek tertentu.
5. Data item/*field*/atribut, merupakan unit terkecil yang disebut data, yaitu sekumpulan *byte* yang mempunyai makna.
6. Data *agregate*, merupakan sekumpulan data item/field/atribut dengan ciri tertentu diberi nama.
7. *Byte*, bagian terkecil yang dialamatkan dalam memori.
8. *Bit*, sistem biner yang terdiri atas dua macam nilai, yaitu 0 dan 1.

2.5. Rangkuman

Pengertian basis data suatu kumpulan data terhubung *(interrelated data)* yang disimpan secara bersama-sama pada suatu media, tanpa mengatap satu sama lain atau tidak perlu suatu kerangkapan data (kalaupun ada maka kerangkapan data tersebut harus seminimal mungkin dan terkontrol *(controlled redudancy)*, data disimpan dengan cara-cara tertentu.

Sistem Basis Data (DBMS) adalah software atau aplikasi yang digunakan untuk memanajemen dan mengelola basis data. DBMS memiliki beberapa fungsi, seperti:

1. Mengelola data dan informasi: DBMS membantu dalam mengelola data dan informasi secara terstruktur, memastikan integritas dan keamanan data, dan mempercepat akses dan analisis data.
2. Mempermudah pengambilan keputusan: DBMS memungkinkan analisis data yang lebih baik, membantu dalam membuat laporan dan memantau kinerja, sehingga mempermudah pengambilan keputusan.
3. Meningkatkan efisiensi bisnis atau organisasi: DBMS membantu dalam mempercepat proses bisnis dan mengurangi redundansi, sehingga meningkatkan efisiensi bisnis atau organisasi.
4. Mendukung keputusan data-driven: DBMS memungkinkan akses dan analisis data yang lebih baik, sehingga membantu dalam membuat keputusan berdasarkan data dan informasi yang akurat.
5. Menjaga integritas dan keamanan data: DBMS memastikan integritas dan keamanan data, seperti memastikan tidak adanya duplikasi data atau kesalahan data, dan melindungi data dan informasi dari akses yang tidak sah atau modifikasi yang tidak sah.

DBMS merupakan bagian penting dari sistem informasi dan membantu dalam memastikan sistem basis data berfungsi dengan baik dan memenuhi kebutuhan bisnis atau organisasi. Beberapa contoh DBMS yang populer adalah Microsoft SQL Server, Oracle, MySQL, dan PostgreSQL.

DBMS (Database Management System) adalah perangkat lunak yang dirancang untuk memfasilitasi pengelolaan basis data. Berikut adalah beberapa contoh perangkat lunak DBMS:

1. MySQL: MySQL adalah salah satu DBMS open-source yang paling populer dan banyak digunakan di seluruh dunia. MySQL mendukung banyak sistem operasi, termasuk Windows, Linux, dan Mac OS, dan dapat digunakan untuk membangun aplikasi web, aplikasi desktop, dan banyak lagi.
2. Oracle Database: Oracle Database adalah salah satu DBMS yang paling canggih dan kompleks yang tersedia. Oracle Database digunakan oleh banyak organisasi besar dan pemerintah, dan mendukung banyak fitur canggih seperti manajemen transaksi, penyimpanan data terdistribusi, dan skalabilitas horizontal.
3. Microsoft SQL Server: Microsoft SQL Server adalah DBMS yang populer dan banyak digunakan di lingkungan bisnis dan perusahaan. SQL Server mendukung banyak fitur canggih seperti manajemen transaksi, pengolahan paralel, dan pemulihan bencana.
4. PostgreSQL: PostgreSQL adalah DBMS open-source yang canggih dan handal. PostgreSQL mendukung banyak fitur canggih seperti manajemen transaksi, pengolahan paralel, dan penyimpanan data terdistribusi.

5. MongoDB: MongoDB adalah DBMS yang populer dan banyak digunakan untuk aplikasi web dan mobile. MongoDB menggunakan model data dokumen, yang memungkinkan pengguna untuk menyimpan data dalam format yang lebih fleksibel dan mudah dikelola.
6. SQLite: SQLite adalah DBMS open-source yang ringan dan sederhana. SQLite digunakan oleh banyak aplikasi desktop dan mobile karena kecepatannya, ukuran file yang kecil, dan kemampuannya untuk menyimpan basis data dalam satu file.
7. IBM DB2: IBM DB2 adalah DBMS yang canggih dan kompleks yang banyak digunakan di lingkungan bisnis dan perusahaan. DB2 mendukung banyak fitur canggih seperti manajemen transaksi, pemulihan bencana, dan penyimpanan data terdistribusi.
8. File, merupakan sekumpulan *record* sejenis secara relasi yang tersimpan dalam media penyimpanan sekunder
9. *Record*, merupakan sekumpulan field/atribut/data item yang saling berhubungan terhadap objek tertentu.
10. Data item/*field*/atribut, merupakan unit terkecil yang disebut data, yaitu sekumpulan *byte* yang mempunyai makna.
11. Data *agregate*, merupakan sekumpulan data item/field/atribut dengan ciri tertentu diberi nama.
12. *Byte*, bagian terkecil yang dialamatkan dalam memori.

13. *Bit*, sistem biner yang terdiri atas dua macam nilai, yaitu 0 dan 1.

2.6. Bahan Diskusi

Berikut beberapa gambaran sistem informasi dalam beberapa perusahaan :

1. Sistem Informasi Inventory Barang: Studi kasus ini membahas bagaimana membuat sistem informasi inventory barang yang menggunakan basis data. Sistem ini harus mampu mengelola data barang, stok barang, dan laporan penjualan.
2. Sistem Informasi Penjualan Tiket: Studi kasus ini membahas bagaimana membuat sistem informasi penjualan tiket yang menggunakan basis data. Sistem ini harus mampu mengelola data tiket, laporan penjualan, dan laporan keuangan.
3. Sistem Informasi Kepegawaian: Studi kasus ini membahas bagaimana membuat sistem informasi kepegawaian yang menggunakan basis data. Sistem ini harus mampu mengelola data pegawai, absensi pegawai, dan laporan gaji.
4. Sistem Informasi Perpustakaan: Studi kasus ini membahas bagaimana membuat sistem informasi perpustakaan yang menggunakan basis data. Sistem ini harus mampu mengelola data buku, peminjaman buku, dan laporan peminjaman.

5. Sistem Informasi Manajemen Rumah Sakit: Studi kasus ini membahas bagaimana membuat sistem informasi manajemen rumah sakit yang menggunakan basis data. Sistem ini harus mampu mengelola data pasien, data dokter, dan laporan kunjungan pasien.

Coba diskusiakan bagaimana membuat sistem informasi menggunakan basis data dan mengatasi masalah yang mungkin terjadi selama pembuatan sistem.

BAB III

TUJUAN DAN KEUNTUNGAN PENGEMBANGAN BASIS DATA

3.1. Pengantar

Tujuan utama dari pengembangan basis data adalah untuk memastikan bahwa data dapat disimpan, dikelola, dan diakses dengan efisien dan memastikan bahwa sistem basis data dapat memenuhi kebutuhan bisnis yang berubah. Beberapa tujuan pengembangan basis data adalah:

1. Efisiensi: Memastikan bahwa data dapat disimpan, dikelola, dan diakses dengan cepat dan mudah.
2. Integritas data: Memastikan bahwa data yang disimpan dalam sistem basis data akurat dan dapat dipercaya.
3. Keamanan data: Memastikan bahwa data yang disimpan dalam sistem basis data aman dan terlindung dari pengaksesan yang tidak sah.
4. Skalabilitas: Memastikan bahwa sistem basis data dapat dikembangkan dan ditingkatkan untuk memenuhi kebutuhan bisnis yang berubah.
5. Keterbukaan: Memastikan bahwa data dapat diakses oleh pemakai yang berbeda dalam organisasi dengan mudah.

6. Fleksibilitas: Memastikan bahwa sistem basis data dapat digunakan untuk menangani berbagai jenis data dan memenuhi berbagai kebutuhan bisnis.
7. Kurva belajar pendek: Memastikan bahwa pemakai baru dapat memahami dan menggunakan sistem basis data dengan cepat dan mudah.

Dengan demikian, pengembangan basis data memainkan peran penting dalam memastikan bahwa sistem basis data dapat berfungsi dengan efisien dan memenuhi kebutuhan bisnis yang berubah. Ini juga membantu untuk memastikan bahwa data dapat diakses dan dikelola dengan mudah dan memastikan bahwa sistem basis data dapat ditingkatkan seiring waktu untuk memenuhi kebutuhan bisnis yang berubah.

Pengembangan basis data memiliki beberapa keuntungan utama, seperti:

1. Efisiensi: Sistem basis data yang dikembangkan dengan baik memastikan bahwa data dapat disimpan, dikelola, dan diakses dengan cepat dan mudah, yang meningkatkan efisiensi bisnis.
2. Integritas data: Pengembangan basis data memastikan bahwa data yang disimpan dalam sistem basis data akurat dan dapat dipercaya, memastikan bahwa bisnis memiliki informasi yang tepat untuk membuat keputusan yang tepat.

3. Keamanan data: Pengembangan basis data memastikan bahwa data yang disimpan dalam sistem basis data aman dan terlindung dari pengaksesan yang tidak sah, memastikan bahwa bisnis memiliki kontrol yang baik atas data mereka.
4. Skalabilitas: Pengembangan basis data memastikan bahwa sistem basis data dapat dikembangkan dan ditingkatkan untuk memenuhi kebutuhan bisnis yang berubah, memastikan bahwa bisnis dapat mengadaptasi dengan mudah dan cepat terhadap perubahan pasar.
5. Keterbukaan: Pengembangan basis data memastikan bahwa data dapat diakses oleh pemakai yang berbeda dalam organisasi dengan mudah, memastikan bahwa bisnis dapat bekerja secara efisien dan terkoordinasi.
6. Fleksibilitas: Pengembangan basis data memastikan bahwa sistem basis data dapat digunakan untuk menangani berbagai jenis data dan memenuhi berbagai kebutuhan bisnis, memastikan bahwa bisnis dapat beradaptasi dengan mudah dan cepat terhadap perubahan pasar.

Dengan demikian, pengembangan basis data membantu bisnis untuk meningkatkan efisiensi, memastikan integritas data, memastikan keamanan data, dan memastikan bahwa sistem basis data dapat dikembangkan dan ditingkatkan untuk memenuhi kebutuhan bisnis yang berubah. Ini juga membantu bisnis untuk bekerja secara efisien dan terkoordinasi dan

memastikan bahwa data dapat diakses dan dikelola dengan mudah.

3.2. Tujuan Pengembangan Basis Data

Pengembangan basis data memiliki beberapat tahap diataranya adalah analisis kebutuhan, disain basis data, impelemtasi, pengujian dan validasi serta pemeiliharaan.

Adapun tujuan pengembangan basis data adalah untuk memastikan basis data berfungsi dengan baik dan memenuhi kebutuhan bisnis atau organisasi, seperti:

1. Mengelola data dan informasi: memastikan data dan informasi dapat diakses, dianalisis, dan dikelola dengan baik.
2. Mempermudah pengambilan keputusan: memastikan data dan informasi dapat digunakan untuk membantu pengambilan keputusan yang data-driven.
3. Meningkatkan efisiensi bisnis atau organisasi: memastikan sistem basis data berfungsi dengan efisien dan membantu dalam mengurangi redundansi.
4. Mendukung keputusan data-driven: memastikan data dan informasi dapat digunakan untuk membuat keputusan yang akurat dan data-driven.
5. Menjaga integritas dan keamanan data: memastikan integritas dan keamanan data, seperti memastikan tidak adanya duplikasi data atau kesalahan data, dan

melindungi data dan informasi dari akses yang tidak sah atau modifikasi yang tidak sah.

Tujuan Pengembangan Basis Data oleh James Martin (1975) dibedakan atas 2 (dua), yaitu :

1. Tujuan Primer, sebagai tujuan utama yang ingin dicapai dalam setiap usaha perancangan dan pengembangan basis data yaitu mamasikan basis data dapat memenuhi kebutuhan bisnis orgnisasi dan memastikan basis data dapat berfungsi dengan baik dan efisien diantaranya :
    - Mengelola dan menyimpan data dan informasi dengan baik dan efisien.
    - Memberikan akses yang cepat dan mudah ke data dan informasi.
    - Mempermudah analisis dan pengambilan keputusan berdasarkan data dan informasi.
    - Mendukung integritas dan keamanan data dan informasi.
    - Memastikan basis data dapat diperbarui dan dikembangkan sesuai dengan perkembangan bisnis atau organisasi.
2. Tujuan Sekunder, merupakan tujuan tambahan untuk mencapai tujuan primer diantaranya adalah untuk memastikan efisiensi, integritas, dan keamanan data. Ini bisa meliputi tugas-tugas seperti mengoptimalkan performa database, memastikan data tidak rusak atau

hilang, dan melindungi data dari akses yang tidak sah. Hal ini juga dapat termasuk mengatur akses ke data dan memastikan bahwa hanya pengguna yang memiliki otoritas yang sesuai yang dapat memodifikasi atau melihat informasi.

3.3. Tujuan Primer Pengembangan Basis Data

Tujuan utama pengembangan basis data (database) adalah untuk menyimpan, mengelola, dan mengatur data sehingga dapat diakses dan dimanfaatkan dengan mudah. Basis data dirancang untuk memungkinkan pengguna untuk menyimpan, mengambil, dan memperbarui informasi dengan mudah dan efisien. Beberapa tujuan primer pengembangan basis data adalah:

1. Efisiensi: Basis data dirancang untuk menyimpan data secara efisien dan mengoptimalkan proses untuk mengambil dan memperbarui data. Dengan menggunakan basis data, pengguna dapat mengakses data dengan cepat dan mudah, tanpa perlu melakukan proses yang rumit dan memakan waktu.
2. Konsistensi: Basis data dirancang untuk memastikan konsistensi data. Hal ini dilakukan dengan memastikan bahwa setiap entitas dalam basis data memiliki atribut yang sama dan bahwa data di dalamnya konsisten dan terstruktur.
3. Keamanan: Basis data dirancang untuk memastikan bahwa data aman dan hanya dapat diakses oleh

pengguna yang berwenang. Ini dilakukan dengan membatasi akses ke data dan menerapkan kontrol keamanan pada basis data.

4. Ketersediaan: Basis data dirancang untuk memastikan bahwa data tersedia setiap saat. Ini dilakukan dengan memastikan bahwa sistem basis data selalu berjalan dan siap digunakan kapan saja.

Dengan menggunakan basis data, perusahaan atau organisasi dapat mengelola data mereka dengan lebih efisien dan efektif, dan membuat keputusan yang lebih baik berdasarkan informasi yang akurat dan terpercaya. Terdapat 14 (empat belas) Tujuan Primer pengembangan Basis Data (James Martin, 1975), yaitu :

1. Data dalam basis data dapat digunakan oleh banyak pemakai.
2. Menjaga investasi intelektual, kebutuhan baru akan pengolahan data dapat terpenuhi dari data yang tersedia saat ini.
3. Penekanan biaya, ada 3 hal yang berkaitan, yaitu : biaya penyimpanan, biaya penggunaan data, dan tingginya biaya ketika membuat perubahan-perubahan pada basis data.
4. Menghilangkan pengembangan sistem ganda *(poliferasi)*
5. Kinerja *(performance)*

6. Kejelasan *(clarity)*
7. Kemudahan pemakaian
8. Fleksibilitas penggunaan *(flexibility)*
9. Kebutuhan data yang tidak terantisipasi dapat dipenuhi dengan cepat.
10. Perubahan yang mudah
11. Akurasi *(accuracy)* dan konsistensi *(consictency)*
12. Privasi *(privacy)*
13. Keamanan *(security)*
14. Ketersediaan *(availability)*

3.4. Tujuan Sekunder Pengembangan Basis Data

Selain tujuan utama seperti yang telah dijelaskan sebelumnya, pengembangan basis data juga memiliki beberapa tujuan sekunder, di antaranya adalah:

1. Meningkatkan produktivitas: Basis data dapat membantu meningkatkan produktivitas karena data dapat diakses dengan cepat dan mudah, dan proses bisnis dapat diotomatisasi dengan lebih baik. Dengan menggunakan basis data, perusahaan dapat menghemat waktu dan tenaga yang sebelumnya digunakan untuk mengelola data secara manual.
2. Meningkatkan kualitas data: Basis data dapat membantu meningkatkan kualitas data dengan memastikan bahwa

data di dalamnya terstruktur dengan baik dan konsisten. Hal ini dapat membantu mengurangi kesalahan manusia dan membuat data lebih akurat dan terpercaya.

3. Memungkinkan integrasi data: Basis data memungkinkan data dari berbagai sumber dan aplikasi dapat diintegrasikan dan digunakan bersama. Ini dapat membantu perusahaan untuk mengambil keputusan yang lebih baik dan memperoleh wawasan yang lebih dalam tentang operasi mereka.
4. Mengurangi biaya: Basis data dapat membantu mengurangi biaya karena data dapat diakses dan dikelola dengan lebih efisien. Ini dapat mengurangi biaya yang terkait dengan penyimpanan, pengolahan, dan pengambilan data.
5. Meningkatkan pelayanan pelanggan: Basis data dapat membantu meningkatkan pelayanan pelanggan dengan memungkinkan perusahaan untuk mengakses informasi pelanggan dengan cepat dan mudah. Hal ini dapat membantu perusahaan untuk memberikan layanan yang lebih baik dan memenuhi kebutuhan pelanggan dengan lebih baik pula.

Secara keseluruhan, pengembangan basis data memiliki banyak tujuan sekunder yang penting bagi perusahaan dan organisasi untuk mencapai keberhasilan bisnis. Terdapat 14 (empat belas) Tujuan Sekunder pengembangan Basis Data (James Martin, 1975), yaitu :

1. Kebebasan Data secara fisik
2. Kebebasan Data secara logika
3. Pengendalian atau minimalisasi kerangkapan data
4. Kecepatan Akses
5. Kecepatan pencarian
6. Standarisasi Data
7. Tersedianya kamus data
8. Antarmuka pemrogram tingkat tinggi
9. Bahasa *end-user*
10. Pengendalian integritas
11. Kecepatan pemulihan dari kerusakan
12. Kemampuan perubahan penyesuaian (*tuning*)
13. Perancangan dan pengawasan alat-alat
14. Pengorganisasian kembali atau migrasi data dapat dilakukan secara otomatis

Berikut ini terdapat beberapa tujuan basis data secara umum terdiri atas:

1. Kecepatan dan Kemudahan *(speed)* yakni agar pengguna basis data bisa:
    a. menyimpan data
    b. melakukan perubahan/manipulasi terhadap data

c. menampilkan kembali data dengan lebih cepat dan mudah dibandingkan dengan cara biasa (baik manual ataupun elektronis).

2. Efisiensi Ruang Penyimpanan *(Space)*

   Dengan basis data kita mampu melakukan penekanan jumlah *redundancy* (pengulangan) data, baik dengan menerapkan sejumlah pengkodean atau dengan membuat relasi-relasi antara kelompok data yang saling berhubungan.

   Agar data sesuai dengan aturan dan batasan tertentu dengan cara memanfaatkan pengkodean atau pembentukan relasi antar data bersama dengan penerapan aturan/batasan *(constraint)* tipe data, domain data, keunikan data dsb.

3. Ketersediaan *(Availability)*

   Agar data bisa diakses oleh setiap pengguna yang membutuhkan, dengan penerapan teknologi jaringan serta melakukan pemindahan/penghapusan data yang sudah tidak digunakan / kadaluwarsa untuk menghemat ruang penyimpanan.

4. Kelengkapan *(Completeness)*

   Agar data yang dikelola senantiasa lengkap baik relatif terhadap kebutuhan pemakai maupun terhadap waktu, dengan melakukan penambahan baris- baris data ataupun melakukan perubahan struktur pada basis data;

yakni dengan menambahkan *field* pada tabel atau menambah tabel baru.

5. Keamanan *(Security)*

   Agar data yang bersifat rahasia atau proses yang vital tidak jatuh ke orang / pengguna yang tidak berhak, yakni dengan penggunaan *account (username* dan *password)* serta menerapkan pembedaan hak akses setiap pengguna terhadap data yang bisa dibaca atau proses yang bisa dilakukan.

6. Kebersamaan *(Sharability)*

   Agar data yang dikelola oleh sistem mendukung lingkungan multiuser (banyak pemakai), dengan menjaga / menghindari munculnya problem baru seperti inkonsistensi data (karena terjadi perubahan data yang dilakukan oleh beberapa user dalam waktu yang bersamaan) atau kondisi *deadlock* (karena ada banyak pemakai yang saling menunggu untuk menggunakan data).

3.5. Keuntungan Pengembagan Basis Data Menurut Para Ahli

Penyusunan basis data dengan maksud untuk mengatasi permasalahan-permasalahan pada saat pengolahan data. Keuntungan dari Pengembangan Basis Data (Martin, 1975) :

1. Kerangkapan data dapat diminimalkan

2. Inkonsistensi data dapat dihindari
3. Data dalam basis data dapat digunakan secara bersama *(multiuser)*
4. Standarisasi data dapat dilakukan
5. Pembatasan untuk keamanan data dapat diterapkan
6. Integritas data dapat dipelihara
7. Perbedaan kebutuhan data dapat diseimbangkan

Keuntungan Pengembangan Basis Data (Kamran Parsaye, Mark Chignell, Setrag Khoshafian, dan Harry Wong, 1989) adalah sebagai berikut :

1. Akses bersama data untuk pengguna yang berbeda
2. Keamanan data
3. Meningkatkan kemudahan dan efisiensi *update* untuk pemeliharaan

Keuntungan Pengembangan Basis Data (Raymond McLeod Jr. dan George Schell, 2001) adalah sebagai berikut :

1. Mengurangi kerangkapan data
2. Menghindari ketergantungan data
3. Memungkinkan integritas data dari banyak file
4. Pemanggilan data dan informasi cepat
5. Meningkatkan keamanan data

Dari beberapa pendapat keuntungan pengembangan menurut para ahli dapat dinyatakkan bahwa keuntungan dari pengembangan basis data, diantaranya:

1. Aksesibilitas: Basis data memungkinkan pengguna untuk mengakses dan menggunakan data dengan cepat dan mudah.
2. Integritas data: Basis data memastikan integritas data dengan menerapkan aturan dan kendali pada proses input, pembaruan, dan penghapusan data.
3. Efisiensi: Basis data mengoptimalkan penyimpanan dan pengolahan data untuk memastikan efisiensi dan kecepatan akses.
4. Scalability: Basis data dapat dikembangkan dan diperluas untuk mengakomodasi pertumbuhan dan perubahan dalam jumlah dan jenis data.
5. Keamanan: Basis data menyediakan fitur keamanan untuk melindungi data dari akses yang tidak sah atau pembaruan yang tidak sesuai.
6. Analisis data: Basis data memungkinkan pengguna untuk melakukan analisis data dan mengambil keputusan bisnis yang didasarkan pada data yang terstruktur dan terorganisir.

3.6. Aspek Manfaat dan Keuntungan Pengembangan Basis Data

Berikut ini terdapat beberapa manfaat dan keuntungan basis data yang meliputi beberapa aspek yang terdiri atas:

1. Kecepatan dan Kemudahan

   Database memiliki kemampuan untuk memilih data sehingga menjadi kelompok diurutkan dengan cepat. Inilah yang ahirnya dapat menghasilkan informasi yang dibutuhkan dengan cepat pula. seberapa cepat diolah oleh *database* juga tergantung pada desain *database*.

2. Dapat Digunakan Bersama

   Database dapat digunakan oleh siapa saja dalam sebuah perusahaan. Misalnya dalam *database* siswa perguruan tinggi diperlukan oleh beberapa bagian, seperti admin, keuangan, bagian akademik. Semua bidang ini memerlukan *database* mahasiswa, tetapi tidak perlu setiap bagian dibuat *database* itu sendiri, cukup dari *database* mahasiswa disimpan pada server pusat. Kemudian aplikasi masing-masing bagian dapat dihubungkan ke *database* siswa.

3. Kontrol data terpusat

   Kontrol data terpusat dapat di lihat pada sebuah perusahaan memiliki banyak bagian atau divisi tapi *database* yang diperlukan tetap menjadi satu saja. Sehinggap proses data dimulai dari input, proses, dan

output terjadi dalam sekali proses dan dapat dilihat oleh masing masing divisi, Ini memfasilitasi data kontrol seperti ketika Anda ingin memperbarui data dalam setiap bagian atau divisi, tapi cukup dalam satu *database* yang ada di server pusat.

4. Perangkat hemat biaya

   Dengan memiliki *database* terpusat maka dalam setiap divisi tidak memerlukan perangkat untuk menyimpan *database* karena *database* hanya diperlukan satu yang disimpan di server pusat, ini akan memotong biaya pembelian perangkat.

5. Keamanan Data

   Hampir semua sekarang memiliki aplikasi manajemen *database* fasilitas manajemen pengguna. Manajemen pengguna ini mampu menciptakan hak akses yang berbeda tergantung disesuaikan dengan kepentingan dan posisi pengguna. selain itu data yang disimpan dalam *database* diperlukan *password* untuk mengaksesnya.

6. Memfasilitasi pembuatan Aplikasi baru

   Pada titik ini *database* dirancang dengan sangat baik, sehingga perusahaan membutuhkan aplikasi baru tidak perlu membuat *database* baru juga, atau tidak perlu mengubah struktur *database* yang sudah ada. Sehingga pengembang aplikasi atau programmer Si hanya cukup untuk membuat atau antarmuka aplikasi regulasi saja.

Dengan segudang manfaat dan kegunaan yang dimiliki oleh *database* maka seharusnya semua perusahaan yang baik Para pengusaha kecil terutama perusahaan besar memiliki *database* dibangun dengan desain yang baik. Ditambah dengan penggunaan teknologi jaringan komputer, manfaat dari *database* ini akan semakin besar.

Penggunaan *database* di teknologi jaringan komputer yang sama telah banyak digunakan oleh berbagai Perusahaan, misalnya, hanya bank-bank yang memiliki cabang di setiap kota. Bank Perusahaan hanya memiliki *database* yang disimpan pada server pusat, sedangkan cabang terhubung melalui jaringan komputer untuk mengakses *database* yang terletak di pusat.

Tujuan pengembangan basis data dapat berbeda-beda tergantung pada jenis organisasi dan kebutuhan bisnisnya. Namun, beberapa contoh tujuan pengembangan basis data yang umum digunakan antara lain:

1. Meningkatkan efisiensi bisnis: Dengan pengembangan basis data yang tepat, perusahaan dapat mengelola dan mengakses data dengan lebih cepat dan efisien. Ini dapat meningkatkan produktivitas karyawan dan mempercepat proses bisnis.
2. Meningkatkan kualitas data: Basis data yang baik dapat membantu memastikan data yang diinput ke dalamnya

akurat dan konsisten. Dengan demikian, perusahaan dapat memastikan bahwa data yang digunakan dalam proses bisnisnya akurat dan andal.

3. Meningkatkan keamanan data: Basis data yang dirancang dengan baik dapat memastikan keamanan data yang tersimpan di dalamnya. Ini termasuk pembaruan regulasi dan kepatuhan terhadap standar keamanan yang ketat.
4. Menyediakan akses data yang lebih mudah: Basis data yang dikembangkan dengan baik dapat memudahkan pengguna untuk mengakses data yang mereka butuhkan. Hal ini dapat membantu meningkatkan kinerja bisnis dan memungkinkan pengambilan keputusan yang lebih baik.
5. Memungkinkan analisis data yang lebih baik: Basis data yang baik dapat membantu perusahaan mengumpulkan data dalam jumlah besar dan menganalisisnya untuk memperoleh insight yang lebih baik tentang pelanggan, produk, atau pasar.

3.7. Rangkuman

Tujuan basis data dapat dikelompokkan menjadi dua yaitu tujuan primer dan tujuan sekunder, secara umum tujuan basis data banyak berhubungan dengan kinerja , kejelasan dana kemudahan pemakai serta bagaimana basis data mampu menjawab kebutuhan user. Tujuan Pengembangan Basis Data oleh James Martin (1975) dibedakan atas 2 (dua), yaitu :

1. Tujuan Primer, sebagai tujuan utama yang ingin dicapai dalam setiap usaha perancangan dan pengembangan basis data yaitu mamasikan basis data dapat memenuhi kebutuhan bisnis orgnisasi dan memastikan basis data dapat berfungsi dengan baik dan efisien diantaranya :
    - Mengelola dan menyimpan data dan informasi dengan baik dan efisien.
    - Memberikan akses yang cepat dan mudah ke data dan informasi.
    - Mempermudah analisis dan pengambilan keputusan berdasarkan data dan informasi.
    - Mendukung integritas dan keamanan data dan informasi.
    - Memastikan basis data dapat diperbarui dan dikembangkan sesuai dengan perkembangan bisnis atau organisasi.
2. Tujuan Sekunder, merupakan tujuan tambahan untuk mencapai tujuan primer diantaranya adalah untuk memastikan efisiensi, integritas, dan keamanan data. Ini bisa meliputi tugas-tugas seperti mengoptimalkan performa database, memastikan data tidak rusak atau hilang, dan melindungi data dari akses yang tidak sah. Hal ini juga dapat termasuk mengatur akses ke data dan memastikan bahwa hanya pengguna yang memiliki otoritas yang sesuai yang dapat memodifikasi atau melihat informasi.

Penyusunan basis data dapat mengatasi permasalahan-permasalahan pada waktu pengolahan data diantaranya kerangkapan data dapat diminimalisasi inkonsistensi dapat dihindari dan data dapat digunakan secara *multi user* sesuai standar yang telah ditetapkan. Begitu juga dengan integrity dan *security* pada basis data. Secara umum dalam pemanfaatan basis data akan didapat beberapa elemen yang diuntungkan diantaranya kemudahan dan kecepatan, dapat digunakan secara bersama dapat dikontrol dari pusat perangkat biaya lebih hemat dan keamanan data.

Berikut ini terdapat beberapa manfaat dan keuntungan basis data yang meliputi beberapa aspek yang terdiri atas:

1. Kecepatan dan Kemudahan

   Database memiliki kemampuan untuk memilih data sehingga menjadi kelompok diurutkan dengan cepat. Inilah yang ahirnya dapat menghasilkan informasi yang dibutuhkan dengan cepat pula. seberapa cepat diolah oleh *database* juga tergantung pada desain *database*.

2. Dapat Digunakan Bersama

   Database dapat digunakan oleh siapa saja dalam sebuah perusahaan. Misalnya dalam *database* siswa perguruan tinggi diperlukan oleh beberapa bagian, seperti admin, keuangan, bagian akademik. Semua bidang ini memerlukan *database* mahasiswa, tetapi tidak perlu setiap bagian dibuat *database* itu sendiri, cukup dari *database* mahasiswa disimpan pada server pusat.

Kemudian aplikasi masing-masing bagian dapat dihubungkan ke *database* siswa.

3. Kontrol data terpusat

   Kontrol data terpusat dapat di lihat pada sebuah perusahaan memiliki banyak bagian atau divisi tapi *database* yang diperlukan tetap menjadi satu saja. Sehinggap proses data dimulai dari input, proses, dan output terjadi dalam sekali proses dan dapat dilihat oleh masing masing divisi, Ini memfasilitasi data kontrol seperti ketika Anda ingin memperbarui data dalam setiap bagian atau divisi, tapi cukup dalam satu *database* yang ada di server pusat.

4. Perangkat hemat biaya

   Dengan memiliki *database* terpusat maka dalam setiap divisi tidak memerlukan perangkat untuk menyimpan *database* karena *database* hanya diperlukan satu yang disimpan di server pusat, ini akan memotong biaya pembelian perangkat.

5. Keamanan Data

   Hampir semua sekarang memiliki aplikasi manajemen *database* fasilitas manajemen pengguna. Manajemen pengguna ini mampu menciptakan hak akses yang berbeda tergantung disesuaikan dengan kepentingan dan posisi pengguna. selain itu data yang disimpan dalam *database* diperlukan *password* untuk mengaksesnya.

6. Memfasilitasi pembuatan Aplikasi baru

   Pada titik ini *database* dirancang dengan sangat baik, sehingga perusahaan membutuhkan aplikasi baru tidak perlu membuat *database* baru juga, atau tidak perlu mengubah struktur *database* yang sudah ada. Sehingga pengembang aplikasi atau programmer Si hanya cukup untuk membuat atau antarmuka aplikasi regulasi saja.

3.8. Bahan Diskusi

Berikut ini adalah beberapa studi kasus yang bisa dibahas untuk menjelaskan tujuan pengembangan basis data:

1. Sistem Informasi Akademik: Studi kasus ini membahas bagaimana membuat sistem informasi akademik yang menggunakan basis data untuk mengelola data mahasiswa, data mata kuliah, dan laporan akademik. Tujuan pengembangan basis data adalah untuk memastikan bahwa data akademik dapat tersimpan dengan benar dan mudah dikelola.
2. Sistem Informasi Pemesanan Hotel: Studi kasus ini membahas bagaimana membuat sistem informasi pemesanan hotel yang menggunakan basis data untuk mengelola data pemesanan, data kamar, dan laporan pemesanan. Tujuan pengembangan basis data adalah untuk memastikan bahwa data pemesanan dapat tersimpan dengan benar dan mudah dikelola.

3. Sistem Informasi Pembelian Barang: Studi kasus ini membahas bagaimana membuat sistem informasi pembelian barang yang menggunakan basis data untuk mengelola data pembelian, data barang, dan laporan pembelian. Tujuan pengembangan basis data adalah untuk memastikan bahwa data pembelian dapat tersimpan dengan benar dan mudah dikelola.
4. Sistem Informasi Manajemen Proyek: Studi kasus ini membahas bagaimana membuat sistem informasi manajemen proyek yang menggunakan basis data untuk mengelola data proyek, data tugas, dan laporan proyek. Tujuan pengembangan basis data adalah untuk memastikan bahwa data proyek dapat tersimpan dengan benar dan mudah dikelola.
5. Sistem Informasi Manajemen Aset: Studi kasus ini membahas bagaimana membuat sistem informasi manajemen aset yang menggunakan basis data untuk mengelola data aset, data pemeliharaan, dan laporan aset. Tujuan pengembangan basis data adalah untuk memastikan bahwa data aset dapat tersimpan dengan benar dan mudah dikelola.

Studi kasus ini menunjukkan bahwa tujuan pengembangan basis data adalah untuk memastikan bahwa data dapat tersimpan dengan benar, mudah dikelola, dan memenuhi kebutuhan bisnis atau organisasi. Berikan gambaran diagram tujuan primer dan tujuan sekunder dari pengembangan basis data diatas.

BAB IV
ARSITEKTUR BASIS DATA DAN PERMODELAN DATA

4.1. Pengantar

Arsitektur basis data adalah desain yang memastikan bagaimana data disimpan, dikelola, dan diakses dalam suatu sistem basis data. Ini termasuk pemilihan teknologi yang tepat, pembuatan model data yang efisien, dan implementasi sistem basis data yang tepat.

Arsitektur basis data terdiri dari beberapa komponen utama, termasuk:

1. Model data: Menentukan bagaimana data disimpan dan dipresentasikan, seperti model relasional, model objek, atau model dokumen.
2. Teknologi penyimpanan data: Memilih teknologi yang tepat untuk menyimpan data, seperti basis data relasional, NoSQL, atau basis data dokumen.
3. Sistem manajemen basis data (DBMS): Menentukan bagaimana basis data dikelola dan diakses, seperti MySQL, Oracle, atau MongoDB.
4. Aplikasi: Menentukan bagaimana data diakses dan digunakan oleh aplikasi, seperti aplikasi web, aplikasi mobile, atau aplikasi desktop.

5. Jaringan: Mengatur bagaimana data diteruskan dan diakses dari lokasi yang berbeda, seperti jaringan lokal, jaringan intranet, atau jaringan internet.

Arsitektur basis data harus memastikan bahwa sistem basis data dapat mengatasi beban kerja yang tinggi, memastikan integritas dan keamanan data, dan memastikan bahwa data dapat diakses dengan cepat dan efisien. Ini juga harus memastikan bahwa sistem basis data dapat dikembangkan dan ditingkatkan seiring waktu untuk memenuhi kebutuhan bisnis yang berubah.

Permodelan data adalah proses pembuatan model yang mewakili data dan hubungan antar data dalam suatu sistem basis data. Ini membantu untuk memahami dan memetakan struktur data dan memastikan bahwa data dapat disimpan, dikelola, dan diakses dengan efisien.

Ada beberapa jenis model data yang digunakan dalam permodelan data, termasuk:

1. Model relasional: Menggambarkan data sebagai tabel-tabel yang saling terkait dengan menggunakan kunci utama dan kunci asing.
2. Model objek: Menggambarkan data sebagai objek-objek yang saling terkait dengan menggunakan hubungan antar objek.
3. Model dokumen: Menggambarkan data sebagai dokumen-dokumen yang saling terkait dengan menggunakan hubungan antar dokumen.

4. Model grafik: Menggambarkan data sebagai node dan edge dalam grafik.

Permodelan data adalah bagian penting dari proses perancangan basis data, memastikan bahwa data dapat disimpan dan dikelola dengan efisien dan memastikan bahwa data dapat diakses dengan cepat dan akurat. Ini juga membantu untuk memastikan integritas dan keamanan data dan memastikan bahwa sistem basis data dapat dikembangkan dan ditingkatkan seiring waktu untuk memenuhi kebutuhan bisnis yang berubah.

4.2. Arsitektur Basis Data

Sebuah basis data dapat dipandang dari 2 (dua) sisi, yaitu sisi pengguna yangn merupakan orang atau program aplikasi yang mengakses sistem basis data baik sendiri maupun secara bersamaan dan Sisi perancang, merupakan seorang perancang dapat memiliki 2 (dua) pandangan yang berbeda, yaitu : Secara Konseptual dan Secara Fisik.

Arsitektur sistem basis data adalah desain konseptual dan fisik dari sistem basis data, yang mencakup organisasi informasi, pemrosesan data, dan infrastruktur teknologi yang digunakan. Arsitektur sistem basis data menentukan bagaimana data dikumpulkan, disimpan, diproses, dan digunakan dalam sistem informasi. Ini termasuk memilih teknologi, merancang skema database, dan memilih bagaimana data akan diakses dan diproses. Arsitektur sistem basis data memainkan peran penting dalam memastikan

kinerja yang efisien, integritas data yang baik, dan fleksibilitas untuk menghadapi perubahan dan pertumbuhan jumlah dan jenis data.

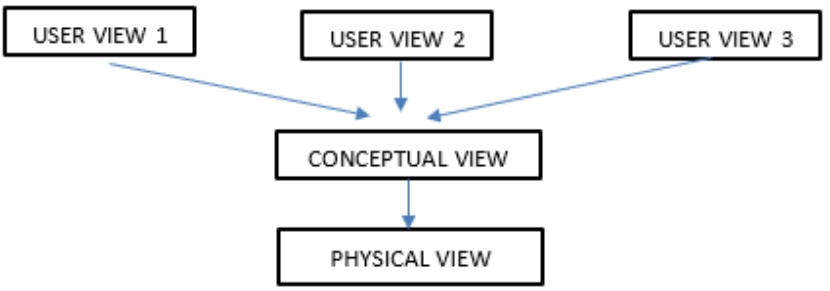
Desain dan pengorganisasian sistem basis data yang menentukan bagaimana data dikumpulkan, disimpan, diproses, dan diterima. Ini mencakup aspek teknis seperti model data, metode penyimpanan dan sistem manajemen basis data, serta aspek organisasi seperti proses bisnis, praktik keamanan informasi, dan prosedur pemeliharaan. Arsitektur sistem basis data berperan penting dalam memastikan efisiensi, skalabilitas, dan keamanan sistem basis data, serta membantu memastikan bahwa data dapat digunakan secara efektif dan efisien dalam berbagai situasi dan kebutuhan bisnis.

Konsep dasar dari arsitektur sistem basis data meliputi:

1. Model data: Menentukan bagaimana data harus disimpan dan diorganisir dalam basis data. Beberapa model data populer meliputi model entitas-relasi, model dokumen, dan model objek-relasi.
2. Sistem manajemen basis data (DBMS): Merupakan perangkat lunak yang mengatur dan mengelola akses dan manipulasi data dalam basis data.
3. Metodologi penyimpanan: Menentukan bagaimana data harus disimpan dan diorganisir dalam basis data, termasuk pemilihan struktur tabel, indeks, dan relasi antar tabel.

4. Prosedur pemeliharaan: Menentukan bagaimana basis data harus dikelola dan dipelihara untuk memastikan konsistensi, akurasi, dan integritas data.
5. Kebijakan keamanan: Menentukan bagaimana data harus dilindungi dari akses yang tidak sah dan pembaruan yang tidak sesuai.
6. Proses bisnis: Menentukan bagaimana data harus digunakan dalam proses bisnis dan bagaimana data harus disinkronkan dengan sistem lain dalam organisasi.

Arsitektur sistem basis data memastikan bahwa sistem basis data bekerja secara efektif dan efisien, memenuhi kebutuhan bisnis, dan memastikan integritas dan keamanan data. Dasar utama dari arsitektur sistem basis data disesuaikan dengan persepsi antara pengguna, perancang dan bagian implementasi. Pandangan sistem basis data sering disebut sebagai arsitektur sistem basis data dapat digambarkan sebagai berikut :



Gambar 5.1 Pandangan Sistem Basis Data

Dari gambar diatas dapat di lihat bahwa level yang paling atas merupakan level penggun yang mempunyai pendangan

bagaiman implementasi basis data yang sesungguhnya sesuai dengan kebutuhan dan proses bisnis yang ada dalam perusahaan atau organisasi, dari pandangan implementasi yang nyata maka dibuat konsep basis data yang mampu menjawab kebutuhan pengguna tersebut dengan berbagai metode pengembangan basis data yang disebut dengan level konseptual, dari konsep yang telah di buat atau dirancang, tinggal bagaimana diterapakan kedalam bentuk fisik sehingga menjadi sebuah objek sistem yang dapat berfungsi sebagai pengelolaan *database* yang ada pada perusahaan tersebut inilan yang menjadi level fisik,

1. Pandangan Level Pengguna (User View)

   Merupakan pandangan pada pengguna basis data dimana masing-masing pengguna basis data dapat memiliki cara pandangan yang berbeda tergantung pada macam data apa saja yang tersedia atau dapat diakses oleh para pengguna (Sutanta, 2004)

2. Pandangan Level Konseptual (*Conceptual View*)

   Merupakan pandangan perancang basis data yang berkaitan dengan data-data apa saja yang perlu disimpan dalam basis data dan penjelasan mengenai bagaimana hubungan antara data yang satu dengan yang lainnya (Martin, 1975).

   Perancang basis data perlu mengetahui macam data apa saja yang diperlukan seluruh pengguna atau

program aplikasi yang ada pada seluruh subsistem pada seluruh level manajemen ada.

3. Pandangan Level Fisikal (Physical View)

   Merupakan bentuk implementasi dari pandangan level pengguna dan pandangan level konseptual. Pandangan ini berkaitan dengan permasalahan teknik penyimpanan data-data basis data ke dalam fisik media penyimpanan data yang digunakan.

   Misalnya jika kita melihat sebuai sistem informasi akademik yang berada pada sebuah perguruan tinggi.maka kitab isa mengelompokkan

   a. Secara level user kita bisa melihat pandangan pengguna mulai dari kebutuha mahasiswa, dosen, tenaga kependidikan dann pihak pengelola mulai dari tinggak ketua program studi, ketua jurusan, dekan fakultas sampai ke pihak rektorat . masing masin pasti membutuhkan aplikasi database yang berbeda beda, namu semua mempuanyai keterhubunan dan keberlajutan tugasnya masing masing.

   b. Secara konseptual masing masing user mempunyai kebutuhan data yang sama, namun mempunyai hak akses pengelolaan yang berbeda, sesuai dengan tugas pokok dan fungsi masing masing. Misalnya data kemahasiswaan :

- Untuk seorang mahasiswa berlaku data profile mahasiwa tersebut beserta mengikuti kegiatan akademik masing masing seperti bimbingan akademik, absensi, ujian, administarasi pembeyaran.
- Untuk seorang dosen, data kemahasiswaan berhubungan denan aktifitas akademik interaksi dengan mahasiswa, seperti pengajaran , penilaian, pelaporan.
- Untuk bagian program studi dan jurusan berhubungan dengan pengelolaan rekap kegiatan pengajaran, akademik dan administrasinya.
- Untuk Dekan fakultas berhubungan dengan strategi pelaksanaan perkuliahan.

Semua dikonsep menjadi kesatuan sehingga tidak terjadi pemborosan/redundancy baik secara proses maupun dalam hal media penyimpan, menghindari dupliksi dan inkonsistensi data.

c. Secara fisical lebih menjelaskan bagaimana perwujutan konsep sistem basis data dapat implementasi, dan bagaiman struktur databasenya serta metode penyimpanan data yang dliakukan.

Berdasarkan teori-teori atau konsep tersebut, maka kita akan bertanya Bagaimana cara kita memodelkan arsitektur basis data tersebut ?

Langkah-langkah dalam memodelkan arsitektur basis data :

1. Melakukan pengumpulan data pada objek/perusahaan yang akan dibuat basis data (lebih luasnya sistem informasi/aplikasi). Dengan menggunakan teknik pengumpulan data seperti : wawancara, observasi, kuesioner.
2. Membuat pemodelan (dalam bentuk gambar) dari hasil pengumpulan data tersebut. Pada tahap inilah dilakukan Conceptual Design dan Physical Design untuk memenuhi *Conceptual View* dan *Physical View*. Pemodelan dengan menggunakan : ERD *(Entity Relationship Diagram), CDM (Conceptual Data Model), PDM (Physical Data Model).*

Misalnya sebuah kasus pada sebuah toko retail memerlukan suatu aplikasi untuk mendukung proses penjualan barangnya. Pemilik toko tersebut menginginkan aplikasi yang mampu menyimpan data pelanggannya, data barang dan data transaksi penjualan. Dimana berdasarkan data-data yang disimpan tersebut, pemilik toko dapat mengetahui laporan mengenai pelanggan, barang, penjualan. Dan dapat melakukan transaksi penjualan dengan mudah dan cepat. Pemrosesan transaksi tersebut dilakukan oleh karyawan.

Langkah-langkah yang harud dilakukan berdasarkan kesimpulan wawancara tersebut adalah :

1. Mencari entitas apa terlibat dalam sistem tersebut

   Entitas yang terlibat dari sistem tersebut adalah pelanggan, barang, penjualan, karyawan.

2. Menentukan atribut yang akan menjelaskan entitas yang ditentukan tersebut, seperti :

   Pelanggan → kode pelanggan, nama, alamat, no. telepon

   Barang → kode barang, nama, harga

   Penjualan → kode penjualan, tanggal, total harga

   Karyawan → kode karyawan, nama, alamat, jenis kelamin

3. Membuat pemodelan ERD (*Entity Relationship Diagram*), diagram ini digunakan untuk memodelkan hubungan antar entitas tersebut.

4. Membuat CDM (*Conceptual Data Model*)

5. Membuat PDM (*Physical Data Model*)

6. Antarmuka pandangan terhadap Basis Data

7. Antarmuka (*interface*) menyediakan layanan yang lengkap untuk lapisan yang lebih tinggi, sehingga setiap lapisan bergantung pada lapisan dibawahnya.

Independensi data merupakan ketidaktergantungan data dalam basis data. Mempunyai 2 (dua) dimensi, yaitu secara fisik dan konseptual. Independensi Data memberikan jaminan berupa fleksibilitas basis data(Sutanta, 2004) :

1. Media dan metode akses dari fisik penyimpanan basis data dapat mengalami perubahan tanpa harus mengubah pandangan konseptual
2. Kebutuhan data-data oleh para pengguna basis data dapat mengalami perubahan tanpa harus mengubah pandangan konseptual.
3. Pengguna tidak perlu tahu kerumitan/kompleksitas yang terjadi berkaitan dengan perancangan dan teknis penyimpanan basis data.

4.3. Permodelan Data

Model Data merupakan suatu cara untuk menjelaskan tentang data-data yang tersimpan dalam basis data dan bagaimana hubungan antar data tersebut untuk para pengguna *(user)* secara logika. Secara garis besar model data dapat dikategorikan dalam 3 (tiga) macam, yaitu : (Martin, 1975)

1. Object based data model - ERD

   Merupakan himpunan data dan prosedur/relasi yang menjelaskan hubungan logik antar data dalam basis data berdasarkan pada objek datanya.

2. Record based data model - CDM

   Model ini mendasar pada record/rekaman untuk menjelaskan kepada pada pengguna tentang hubungan logik antar data dalam basis data.

3. Physical data model - PDM

   Model ini mendasar pada teknis penyimpanan record/rekaman pada basis data

Berikut beberapa contoh Perangkat lunak model data :

1. DB2 dari IBM
2. Microsoft Access dari Microsoft
3. Microsoft SQL dari Microsoft Corporation
4. Postgre SQL dari Postgre SQL Global Development Group
5. Visual dbase dari Borland International

Arsitektur sistem basis data mengacu pada cara sistem basis data dirancang dan diatur untuk mendukung kebutuhan bisnis dan teknologi organisasi. Arsitektur sistem basis data yang baik harus dapat memastikan data yang disimpan dan diakses dengan aman, cepat, dan efisien. Misalnya Perusahaan e-commerce yang berkembang pesat memiliki masalah dengan kinerja sistem basis data mereka. Sistem basis data mereka sering mengalami downtime, dan data pelanggan yang disimpan tidak selalu akurat. Perusahaan ingin memperbaiki kinerja sistem basis data dan memastikan data pelanggan yang tersimpan akurat dan aman.

Untuk memecahkan masalah ini, perusahaan harus merancang dan mengimplementasikan arsitektur sistem basis data yang lebih baik. Berikut adalah beberapa faktor yang harus dipertimbangkan:

1. Skalabilitas: Sistem basis data harus dirancang agar mudah ditingkatkan untuk menangani pertumbuhan data dan transaksi bisnis.
2. Keamanan: Sistem basis data harus dirancang dengan standar keamanan yang ketat untuk melindungi data pelanggan dan transaksi bisnis.
3. Performa: Sistem basis data harus dirancang agar kinerjanya cepat dan efisien, sehingga data dapat diakses dengan cepat dan transaksi bisnis dapat diselesaikan dengan efisien.
4. Redundansi dan pemulihan bencana: Sistem basis data harus dirancang agar memiliki cadangan data yang dapat digunakan saat terjadi kegagalan sistem atau bencana.
5. Pengelolaan data: Sistem basis data harus dirancang dengan struktur data yang baik dan memungkinkan manajemen data yang efektif, termasuk pemeliharaan data, pembaruan, dan penghapusan data.

Setelah merancang arsitektur sistem basis data yang lebih baik, perusahaan dapat mulai mengimplementasikannya dan memastikan bahwa data pelanggan yang disimpan aman dan akurat. Selain itu, kinerja sistem basis data yang ditingkatkan dapat membantu perusahaan meningkatkan efisiensi bisnis dan memperbaiki pengalaman pelanggan.

4.4. Rangkuman

Dalam arsitektur basis data terdapat cara pandang terhadap rancangan sistem basis data mulai dari level user level konseptual Dan lokal fisik semuanya mempunyai karakteristik yang berbeda-beda bisanya level konseptual lebih mengarah kepada Bagaimana membangun disimpaksi data secara kerangka struktur konsepnya dan dilanjutkan dengan level fisik yang bagaimana membangun sebuah sistem basis data dari perangkat-perangkat yang ada misalnya bahasa pemrograman perangkat keras data dan aspek-aspek lainnya kemudian jika kelak siap atau jadi maka muncul level user sebagai pengguna dari sistem basis data

Konsep dasar dari arsitektur sistem basis data meliputi:

1. Model data: Menentukan bagaimana data harus disimpan dan diorganisir dalam basis data. Beberapa model data populer meliputi model entitas-relasi, model dokumen, dan model objek-relasi.
2. Sistem manajemen basis data (DBMS): Merupakan perangkat lunak yang mengatur dan mengelola akses dan manipulasi data dalam basis data.
3. Metodologi penyimpanan: Menentukan bagaimana data harus disimpan dan diorganisir dalam basis data, termasuk pemilihan struktur tabel, indeks, dan relasi antar tabel.

4. Prosedur pemeliharaan: Menentukan bagaimana basis data harus dikelola dan dipelihara untuk memastikan konsistensi, akurasi, dan integritas data.
5. Kebijakan keamanan: Menentukan bagaimana data harus dilindungi dari akses yang tidak sah dan pembaruan yang tidak sesuai.
6. Proses bisnis: Menentukan bagaimana data harus digunakan dalam proses bisnis dan bagaimana data harus disinkronkan dengan sistem lain dalam organisasi.

Selain dari aspek dari arsitektur basis data dapat juga dikembangkan berbagai model data untuk membangun sistem basis data ada beberapa kategori dalam model data diantaranya objek yang berdasarkan data model atau *entity relationship diagram* kemudian ada lagi dengan menggunakan rekor basis data model atau cdm yang merupakan bentuk model data berdasarkan dari rekaman yang ada kemudian terakhir dari *physical data model* yang berdasarkan pada teknik penyimpanan data model. Secara garis besar model data dapat dikategorikan dalam 3 (tiga) macam, yaitu : (Martin, 1975)

1. Object based data model - ERD

   Merupakan himpunan data dan prosedur/relasi yang menjelaskan hubungan logik antar data dalam basis data berdasarkan pada objek datanya.

2. Record based data model - CDM

   Model ini mendasar pada record/rekaman untuk menjelaskan kepada pada pengguna tentang hubungan logik antar data dalam basis data.

3. Physical data model - PDM

   Model ini mendasar pada teknis penyimpanan record/rekaman pada basis data

Berikut beberapa contoh Perangkat lunak model data :

1. DB2 dari IBM
2. Microsoft Access dari Microsoft
3. Microsoft SQL dari Microsoft Corporation
4. Postgre SQL dari Postgre SQL Global Development Group
5. Visual dbase dari Borland International

4.5. Bahan Diskusi

Beberapa Bahan Diskusi mengenai arsitektur sistem basis data:

1. Jenis arsitektur basis data: diskusikan tentang berbagai jenis arsitektur basis data, seperti arsitektur basis data terpadu, terdistribusi, dan basis data cloud. Bandingkan dan bedakan kelebihan dan kekurangan dari masing-masing jenis arsitektur.

2. Skalabilitas dan performa: diskusikan bagaimana arsitektur basis data mempengaruhi skalabilitas dan performa sistem. Bagaimana arsitektur basis data dapat diterapkan untuk mengatasi masalah performa dan skalabilitas pada basis data besar?
3. Keamanan: diskusikan bagaimana arsitektur basis data mempengaruhi tingkat keamanan data. Bagaimana arsitektur basis data dapat memastikan bahwa data yang disimpan aman dan terlindung dari akses yang tidak sah?
4. Keterbukaan dan interoperabilitas: diskusikan bagaimana arsitektur basis data mempengaruhi keterbukaan dan interoperabilitas antar sistem. Bagaimana arsitektur basis data dapat memastikan bahwa data dapat dibagi dan diakses oleh sistem lain dengan mudah?
5. Biaya: diskusikan bagaimana biaya mempengaruhi pemilihan arsitektur basis data. Apakah lebih baik untuk memilih arsitektur basis data berbayar atau gratis? Bagaimana arsitektur basis data dapat mempengaruhi biaya pemeliharaan dan pengembangan sistem?

BAB V

MODEL DATA ENTITY REALATIONSHIP DIAGRAM

5.1. Pengantar

ERD adalah singkatan dari Entity-Relationship Diagram, yaitu diagram yang digunakan untuk menggambarkan hubungan antar entitas dalam sistem basis data. ERD adalah alat yang berguna dalam proses perancangan basis data, membantu para analis dan pengembang untuk memahami hubungan antar entitas dan membuat model logis dari sistem.

ERD menggambarkan entitas sebagai simbol oval dan hubungan antar entitas sebagai garis antara simbol. Setiap entitas memiliki atribut yang ditunjukkan sebagai kotak di sekitar simbol entitas. Hubungan antar entitas dapat berupa satu-ke-satu, satu-ke-banyak, atau banyak-ke-banyak.

ERD juga menunjukkan kardinalitas dari hubungan antar entitas, yaitu jumlah maksimal dan minimal dari entitas yang dapat terkait dengan entitas lain. Misalnya, kardinalitas satu-ke-satu menunjukkan bahwa hanya ada satu entitas dari jenis tertentu yang dapat terkait dengan entitas lain.

ERD sangat berguna dalam memahami struktur dan hubungan dalam sistem basis data dan memastikan bahwa model logis dari sistem tersebut dapat diterjemahkan ke

dalam basis data yang efisien dan dapat dikelola. Ini juga membantu dalam memastikan bahwa data dapat dikelola dan diterima secara konsisten dan akurat.

5.2. Konsep ERD

ERD (Entity Relationship Diagram) adalah sebuah model data yang menggambarkan hubungan antar entitas dalam sistem informasi. Dalam model ini, entitas dipresentasikan sebagai objek yang memiliki atribut dan hubungan. Beberapa konsep dasar dari ERD antara lain:

1. Entitas: Setiap objek dalam sistem informasi dipresentasikan sebagai entitas, yang memiliki atribut dan membentuk hubungan dengan entitas lain.
2. Atribut: Setiap entitas memiliki beberapa atribut, yang mewakili karakteristik dari entitas tersebut.
3. Hubungan: Hubungan dalam ERD menggambarkan bagaimana entitas berinteraksi satu sama lain. Terdapat berbagai tipe hubungan, seperti hubungan satu-ke-satu, satu-ke-banyak, dan banyak-ke-banyak.
4. Kunci utama: Setiap entitas memiliki satu atau lebih kunci utama, yang digunakan untuk membedakan antar entitas.
5. Entitas terkait: Entitas terkait adalah entitas yang memiliki hubungan dengan entitas lain dalam sistem informasi.

ERD membantu dalam menentukan bagaimana data harus disimpan dan diorganisir dalam basis data, mempermudah dalam proses analisis dan perancangan sistem informasi, dan memastikan integritas dan akurasi data.

5.3. Komponen ER_Diagram

Dalam sebuah kumpulan data, terdapat susunan filed yang membentuk *database* dalam suatu program komputer saat informasi tertentu sedang dibutuhkan. Agar dapat menjadi sistem *database* yang rapi dan terstruktur kita membutuhkan Entity Relationship Diagram (ERD), yaitu sebuah model untuk menyusun *database* agar dapat menggambarkan data yang mempunyai relasi dengan *database* yang akan didesain.

Entity Relationship Model/ER_M merupakan suatu model data yang dikembangkan berdasarkan objek. ER_M digunakan untuk menjelaskan hubungan antar data dalam basis data kepada pengguna secara logik. ER_M digambarkan dalam bentuk diagram yang dikenal dengan nama ER_D (Entity Relationship Diagram). Bagi Perancang ER_D bergunan memodelkan sistem yang nantinya basis datanya akan dikembangkan. Bagi Pengguna Model ini sangat membantu dalam hal pemahaman model sistem dan rancangan basis data yang akan dikembangkan oleh perancang/analis sistem (Sutanta, 2004).

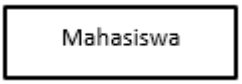
ER_D tersusun atas 3 (tiga) komponen yaitu :

1. Entitas

   Menunjukkan obyek-obyek dasar yang terkait di dalam sistem. Obyek dasar dapat berupa orang, benda, atau hal yang keterangannya perlu disimpan di dalam basis data. Untuk keterangan sebuah entitas dapat dijabarkan sebagai berikut :

   a. Entitas dinyatakan dengan simbol persegi panjang
   b. Nama entitas dituliskan dalam simbol persegi panjang
   c. Nama entitas berupa kata benda, tunggal
   d. Nama entitas sedapat mungkin menggunakan nama yang mudah dipahami dan dapat menyatakan maknanya jelas

   Contoh gambar entitas :



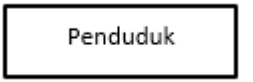
Gambar 6.1 Contoh Entitas

Jika nama entitas tersusun lebih dari satu kata, maka digunakan tanda _ (garis bawah/under score) yang dimaksudkan untuk menyatakan bahwa beberapa kata tersebut dianggap sebagai kata tunggal. Penentuan entitas dalam suatu sistem perlu dilakukan dengan cermat dan hati-hati. Tidak semua orang, benda, atau hal dapat disebut entitas. Hanya orang, benda, dan hal yng

terkait dengan dan keterangannya perlu disimpan dalam basis data yang dapat disebut entitas.

Misalnya untuk data kependudukan , dapat dinyatakan sebuah entitas penduduk yang nantinya data kependudukan itu dapat berkolaborasin dengan data aktifitas penduduk lain seperti pengurusan SIM, pengurusan Pasport di Imigrasi, atau pengurusan yang berhubungan dengan dunia perbangkan sebagai nasabah. Sehingga dinyatakan entitatas Penduduk.yang dapat nyatakan sebagai berikut :

Entitas Penduduk :



2. Atribut

Atribut sering pula disebut sebagai properti *(property)*, merupakan keterangan-keterangan yang terkait pada sebuah entitas yang perlu disimpan dalam basis data. Atribut berfungsi sebagai penjelas pada sebuah entitas. Gambaran untuk sebuah atribut dapat dinyatakan dalam keterangan berikut :

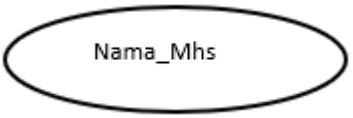
a. Atribut dinyatakan dengan simbol elips
b. Nama atribut dituliskan di dalam simbol elips
c. Nama atribut berupa kata benda, tunggal

d. Nama atribut sedapat mungkin menggunakan nama yang mudah dipahami dan dapat menyatakan maknya dengan jelas

e. Atribut dihubungkan dengan entitas yang bersesuaian dengan menggunakan sebuah garis (seyoggianya menggunakan garis lurus, namun dalam kondisi yang tidak memungkinkan dapat juga tidak menggunakan garis lurus).

Nama-nama yang digunakan sebagai atribut juga harus jelas, menunjukkan maknanya. Jika perlu, penggunaan tanda _ (garis bawah/*underscore*) atau penggunaan singkatan juga bisa digunakan, sepanjang lebih mudah dipahami.

Contoh atribut :



Gambar 6.2 Contoh atribut

Gambar diatas merupakan atribut dari entitas mahasiswa, karena nama mahasiwa adalah keterangan dari entitas mahasiswa, Adapun atribut lainya untuk entitas mahasiswa adalah no_mhas, program_studi, Angkatan,

Dalam penulisan atribut untuk dua suku kata atau lebih di hubungkan dengan garis bawah atau underscore yang bertujuan agar nanti dalam pembacaan variable

disaat permrograman dapat dibaca sebagai satu variable.

Misalnya dari entitas penduduk dalam E-KTP dapat gambaran bahwa setiap entitas pada e-KTP (Kartu Tanda Penduduk Elektronik) memiliki atribut yang berisi informasi spesifik tentang entitas tersebut. Berikut adalah atribut-atribut dari beberapa entitas pada e-KTP:

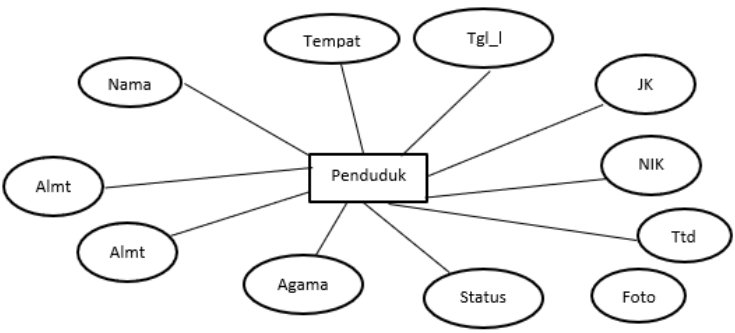
a. Nama lengkap: Atribut-atribut dari entitas ini meliputi nama depan, nama tengah, nama belakang, dan gelar.
b. Tempat, tanggal lahir: Atribut-atribut dari entitas ini meliputi tempat lahir, tanggal lahir, bulan lahir, dan tahun lahir.
c. Jenis kelamin: Atribut dari entitas ini hanya berisi informasi tentang jenis kelamin pemegang e-KTP.
d. Alamat: Atribut-atribut dari entitas ini meliputi alamat rumah, RT/RW, kelurahan/desa, kecamatan, kabupaten/kota, provinsi, dan kode pos.
e. Nomor KTP: Atribut dari entitas ini hanya berisi nomor KTP pemegang e-KTP.
f. Nomor NIK: Atribut dari entitas ini hanya berisi nomor NIK pemegang e-KTP.
g. Agama: Atribut dari entitas ini hanya berisi informasi tentang agama pemegang e-KTP.

h. Status perkawinan: Atribut dari entitas ini hanya berisi informasi tentang status perkawinan pemegang e-KTP, seperti lajang, menikah, cerai, atau janda/duda.

i. Foto: Atribut dari entitas ini hanya berisi foto pemegang e-KTP.

j. Tanda tangan: Atribut dari entitas ini hanya berisi tanda tangan pemegang e-KTP.

Atribut-atribut tersebut digunakan untuk mengidentifikasi dan memverifikasi identitas pemegang e-KTP dalam berbagai proses administrasi dan transaksi di Indonesia. Dengan atribut-atribut tersebut, e-KTP dapat menjadi dokumen identitas yang akurat dan valid. Dari atribut diatas dapat dinyatakan dalam diagram ERD sebagai berikut:



3. Kerelasian antar Entitas

Kerelasian antar entitas mendefinisikan hubungan antara dua buah entitas. Kerelasian adalah kejadian atau transaksi yang terjadi diantara dua buah entitas yang

keterangannya perlu disimpan dalam basis data. Kejadian atau transaksi yang tidak perlu disimpan dalam basis data (sekalipun benar-benar terjadi) bukan termasuk kerelasian. Aturan penggambaran kerelasian antar entitas adalah sebagai berikut :

a. Kerelasian dinyatakan dengan simbol belah ketupat
b. Nama kerelasian dituliskan di dalam simbol belah ketupat
c. Kerelasian menghubungkan dua entitas
d. Nama kerelasian berupa kata kerja aktif (diawali dengan awalan me-), tunggal
e. Nama kerelasian sedapat mungkin menggunakan nama yang mudah dipahami dan dapat menyatakan maknanya dengan jelas

Bentuk kerelasian yang menghubung antar entitas dapat dijadikan sebagai objek transaksi antar entitas yang dihubungkan oleh kunci penghubug antar entitas. Kerelasian antar entitas dikelompokkan kedalam 3 (tiga)jenis, yaitu :

a. Kerelasian jenis 1 ke 1/satu ke satu *(one to one)*
b. Kerelasian jenis n ke 1/banyak ke satu *(many to one)*
c. Kerelasian jenis n ke n/banyak ke banyak *(many to many)*

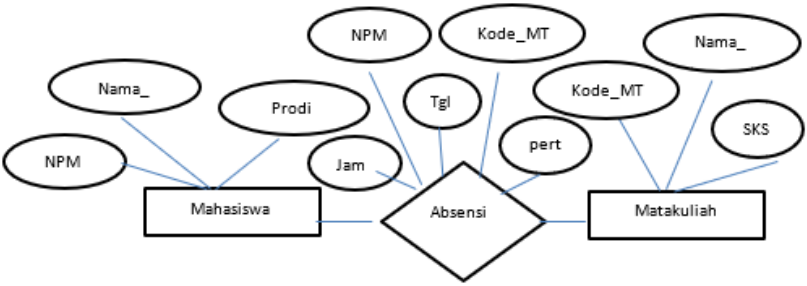
Contoh kerelasian :

Gambar 6.3 Contoh kerelasian

Gambar diatas menampilkan simbol relasi antara entitas mahasiswa dengan matakuliah, di mana aktifitas perkuliahan di buktikan dengan absensi, jadi kerelasian menjadi simbol aktifitas interaksi antara dua entitas atau lebih.

5.4. Contoh ER_Diagram

Dengan menggunakan simbol simbol ER_Diagram yang telah dijelaskan diatas maka kita dapat menbangun sebuah relasi antar entitas yang dilengkapi dengan atribut atribut dan jenis kerelasianya seperti yang ada pada contoh ER_Diagram yang menggambarkan hubungan antar entitas yang terjadi pada kegiatan akademik dengan kerelasian pada aktifitas perkuliahan seperti gambar dibawah ini:



Gambar 6.4 ER-Diagram

Gambar ER_Diagram diatas menampilkan hubungan antara entitas mahasiswa dengan entitas matakuliah dengan menggunakan kerelasian absensi, entitas mahasiswa di lengkapi dengan atribut Nama_mahasiswa, No_mhs, prodi , sedangankan entitas matakuliah dilangkapi dengan atribut Kode matakuliah, nama matakulia, SKS, dalam kerelasian kunci penguhubungnya adalah No_mhs, dan Kode Mtk, kemudian ditambahkan dengan tanggal, waktu, dan pertemuan.

Ada beberapa hal yang perlu diperhatikan dalam pengembangan basis data dengan model ER-D, diantaranya :

1. Kita harus mengetahui model proses bisnisnya
2. Kita harus mengetahui urutan proses bisnisnya
3. Kita harus mengetui entitas, atribut, dan relasi yang terlibat dalam proses bisninya masing masing.
4. Menghubungkan setiap entitas kedalam satu bagian proses bisnis

5. Tambahkan atribut untuk masing masing entitas dan relasi

Berikut dimisalkan sebuah kasus kegiatan yang terdi pada kegiatan yang berada pada sebuah benkel dengan pelayanan proses bisnis service kendaraan bermotor.

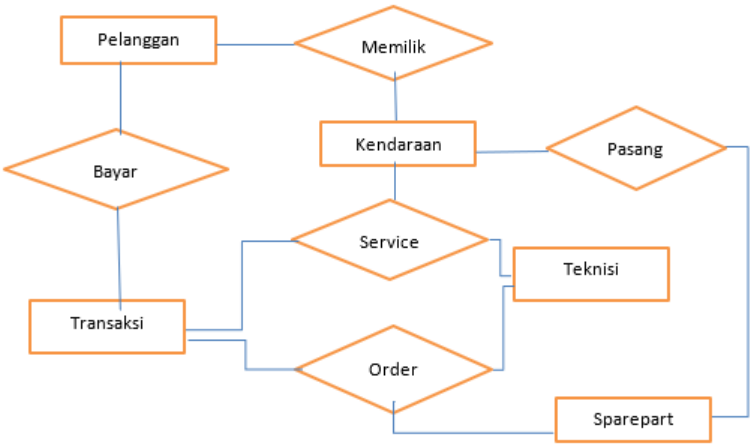
1. Model proses bisnisnya meliput :
   a. Pelayana service kendaraan
   b. Pelayanan pemasangan dan pembelian sparepart
2. Urutan proses bisnisnya meliputi :
   a. Pelanggan memiliki kendaraan
   b. Kendaraan di service teknisi
   c. Teknisi memesan sparepart dan memasang kekendaraan.
   d. Pelanggan membayar biaya service dan biaya sparepart.
3. Penentuan Entitas,atribut dan relasi
   a. Pelanggan memiliki kendaraan

   b. Kendaraan disservice teknisi

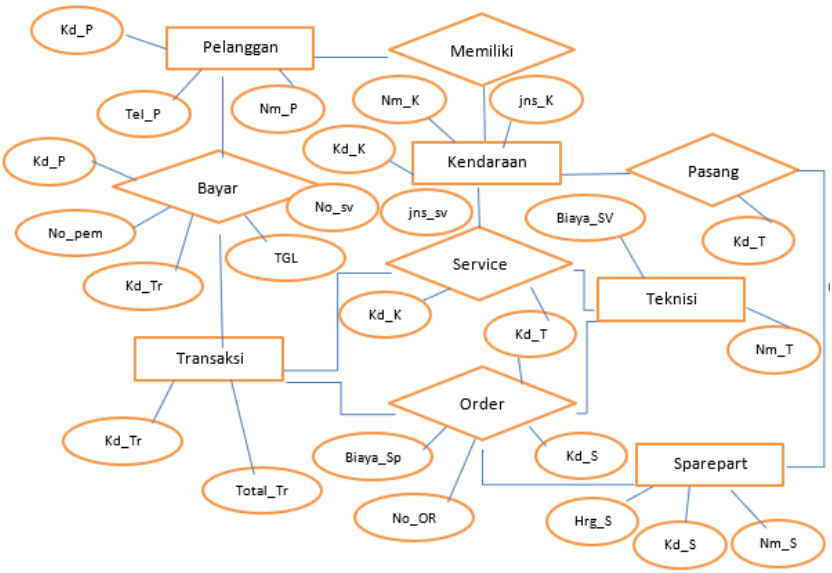
c. Teknisi pesan sparepart

d. Pelanggan memiliki kendaraan

4. Menghubungkan setiap entitas kedalam satu bagian proses bisnis



5. Tambahkan atribut untuk masing masing entitas dan relasi.



5.5. Kelebihan dan Kelemahan ER_Diagram

Kelebihan ER_D

Jika dapat diterapkan dengan benar dan tepat maka penggunaan ERD dalam pemodelan data akam memberikan keuntungan, bagi perancang maupun pemakai yaitu:

1. Dengan memakai ERD ini, pengguna lebih mudah memahami sistem basis data yang sudah di rancang oleh perancang.
2. Visualisasi yang baik: ERD mempermudah visualisasi hubungan antar entitas dalam sistem informasi, sehingga memudahkan dalam proses analisis dan perancangan.

3. Meningkatkan kualitas perancangan: ERD membantu dalam menentukan bagaimana data harus disimpan dan diorganisir dalam basis data, sehingga memastikan integritas dan akurasi data.
4. Komunikasi yang lebih baik: ERD mempermudah komunikasi antar tim dalam proses perancangan sistem informasi, karena memudahkan untuk memahami dan membagikan informasi tentang hubungan antar entitas.
5. Dapat digunakan sebagai dokumentasi: ERD dapat digunakan sebagai dokumentasi proses perancangan sistem informasi, sehingga mempermudah dalam pemeliharaan dan pengembangan lebih lanjut.

Kelemahan ERD adalah:

1. Keterbatasan dalam menggambarkan proses bisnis: ERD lebih menitikberatkan pada hubungan antar entitas, sehingga keterbatasan dalam menggambarkan proses bisnis dan logika sistem.
2. Membutuhkan skill khusus: Membuat ERD membutuhkan skill dan pemahaman khusus tentang model data dan basis data, sehingga tidak semua orang dapat melakukannya dengan baik.
3. Kemungkinan kurang akurat: ERD mungkin kurang akurat jika tidak dipelihara dengan baik, sehingga memerlukan revisi dan perbaruan secara berkala.

Meskipun memiliki kelemahan, ERD masih menjadi model data yang populer dan banyak digunakan dalam proses perancangan sistem informasi. Dengan mengetahui kelebihan dan kelemahannya, ERD dapat digunakan dengan lebih efektif dan efisien.

5.6. Rangkuman

Entity relationship diagram sangat bermanfaat sekali dalam menggambarkan sebuah sistem basis data dimana kita bisa dengan mudah mempelajari elemen-elemen apa saja yang terjadi dalam proses abstraksi data dengan memanfaatkan ERD maka bentuk alur sistem yang sudah ada dengan mudah di gambarkan di dalam ERD ada 3 bagian yaitu itu ada entitas, atribut dan relasi. Entitas merupakan objek yang terlibat pada sebuah proses dan diterangkan dalam bentuk objek entity kemudian keterangan dari entitas berupa atribut atribut yang mendukung, hubungan antara satu entitas disebut dengan relasi ada banyak bentuk hubungan diantaranya diantaranya satu ke banyak, satu ke satu, dan banyak ke banyak.

Dalam model ERD, entitas dipresentasikan sebagai objek yang memiliki atribut dan hubungan. Beberapa konsep dasar dari ERD antara lain:

1. Entitas: Setiap objek dalam sistem informasi dipresentasikan sebagai entitas, yang memiliki atribut dan membentuk hubungan dengan entitas lain.
2. Atribut: Setiap entitas memiliki beberapa atribut, yang mewakili karakteristik dari entitas tersebut.

3. Hubungan: Hubungan dalam ERD menggambarkan bagaimana entitas berinteraksi satu sama lain. Terdapat berbagai tipe hubungan, seperti hubungan satu-ke-satu, satu-ke-banyak, dan banyak-ke-banyak.
4. Kunci utama: Setiap entitas memiliki satu atau lebih kunci utama, yang digunakan untuk membedakan antar entitas.
5. Entitas terkait: Entitas terkait adalah entitas yang memiliki hubungan dengan entitas lain dalam sistem informasi.

Jika dapat diterapkan dengan benar dan tepat maka pe nggunaan ERD dalam pemodelan data akam memberikan keu ntungan, bagi perancang maupun pemakai yaitu:

1. Dengan memakai ERD ini, pengguna lebih mudah mema hami sistem basis data yang sudah di rancang oleh perancang.
2. Visualisasi yang baik: ERD mempermudah visualisasi hubungan antar entitas dalam sistem informasi, sehingga memudahkan dalam proses analisis dan perancangan.
3. Meningkatkan kualitas perancangan: ERD membantu dalam menentukan bagaimana data harus disimpan dan diorganisir dalam basis data, sehingga memastikan integritas dan akurasi data.

4. Komunikasi yang lebih baik: ERD mempermudah komunikasi antar tim dalam proses perancangan sistem informasi, karena memudahkan untuk memahami dan membagikan informasi tentang hubungan antar entitas.
5. Dapat digunakan sebagai dokumentasi: ERD dapat digunakan sebagai dokumentasi proses perancangan sistem informasi, sehingga mempermudah dalam pemeliharaan dan pengembangan lebih lanjut.

Kelemahan ERD adalah:

1. Keterbatasan dalam menggambarkan proses bisnis: ERD lebih menitikberatkan pada hubungan antar entitas, sehingga keterbatasan dalam menggambarkan proses bisnis dan logika sistem.
2. Membutuhkan skill khusus: Membuat ERD membutuhkan skill dan pemahaman khusus tentang model data dan basis data, sehingga tidak semua orang dapat melakukannya dengan baik.
3. Kemungkinan kurang akurat: ERD mungkin kurang akurat jika tidak dipelihara dengan baik, sehingga memerlukan revisi dan perbaruan secara berkala.

Meskipun memiliki kelemahan, ERD masih menjadi model data yang populer dan banyak digunakan dalam proses perancangan sistem informasi. Dengan mengetahui kelebihan dan kelemahannya, ERD dapat digunakan dengan lebih efektif dan efisien.

5.7. Bahan Diskusi

Berikut beberapa Bahan Diskusi mengenai Model Data Entity Relationship (ERD), Jika ada dua kasus proses bisnis dibawah ini :

1. Sistem Informasi Pembelian Barang: Studi kasus ini membahas bagaimana membuat sistem informasi pembelian barang yang menggunakan basis data untuk mengelola data pembelian, data barang, dan laporan pembelian. Tujuan pengembangan basis data adalah untuk memastikan bahwa data pembelian dapat tersimpan dengan benar dan mudah dikelola.
2. Sistem Informasi Manajemen Proyek: Studi kasus ini membahas bagaimana membuat sistem informasi manajemen proyek yang menggunakan basis data untuk mengelola data proyek, data tugas, dan laporan proyek. Tujuan pengembangan basis data adalah untuk memastikan bahwa data proyek dapat tersimpan dengan benar dan mudah dikelola.

Dari kasus diatas jawab pertanyaan dibawah ini untuk memahami konsep dan pengembangan basis data menggunakan model ERD :

1. Konsep Dasar ERD: diskusikan konsep dasar dari ERD, seperti entitas, hubungan, dan atribut. Jelaskan bagaimana ERD digunakan untuk mengungkapkan hubungan antar entitas dalam suatu sistem basis data.

2. Keuntungan dan Kerugian ERD: diskusikan keuntungan dan kerugian dari menggunakan ERD sebagai model data. Bagaimana ERD membantu dalam proses desain basis data dan mempermudah komunikasi antar tim? Apa saja kerugian dari menggunakan ERD, seperti keterbatasan dalam mengungkapkan relasi antar entitas?
3. Standar ERD: diskusikan standar yang digunakan dalam pembuatan ERD, seperti Crow's Foot Notation dan Bachman Notation. Bandingkan dan bedakan kelebihan dan kekurangan dari masing-masing standar.
4. Normalisasi: diskusikan bagaimana normalisasi mempengaruhi desain ERD. Jelaskan bagaimana normalisasi membantu mengatasi masalah redundansi dan memastikan integritas data.
5. Proses pembuatan ERD: diskusikan proses pembuatan ERD, mulai dari pengumpulan informasi hingga validasi dan pengujian. Jelaskan bagaimana ERD dapat diperbaiki dan diperbarui seiring dengan perubahan dalam sistem basis data.

BAB VI
MODEL DATA SEMANTIC

6.1. Pengantar

Model data semantik adalah salah satu cara untuk menyimpan dan mengolah data sehingga lebih mudah diterjemahkan dan dipahami oleh mesin. Dalam model ini, data disimpan dalam bentuk yang menggambarkan makna dan hubungannya, bukan hanya sebagai string atau angka tunggal.

Model data semantik menggunakan ontologi untuk menentukan konsep dan relasi antar konsep. Ontologi adalah model formal dari konsep dan hubungan dalam domain tertentu. Model data semantik juga menggunakan Resource Description Framework (RDF), yang merupakan bahasa standar untuk menyimpan dan mengolah data semantik.

Dengan model data semantik, data dapat diterjemahkan dan dipahami oleh mesin seperti manusia, memungkinkan mesin untuk melakukan tugas-tugas seperti pencarian, analisis, dan pengambilan keputusan yang didasarkan pada makna data dan hubungan antar data. Model data semantik sangat berguna dalam aplikasi seperti pemrosesan bahasa alami, analisis sentiment, dan sistem rekomendasi.

6.2. Konsep Diagram Semantik

Semantic model merupakan suatu model data yang dikembangkan berdasarkan objek. *Semantic model* digunakan untuk menjelaskan hubungan antar data dalam basis data kepada pengguna secara logik. Grafik semantik adalah grafik yang menggambarkan arti atau signifikansi data yang disimpan dalam database. Tujuan diagram semantik adalah untuk memfasilitasi pemahaman tentang bagaimana data diatur dan bagaimana entitas dan atributnya berinteraksi satu sama lain.

Diagram semantik umumnya menggunakan simbol dan notasi tertentu untuk menggambarkan entitas, atribut, dan hubungan antar entitas. Ini membantu dengan analisis dan desain database, memfasilitasi komunikasi antar tim dan memastikan integritas dan akurasi data.

Diagram semantik dapat dibuat menggunakan berbagai teknik, mis. B. dengan Entity Relationship Diagram (ERD), Unified Modelling Language (UML) dan lain-lain. Pilihan teknologi yang tepat tergantung pada kompleksitas sistem dan tujuan dari proses desain.

Diagram ini biasanya menggunakan simbol atau ikon untuk mewakili konsep dan garis atau fleksi untuk menunjukkan hubungan antar konsep. Beberapa jenis diagram semantik yang populer meliputi diagram ontologi, diagram konsep, dan diagram entitas-hubungan. Tujuan dari diagram semantik adalah untuk memvisualisasikan dan

memahami konsep dan hubungan antar konsep secara lebih jelas dan terstruktur.

Semantic model hampir sama dengan *Entity Relationship Model* - ERD, perbedaannya ada pada kerelasian antar objek tidak digambarkan dengan simbol, tetapi dengan kata-kata *(semantic).* Ilustrasi model *Semantic* :

Diagram *semantic* tersusun atas 3 komponen, yaitu :

1. Entitas (*Entity*)
2. Atribut (*Attribute*)
3. Kerelasian Antar Entitas (*Relationship*)

6.3. Komponen Diagram Semantic

Berikut ini akan dijabarkan komponen diagram semantic yang secara garis besar memiliki bagian yang hampir sama dengan komponen Entity Relationship Diagram, diantaranya adalah Objek-objek dasar yang terkait di dalam sistem, dapat berupa orang, benda, atau hal yang keterangannya perlu disimpan di dalam basis data. Dalam diagram semantic, entitas dapat digambarkan dengan cara berikut (Sutanta, 2004) :

1. Entitas dinyatakan dengan simbol persegi panjang atau elips
2. Nama entitas dituliskan di dalam simbol persegi panjang
3. Nama entitas berupa kata benda, tunggal
4. Nama entitas sedapat mungkin menggunakan nama yang mudah dipahami dan dapat menyatakan maknanya dengan jelas

Penggunaan tanda _ (*underscore*) juga digunakan sebagai pemisah kata pada diagram semantic.

1. Atribut (*Attribute*)

   Atribut atau properti (*property*) merupakan keterangan-keterangan yang terkait pada sebuah entitas yang perlu disimpan sebagai basis data. Atribut dapat digambarkan dengan cara sebagai berikut (Sutanta, 2004) :

   a. Atribut dinyatakan dengan simbol elips
   b. Nama atribut dituliskan di dalam simbol elips
   c. Nama atribut berupa kata benda, tunggal
   d. Nama atribut sedapat mungkin menggunakan nama yang mudah dipahami dan dapat menyatakan maknanya dengan jelas
   e. Atribut dihubungkan dengan entitas yang bersesuaian dengan menggunakan sebuah garis

   Atribut-atribut dari suatu entitas yang sama, akan berbeda jika berada pada sistem yang berbeda. Karena setiap sistem akan dipengaruhi dari kebutuhan akan sistem dan gaya manajemen masing-masing perusahaan/instusi/organisasi.

2. Kerelasian antar Entitas (*Relationship*)

   Kerelasian antar entitas yang menyatakan kejadian atau transaksi yang terjadi diantara dua buah entitas yang keterangannya perlu disimpan dalam basis data,

dalam semantic model dapat digambarkan dengan cara sebagai berikut (Sutanta, 2004) :

a. Kerelasian dinyatakan dengan simbol garis dengan sebuah mata panah
b. Nama kerelasian dituliskan di samping garis kerelasian
c. Kerelasian menghubungkan dua entitas
d. Nama kerelasian berupa : kata kerja aktif (diawali dengan awalan me-), tunggal
e. Nama kerelasian sedapat mungkin menggunakan nama yang mudah dipahami dan dapat menyatakan maknanya dengan jelas

Kerelasian antar entitas dalam semantic model juga dikelompokkan dalam 3 (tiga) jenis, yaitu (Sutanta, 2004) :

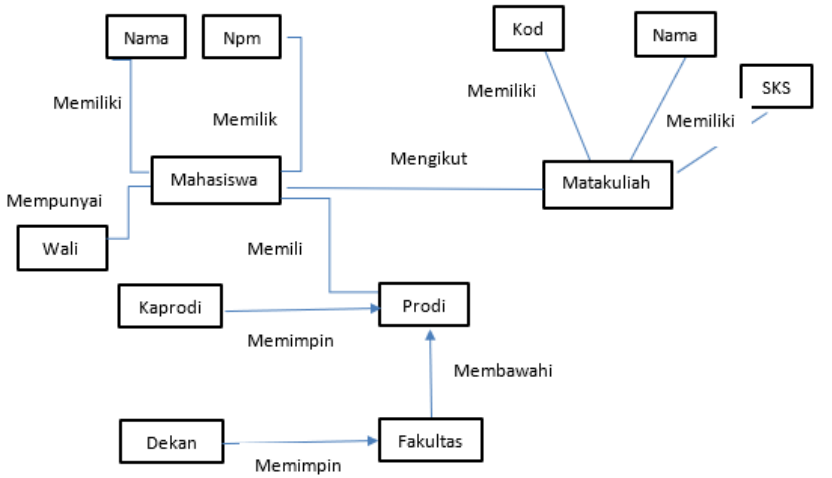
a. Kerelasian jenis 1- ke -1 (*one to one*)
b. Kerelasian jenis n- ke -1 (*many to one*)
c. Kerelasian jenis n- ke –n (*many to many*)

6.4. Menggambar Diagram Semantic

Menggambarkan diagram semantic (*semantic diagram*) secara lengkap dapat dilakukan dengan langkah sebagai berikut (Sutanta, 2004) :

1. Identifikasi setiap entitas yang terlibat, hal ini bertujuan untuk mencari komponem komponem pendukung utama yang membangun sebuah sistem basis data.
2. Identifikasi setiap atribut pada entitas, hal ini bertujuan menggambarkan keterangan keterangan yang lebih detail dari informasi sebuah entitas yang terlibat tadi
3. Identifikasi setiap kerelasian berikut jenisnya yang terjadi diantara entitas, hal ini menggambarkan bagaimana relasi yang membentuk transaksi transaksi apa saja yang terjadi.
4. Gambarkan simbol-simbol entitas, atribut, dan kerelasian antar entitas sedemikian sehingga simbol kerelasian dapat digambarkan dengan jelas/tidak saling bertabrakan
5. Cek diagram semantic yang terbentuk, dalam hal :
    a. Kelengkapan entitas
    b. Kelengkapan atribut
    c. Kelengkapan kerelasian antar entitas
    d. Jenis kerelasian antar entitas

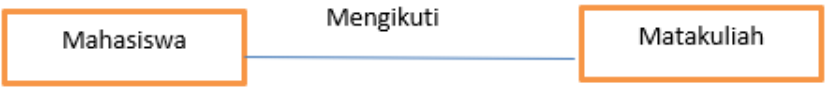
Berikut ini contoh gambaran diaram semantic dalam bidang akademik di sebuah perguruan tingggi :



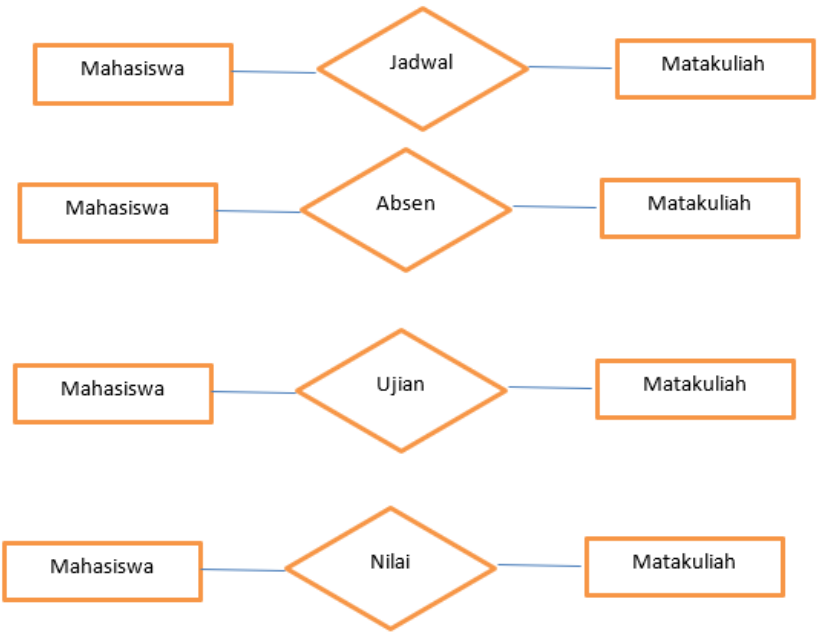
Gambar 7.1 Diagram Semantic

Dari gambar diatas untuk entitas mahasiswa mempunya semantic mengikuti matakuliah yang memiliki kode, nama dan SKS. Untuk entitas mahasiswa memiliki nama, npm dan mempunyai wali, juga ada semantic memilih prodi, sedangkan prodi di pimpim oleh kaprod berda di bawah fakultas yang dipimpin oleh Deka fakultas.

Dari model semantic dapat dijabarkan lebih detail lagi menjadi moden ERD atau madel relasi lainya. Misalnya secara semantic dosen mengambil matakuliah.



Kata mengikuti dapat dibuktikan direpresentasikan manjadi : Jadwal, Absensi, Ujian, Nilai. Sehingga menjadi



6.5. Kelebihan dan Kelemahan Diagram Semantic

Kelebihan dan kelemahan dari diagram semantic adalah sebagai berikut (Sutanta, 2004) :

1. Kelebihan :
    a. Visualisasi yang jelas: Membantu memvisualisasikan dan memahami hubungan antar konsep secara jelas dan terstruktur.
    b. Komunikasi yang efektif: Mempermudah komunikasi dan pemahaman konseptual antar tim dan stakeholder.
    c. Pemodelan yang mudah: Mempermudah proses pemodelan konseptual dan mengurangi kemungkinan salah paham.

d. Dokumentasi yang baik: Dapat digunakan sebagai dokumentasi formal untuk memperjelas konsep dan hubungan antar konsep.

2. Kelemahan :

a. Kurva belajar yang tinggi: Membutuhkan waktu dan latihan untuk memahami dan membuat diagram semantik dengan baik.

b. Keterbatasan dalam representasi kompleksitas: Mungkin sulit untuk menggambarkan hubungan konseptual yang sangat kompleks atau abstrak.

c. Kemungkinan ambiguitas: Diagram yang tidak jelas atau ambigu dapat menyebabkan salah paham.

d. Kemungkinan over-generalisasi: Mungkin terjadi over-generalisasi dalam

6.6. Rangkuman

Semantic model merupakan suatu model data yang dikembangkan berdasarkan objek. *Semantic* model digunakan untuk menjelaskan hubungan antar data dalam basis data kepada pengguna secara logik.

Diagram ini biasanya menggunakan simbol atau ikon untuk mewakili konsep dan garis atau fleksi untuk menunjukkan hubungan antar konsep. Beberapa jenis diagram semantik yang populer meliputi diagram ontologi, diagram konsep, dan diagram entitas-hubungan. Tujuan dari diagram semantik adalah untuk memvisualisasikan dan

memahami konsep dan hubungan antar konsep secara lebih jelas dan terstruktur.

Semantic model hampir sama dengan *Entity Relationship Model* - ERD, perbedaannya ada pada kerelasian antar objek tidak digambarkan dengan simbol, tetapi dengan kata-kata *(semantic)* ,Ilustrasi model *Semantic* :

Diagram *semantic* tersusun atas 3 komponen, yaitu :

1. Entitas (*Entity*)
2. Atribut (*Attribute*)
3. Kerelasian Antar Entitas *(Relationship)*

Kelebihan diagram semantic :

1. Visualisasi yang jelas: Membantu memvisualisasikan dan memahami hubungan antar konsep secara jelas dan terstruktur.
2. Komunikasi yang efektif: Mempermudah komunikasi dan pemahaman konseptual antar tim dan stakeholder.
3. Pemodelan yang mudah: Mempermudah proses pemodelan konseptual dan mengurangi kemungkinan salah paham.
4. Dokumentasi yang baik: Dapat digunakan sebagai dokumentasi formal untuk memperjelas konsep dan hubungan antar konsep.

Kelemahan diagram semantic :

1. Kurva belajar yang tinggi: Membutuhkan waktu dan latihan untuk memahami dan membuat diagram semantik dengan baik.
2. Keterbatasan dalam representasi kompleksitas: Mungkin sulit untuk menggambarkan hubungan konseptual yang sangat kompleks atau abstrak.
3. Kemungkinan ambiguitas: Diagram yang tidak jelas atau ambigu dapat menyebabkan salah paham.
4. Kemungkinan over-generalisasi: Mungkin terjadi over-generalisasi dalam representasi konseptual, menyebabkan kehilangan detail penting.

6.7. Bahan Diskusi

Berikut beberapa Bahan Diskusi mengenai Model Data Semantik, Jika ada dua kasus proses bisnis dibawah ini :

1. Sistem Informasi Pembelian Barang: Studi kasus ini membahas bagaimana membuat sistem informasi pembelian barang yang menggunakan basis data untuk mengelola data pembelian, data barang, dan laporan pembelian. Tujuan pengembangan basis data adalah untuk memastikan bahwa data pembelian dapat tersimpan dengan benar dan mudah dikelola.
2. Sistem Informasi Manajemen Proyek: Studi kasus ini membahas bagaimana membuat sistem informasi manajemen proyek yang menggunakan basis data untuk

mengelola data proyek, data tugas, dan laporan proyek. Tujuan pengembangan basis data adalah untuk memastikan bahwa data proyek dapat tersimpan dengan benar dan mudah dikelola.

Dari kasus diatas jawab pertanyaan dibawah ini untuk memahami konsep dan pengembangan basis data menggunakan model data semantik :

1. Konsep Dasar Model Data Semantik: diskusikan konsep dasar model data semantik, seperti ontologi, triplestore, dan RDF. Jelaskan bagaimana model data semantik membantu mengungkapkan hubungan antar konsep dalam suatu sistem basis data.
2. Keuntungan dan Kerugian Model Data Semantik: diskusikan keuntungan dan kerugian dari menggunakan model data semantik. Bagaimana model data semantik membantu dalam proses pemahaman konsep dan mempermudah komunikasi antar aplikasi? Apa saja kerugian dari menggunakan model data semantik, seperti keterbatasan dalam skalabilitas dan performa?
3. Linked Data: diskusikan bagaimana Linked Data mempengaruhi desain model data semantik. Jelaskan bagaimana Linked Data membantu mengatasi masalah interoperabilitas antar sistem dan mempermudah pembagian dan akses data.
4. Standar Model Data Semantik: diskusikan standar yang digunakan dalam pembuatan model data semantik,

seperti OWL dan RDFS. Bandingkan dan bedakan kelebihan dan kekurangan dari masing-masing standar.

BAB VII
MODEL DATA HIERARCHYCAL DAN MODEL DATA NETWORK

7.1. Pengantar

Model data hierarki adalah salah satu dari beberapa model data yang digunakan dalam pemodelan basis data. Model ini mewakili data dalam bentuk hierarki atau tingkatan, di mana setiap tingkatan memiliki hubungan parent-child dengan tingkatan di atasnya. Dalam model data hierarki, data disimpan dalam tingkatan yang terpisah dan setiap tingkatan memiliki entitas atau objek yang terkait.

Keuntungan dari model data hierarki adalah kemudahannya dalam memahami dan memvisualisasikan hubungan antar entitas dan memastikan integritas data dengan menghindari duplikasi data. Namun, model ini juga memiliki beberapa kekurangan, seperti keterbatasan dalam merepresentasikan hubungan antar entitas yang lebih kompleks dan sulit untuk memodifikasi atau memperbarui hubungan antar entitas. Sebagai alternatif, model data relasional, model data objek, dan model data jaringan juga dapat digunakan untuk memodelkan data dalam basis data.

Model data jaringan adalah salah satu dari beberapa model data yang digunakan dalam pemodelan basis data.

Model ini menggambarkan data sebagai jaringan atau grafik yang terdiri dari entitas atau objek dan hubungan antar entitas.

Alam model data jaringan, setiap entitas dapresentasikan sebagai node atau titik, dan hubungan antar entitas dapresentasikan sebagai garis atau busur. Setiap entitas dapat memiliki beberapa hubungan dengan entitas lain, dan hubungan antar entitas dapat berlangsung dalam berbagai arah.

Keuntungan dari model data jaringan adalah kemampuannya untuk merepresentasikan hubungan antar entitas yang lebih kompleks dan dinamis dibandingkan dengan model data hierarki. Model ini juga mudah dimodifikasi dan memperbarui hubungan antar entitas.

Namun, model data jaringan juga memiliki beberapa kekurangan, seperti kesulitan dalam memastikan integritas data dan keterbatasan dalam memvisualisasikan hubungan antar entitas.

7.2. Model Data Hierarchycal

1. Pengertian Model data hierarchical

   Model hierarki *(Hierarchycal Model)* merupakan salah satu model data yang didasarkan pada *record* (*Record Based Data Model*/*RDBM*). Model ini digunakan untuk menjelaskan kepada pengguna tentang hubungan logik

antar data dalam basis data dalam bentuk bertingkat (Silberschatz, 2001).

Hierarchycal model sering juga disebut sebagai struktur pohon *(Tree Structure)* (Martin, 1975) karena bentuknya dapat dianalogikan sebagai pohon terbalik, atau sebuah model data yang menggambarkan relasi antar entitas sebagai hierarki, dimana setiap entitas memiliki satu atau lebih entitas anak yang merupakan entitas turunan dari entitas yang lebih tinggi. Dalam model data ini, data disimpan dan diorganisir dalam bentuk pohon, di mana setiap node mewakili sebuah entitas dan garis yang menghubungkan node-node menggambarkan hubungan antar entitas yaitu terdiri atas bagian akar, batang, dahan, ranting, dan daun.

Keterangan :

Node → A sampai S

Root → A

Leaves → E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S

Parent → B terhadap E, F, G | C terhadap H, I | D terhadap J, K, L

Child → E, F, G dari B | H, I dari C | J, K, L dari D

Bentuk pohon dapat memiliki 3 (tiga) kemungkinan, yaitu (Sutanta, 2004) :

a. Pohon tidak seimbang (*Unbalanced Tree*), yaitu jika node-node dalam pohon mempunyai jumlah cabang yang berbeda-beda

b. Pohon setimbang (*Balanced Tree*), yaitu jika setiap node seluruh level kecuali level (kecuali leaves) mempunyai jumlah cabang yang sama

c. Pohon biner (*Binary Tree*), yaitu jika setiap node pada seluruh level dalam pohon (kecuali *leaves*) mempunyai dua cabang

Model Hierarchycal digunakan untuk menggambarkan jenis kerelasian 1- ke –n dalam hubungan agregat data (Sutanta, 2004)

2. Contoh model data hierarchical

Tanda 1 mata panah pada diagram schema kerelasian antar-record menunjukkan satu kemungkinan kejadian dan tanda 2 mata panah menunjukkan lebih dari satu kemungkinan kejadian. Beberapa contoh dari model data hierarchical adalah file system komputer, organisasi pekerjaan, dan sistem klasifikasi buku. Dalam file system komputer, direktori dapat berisi subdirektori atau file, dan setiap subdirektori dapat berisi subdirektori lain atau file. Dalam organisasi pekerjaan, setiap karyawan dapat memiliki atasan atau bawahan, dan setiap divisi dapat memiliki divisi turunan. Dalam sistem klasifikasi buku, setiap kategori dapat memiliki subkategori, dan setiap subkategori dapat memiliki subkategori lain

3. Kelebihan dan kekurangan model data hierarchical

Kelebihan model data hierarchycal adalah sangat mudah dalam penyusunan definisi struktur *database,*

memudahkan dalam mengakses dan mengelola data yang bersifat hierarkis, memudahkan dalam memodelkan hubungan antar entitas, dan memudahkan dalam memahami hubungan antar entitas.

Kelemahan model data hierarchycal adalah Tidak dapat menggambarkan hubungan antar data selain jenis kerelasian 1- ke -n, membutuhkan duplikasi data untuk menjaga konsistensi data, sulit untuk memodelkan hubungan antar entitas yang tidak hierarkis, dan memiliki keterbatasan dalam mengatasi hubungan antar entitas yang sangat kompleks.

7.3. Model Data Network

1. Pengertian Model data Network

Model jaringan dapat dideskripsikan ke dalam struktur hubungan beringkat yang tersusun atas komponen berupa node. Teknik *levelling* dan penyebutan node pada model jaringan sama dengan teknik dan penyebutan dalam model hierarki. (Silberschatz, 2001) atau sebuah model data yang menggambarkan relasi antar entitas sebagai jaringan, dimana setiap entitas dapat memiliki hubungan dengan beberapa entitas lain. Dalam model data ini, data disimpan dan diorganisir sebagai node dalam jaringan, dan hubungan antar entitas ditunjukkan oleh garis yang menghubungkan node-node.

Berbeda dengan model hierarki yang hanya menggunakan sebuah *pointer*, model jaringan menambahkan sebuah pointer untuk meningkatkan fleksibilitas model *network*, artinya model jaringan menggunakan 2 buah *pointer*, satu digunakan untuk menghubungkan dengan *record* sebelumnya (*Previous* = P) dan yang lain digunakan untuk menghubungkan dengan *record* selanjutnya (*Next* = N).

Contoh dari model data network adalah sistem perpustakaan, sistem jaringan telepon, dan sistem manajemen relasi pelanggan. Dalam sistem perpustakaan, buku dapat dipinjam oleh beberapa peminjam, dan setiap peminjam dapat meminjam beberapa buku. Dalam sistem jaringan telepon, setiap telepon dapat terhubung dengan beberapa telepon lain, dan setiap telepon dapat menerima panggilan dari beberapa telepon lain. Dalam sistem manajemen relasi pelanggan, setiap pelanggan dapat memiliki beberapa hubungan dengan pelanggan lain, dan setiap pelanggan dapat memiliki beberapa hubungan dengan perusahaan.

2. Kelebihan dan kekurangan model data network

   Kelebihan model data network adalah sangat memudahkan perancang pada saat mendefinisikan struktur file database, memudahkan dalam menggambarkan hubungan antar entitas yang tidak terbatas, memudahkan dalam mengatasi hubungan antar entitas yang sangat kompleks, dan memudahkan

dalam mengakses data dengan cara mengikuti jalur dalam jaringan

Kelemahan model data network adalah tidak dapatigunakan untuk menggambarkan hubungan antar data selain jenis kerelasian n- ke -n dan sulit untuk memodelkan hubungan antar entitas yang bersifat hierarkis, sulit untuk mengatur akses data dan mengelola data secara efisien, dan memiliki keterbatasan dalam mengatasi duplikasi data.

7.4. Rangkuman

Model hierarki (*Hierarchycal Model*) merupakan salah satu model data yang didasarkan pada *record* (*Record Based Data Model*/*RDBM*). Model ini digunakan untuk menjelaskan kepada pengguna tentang hubungan logik antar data dalam basis data dalam bentuk bertingkat (Silberschatz, 2001).

Model jaringan dapat dideskripsikan ke dalam struktur hubungan beringkat yang tersusun atas komponen berupa node. Teknik *levelling* dan penyebutan node pada model jaringan sama dengan teknik dan penyebutan dalam model hierarki. (Silberschatz, 2001).

Berbeda dengan model hierarki yang hanya menggunakan sebuah pointer, model jaringan menambahkan sebuah pointer untuk meningkatkan fleksibilitas model network, artinya model jaringan menggunakan 2 buah pointer, satu digunakan untuk menghubungkan dengan

record sebelumnya (Previous = P) dan yang lain digunakan untuk menghubungkan dengan record selanjutnya (Next = N).

7.5. Bahan Diskusi

Berikut beberapa Bahan Diskusi mengenai Model Data Hierarki dan Network, Jika ada dua kasus proses bisnis dibawah ini :

1. Sistem Informasi Pembelian Barang: Studi kasus ini membahas bagaimana membuat sistem informasi pembelian barang yang menggunakan basis data untuk mengelola data pembelian, data barang, dan laporan pembelian. Tujuan pengembangan basis data adalah untuk memastikan bahwa data pembelian dapat tersimpan dengan benar dan mudah dikelola.
2. Sistem Informasi Manajemen Proyek: Studi kasus ini membahas bagaimana membuat sistem informasi manajemen proyek yang menggunakan basis data untuk mengelola data proyek, data tugas, dan laporan proyek. Tujuan pengembangan basis data adalah untuk memastikan bahwa data proyek dapat tersimpan dengan benar dan mudah dikelola.

Dari kasus diatas jawab pertanyaan dibawah ini untuk memahami konsep dan pengembangan basis data menggunakan model data hierarki dan network :

1. Konsep Dasar Model Data Hierarki dan Network: diskusikan konsep dasar model data hierarki, seperti

parent-child relationship dan node. Jelaskan bagaimana model data hierarki digunakan untuk mengungkapkan hubungan antar elemen dalam suatu sistem basis data.

2. Keuntungan dan Kerugian Model Data Hierarki dan Network: diskusikan keuntungan dan kerugian dari menggunakan model data hierarki dan network . Bagaimana model data hierarki dan network membantu dalam proses navigasi dan mempermudah komunikasi antar elemen? Apa saja kerugian dari menggunakan model data hierarki, seperti keterbatasan dalam mengungkapkan relasi antar elemen?
3. Struktur Model Data Hierarki dan Network: diskusikan bagaimana struktur model data hierarki dan network mempengaruhi desain sistem basis data. Jelaskan bagaimana struktur model data hierarki membantu mengatasi masalah redundansi dan memastikan integritas data.
4. Proses pembuatan Model Data Hierarki dan Network: diskusikan proses pembuatan model data hierarki, mulai dari pengumpulan informasi hingga validasi dan pengujian. Jelaskan bagaimana model data hierarki dan network dapat diperbaiki dan diperbarui seiring dengan perubahan dalam sistem basis data.
5. Aplikasi Model Data Hierarki dan Network: diskusikan aplikasi dari model data hierarki dan network, seperti dalam bidang organisasi, manajemen sumber daya, dan

lainnya. Bagaimana model data hierarki membantu dalam proses pengambilan keputusan dan mempermudah pemahaman struktur?

BAB VIII
MODEL DATA RELASIONAL

8.1. Pengantar

Model data relasional adalah salah satu dari beberapa model data yang digunakan dalam pemodelan basis data. Model ini mewakili data dalam bentuk tabel-tabel yang saling terkait melalui relasi atau hubungan antar tabel.

Dalam model data relasional, setiap entitas dapresentasikan sebagai tabel yang terdiri dari baris dan kolom. Setiap kolom menyimpan atribut atau informasi tentang entitas terkait, dan setiap baris menyimpan data atribut untuk instance spesifik dari entitas. Relasi antar tabel didefinisikan melalui kunci atau kolom yang berisi nilai yang unik dan dapat diterapkan pada tabel lain sebagai referensi.

Keuntungan dari model data relasional adalah keterbacaan yang baik dan kemudahannya dalam memastikan integritas data melalui teknik seperti normalisasi. Model ini juga mudah dimodifikasi dan memperbarui data.

Namun, model data relasional juga memiliki beberapa kekurangan, seperti kesulitan dalam merepresentasikan hubungan antar entitas yang lebih kompleks dan sulit untuk memvisualisasikan hubungan antar entitas.

8.2. Terminologi RDBM

Model RDBM *(Relational Database Model)* diperkenalkan oleh E.F. Codd pada tahun 1970. RDBM menjelaskan kepada pengguna tentang hubungan logik antar data dalam basis data dengan merepresentasikannya ke dalam bentuk relasi-relasi berupa tabel mendatar *(flat file)* yang terdiri atas sejumlah baris yang menunjukkan *record* dan kolom yang menunjukkan atribut tertentu (Martin, 1975). Atau sebuah model data yang menggambarkan entitas dan hubungan antar entitas sebagai tabel-tabel dalam basis data. Dalam model data ini, setiap entitas diterima sebagai tabel yang memiliki beberapa kolom, yang mewakili atribut dari entitas tersebut, dan baris, yang mewakili instance dari entitas. Hubungan antar entitas diterima sebagai hubungan antar tabel, dimana satu tabel dapat terhubung dengan tabel lain melalui kolom kunci asing. Berikut istilah-istilah dalam Terminologi RDBM :

1. **Elemen Data** : Unit terkecil yang disebut data
2. **Atribut** : Sekelompok rinci data yang mempunyai arti
3. ***Record*** : Sekumpulan atribut yang mempunyai hubungan terhadap objek tertentu
4. **Tabel** / **Relasi** : Sekumpulan *record* yang sejenis secara relasi
5. **Derajat** : Jumlah atribut dalam sebuah relasi
6. **Kardinalitas** : Jumlah *record* dalam sebuah relasi

7. **Kerelasian :** Hubungan antar relasi
8. ***Unary Relation* :** Relasi yang tersusun oleh 1 (satu) atribut
9. ***Binary Relation* :** Relasi yang tersusun oleh 2 (dua) atribut
10. ***Ternary Relation* :** Relasi yang tersusun oleh 3 (tiga) atribut
11. ***N-ary Relation* :** Relasi yang tersusun oleh n atribut
12. **Key :** Satu gabungan atribut bersifat unik yang digunakan untuk mengidentifikasi setiap *record* dalam relasi
13. **Candidate Key :** Satu atau gabungan minimal atribut bersifat unik yang dapat digunakan untuk mengidentifikasi setiap *record* dalam relasi
14. **Primary Key :** Bagian dari *Candidate Key* yang dipilih/ digunakan sebagai kunci utama dalam relasi
15. ***Alternate Key* :** Bagian dari *Candidate Key* yang tidak dipilih/ digunakan sebagai kunci utama dalam relasi
16. ***Foreign Key* :** Satu atau gabungan sembarang atribut yang menjadi *Primary Key* dalam relasi lain yang mempunyai hubungan logik
17. **Domain :** himpunan nilai yang memenuhi syarat
18. **Schema :** Deskipsi hubungan logik secara global, termasuk di dalamnya nama dan deskripsi tipe dan

ukuran atribut dan hubungan logik antar relasi basis data dalam lingkup sebuah sistem

19. **Subschema** : deskripsi hubungan logik secara terpisah, termasuk di dalamnya nama dan hubungan logik antar relasi basis data dalam lingkup sebuah sub sistem aplikasi basis data

Beberapa contoh dari model data relasional adalah sistem manajemen basis data perusahaan, sistem informasi pelanggan, dan sistem informasi produk. Dalam sistem manajemen basis data perusahaan, setiap departemen dapat diterima sebagai tabel, dan setiap karyawan dapat diterima sebagai tabel lain. Hubungan antar departemen dan karyawan dapat diterima sebagai hubungan antar tabel melalui kolom kunci asing. Dalam sistem informasi pelanggan, setiap pelanggan dapat diterima sebagai tabel, dan setiap pesanan dapat diterima sebagai tabel lain. Hubungan antar pelanggan dan pesanan dapat diterima sebagai hubungan antar tabel melalui kolom kunci asing.

8.3. Karakteristik Basis Data Model RDBM

Relasi dalam model data RDBM mempunyai beberapa karakteristik yang harus dipenuhi. Karakteristik yang dimaksud adalah sebagai berikut :

1. Semua elemen data/entri pada suatu *record* dan atribut tertentu harus mempunyai nilai tunggal (*single value*),

bukan suatu larik atau grup perulangan dan harus berupa nilai yang tidak dapat dibagi lagi (*atomic value*)

2. Semua elemen data/entri pada suatu atribut tertentu dalam sebuah relasi harus mempunyai tipe dan ukuran yang sama
3. Masing-masing atribut dalam sebuah relasi mempunyai nama yang unik
4. Pada sebuah relasi tidak ada dua *record* yang identic

Model data relasional memiliki beberapa kelebihan, seperti memudahkan dalam mengelola data dengan cara memodelkan hubungan antar entitas, memudahkan dalam memastikan integritas data dengan cara mengatasi duplikasi data, dan memudahkan dalam mengakses dan mengolah data dengan cara menggunakan bahasa query standar seperti SQL. Namun, model data relasional juga memiliki beberapa kekurangan, seperti sulit untuk memodelkan hubungan antar entitas yang tidak terbatas, sulit untuk memodelkan hubungan antar entitas yang bersifat hierarkis, dan memiliki keterbatasan dalam mengatasi hubungan antar entitas yang sangat kompleks.

8.4. Komponen Relasi

Setiap relasi RDBM selalu tersusun atas 2 komponen, yaitu (Sutanta, 2004) :

1. Intensi (*intension*), menunjukkan definisi struktur penamaan (*naming structure*) pada relasi dan batasan

integritas *(integrity contraint)* yang meliputi batasan integritas kesatuan *(entity integrity)* pada kunci primer dan batasan integritas referensial pada kunci penghubung.

2. Ekstensi *(extension)*, menunjukkan nilai-nilai aktual elemen data/entri yang tersimpan dalam berkas pada suatu saat tertentu, komponen ekstensi cenderung tidak stabil/berubah karena nilai-nilai elemen data/entri selalu ditambah, diperbaharui, atau dihapus.

8.5. Kunci Relasi

Kunci Relasi diperlukan dalam rangka pengaksesan data dari dalam relasi atau untuk menyusun kerelasian antar relasi. Kunci Relasi harus bersifat unik Tidak diizinkan adanya kerangkapan pada kunci relasi Berdasarkan Jumlah Atribut Penyusunnya, kunci relasi diklasifikasikan menjadi 2 (dua) jenis, yaitu (Martin, 1975) :

1. Kunci Sederhana *(simple key)*, merupakan kunci relasi yang tersusun atas sebuah atribut. Contoh : NPM, NIP, NIK.

2. Kunci komposit *(composite key)*, merupakan kunci relasi yang tersusun atas gabungan atribut. Hal ini terjadi jika untuk mencapai sifat unik tidak dapat dipenuhi oleh sebuah atribut, tetapi harus menggabungkan lebih dari satu/beberapa atribut.

Secara umum kunci relasi untuk menghubungkan antar realasi maka kunci relasi terdiri atas (Martin, 1975) :

1. Kunci Kandidat (*Candidate Key/CK*), merupakan 1 (satu) atau gabungan minimal atribut yang bersifat unik yang dapat digunakan untuk mengidentifikasi/membedakan setiap *record* dalam relasi
2. Kunci Primer (*Primary Key/PK*), merupakan bagian/salah satu dari CK yang dipilih/digunakan sebagai kunci utama untuk mengidentifikasi/membedakan setiap record dalam relasi.
3. Kunci alternatif *(Alternate Key/AK),* merupakan bagian dari CK yang tidak dipilih/digunakan sebagai PK.
4. Kunci Penghubung *(Foreign Key/FK),* merupakan 1 (satu) atau gabungan sembarang atribut yang menjadi PK dalam relasi lain yang mempunyai hubungan secara logik.

8.6. Aturan-aturan (*rules*) pada Kunci Relasi

Aturan-aturan pada kunci relasi utamanya berkaitan dengan 2 (dua) macam batasan integritas kesatuan/integritas entitas (*entity integrity*) dan integritas referensial (*referential integrity*) (Parsaye, dkk 1989) :

1. Integritas kesatuan/integritas entitas (*entity integrity*), aturan ini memberikan batasan secara kesatuan nilai-nilai elemen data/entri pada atribut yang dipilih/digunakan sebagai PK tidak boleh null (kosong).

2. Integritas referensial (*referential integrity*), aturan ini memberikan batasan bahwa di dalam kerelasian antar 2 (dua) atau lebih relasi dalam basis data yang dihubungkan dengan suatu kunci penghubung (*foreign* key/FK).

8.7. Kerelasian antar Relasi

Jenis-jenis kerelasian dalam RDBM adalah (Date, 1995) sebagai berikut :

1. Kerelasian 1 ke 1
2. Kerelasian 1 ke banyak
3. Kerelasian banyak ke 1
4. Kerelasian banyak ke banyak

Beberapa definisi yang berkaitan dengan penyebutan relasi (tabel) adalah sebagai berikut (Sutanta, 2004) :

1. Relasi tak gayut, merupakan relasi yang tidak memiliki FK, relasi induk
2. Relasi asosiatif, relasi yang memiliki FK lebih dari 1
3. Relasi karakteristik, menyatakan sebuah relasi yang berasal dari entitas dependen/tak gayut. Ciri relasi memiliki kerelasian n ke 1.
4. Subrelasi, menyatakan sebuah relasi yang berasal dari sub type entity.

8.8. Penyimpangan-penyimpangan (*anomalies*) dalam pengolahan data

Menurut Gio Wiederhold (1998), penyimpangan-penyimpangan (*anomalies*) yang harus dihindari dalam modifikasi data meliputi :

1. Penyimpangan penghapusan (*Delete Anomally*)

   Merupakan suatu proses penghapusan nilai rinci data yang mengakibatkan hilangnya informasi rinci data lain yang tidak mempunyai kerelasian secara logik

2. Penyimpangan penyisipan (*Insert Anomally*)

   Merupakan proses penyisipan suatu nilai rinci data yang mengakibatkan perlunya penyisipan pada nilai rinci data lain yang tidak mempunyai kerelasian secara logik

3. Penyimpangan pembaharuan (*Update Anomally*)

   Merupakan proses mengubah suatu nilai rinci data yang mengakibatkan perlunya pengubahan pada nilai rinci data lain yang tidak mempunyai kerelasian secara logic

8.9. Ketergantungan Data

Penyimpangan yang terjadi dalam modifikasi selain akibat kerangkapan data, juga terjadi karena kenyataan suatu nilai rinci data dalam relasi bergantung kepada nilai rinci data yang lain, selama rinci data tersebut dalam struktur tidak tergantung secara logik.

Relasi : tabel : Tabel_data_mahasiswa

Nilai rinci : atribut : nama, umur, alamat, dsbnya

8.10. Rangkuman

RDBM menjelaskan kepada pengguna tentang hubungan logik antar data dalam basis data dengan merepresentasikannya ke dalam bentuk relasi-relasi berupa tabel mendatar (*flat file*) yang terdiri atas sejumlah baris yang menunjukkan *record* dan kolom yang menunjukkan atribut tertentu (Martin, 1975).

Berdasarkan **Jumlah Atribut Penyusunnya**, kunci relasi diklasifikasikan menjadi 2 (dua) jenis, yaitu (Martin, 1975) :

1. Kunci Sederhana (*simple key*), merupakan kunci relasi yang tersusun atas sebuah atribut. Contoh : NPM, NIP, NIK.
2. Kunci komposit (*composite key*), merupakan kunci relasi yang tersusun atas gabungan atribut. Hal ini terjadi jika untuk mencapai sifat unik tidak dapat dipenuhi oleh sebuah atribut, tetapi harus menggabungkan lebih dari satu/beberapa atribut.

Berdasarkan macamnya, kunci relasi terdiri atas (Martin, 1975) :

1. Kunci Kandidat (*Candidate Key/CK*), merupakan 1 (satu) atau gabungan minimal atribut yang bersifat unik yang

dapat digunakan untuk mengidentifikasi/membedakan setiap *record* dalam relasi

2. Kunci Primer (*Primary Key*/*PK*), merupakan bagian/salah satu dari CK yang dipilih/digunakan sebagai kunci utama untuk mengidentifikasi/membedakan setiap record dalam relasi.
3. Kunci alternatif (*Alternate Key*/*AK*), merupakan bagian dari CK yang tidak dipilih/digunakan sebagai PK.
4. Kunci Penghubung (*Foreign* Key/FK), merupakan 1 (satu) atau gabungan sembarang atribut yang menjadi PK dalam relasi lain yang mempunyai hubungan secara logik.

8.11. Bahan Diskusi

Berikut beberapa Bahan Diskusi mengenai Model Data Relasional, Jika ada dua kasus proses bisnis dibawah ini :

1. Sistem Informasi Pembelian Barang: Studi kasus ini membahas bagaimana membuat sistem informasi pembelian barang yang menggunakan basis data untuk mengelola data pembelian, data barang, dan laporan pembelian. Tujuan pengembangan basis data adalah untuk memastikan bahwa data pembelian dapat tersimpan dengan benar dan mudah dikelola.
2. Sistem Informasi Manajemen Proyek: Studi kasus ini membahas bagaimana membuat sistem informasi manajemen proyek yang menggunakan basis data untuk

mengelola data proyek, data tugas, dan laporan proyek. Tujuan pengembangan basis data adalah untuk memastikan bahwa data proyek dapat tersimpan dengan benar dan mudah dikelola.

Dari kasus diatas jawab pertanyaan dibawah ini untuk memahami konsep dan pengembangan basis data menggunakan model data relasional :

1. Konsep Dasar Model Data Relasional: diskusikan konsep dasar model data relasional, seperti tabel, kolom, baris, dan relasi. Jelaskan bagaimana model data relasional digunakan untuk mengungkapkan hubungan antar elemen dalam suatu sistem basis data.
2. Keuntungan dan Kerugian Model Data Relasional: diskusikan keuntungan dan kerugian dari menggunakan model data relasional. Bagaimana model data relasional membantu dalam proses pengelolaan data dan mempermudah komunikasi antar aplikasi? Apa saja kerugian dari menggunakan model data relasional, seperti keterbatasan dalam mengungkapkan hubungan antar elemen?
3. Normalisasi Model Data Relasional: diskusikan bagaimana normalisasi mempengaruhi desain model data relasional. Jelaskan bagaimana normalisasi membantu mengatasi masalah redundansi dan memastikan integritas data.

4. SQL dan Model Data Relasional: diskusikan bagaimana SQL mempengaruhi dan membantu dalam desain model data relasional. Jelaskan bagaimana SQL membantu dalam proses pembuatan tabel, memasukkan data, dan memodifikasi data.
5. Aplikasi Model Data Relasional: diskusikan aplikasi dari model data relasional, seperti dalam bidang bisnis, teknologi, dan lainnya. Bagaimana model data relasional membantu dalam proses pengambilan keputusan dan mempermudah pemahaman hubungan antar elemen?

BAB IX
NORMALISASI

9.1. Pengantar

Normalisasi basis data adalah proses memecahkan tabel besar dalam basis data menjadi tabel yang lebih kecil dan lebih ternormalisasi untuk mengurangi data yang redundan dan memastikan integritas data. Proses ini membantu dalam menjaga konsistensi data dan mempermudah pembaruan dan modifikasi data di masa depan.

Normalisasi basis data dilakukan dengan memastikan bahwa setiap tabel memiliki tugas yang spesifik dan hanya menyimpan informasi yang unik. Setiap tabel harus ternormalisasi sampai tingkat tertentu, yang dikenal sebagai tingkat normalisasi. Tiga tingkat normalisasi yang paling umum adalah tingkat pertama (1NF), tingkat kedua (2NF), dan tingkat ketiga (3NF).

Normalisasi basis data memiliki beberapa keuntungan, seperti mengurangi redundansi data, memastikan integritas data, mempermudah pembaruan dan modifikasi data, dan membantu dalam menjaga konsistensi data. Namun, normalisasi basis data juga memiliki beberapa kekurangan, seperti membuat sistem basis data menjadi lebih kompleks

dan memerlukan lebih banyak query untuk mengambil data dari beberapa tabel.

Oleh karena itu, penting bagi pengembang basis data untuk mempertimbangkan tingkat normalisasi yang sesuai untuk setiap proyek dan memastikan bahwa normalisasi tidak mempengaruhi performa sistem secara negatif.

9.2. Pengertian Normalisasi

Normalisasi adalah suatu teknik yang menstrukturkan/memecah atau mendekomposisi data dalam cara-cara tertentu untuk mencegah / meminimalisir timbulnya permasalahan pengolahan data dalam basis data atau proses yang digunakan untuk memastikan bahwa sebuah basis data memiliki desain yang baik dan ternormalisasi. Normalisasi membantu mencegah redundansi data dan memastikan integritas data. Ini dilakukan dengan membagi tabel dalam basis data menjadi tabel-tabel yang lebih kecil dan lebih terfokus, dengan setiap tabel memiliki tujuan dan fungsinya sendiri. Normalisasi juga memastikan bahwa data dalam tabel dipisahkan dari dependensi logis dan dipindahkan ke tabel yang sesuai.

Berikut keuntungan Proses Normalisasi :

1. Meminimalkan ukuran penyimpanan yang diperlukan untuk menyimpan data
2. Meminimalkan resiko inkonsistensi data pada basis data
3. Meminimalkan kemungkinan anomali pembaruan

4. Memaksimalkan stabilitas struktur basis data

Proses dalam normalisasi yang umum dilakukan untuk memudahkan dalam proses normalisasi dapat di uraikan dalam pernyataan berikut :

1. Data diuraikan ke dalam tabel, selanjutnya dianalisis berdasarkan persyaratan tertentu ke dalam beberapa tingkat
2. Apabila tabel yang diuji belum memenuhi persyaratan tertentu, maka tabel tersebut dipecah menjadi beberapa tabel yang lebih sederhana sampai ke bentuk yang optimal.

9.3. Tahapan-tahapan Normalisasi

Tahapan-tahapan normalisasi dapat dilakukan dengan beberapa langkah yang disebut dengan bentuk normal atau *Normal Form*, Ada beberapa tingkatan normalisasi, mulai dari Normal Form 1 (NF1) hingga Normal Form 5 (NF5), yang masing-masing menentukan tingkat normalisasi yang lebih tinggi dan menetapkan aturan yang lebih ketat untuk desain tabel. Dalam praktik, beberapa aplikasi hanya membutuhkan normalisasi sampai Normal Form 3 (NF3), meskipun Normal Form 5 (NF5) memiliki tingkat normalisasi yang sangat tinggi dan memastikan integritas data yang sangat ketat. Berikut bentuk normal yang terjadi dalam tahapan-tahapan normalisasi diantaranya adalah:

1. Bentuk tidak normal (UNF)

2. Bentuk normal pertama (1NF)
3. Bentuk normal kedua (2NF)
4. Bentuk normal ketiga (3NF)
5. Bentuk normal Boyce-Codd (BCNF)
6. Bentuk normal keempat (4NF)
7. Bentuk normal kelima (5NF)

Namun secara umum yang biasa dilakukan beberapa tahapan dalam proses normalisasi basis data, yaitu:

1. Analisis data: Tahap ini melibatkan penelaahan data yang akan diterapkan dalam basis data dan identifikasi atribut atau kolom yang akan digunakan dalam tabel.
2. Pembuatan tabel sementara: Tahap ini melibatkan pembuatan tabel sementara yang berisi atribut-atribut yang telah didentifikasi selama tahap analisis data.
3. Aplikasi Normal Form 1 (NF1): Tahap ini melibatkan penentuan apakah setiap tabel sementara memenuhi aturan NF1, yaitu setiap kolom harus memiliki tipe data yang sama dan tidak boleh memiliki duplikat data.
4. Aplikasi Normal Form 2 (NF2): Tahap ini melibatkan penentuan apakah setiap tabel sementara memenuhi aturan NF2, yaitu setiap kolom yang tidak berhubungan dengan kunci utama harus dipisahkan ke dalam tabel yang berbeda.

5. Aplikasi Normal Form 3 (NF3) dan seterusnya: Tahap ini melibatkan penentuan apakah setiap tabel sementara memenuhi aturan NF3 dan seterusnya, yang melibatkan penentuan apakah setiap kolom yang tidak berhubungan dengan kunci utama atau kunci asing lainnya harus dipisahkan ke dalam tabel yang berbeda.
6. Uji validasi: Tahap ini melibatkan uji validasi untuk memastikan bahwa basis data memenuhi kriteria normalisasi yang telah ditentukan.
7. Implementasi: Tahap terakhir melibatkan implementasi dari desain tabel yang telah diterapkan dan normalisasi yang telah dilakukan.

Perlu diingat bahwa proses normalisasi basis data bertujuan untuk memastikan integritas data dan menghilangkan redundansi data, dan bukan untuk membuat basis data lebih mudah dikelola atau diterapkan.

9.4. Kunci Entitas

Karena pada saat proses Normalisasi kita membutuhkan pemahaman mengenai kunci (key) entitas, Kunci entitas adalah kolom atau kombinasi kolom dalam tabel yang digunakan untuk membedakan antar baris dalam tabel. Ini memastikan bahwa setiap baris dalam tabel memiliki identitas yang unik. Dalam normalisasi basis data, kunci entitas memainkan peran penting dalam menentukan normal form

sebuah tabel dan memastikan integritas data. maka berikut jenis-jenis kunci (key) dari entitas :

1. Kunci Super (*Super Key/SK*)

   Himpunan satu atribut atau lebih yang memungkinkan identifikasi secara unik entitas pada himpunan entitas tersebut

   - Calon Kunci *(Candidate Key/CK)*

     Super Key yang minimal

   - Kunci Utama *(Primary Key/PK)*

     Candidate Key yang dipilih perancang basis data sebagai alat utama untuk mengidentifikasi entitas pada himpunan entitas

2. Kunci Alternatif (*Alternate Key/AK*)

   Candidate Key yang tidak dipilih sebagai *Primary Key*

3. Kunci Asing (*Foreign Key/FK*)

   Jika suatu *Primary Key* pada relasi A berada pada relasi B, dimana *Primary Key* tersebut berfungsi sebagai *Reference*

Penentuan kunci entitas yang tepat dan implementasinya selama proses normalisasi basis data membantu memastikan integritas data dan menghilangkan redundansi data.

9.5. Bentuk Normalisasi

Bentuk normalisasi database adalah tingkat keteraturan data dalam tabel dalam basis data. Ada beberapa bentuk normalisasi, masing-masing memiliki aturan yang berbeda untuk memastikan integritas data dan menghilangkan redundansi data. erikut adalah beberapa bentuk normalisasi yang umum digunakan dan dilakukan dalam beberapa level :

1. Bentuk Normal 1NF

   Kriteria 1NF : aturan NF1 menyatakan bahwa setiap kolom dalam tabel harus memiliki tipe data yang sama dan tidak boleh memiliki duplikat data.

   - Jika seluruh atribut dalam relasi bernilai atomik *(atomic value)*
   - Jika seluruh atribut dalam relasi bernilai tunggal *(single value)*
   - Jika relasi tidak memuat set atribut berulang

   Jika semua record mempunyai sejumlah atribut yang sama

2. Bentuk 2NF

   Kriteria 2NF: aturan NF2 menyatakan bahwa setiap kolom yang tidak berhubungan dengan kunci utama harus dipisahkan ke dalam tabel yang berbeda.

   - Jika memenuhi kriteria 1NF

- Jika semua atribut non kunci FD pada PK *(primary key)* dengan kata lainnya atribut tidak primer bergantung penuh terhadap CK *(Candidate Key)*

  Contoh :

  NPM – Nama – Mata_Kuliah

  NPM dan Mata_Kuliah → CK *(Candidate Key)*

  NPM → PK *(Primary key)*

  Relasinya :

  NPM – Nama

  NPM – Mata_Kuliah

3. Bentuk 3NF

   Kriteria 3NF : aturan NF3 menyatakan bahwa setiap kolom yang tidak berhubungan dengan kunci utama atau kunci asing lainnya harus dipisahkan ke dalam tabel yang berbeda.

   - Jika memenuhi kriteria 2NF
   - Jika semua atribut non kunci tidak TDF (non TDF) terhadap PK

4. Bentuk BCNF

   Kriteria BCNF : aturan BCNF menyatakan bahwa setiap tabel harus memiliki kunci utama yang memenuhi syarat sebagai kunci superkey, yaitu memiliki relasi fungsional yang unik dengan setiap kolom lain dalam tabel

- Jika memenuhi bentuk 3NF
- Jika determine (penentu) pada relasi adalah kunci relasi *(Primary Key/PK)*

Secara umumnya bentuk BCNF sama dengan 3NF, sehingga pada bentuk 3NF masih terdapat ketergantungan data, maka dilakukan dekomposisi sehingga menjadi bentuk BCNF

5. Bentuk 4NF

   Kriteria 4NF : aturan 4NF menyatakan bahwa setiap tabel harus memenuhi syarat NF3 dan tidak memiliki relasi fungsional yang tidak dapat diterima.

6. Bentuk 5NF

   Kriteria 5 NF :Aturan 5NF menyatakan bahwa setiap tabel harus memenuhi syarat 4NF dan tidak memiliki relasi multivalued.

Penting untuk diingat bahwa tujuan dari normalisasi basis data adalah memastikan integritas data dan menghilangkan redundansi data, bukan untuk membuat basis data lebih mudah dikelola atau diterapkan. Oleh karena itu, tingkat normalisasi yang tepat harus dipilih berdasarkan konteks aplikasi dan kebutuhan data.

9.6. Rangkuman

Normalisasi data adalah suatu teknik yang menstrukturkan/memecah atau mendekomposisi data dalam

cara-cara tertentu untuk mencegah / meminimalisir timbulnya permasalahan pengolahan data dalam basis data

Tahapan-tahapan normalisasi :

1. Bentuk tidak normal (UNF)
2. Bentuk normal pertama (1NF)
3. Bentuk normal kedua (2NF)
4. Bentuk normal ketiga (3NF)
5. Bentuk normal *Boyce-Codd* (BCNF)
6. Bentuk normal keempat (4NF)
7. Bentuk normal kelima (5NF)

Pada saat proses Normalisasi kita membutuhkan pemahaman mengenai kunci (key) entitas, berikut jenis-jenis kunci (key) dari entitas :

1. Kunci Super *(Super Key/SK)*

   Himpunan satu atribut atau lebih yang memungkinkan identifikasi secara unik entitas pada himpunan entitas tersebut

   - Calon Kunci *(Candidate Key/CK)*
   - Kunci Utama *(Primary Key/PK)*
2. Kunci Alternatif (*Alternate Key/AK*)
3. Kunci Asing (*Foreign Key/FK*)

9.7. Bahan Diskusi

Berikut beberapa Bahan Diskusi mengenai Normalisasi database, Jika ada dua kasus proses bisnis dibawah ini :

1. Sistem Informasi Pembelian Barang: Studi kasus ini membahas bagaimana membuat sistem informasi pembelian barang yang menggunakan basis data untuk mengelola data pembelian, data barang, dan laporan pembelian. Tujuan pengembangan basis data adalah untuk memastikan bahwa data pembelian dapat tersimpan dengan benar dan mudah dikelola.
2. Sistem Informasi Manajemen Proyek: Studi kasus ini membahas bagaimana membuat sistem informasi manajemen proyek yang menggunakan basis data untuk mengelola data proyek, data tugas, dan laporan proyek. Tujuan pengembangan basis data adalah untuk memastikan bahwa data proyek dapat tersimpan dengan benar dan mudah dikelola.

Dari kasus diatas jawab pertanyaan dibawah ini untuk memahami konsep dan pengembangan basis data dalam aspek normalisasi: Normalisasi: diskusikan bagaimana normalisasi mempengaruhi desain ERD. Jelaskan bagaimana normalisasi membantu mengatasi masalah redundansi dan memastikan integritas data.

BAB X

SCHEMA DAN SUBSCHEMA

10.1. Pengantar

Dalam basis data, schema adalah struktur yang menggambarkan skema logis dari basis data. Schema menentukan bagaimana data dalam basis data terorganisasi, termasuk informasi tentang tabel, atribut, relasi, dan integritas data. Schema dapat dipahami sebagai sebuah abstraksi dari struktur fisik dari basis data.

Subschema, pada gilirannya, adalah bagian dari schema yang menentukan bagaimana aplikasi tertentu akan mengakses dan memanipulasi data dalam basis data. Subschema menentukan bagaimana aplikasi memanipulasi data, seperti bagaimana tabel harus digabungkan, atribut yang harus ditampilkan, dan batasan integritas data.

Dengan memiliki schema dan subschema yang berbeda, pengembang basis data dapat memastikan bahwa data dalam basis data terorganisasi dengan benar dan mudah diakses oleh aplikasi yang membutuhkan. Ini juga membantu dalam memastikan bahwa data tetap integritas dan konsisten, meskipun ada beberapa aplikasi yang memanipulasinya secara bersamaan.

Schema dan subschema sangat penting bagi pengembangan basis data karena mereka memastikan bahwa data dalam basis data terorganisasi dengan benar, mudah diakses, dan tetap integritas dan konsisten, meskipun ada beberapa aplikasi yang memanipulasinya secara bersamaan.

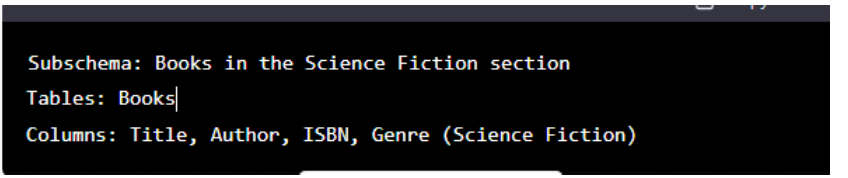
10.2. Konsep *Schema* dan *Subschema* Basis Data

Schema dan *Subschema* adalah konsep yang penting dalam basis data yang mengacu pada bagaimana data dalam basis data diorganisasi dan diatur.*Scema* memberikan deskripsi hubungan logik antar data dalam basis data secara lengkap, termasuk nama dan deskripsi dari semua *attribute , record* dan batasan nilai karena Schema adalah representasi abstrak dari struktur data dalam basis data. Schema menggambarkan bagaimana data dalam basis data terorganisasi dan bagaimana relasi antar tabel didefinisikan. Schema biasanya digunakan oleh administrator basis data atau pembuat basis data untuk membuat dan mengelola struktur data.

```
Schema: Library
Tables: Books, Members, Loans
```

Subscema merupakan deskripsi terpisah dari atribut, *record* dan batasan nilai yang digunakan oleh sebuah program aplikasi. Karena *subschema* adalah bagian dari *schema* yang diterapkan untuk aplikasi atau pengguna tertentu. *Subschema* menentukan bagaimana data akan ditampilkan dan diakses

oleh pengguna atau aplikasi tertentu. Subschema membatasi akses dan visibilitas data yang dapat dilihat dan digunakan oleh pengguna atau aplikasi.



Raymon McLeod dan George Schell menyatakan bahwa Scema memuat deskripsi yang meliputi :

1. Nama Field data.
2. Alias atau nama lain yg digunakan utk field data yang sama.
3. Tipe data.
4. Jumlah digit pada posisi angka.
5. Jumlah digit pada posisi desimal.
6. Sejumlah aturan integritas.

Sebuah relasi RDBM dipresentasikan sebagai sebuah tabel datar (flat file) yang memuat beberapa keterangan yaitu :

1. Nama relasi.
2. Kunci-kunci relasi.
3. Kunci-kunci indeks.
4. Sejumlah record.
5. Nama-nama atribut.

Berikut Contoh Schema: Schema dalam basis data bank dapat memiliki tabel untuk menyimpan informasi tentang nasabah, rekening, dan transaksi. Schema akan menggambarkan relasi antara tabel-tabel, seperti relasi antara tabel nasabah dan tabel rekening, atau antara tabel rekening dan tabel transaksi.

Contoh Subschema: Dalam konteks aplikasi internet banking, sub-schema akan diterapkan untuk setiap nasabah. Subschema akan menentukan informasi apa yang dapat dilihat dan diakses oleh nasabah, seperti informasi tentang saldo rekening, riwayat transaksi, dan informasi pribadi.

Dalam contoh ini, schema menggambarkan struktur data yang tersedia dalam basis data, sementara sub-schema menentukan bagaimana data akan diterapkan dan ditampilkan untuk setiap nasabah. Ini memastikan bahwa setiap nasabah hanya dapat melihat informasi yang relevan dan membatasi akses ke informasi yang tidak penting atau sensitif.

10.3. Aspek Pengembangan Basis Data

Aspek pengembangan basis data merupakan hal-hal yang harus diperhatikan dalam proses pembuatan dan pengembangan basis data. Berikut adalah beberapa aspek pengembangan basis data:

1. Analisis Kebutuhan: Analisis kebutuhan sangat penting dalam pengembangan basis data. Ini melibatkan

identifikasi dan evaluasi kebutuhan data dan informasi yang dibutuhkan oleh organisasi. Analisis kebutuhan akan membantu memastikan bahwa basis data yang dikembangkan sesuai dengan kebutuhan dan harapan pengguna.

2. Desain Database: Desain database adalah proses membuat representasi visual dari basis data yang akan dikembangkan. Desain database menentukan bagaimana data akan diorganisasi dan dalam bentuk apa. Desain database juga menentukan relasi antar tabel dan bagaimana data akan disimpan dan diterapkan.
3. Implementasi: Implementasi adalah proses membuat dan mengimplementasikan basis data sesuai dengan desain database. Ini melibatkan pembuatan tabel, relasi antar tabel, dan program akses basis data.
4. Integrasi Aplikasi: Integrasi aplikasi adalah proses memastikan bahwa basis data dapat bekerja dengan baik dengan aplikasi yang ada atau akan dikembangkan. Ini melibatkan pembuatan koneksi antara basis data dan aplikasi, serta memastikan bahwa data yang diterima dan disimpan oleh aplikasi sesuai dengan spesifikasi.
5. Pengujian: Pengujian adalah proses memastikan bahwa basis data bekerja dengan baik dan sesuai dengan spesifikasi. Ini melibatkan pengujian fungsionalitas dan performa basis data.

6. Dukungan dan Pemeliharaan: Dukungan dan pemeliharaan adalah hal yang penting setelah basis data sudah dikembangkan dan diimplementasikan. Ini melibatkan pemantauan dan pemeliharaan basis data untuk memastikan bahwa data tetap akurat dan tersimpan dengan benar.

Dengan memperhatikan aspek-aspek ini dalam pengembangan basis data, organisasi dapat memastikan bahwa basis data yang dikembangkan sesuai dengan kebutuhan dan harapan pengguna, dan dapat bekerja dengan baik dan efisien.Tujuan Basis Data Meliputi

1. Penyediaan sarana akses yang fleksibel.
2. Pemeliharaan integritas data.
3. Proteksi data dari kerusakan dan penggunaan yang ilegal.
4. Penyediaan sarana untuk penggunaan bersama.
5. Penyediaan sarana untuk kelerasian antar data.
6. Minimalisasi kerangkapan data.
7. Menghilangkan ketergantungan data.
8. Menstandarkan defenisi rinci data.
9. Meningkatkan produktivitas personal.

Pengembangan Basis Data terdiri dari 3 Tahapan

1. Analisis
2. Desain atau perancangan

3. Implementasi

Tahapan perancanngan meliputi

1. Review kebutuhan.
2. Desain umum.
3. Desain terperinci meliputi :
    a. Desain input
    b. Desain proses
    c. Desain output
    d. Desain Basis DataDesain Dialog
4. Laporan ke manajemen.

Tahapan implementasi meliputi

1. *Review* desain.
2. Penjadwalan tugas pengembangan.
3. *Coding program*.
4. Testing program meliputi :
    a. *Testing* modul.
    b. *Testing* menyeluruh.
5. Pelatihan Petugas.
6. Konversi sistem.
7. Laporan ke manajemen.

Pengembangan Basis Data terdiri dari 5 Tahapan

1. Perencanaan
2. Analisis
3. Desain / perancangan
4. Implementasi
5. Penggunaan /evaluasi

Tahapan perencanaan meliputi

1. Mengenali Masalah
2. Menentukan maslaah.
3. Menentukan tujuan
4. Mengenali kendala
5. Studi kelayakan.
6. Laporan ke manajemen.

Tahapan analisis meliputi

1. Menentukan kebutuhan informasi
2. Menentukan kriteria kinerja sistem
3. Laporan ke manajemen.

Tahapan implementasi meliputi

1. Menyiapkan PK.
2. Menyiapkan PL.
3. Menyiapkan Basis data.

4. Menyiapkan fasilitas fisik.
5. Melatih pemakai.
6. Laporan ke manajemen.

Tahapan evalusi meliputi

1. Operasional sistem.
2. Evaluasi sistem.
3. Memelihara sistem.
4. Mempertahankan kinerja sistem.
5. Meningkatkan kinerja sistem.
6. Laporan ke manajemen.

Pendekatan umum yg dilakukan Meliputi

1. Kelayakan teknis.
2. Kelayakan operasional.
3. Kalayakan ekonomis.
4. Kelayakan hukum.
5. Kelayakan Jadwal.

10.4. Rangkuman

Skema memberikan deskripsi hubungan logik antar data dalam basis data secara lengkap, termasuk nama dan deskripsi dari semua *attribute , record* dan batasan nilai. Subscema

merupakan deskripsi terpisah dari atribut, *record* dan batasan nilai yang digunakan oleh sebuah program aplikasi

Aspek pengembangan basis data terdiri dari tiga tahapan diantaranya analisis, perancangan dan impelmentasi. Analisis dilakukan untuk mempelajari permasalahan yang ada pada sistem , perancangan merupakan mencari jawaban dari permasalahan yang ada,sedangkan implementasi merupakan bagian pelaksanaan dari hasil perancangan yang telah dilakukan.

10.5. Bahan Diskusi

Berikut adalah beberapa Bahan Diskusi mengenai Schema dan Subschema:

1. Konsep Dasar Schema dan Subschema: diskusikan konsep dasar schema dan subschema, seperti definisi, fungsi, dan tujuan. Jelaskan bagaimana schema dan subschema digunakan untuk mengungkapkan hubungan antar elemen dalam suatu sistem basis data.
2. Schema sebagai Abstraksi: diskusikan bagaimana schema menyediakan abstraksi dari sistem basis data dan membantu mempermudah komunikasi antar aplikasi. Jelaskan bagaimana schema membantu dalam proses pengelolaan data dan memastikan integritas data.
3. Fungsi Subschema: diskusikan bagaimana subschema membantu mengatur akses ke sistem basis data.

Jelaskan bagaimana subschema membantu dalam proses pembuatan view dan memastikan keamanan data.

4. Perbandingan Schema dan Subschema: diskusikan perbandingan antara schema dan subschema, termasuk kelebihan dan kekurangan masing-masing. Jelaskan bagaimana kedua konsep ini membantu dalam proses pengambilan keputusan dan mempermudah pemahaman hubungan antar elemen.
5. Aplikasi Schema dan Subschema: diskusikan aplikasi dari schema dan subschema, seperti dalam bidang bisnis, teknologi, dan lainnya. Bagaimana schema dan subschema membantu dalam proses pengambilan keputusan dan mempermudah pemahaman hubungan antar elemen?

BAB XI

STRUCTURE QUERY LANGUAGE

11.1. Pengantar

SQL (*Structured Query Language*) adalah bahasa standar untuk berinteraksi dengan basis data relasional. Ini digunakan untuk mengakses, mengelola, dan memodifikasi data dalam basis data relasional.

SQL menyediakan berbagai perintah untuk membuat, mengubah, dan menghapus tabel, memasukkan, memperbarui, dan menghapus data, serta melakukan operasi seperti penggabungan tabel, pencarian data, dan agregasi data (seperti jumlah, rata-rata, maksimum, dll.).

SQL juga memiliki fitur yang membantu untuk memastikan integritas data, seperti batasan integritas referensial, yang memastikan bahwa data dalam tabel terkait tetap sesuai.

SQL merupakan bahasa yang sangat populer dan umum digunakan dalam pengembangan basis data, karena mudah dipahami dan memiliki banyak fitur yang membantu dalam pengembangan aplikasi yang memanfaatkan basis data. Keuntungan lain dari SQL adalah bahwa banyak perusahaan menawarkan produk dan layanan basis data relasional yang menggunakan SQL, sehingga mudah bagi pengembang untuk

menemukan sumber daya dan dukungan untuk membangun aplikasi yang menggunakan basis data relasional.

11.2. Pengertian *Structure Query Language*

SQL (Structured Query Language) adalah bahasa pemrograman khusus yang digunakan untuk memanajemen data dalam RDBMS atau bahasa standar untuk bekerja dengan basis data relasional. SQL digunakan untuk membuat, memodifikasi, dan mengakses data dalam basis data relasional. SQL memungkinkan pengguna untuk melakukan operasi seperti menambah, mengubah, dan menghapus data; memfilter dan mengaggregasi data; dan membuat dan mengubah struktur data seperti tabel dan relasi. *SQL* biasanya berupa perintah sederhana yang berisi instruksi-instruksi untuk manipulasi data. Perintah *SQL* ini sering juga disingkat dengan sebutan ***'query'***.

SQL adalah bahasa yang kuat dan fleksibel, dan dapat digunakan untuk membuat basis data untuk berbagai jenis aplikasi, seperti aplikasi e-commerce, sistem manajemen inventori, dan sistem manajemen informasi. SQL juga memiliki fitur seperti transaksi, keamanan, dan pemulihan basis data, yang memastikan bahwa data tetap aman dan tersimpan dengan benar.

Dalam rangka menjamin standarisasi, SQL memiliki spesifikasi standar yang diterbitkan oleh ISO dan ANSI. Ini memastikan bahwa SQL yang digunakan oleh berbagai sistem

basis data relasional dapat saling berbicara dan bekerja dengan baik bersama.

11.3. Sejarah *Structure Query Language*

Pada tahun 1970, team yang beranggotakan peneliti IBM Donald D. Chamberlin dan Raymond F. Boyce, mengembangkan *SQUARE* lebih lanjut menjadi *SEQUEL (Structured English Query Language). SEQUEL* digunakan untuk mengoperasikan prototipe RDBMS pertama IBM, System R. Di kemudian hari, *SEQUEL* berubah nama menjadi *SQL* karena permasalahan merk dagang (trademark) dengan sebuah perusahaan pesawat di inggris yang terlebih dahulu telah memakai nama *SEQUEL*.

Pada akhir 1970an, perusahaan Relational Software, Inc. (sekarang Oracle Corporation) melihat potensi bahasa *SQL* dan mengembangkan sendiri versi *SQL* untuk RDBMS mereka. Oracle V2 (versi 2) yang dirilis Juni 1979 adalah RDBMS komersial pertama yang mengimplementasikan *SQL*.

Dengan kemudahan yang ditawarkan, *SQL* mulai diimplementasikan oleh berbagai RDBMS dengan versi *SQL* mereka masing-masing. Namun hal ini menimbulkan permasalahan karena perbedaan penerapan *SQL* dari satu aplikasi dengan aplikasi *database* lainnya yang tidak seragam.Sehingga pada tahun 1986, badan standar amerika, *ANSI (American National Standards Institute)* merancang sebuah standar untuk *SQL*. Satu tahun setelahnya, *ISO (International Organization for*

Standardization) juga mengeluarkan standar untuk *SQL*. Versi terakhir standar *SQL* dirilis pada 2011, yang dinamakan *SQL* 2011. Dengan standar ini diharapkan ada keseragaman *SQL* antar aplikasi RDBMS.

11.4. Data Definition Language

Jenis perintah dasar yang pertama adalah Data Definition Language atau biasa disingkat dengan DDL. DDL adalah singkatan dari Data Definition Language. DDL adalah bagian dari SQL yang digunakan untuk memodelkan dan membuat struktur data dalam basis data relasional. DDL menyediakan perintah untuk membuat, mengubah, dan menghapus tabel, kolom, relasi, dan objek basis data lainnya. Ini juga menyediakan perintah untuk memodifikasi struktur data yang ada, seperti menambah atau menghapus kolom dari tabel, atau mengubah tipe data kolom. Perintah dasar ini sebenarnya merupakan perintah paling mendasar dari bahasa *SQL*. Tujuannya untuk membuat struktur sebuah *database*. Kemudian, perintah dasar DDL masih dibedakan lagi ke dalam setidaknya lima jenis perintah yakni bisa dilihat di bawah ini.

1. Perintah ***Create***: sebuah perintah yang bisa digunakan ketika membuat sebuah *database* yang baru, baik itu berupa tabel baru atau sebuah kolom baru. contoh '*CREATE DATABASE* nama_*database*.

```
CREATE TABLE table_name (
  column1 data_type constraint,
  column2 data_type constraint,
  ...
);
```

2. Perintah ***Alter***: biasa digunakan ketika seseorang ingin mengubah struktur tabel yang sebelumnya sudah ada. Bisa jadi dalam hal ini adalah seperti nama tabel, penambahan kolom, mengubah, maupun menghapus kolom serta menambahkan atribut lainnya.

```
ALTER TABLE table_name
ADD column data_type constraint;
```

3. Perintah ***Rename***: dapat digunakan untuk mengubah sebuah nama di sebuah tabel ataupun kolom yang ada. Bila digunakan perintah ini maka *query*-nya menjadi '*RENAME TABLE* nama_tabel_lama TO nama_tabel_baru"

```
RENAME TABLE old_table_name TO new_table_name;
```

4. Perintah ***Drop*:** Bisa digunakan dalam menghapus baik itu berupa *database*, table maupun kolom hingga index.

```
DROP TABLE table_name;
```

5. Perintah ***TRUNCATE*:** perintah DDL ini digunakan untuk menghapus semua data dalam table tanpa menghapus tabel yang ada.

```
TRUNCATE TABLE table_name;
```

6. Perintah ***Show*:** perintah DDL ini digunakan untuk menampilkan sebuah tabel yang ada.

```
SHOW [object_type];
```

DDL sangat penting dalam proses pengembangan basis data, karena menentukan bagaimana data akan disimpan dan diorganisasi dalam basis data. Dalam rangka menjamin integritas dan konsistensi data, DDL harus digunakan dengan hati-hati dan memperhitungkan berbagai faktor seperti performa, keamanan, dan pemulihan data.

Misalnya ada sebuah kasus yang meminta untuk membuat sebuah table barang dengan urutan sebagai berikut :

1. Buatlah atau defenisikanlah sebuah table barang berikut dengan atribut [kade_B, Nama_B, Harga_B, Jumlah],
2. Tambahkan atribut pemasok/supplier
3. Ganti table barang dengan TB_BRG
4. Hapus semua atribut table barang
5. Tampilkan table barang

6. Hampus table barang

Dari instsruksi tersebut dapat dibuat perintah :

1. Create Tabel_Barang (

   kade_B : char[8]

   Nama_B : char[18]

   Harga_B : numeric

   Jumlah : Numeric)
2. Alter Tabel_Barang

   ADD Suplier : : char[18]
3. Rename Tabel_Barang to TB_BRG
4. Drop TB_BRG
5. Truncate TB_BRG
6. Show (TB_BRG)

11.5. Data Manipulation Language

DML adalah singkatan dari Data Manipulation Language. DML adalah bagian dari SQL yang digunakan untuk memanipulasi data dalam basis data relasional. DML menyediakan perintah untuk menambah, memodifikasi, dan menghapus data dalam tabel. Ini juga menyediakan perintah untuk memfilter dan mengaggregasi data, seperti mencari data yang memenuhi syarat tertentu atau menghitung jumlah data yang sesuai. Dengan adanya Data Manipulation Language (DML). Seperti namanya, perintah dasar *SQL* ini

bertujuan untuk memanipulasi data yang ada dalam sebuah *database*. Perintah dalam DML juga terbagi ke dalam empat jenis. Beberapa di antaranya adalah *insert, select, update, dan delete.*

1. Perintah ***Insert***: penggunaan perintah ini untuk memasukkan sebuah *record* baru di dalam sebuah tabel *database*.

```
INSERT INTO table_name (column1, column2, ...)
VALUES (value1, value2, ...);
```

2. Perintah ***Select***: Pada perintah digunakan untuk menampilkan maupun mengambil sebuah data pada tabel. Data yang diambil pun tidak hanya terbatas pada satu jenis saja melainkan lebih dari satu tabel dengan memakai relasi.

```
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

3. Perintah ***update***: perintah ini digunakan ketika ingin melakukan pembaruan data di sebuah tabel. Contohnya saja jika ada kesalahan ketika memasukkan sebuah *record*. Kita tidak perlu menghapusnya dan bisa diperbaiki menggunakan perintah ini.

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

4. Perintah ***Delete***: Perintah DML ini dapat digunakan ketika kita ingin menghapus sebuah *record* yang ada dalam sebuah tabel.

```
DELETE FROM table_name
WHERE condition;
```

DML sangat penting dalam proses pengembangan basis data, karena memungkinkan pengembang untuk memanipulasi data dalam basis data dan memastikan bahwa data terus tersimpan dengan benar dan dapat diakses dengan mudah. Dalam rangka menjamin integritas dan konsistensi data, DML harus digunakan dengan hati-hati dan memperhitungkan berbagai faktor seperti performa, keamanan, dan pemulihan data.

Berikut ada beberapa contoh penggunaan perintah SQL dalam bentuk *data manipulation language* :

1. Tambahkan data barang dengan kode B001, untuk sembako beras dengan harga 10000 untuk jumlah stok 50 kilo.
2. Cari berapa harga beras B001
3. Robah harga beras menjadi 12000 per kilo
4. Hapus data barang B005.

Dari instruksi diatas dapat menggunakan perintah DML sebagai berikut :

1. Insert into Tb_barang (Kode_B, Nama_B, Hrg_B, Jumlah)

   Value (B001, B eras, 10000, 50)

2. Select Hrg_B

   from Tb_barang

   whrene Kode_B = "B001"

3. Update Tb_barang

   Set Hrg_B = 12000

   Where Kode_B = "B001"

4. Delate form Tb_Barang

   Where Kode_B = "B005"

Apabilah telah terbiasa menggunakan perintah SQL maka kita akan mudah menggunakan bahasa sehari hari kedalam bentuk data manipulasi DDL yang di kombinasikan dengan fungsi agregate. misalnya

Cari jumlah mahasiswa yang belum membayar perkuliahan.

Maka kita dapat menggunakan perintah :

Select count nama_mahasiswa

From table_pembayaran

Where pembayaran = 0

Dari pernyataan diatas maka muncul jumlah mahasisw yang belum melakukan pembayaran. Begitu juga jika kita ingin mencari nilai rata rata , nilai maksimum dan minimum, dengan menambahkan perintah fungsi agregasi setelah perintah select.

11.6. Rangkuman

SQL (Structured Query Language) *adalah bahasa pemrograman khusus yang digunakan untuk memanajemen data dalam* RDBMS. *SQL* biasanya berupa perintah sederhana yang berisi instruksi-instruksi untuk manipulasi data.

Dalam *SQL* terdapat dua kelompok perintah dasar yaitu DDL atau *Data Definition Language* dan DML atau *Data Manipulation Language*.

Data Definition Language biasa disingkat dengan DDL. Perintah dasar ini sebenarnya merupakan perintah paling mendasar dari bahasa *SQL*. Tujuannya untuk membuat struktur sebuah *database*. Kemudian, perintah dasar DDL masih dibedakan lagi ke dalam setidaknya lima jenis perintah yakni bisa dilihat di bawah ini.

1. Perintah ***Create***: sebuah perintah yang bisa digunakan ketika membuat sebuah *database* yang baru, baik itu berupa tabel baru atau sebuah kolom baru. contoh '*CREATE DATABASE* nama_*database*.
2. Perintah ***Alter***: biasa digunakan ketika seseorang ingin mengubah struktur tabel yang sebelumnya sudah ada.

Bisa jadi dalam hal ini adalah seperti nama tabel, penambahan kolom, mengubah, maupun menghapus kolom serta menambahkan atribut lainnya.

3. Perintah ***Rename***: dapat digunakan untuk mengubah sebuah nama di sebuah tabel ataupun kolom yang ada. Bila digunakan perintah ini maka *query*-nya menjadi 'RENAME TABLE nama_tabel_lama TO nama_tabel_baru''
4. Perintah ***Drop*:** Bisa digunakan dalam menghapus baik itu berupa *database*, tabel maupun kolom hingga *index*.
5. Perintah ***Show*:** perintah DDL ini digunakan untuk menampilkan sebuah tabel yang ada.

Dengan adanya Data Manipulation Language (DML). Seperti namanya, perintah dasar *SQL* ini bertujuan untuk memanipulasi data yang ada dalam sebuah *database*. Perintah dalam DML juga terbagi ke dalam empat jenis. Beberapa di antaranya adalah *insert, select, update, dan delete.*

1. Perintah ***Insert***: penggunaan perintah ini untuk memasukkan sebuah *record* baru di dalam sebuah tabel *database*.
2. Perintah ***Select***: Pada perintah digunakan untuk menampilkan maupun mengambil sebuah data pada tabel. Data yang diambil pun tidak hanya terbatas pada satu jenis saja melainkan lebih dari satu tabel dengan memakai relasi.

3. Perintah ***update***: perintah ini digunakan ketika ingin melakukan pembaruan data di sebuah tabel. Contohnya saja jika ada kesalahan ketika memasukkan sebuah record. Kita tidak perlu menghapusnya dan bisa diperbaiki menggunakan perintah ini.
4. Perintah ***Delete***: Perintah DML ini dapat digunakan ketika kita ingin menghapus sebuah record yang ada dalam sebuah tabel.

11.7. Bahan Diskusi

Berikut adalah beberapa Bahan Diskusi untuk mahasiswa agar mempunuyai pemahaman yang lebih baik tentang SQL:

1. Struktur dasar SQL: membahas tentang bagaimana membuat database, tabel, dan query dasar dalam SQL.
2. Operasi CRUD (Create, Read, Update, Delete): membahas tentang bagaimana melakukan operasi CRUD pada data dalam database SQL.
3. Fungsi dan operator SQL: membahas tentang fungsi-fungsi dan operator-operator yang tersedia dalam SQL dan bagaimana menggunakannya untuk mengolah data.
4. Join pada tabel: membahas tentang bagaimana menggabungkan data dari beberapa tabel menjadi satu dengan menggunakan join.

5. Subquery dan kueri anggota: membahas tentang bagaimana membuat subquery dan kueri anggota dalam SQL untuk mengambil data dari tabel yang berbeda.
6. Optimisasi kueri SQL: membahas tentang bagaimana mengoptimalkan kinerja kueri SQL agar lebih cepat dan efisien.
7. Transaksi dan keamanan data: membahas tentang bagaimana melakukan transaksi dan menjaga keamanan data dalam database SQL.

BAB XII
NOT ONLY STRUCTURE QUERY LANGUAGE

12.1. Pengantar

NoSQL adalah singkatan dari "Not Only SQL", yang mengacu pada model basis data yang tidak mengikuti model basis data relasional yang tradisional. Ini berbeda dari model basis data relasional yang menggunakan SQL karena tidak memiliki struktur tabel yang tetap dan memungkinkan untuk menyimpan data dalam bentuk yang tidak terstruktur, seperti dokumen, grafik, atau key-value.

NoSQL memiliki beberapa keuntungan dibandingkan dengan model basis data relasional tradisional:

1. Skalabilitas: Beberapa implementasi NoSQL dapat dikelola secara horizontal (menambah lebih banyak node ke sistem) untuk menangani tingkat peningkatan trafik yang tinggi.
2. Flexibilitas: NoSQL memungkinkan untuk menyimpan data dalam bentuk yang tidak terstruktur dan tidak memiliki aturan integritas yang ketat, sehingga mudah untuk memodelkan data yang berubah dan tidak terstruktur.

3. Performa: Beberapa implementasi NoSQL dapat menangani tingkat akses yang tinggi dan menyediakan waktu respon yang cepat untuk aplikasi real-time.
4. Biaya: Beberapa implementasi NoSQL memiliki biaya perangkat keras dan software yang lebih rendah dibandingkan dengan model basis data relasional yang tradisional.

Meskipun memiliki beberapa keuntungan, NoSQL juga memiliki beberapa kelemahan, seperti keterbatasan dalam hal pemrograman transaksi dan ketergantungan pada aplikasi untuk menangani integritas data. Oleh karena itu, pemilihan antara model basis data relasional dan NoSQL harus didasarkan pada kebutuhan aplikasi dan jenis data yang akan disimpan.

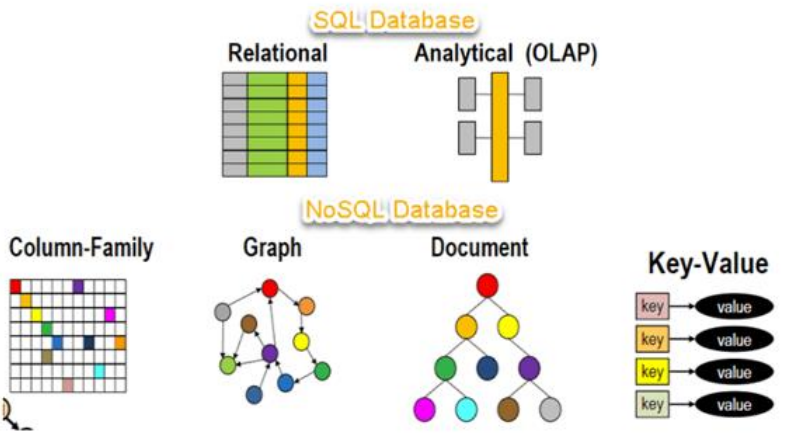
12.2. Pengertian *Not Only Structure Query Language*

Selain *SQL* ada juga istilah *NoSQL* atau *Not Only SQL* yang sudah berkembang sejak tahun 1960 an, namum baru menjadi perhatian sejak tahun 2000 an, semenjak meningkatnya kebutuhan data skala besar. *NoSQL* adalah sebuah sistem manajemen data non-relasional yang tidak memerlukan skema tetap. Tujuan utama dari penggunaan *database NoSQL* adalah untuk penyimpanan data yang terdistribusi dengan kebutuhan penyimpanan data yang besar. Perusahaan yang memerlukan data dalam volume yang besar akan memilih menggunakan *NoSQL* untuk mengumpulkan datanya. Umumnya *NoSQL* digunakan untuk penggunaan *big data* dan

aplikasi web yang *real-time*. Perusahaan seperti Twitter, Facebook, hingga Google yang pasti akan mengumpulkan data penggunanya dalam jumlah yang sangat besar setiap harinya

12.3. Jenis *Database NoSQL*

Berikut dapat di lihat perbedaan antara *database SQL* dengan *database NoSQL* dari gambar berikut :



Gambar 12.1. Perbedaan *database SQL* dengan *database NoSQL*

Dari gambar diatas untuk *database SQL* dapat berbentuk r*elational* dan *Analitical* dengan data yang sudah terstruktur sedangkan *database NoSQL* mempunyai 4 jenis *database* yaitu :

1. *Column based database* : memberikan banyak fleksibilitas daripada *database* yang relasional karena setiap baris tidak diharuskan memiliki kolom yang sama. Setiap kolom dibuat secara terpisah dan nilai

dalam *database* kolom tunggal disimpan secara berdekatan. Jenis *database* ini memberikan kinerja tinggi pada *aggregation queries* seperti *SUM, Count, AVG*, hingga *MIN* karena datanya sudah tersedia di kolom. *Column based database* ini banyak digunakan untuk mengelola *data warehouse, business intelligence*, hingga *CRM.*

2. *Graph database* : menyimpan data dalam *node* dan e*dge*. *Node* biasanya menyimpan informasi tentang orang, tempat, dan benda-benda. Sementara itu, *edge* menyimpan informasi tentang hubungan antar *node*. Jenis *database* yang satu ini lebih unggul dalam penggunaan di mana pengguna perlu mencari tahu hubungan atau pola. Misalnya untuk *social network*, deteksi penipuan, logistik hingga rekomendasi.

3. *Document database* : menyimpan data dalam dokumen yang mirip dengan objek *JSON (JavaScript Object Notation)*. **D**ari *document database* ini lebih efisien dan fleksibel dan program akan lebih mudah dikembangkan karena *document database* akan menyesuaikan penyimpanan data berdasarkan kebutuhan program. *database* ini sangat cocok digunakan untuk *database* yang bertujuan umum. Selain itu, *document database* juga mampu mengakomodasi volume data yang besar.

4. *Key-value dataabase* : Jenis *database* ini lebih sederhana karena setiap item berisi key dan value sebagai tempat

akses data. Sebuah value atau nilai biasanya hanya diambil dengan mereferensikan dari key atau kuncinya. cara membuat queri untuk *key-value* tertentu bisa lebih sederhana. *Key-value database* ini lebih sesuai untuk penyimpanan data dalam jumlah besar yang tidak perlu kueri yang rumit untuk mengambilnya.

12.4. Kelebihan dan Kekurangan *NoSQL*

Setelah mengetahui pengertian dari *NoSQL* dan bagian, dilanjutkan dengan mengetahui kelebihan dan kekurangan *NoSQL* sebagai berikut :

1. Kelebihan *NoSQL*.

   Berikut ini yang menjadi kelebihan *NoSQL* dari berbagai aspek meliputi :

   a. Performa : Memiliki performa yang lebih unggul dari pada *SQL* karena semua informasi dapat dalam satu *database* saja, sedangkan *database SQL* harus menggunakan Query denga berbagai table yang harus didefenisikan terlebih dahulu. *Database NoSQL* dapat melakukan puluhan ribu Query dalam 1 detik.

   b. Skalabilitas : Menggunakan penskalaan horizotal yang lebih mudah diterapkan dalam berbagai aplikasi.

   c. Fleksibilitas : Lebih fleksibel sehingga lebih mudah dilakukan untuk pembaruan *database* tanpa harus merubah struktu data.

2. Kekurangan *NoSQL*

Berikut yang menjadi kekurangan *NoSQL* dari berbagai aspek sebagai berikut :

a. Membutuhkan banyak *database* : Diperlukan berbagai *database* dan model data untuk menggunakannya, bahkan diperlukan bebarapa bentuk *SQL* untuk memudahkan mempersingkat prosesnya.

b. Ukuran *database* bisa lebih besar: *NoSQL* tidak dirancang untuk menghapus data duplikasi, Hal itu membutuhkan lebih banyak tempat penyimpanan data untuk dipersiapkan jika ingin menggunakan *NoSQL*.

c. Pengelolaan tidak mudah : Mengelola data dalam jumlah sangat besar bukanlah hal yang mudah. Itulah mengapa pengelolaan data di *NoSQL* menjadi lebih kompleks. Meskipun tujuan menggunakan *NoSQL* adalah untuk mengelola data dalam jumlah besar menjadi lebih sesederhana mungkin, tapi hal tersebut bukanlah hal yang mudah. Karena itu, dalam mengelola hal yang satu ini dibutuhkan lebih banyak usaha ekstra karena memang sukup sulit.

12.5. Pengenalan *Database MongoDB*

Database mongoDB merupakan salah satu aplikasi *NoSQL* yang berbasis Javascript. Didalam *MongoDB* konsep tabel

sebagai tempat penyimpanan data yang biasa dipakai dalam *SQL* di nyatakan sebagai *Collection*. Semua *Collection* disimpan dalam *database*. MongoDB adalah sebuah sistem manajemen basis data NoSQL yang dikenal dengan basis data dokumentasi. Ini berarti bahwa data disimpan dalam bentuk dokumen JSON (JavaScript Object Notation), bukan tabel seperti yang ditemukan dalam basis data relasional.

MongoDB sangat fleksibel dan mudah digunakan, karena tidak memerlukan pemahaman tentang skema terlebih dahulu. Skema dapat diterapkan seiring waktu, seiring dengan tumbuhnya kebutuhan aplikasi. MongoDB juga menawarkan skalabilitas horizontal melalui teknik sharding, yang memungkinkan untuk mengelola database yang sangat besar dan mengelola beban yang sangat tinggi.

Dengan MongoDB, aplikasi dapat dengan mudah menyimpan dan mengambil data yang kompleks, seperti array, nested document, dan lainnya. MongoDB juga menyediakan banyak fitur bantuan seperti indexing, pencarian, dan agregasi yang membantu developer membuat aplikasi yang kuat dan tangguh.

Dalam MongoDB, koleksi seperti tabel dalam basis data relasional, dan dokumen adalah baris dalam tabel. Koleksi dapat berisi jumlah dokumen yang tidak terbatas, dan setiap dokumen dapat memiliki atribut yang berbeda.

Secara umum, MongoDB memiliki keuntungan seperti skalabilitas, fleksibilitas, dan kecepatan yang membuatnya menjadi pilihan populer bagi aplikasi web modern.

1. Perintah *mongoDB*.

   Berikut beberapa perintah yang biasa digunakan dalam *database mongoDB* :

   a. Membuat *database* : Untuk membuat *database* tidak perlu di defenisikan, karena *database* secara otomatis dengan memanggil *database* yang kita pilih. Untuk memilih *database* bisa digunakan perinta "*use* nama_*database*"

Gambar 12.2. Memilih *Database*

   b. Metode *database* : Ada berapa perintah manipulasi *database* dalam *database mongoDB* seperti menghapus *database*, mengambil informasi databe, informasi versi *database* dan lainya. Berukut beberapa jenis metode *database*:

1) *Db.dropDatabase ()* : digunakan untuk menghapus *database*

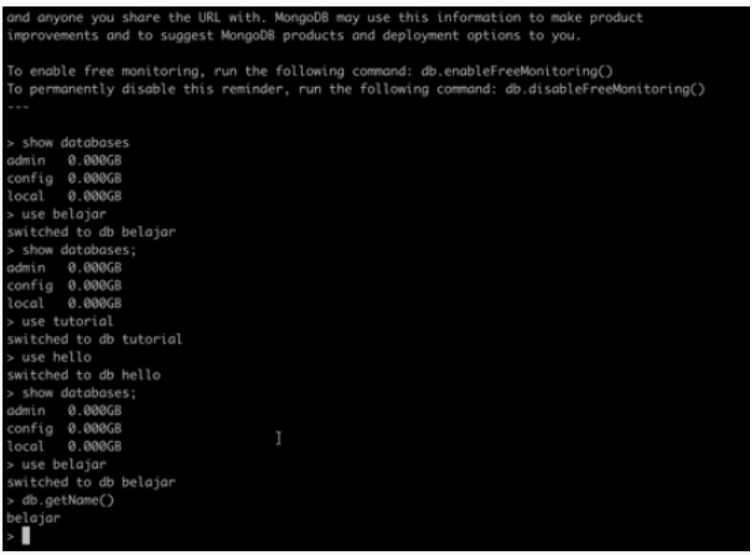
2) *Db.getName()* : digunakan untuk mengambil nama *database*

3) *Db.hostinfor()* : digunakan untuk mengambil informasi *host* tempat *mongoDB*.

4) *Db.version()* : digunakan untuk mengambil versi *database*.

5) *Db.stat()* : digunakan untuk mengambil statstik pengguna *database*.

Berikut contoh implementasi penggunaan metode *database*



Gambar 12.3. Penggunaan *Database*

2. *Collection*.

   Adalah tempat penyimpanan dokumen, maksimum per dokumen dapat disimpan 16 MB. Maksimum level *nested* per dokumen dapat menyimpan 100 level. Ada beberapa fungsi dalam dalam *database* untuk manipulasi *Collection* sebagai berikut:

   a. Db.getCollection.Names() : digunakan untuk mengambil semua nama Collection
   b. Db.createCollection(name) : digunakan untuk membuat Collection baru
   c. Db.getCollection(name) : digunakan untuk mendapatkan objek Collection
   d. Db.Name : sama dengan perintah Db.getCollection(name)
   e. Db.getCollection.Infos() : digunakan untuk mendapatkan informasi semua Collection
   f. Db.<Collection>.find() : digunakan untuk mengambil semua dokumen
   g. Db.<Collection>.count() : digunakan untuk mengambil jumlah dokumen
   h. Db.<Collection>.drop() : digunakan untuk menghapus Collection
   i. Db.<Collection>.totaSize() : digunakan untuk mengambil ukuran Collection

j. Db.<Collection>.stats() : digunakan untuk informasi statistic Collection

3. Data Model

Data modeling dalam MongoDB adalah proses untuk menentukan bagaimana data harus ditempatkan dan disimpan dalam basis data untuk memenuhi kebutuhan aplikasi. Dalam MongoDB, data ditempatkan dalam dokumen, bukan tabel seperti dalam basis data relasional.

Berikut adalah beberapa konsep data modeling yang umum dalam MongoDB:

a. Dokumen Embedded: Dokumen yang menyimpan data yang saling berhubungan dalam satu dokumen. Ini membuat akses data lebih cepat dan efisien, tetapi juga memiliki batasan skalabilitas.

b. Dokumen Terpisah: Dokumen yang menyimpan data yang saling berhubungan dalam koleksi yang berbeda. Ini memungkinkan skalabilitas yang lebih baik, tetapi memerlukan lebih banyak query untuk mengambil data yang saling berhubungan.

c. Referensi Dokumen: Menyimpan ID dari dokumen lain sebagai atribut dalam dokumen. Ini memungkinkan Anda untuk mengacu pada dokumen lain dengan mudah.

d. Denormalisasi: Menyimpan data yang sama dalam beberapa dokumen untuk mengurangi jumlah query yang diperlukan untuk mengambil data.

e. Normalisasi: Menyimpan data yang saling berhubungan dalam dokumen terpisah untuk mengurangi duplikasi data dan membuat basis data lebih mudah dikelola.

Penting untuk mempertimbangkan kebutuhan aplikasi dan memilih model data yang sesuai. Beberapa aplikasi mungkin membutuhkan dokumen embedded, sementara yang lain mungkin membutuhkan referensi dokumen atau denormalisasi. Ada beberapa faktor yang perlu dipertimbangkan, seperti skalabilitas, performa, dan pemeliharaan, dalam memilih model data yang tepat.

Pemahaman data model sangat penting sekali kerena ada perbedaan konsep relasional *database* dengan dokumen *database* yaitu perpindahan penggunaan tabel ke penggunaan *Collection*. Pada saat memodelkan *database* relasional mengacu kepada hasil normalisasi sedangkan untuk memodelkan dokumen *database* mengacu kepada penggunaan aplikasi kepada *query, update* dan memproses data.

Didalam mongoDB kita dapat langsung memasukkan data Collection, sehingga schema pada mongoDB sangat fleksibel tiap dokumen bisa berbeda

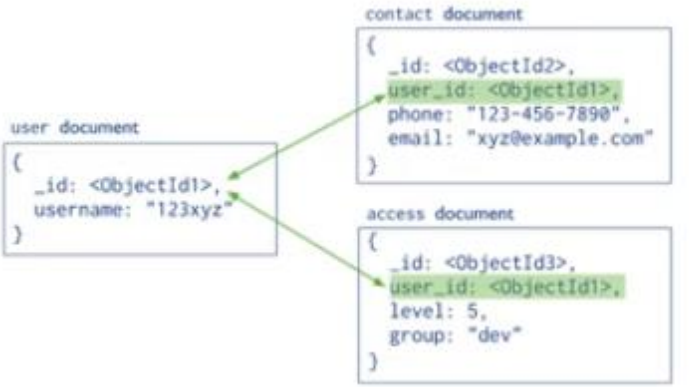
bisa masuk tampa mengharuskan membuat struktur data dulu yang harus sama pada setiap record. Namun pada prakteknya di sarankan memasukkan data dengan tipe yang sama, meskipun bisa menampung data yang berbeda. Yang paling penting di dalam mongoDB semua harus memiliki primary key hanya untuk 1 field saja , sehingga untuk mengakalinya kita bisa menggunakan 2 data untuk 1 field. Berikut ada 2 struktur dokumen dalam mangodb yaitu :

a. Struktur dokumen *embedded* yaitu struktur dokume bersarang dimana dalam *Collection* terdapat dokumen bersarang seperti pada gamber dibawah ini.

Gambar 12.4. Struktur Dokumen *Embedded*

b. Struktur dokumen *reterence* yaitu struktur dokumen dalam satu collectin mempunyai koneksi referensi

dengan dokumen lain. Dapat dilihat pada gambar dibawah ini.



Gambar 12.5. Struktur Dokumen *Reference*

Struktur dokumen embedded digunakan jika :

a. Antar dokumen saling ketergantungan
b. Tidak bisa langsung melakukan perubahan ke *embedded* dokumen
c. *Embedded* dokumen selalu dibutuhkan Ketika mengambil data dokumen

Sedangkan penggunaan dokumen *reference* sebagai berikut :

a. Antar dokumen bisa berdiri sendiri dan tidak terlalu ketergantungan satu sama lain.
b. Bisa melakukan manipulasi data langsung terhadap dokumen *reference*

c. Dokumen *reference* tidak selalu dibutuhkan saat mengambil dokumen.

4. *BSON*

BSON adalah singkatan dari *Binary* JSON yaitu *binary-encoded serialization* dokumen seperti JSON yang juga bisa menggunakan *embedded object, array* dan lain-lain. BSON menawarkan fleksibilitas dan lebih kecil dari JSON standar, membuat MongoDB lebih cepat dan lebih efisien saat menyimpan dan membaca data.

BSON mendukung beberapa tipe data yang berbeda, termasuk string, angka, tanggal, dan array. Ini membuat BSON sangat cocok untuk menyimpan data semi-terstruktur yang biasa ditemukan dalam aplikasi modern.

Di MongoDB, setiap dokumen disimpan sebagai catatan BSON individual. Dokumen ini menyimpan data sebagai *key-value pair*, dimana *key* adalah nama atribut dan *value* adalah nilai atribut. BSON juga mendukung konsep penyematan dan referensi, yang memungkinkan kita menyimpan informasi terkait dalam satu dokumen atau mereferensikan dokumen lain.

Dengan BSON, MongoDB membuat pengelolaan data menjadi fleksibel, cepat, dan efisien. Kita dapat menggunakan BSON untuk membuat model data yang sesuai dengan kebutuhan aplikasi, menyimpan dan

membaca data dengan cepat, dan menyimpan data semi-terstruktur dengan mudah.

Berikut tipe data BSON yang biasa digunakan :

a. *Double* di implementasikan dengan *doubel*
b. *String* di implementasikan dengan *string*
c. *Object* di implementasikan denga *object*
d. *Array* di implementasikan dengan *array*
e. *Bianary Data* di implementasikan dengan *bindata*
f. *Objectld* di implementasikan dengan *objectld*
g. *32bit integer* di implementasikan dengan *int*
h. *Timestamp* di implementasiakan dengan *timestamp*
i. *64bit integer* di implementasikan dengan *long*
j. *Decimal 128* di impelmentaskan dengan *decimal*
k. *Min Key* di implementasikan denga *minkey*
l. *Max Key* di implementasikan dengan *maxkey*

Objectld adalah random data yang unik, cepat untuk di *generate*, dan terurut memiliki ukuran panjang 12 *byte*, konsisten terdiri dari 4 *byte* timestamp, 4 *byte random value*, dan 3 *byte incrementing* counter. *Objectld* digunakan sebagai *default_id(primary key)* di dokumen jika kita tidak secara ekplisit menyebutkan id dokumennya.

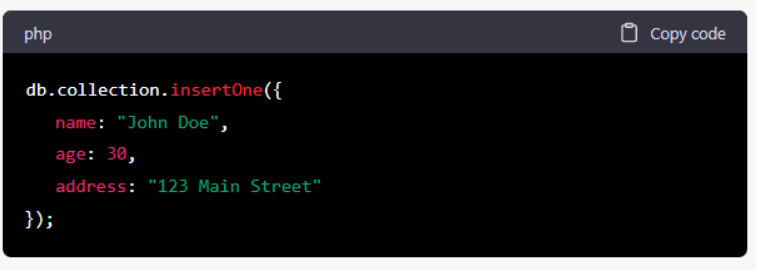
Data dan *ISOdate*.

BSONDate adalah 64 *bit integer* yang merepresentasikan angak milisecon sejak *unix epoch,* nilai ini bisa merepresentasikan waktu dengan jarak 290 juta tahun sebelum dan sesudah *unix epoch* (1 januari 1970). *ISODate* merupakan representasi waktu yang digunakan oleh *mongoDB*. Date ini kompatibel dengan *Java Script*.

5. *Insert Document*

Untuk menambahkan dokumen ke dalam MongoDB, Anda dapat menggunakan metode "insertOne()" atau "insertMany()".

Metode "insertOne()" digunakan untuk menambahkan satu dokumen ke dalam koleksi. Contohnya seperti ini:



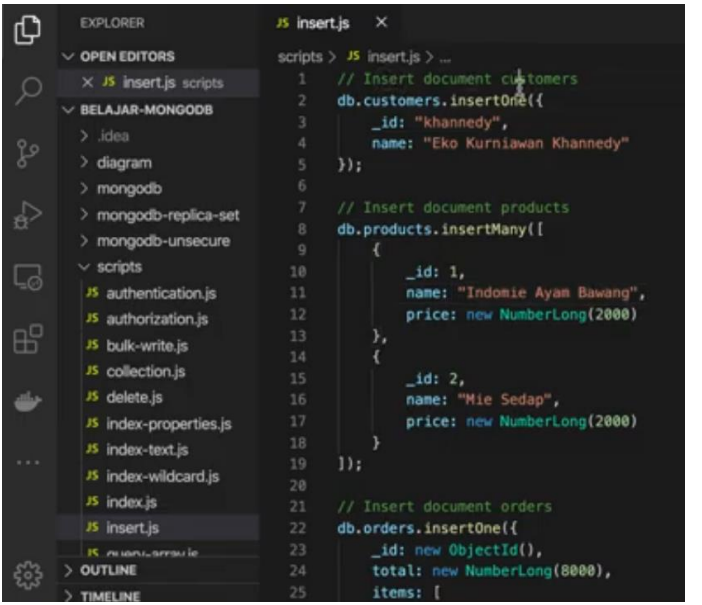
Pastikan bahwa kita sudah terhubung ke database MongoDB dan koleksi yang sesuai sebelum menjalankan perintah ini. Kita juga dapat memverifikasi bahwa dokumen berhasil ditambahkan dengan menjalankan perintah "find()" pada koleksi tersebut.

Metode "insertMany()" digunakan untuk menambahkan beberapa dokumen sekaligus ke dalam koleksi. Contohnya seperti ini:

Untuk menyimpan data kedalam *mongoDB*, kita perlu membuat dokumen dalam bentuk *JSON*, dimana *Field_id* tidak wajib dimasukkan, jika kita tidak memasukkan *field_id* maka secara otomatis *MongoDB* akan membuat id baru secara random dengan tipe data *ObjectId*. Atau kita juga secara eksplisit membuat *ObjectId* baru dengan menggunakan perintah *"new ObjectId()"*. Berikut bentuk *insert document function* adalah :

- Db<Collection>insertOne(document): menambah dokumen ke Collection.
- Db<Collection>insertMany(array<document>): menambah semua dokumen di array ke Collection.

- Db<Collection>insertOne(document/array): menambah satu dokumen atau banyak ke Collection.
- Berikut ini bentuk kode program untuk menambah data



Gambar 12.6. Program menambahikan data

12.6. Rangkuman

NoSQL adalah sebuah sistem manajemen data non-relasional yang tidak memerlukan skema tetap. Tujuan utama dari penggunaan *database NoSQL* adalah untuk penyimpanan data yang terdistribusi dengan kebutuhan penyimpanan data yang besar. Perusahaan seperti Twitter, Facebook, hingga Google yang pasti akan mengumpulkan data penggunanya dalam jumlah yang sangat besar setiap harinya.

Pada *database SQL* dapat berbentuk r*elational* dan *Analitical* dengan data yang sudah terstruktur sedangkan *database NoSQL* mempunyai 4 jenis *database* yaitu :

1. *Column based database* : memberikan banyak fleksibilitas karena setiap baris tidak diharuskan memiliki kolom yang sama. Setiap kolom dibuat secara terpisah dan nilai dalam *database* kolom tunggal disimpan secara berdekatan.
2. *Graph database* : menyimpan data dalam *node* dan e*dge*. *Node* biasanya menyimpan informasi tentang orang, tempat, dan benda-benda. Sementara itu, *edge* menyimpan informasi tentang hubungan antar *node*. Jenis *database* yang satu ini lebih unggul dalam penggunaan di mana pengguna perlu mencari tahu hubungan atau pola. Misalnya untuk *social network*, deteksi penipuan, logistik hingga rekomendasi.
3. *Document database* : menyimpan data dalam dokumen yang mirip dengan objek *JSON (JavaScript Object Notation)*. Dari *document database* ini lebih efisien dan fleksibel dan program akan lebih mudah dikembangkan karena *document database* akan menyesuaikan penyimpanan data berdasarkan kebutuhan program.
4. *Key-value database* : Jenis *database* ini lebih sederhana karena setiap item berisi key dan value sebagai tempat akses data.

Kelebihan dan kekurangan *NoSQL* sebagai berikut :

1. Kelebihan *NoSQL*.

   Berikut ini yang menjadi kelebihan *NoSQL* dari berbagai aspek meliputi :

   a. Memiliki performa yang lebih unggul dari pada *SQL* karena semua informasi dapat dalam satu *database* saja, sedangkan *database SQL* harus menggunakan Query denga berbagai table yang harus didefenisikan terlebih dahulu. *Database NoSQL* dapat melakukan puluhan ribu Query dalam 1 detik.

   b. Menggunakan penskalaan horizotal yang lebih mudah diterapkan dalam berbagai aplikasi.

   c. Lebih fleksibel sehingga lebih mudah dilakukan untuk pembaruan *database* tanpa harus merubah struktu data.

2. Kekurangan *NoSQL*

   Berikut yang menjadi kekurangan *NoSQL* dari berbagai aspek sebagai berikut :

   a. Membutuhkan banyak *database* : Diperlukan berbagai *database* dan model data untuk menggunakannya, bahkan diperlukan bebarapa bentuk *SQL* untuk memudahkan mempersingkat prosesnya.

   b. Ukuran *database* bisa lebih besar: *NoSQL* tidak dirancang untuk menghapus data duplikasi, Hal itu

membutuhkan lebih banyak tempat penyimpanan data untuk dipersiapkan jika ingin menggunakan *NoSQL*.

c. Pengelolaan tidak mudah : Mengelola data dalam jumlah sangat besar bukanlah hal yang mudah. Itulah mengapa pengelolaan data di *NoSQL* menjadi lebih kompleks. Meskipun tujuan menggunakan *NoSQL* adalah untuk mengelola data dalam jumlah besar menjadi lebih sesederhana mungkin, tapi hal tersebut bukanlah hal yang mudah. Karena itu, dalam mengelola hal yang satu ini dibutuhkan lebih banyak usaha ekstra karena memang sukup sulit.

Database mongoDB merupakan salah satu aplikasi *NoSQL* yang berbasis Javascript. Didalam *MongoDB* konsep tabel sebagai tempat penyimpanan data yang biasa dipakai dalam *SQL* di nyatakan sebagai *Collection*. Semua *Collection* disimpan dalam *database.* Berikut beberapa perintah yang biasa digunakan dalam *database mongoDB* :

a. Membuat *database* : Untuk membuat *database* tidak perlu di defenisikan, karena *database* secara otomatis dengan memanggil *database* yang kita pilih. Untuk memilih *database* bisa digunakan perinta "*use* nama_*database*"

b. Metode *database* : Ada berapa perintah manipulasi *database* dalam *database mongoDB* seperti menghapus *database*, mengambil informasi databe, informasi versi

database dan lainya. Berukut beberapa jenis metode *database*:

1) *Db.dropDatabas ()* : digunakan untuk menghapus *database*
2) *Db.getName()* : digunakan untuk mengambil nama *database*
3) *Db.hostinfor()* : digunakan untuk mengambil informasi *host* tempat *mongoDB*.
4) *Db.version()* : digunakan untuk mengambil versi *database*.
5) *Db.stat()* : digunakan untuk mengambil statstik pengguna *database*.

c. Collection adalah tempat penyimpanan dokumen, maksimum per dokumen dapat disimpan 16 MB. Maksimum level nested per dokumen dapat menyimpan 100 level. Ada beberapa fungsi dalam dalam database untuk manipulasi Collection sebagai berikut :

1) Db.getCollection.Names() : digunakan untuk mengambil semua nama Collection
2) Db.createCollection(name) : digunakan untuk membuat Collection baru
3) Db.getCollection(name) : digunakan untuk mendapatkan objek Collection
4) Db.Name : sama dengan perintah Db.getCollection(name)

5) Db.getCollection.Infos() : digunakan untuk mendapatkan informasi semua Collection

6) Db.<Collection>.find() : digunakan untuk mengambil semua dokumen

7) Db.<Collection>.count() : digunakan untuk mengambil jumlah dokumen

8) Db.<Collection>.drop() : digunakan untuk menghapus Collection

9) Db.<Collection>.totaSize() : digunakan untuk mengambil ukuran Collection

10) Db.<Collection>.stats() : digunakan untuk informasi statistic Collection

Pemahaman data model sangat penting sekali kerena ada perbedaan konsep relasional *database* dengan dokumen *database* yaitu perpindahan penggunaan tabel ke penggunaan *Collection*. Didalam *mongoDB* kita dapat langsung memasukkan data *Collection*, sehingga *schema* pada *mongoDB* sangat fleksibel tiap dokumen bisa berbeda bisa masuk tampa mengharuskan membuat struktur data dulu yang harus sama pada setiap *record*. Namun pada prakteknya di sarankan memasukkan data dengan tipe yang sama, meskipun bisa menampung data yang berbeda. Yang paling penting di dalam *mongoDB* semua harus memiliki primary key hanya untuk 1 *field* saja, sehingga untuk mengakalinya kita bisa mengkgunakan 2 data untuk 1 *field*. Berikut ada 2 struktur dokumen dalam mangodb yaitu :

a. Struktur dokumen *embedded* yaitu struktur dokume bersarang dimana dalam *Collection* terdapat dokumen bersarang seperti pada gamber dibawah ini.

b. Struktur dokumen *reterence* yaitu struktur dokumen dalam satu collectin mempunyai koneksi referensi dengan dokumen lain. Dapat dilihat pada gambar dibawah ini.

BSON adalah singkatan dari *Binary JSON* yaitu *binary-encoded serialization* dokumen seperti *JSON* yang juga bisa menggunakan *embedded object, array* dan lain-lain. Berikut tipe data *BSON* yang biasa digunakan :

a. *Double* di implementasikan dengan *doubel*

b. *String* di implementasikan dengan *string*

c. *Object* di implementasikan denga *object*

d. *Array* di implementasikan dengan *array*

e. *Bianary Data* di implementasikan dengan *bindata*

f. *Objectld* di implementasikan dengan *objectld*

g. *32bit integer* di implementasikan dengan *int*

h. *Timestamp* di implementasiakan dengan *timestamp*

i. *64bit integer* di implementasikan dengan *long*

j. *Decimal 128* di impelmentaskan dengan *decimal*

k. *Min Key* di implementasikan denga *minkey*

l. *Max Key* di implementasikan dengan *maxkey*

12.7. Bahan Diskusi

Berikut adalah beberapa Bahan Diskusi tentang NoSQL:

1. Pengertian NoSQL: membahas tentang apa itu NoSQL dan bagaimana ia berbeda dari database relasional (SQL).
2. Jenis-jenis NoSQL: membahas tentang jenis-jenis NoSQL seperti document-oriented, key-value, columnar, dan graph databases.
3. Kelebihan dan kekurangan NoSQL: membahas tentang kelebihan dan kekurangan dari menggunakan NoSQL dibandingkan dengan database relasional.
4. Model data NoSQL: membahas tentang bagaimana menyimpan dan mengolah data dalam NoSQL, termasuk bagaimana mengatasi masalah seperti normalisasi dan integrity.
5. Integrasi NoSQL dengan teknologi lain: membahas tentang bagaimana NoSQL dapat digabungkan dengan teknologi lain seperti Hadoop, Spark, dan Cloud Computing.
6. Keamanan data dalam NoSQL: membahas tentang bagaimana menjaga keamanan data dalam NoSQL, termasuk enkripsi, autentikasi, dan authorization.

DAFTAR PUSTAKA

Dantes, Gede Rasben, dkk. 2019. Pengantar Basis Data. Depok : Penerbit PT.Raja Grafindo Persada.

Elmasri, R., & Navathe, S. B. (2006). Fundamentals of Database Systems. Addison-Wesley.

Fathansyah, 2015. Sistem Basis Data, Penerbit Informatika

Hernandez, M. J. (2019). Database Design for Mere Mortals: A Hands-On Guide to Relational Database Design. Addison-Wesley.

Ramakrishnan, R., & Gehrke, J. (2003). Database Management Systems. McGraw-Hill.

Schwartz, B., Zaitsev, P., & Tkachenko, V. (2012). High Performance MySQL: Optimization, Backups, and Replication. O'Reilly Media.

White, T. (2015). Hadoop: The Definitive Guide. O'Reilly Media.

Winand, M. (2014). SQL Performance Explained. Markus Winand.

Date, C. J. (2009). A Guide to the SQL Standard. Addison-Wesley.

Chodorow, K., & Dirolf, M. (2010). MongoDB: The Definitive Guide. O'Reilly Media.

Bryla, B., & Loney, K. (2015). Oracle Database 12c: The Complete Reference. McGraw-Hill Education.

Fowler, M., & Sadalage, P. J. (2012). NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence. Addison-Wesley.

Hernandez, M. J. (2003). Database Design for Mere Mortals: A Hands-On Guide to Relational Database Design. Addison-Wesley.

Ramakrishnan, R., & Gehrke, J. (2008). Database Management Systems. McGraw-Hill.

Reksoatmodjo Wahyuno. 2018. Analisis dan Perancangan Sistem Basis Data. Yogyakarta : Penerbit Andi

Schwartz, B., Zaitsev, P., & Tkachenko, V. (2018). High Performance MySQL: Optimization, Backups, and Replication. O'Reilly Media.

Sutanta, Edhy. 2011. Basis Data dalam Tinjauan Konseptual. Yogyakarta : Penerbit Andi

Simarmata Janner.2006. Basis Data, Yogyakarta: Penerbit Andi

White, T. (2019). Hadoop: The Definitive Guide. O'Reilly Media.

Winand, M. (2017). SQL Performance Explained. Markus Winand.

Date, C. J. (2016). A Guide to the SQL Standard. Addison-Wesley.

Chodorow, K., & Dirolf, M. (2014). MongoDB: The Definitive Guide. O'Reilly Media.

Bryla, B., & Loney, K. (2013). Oracle Database 12c: The Complete Reference. McGraw-Hill Education.

Fowler, M., & Sadalage, P. J. (2016). NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence. Addison-Wesley.

GLOSARIUM

1. **Basis Data:** Kumpulan data terorganisir yang dapat diakses, dikelola, dan diperbarui dengan bantuan sistem basis data.
2. **Manajemen Basis Data:** Proses merencanakan, mengorganisir, dan mengelola basis data untuk memastikan integritas, ketersediaan, dan keamanan data.
3. **Normalisasi:** Proses desain basis data untuk mengurangi redundansi dan meningkatkan efisiensi dengan memisahkan data ke dalam tabel yang terorganisir.
4. **Model Data Relasional:** Representasi struktur data menggunakan tabel, kolom, dan hubungan antartabel.
5. **Transaksi:** Serangkaian operasi database yang dianggap sebagai unit tunggal, yang harus dilakukan sepenuhnya atau tidak dilakukan sama sekali.
6. **SQL (Structured Query Language):** Bahasa pemrograman khusus untuk berinteraksi dengan basis data relasional, digunakan untuk mengambil, menyisipkan, mengupdate, dan menghapus data.

7. **Optimisasi Kinerja Query:** Proses meningkatkan efisiensi eksekusi query untuk memastikan respons cepat dari basis data.
8. **NoSQL:** Pendekatan basis data yang tidak memanfaatkan SQL dan sering digunakan untuk mengelola data non-relasional atau semi-struktur.
9. **Pengindeksan:** Proses menambahkan struktur pengindeksan pada tabel untuk meningkatkan kecepatan pencarian dan pengurangan beban kueri.
10. **Data Terdistribusi:** Penyebaran data di beberapa lokasi fisik untuk meningkatkan ketersediaan dan kinerja.
11. **Sistem Terdistribusi:** Sistem yang terdiri dari beberapa komputer atau server yang bekerja bersama untuk menangani beban kerja dan meningkatkan ketersediaan.
12. **Integritas Data:** Menjamin bahwa data dalam basis data akurat, konsisten, dan sesuai dengan batasan yang ditetapkan.
13. **Keamanan Basis Data:** Langkah-langkah untuk melindungi basis data dari akses yang tidak sah, kerusakan, atau ancaman keamanan lainnya.
14. **Desain Fisik Basis Data:** Proses menerjemahkan desain konseptual ke dalam struktur fisik yang dapat diimplementasikan di sistem basis data.

15. **Sistem Manajemen Basis Data (SMBD):** Perangkat lunak yang memungkinkan pengelolaan dan interaksi dengan basis data, termasuk manajemen transaksi, keamanan, dan optimisasi kueri.

16. **Triggers:** Aturan atau tindakan otomatis yang diaktifkan oleh perubahan data tertentu dalam basis data.

17. **Big Data:** Volume data yang sangat besar dan kompleks yang sulit dielola dengan cara tradisional.

18. **Backup dan Pemulihan:** Proses mencadangkan data untuk mencegah kehilangan data dan memulihkannya setelah bencana atau kegagalan sistem.

19. **Sistem Basis Data Terdistribusi:** Sistem yang mendistribusikan data di beberapa server atau lokasi untuk meningkatkan ketersediaan dan kinerja.

20. **Polyglot Persistence:** Pendekatan menggunakan berbagai tipe penyimpanan data (SQL, NoSQL) dalam satu aplikasi untuk memenuhi kebutuhan yang berbeda.

21. **Pembaruan Atomic, Konsisten, Isolasi, dan Durabilitas (ACID):** Sifat-sifat transaksi yang menjamin keandalan dan konsistensi data dalam basis data.

22. **Skema Basis Data:** Struktur logikal yang mendefinisikan bagaimana data diorganisir dan berinteraksi dalam sebuah basis data.

23. **Data Mart:** Subsistem basis data yang berfokus pada menyediakan data untuk kebutuhan dan analisis spesifik suatu departemen atau unit bisnis.

24. **Data Warehouse:** Sistem terpusat untuk menyimpan, menggabungkan, dan menganalisis data dari berbagai sumber untuk mendukung pengambilan keputusan.

25. **OLAP (Online Analytical Processing):** Pendekatan pengolahan data yang mendukung analisis kompleks dan multidimensional.

26. **OLTP (Online Transaction Processing):** Pendekatan pengolahan data yang fokus pada pengelolaan dan eksekusi transaksi online secara cepat.

27. **Pemodelan Entity-Relationship:** Pendekatan visual untuk merancang struktur basis data dengan menggunakan entitas, atribut, dan hubungan antar entitas.

28. **Primary Key:** Kolom atau kumpulan kolom yang unik mengidentifikasi setiap baris dalam sebuah tabel.

29. **Foreign Key:** Kolom dalam sebuah tabel yang merujuk ke primary key di tabel lain, menciptakan hubungan antar tabel.

30. **NoSQL:** Pendekatan basis data yang fleksibel dan dapat menyimpan data semi-struktur atau tidak terstruktur.

31. **Data Mining:** Proses mengekstraksi pola atau pengetahuan yang bermanfaat dari data besar untuk mendukung pengambilan keputusan.

32. **Cloud Database:** Basis data yang dihosting dan diakses melalui layanan cloud computing.

33. **Replication:** Proses menyalin dan mendistribusikan data dari satu server basis data ke server lain untuk meningkatkan ketersediaan dan keandalan.

34. **Blob (Binary Large Object):** Jenis data yang dapat menyimpan informasi biner, seperti gambar atau file.

35. **Partisi Tabel:** Membagi tabel besar menjadi bagian-bagian yang lebih kecil untuk meningkatkan kinerja dan manajemen data.

36. **Basis Data In-Memory:** Sistem basis data yang menyimpan dan memproses data utama dalam memori untuk meningkatkan kecepatan akses.

37. **Truncate:** Perintah SQL untuk menghapus semua data dari sebuah tabel tanpa mempengaruhi struktur tabel.

38. **ACID Properties:** Singkatan dari Atomicity, Consistency, Isolation, dan Durability, standar untuk transaksi dalam sistem basis data relasional.

39. **View:** Pemandangan virtual dari satu atau lebih tabel, menampilkan data yang memenuhi kriteria tertentu tanpa menyimpan data fisik.

40. **Index Clustered dan Non-Clustered:** Struktur penyimpanan tambahan yang meningkatkan kecepatan pencarian data dalam sebuah tabel.
41. **Entity:** Objek yang dapat dibedakan dan diidentifikasi, misalnya, dalam model data relasional, entitas dapat menjadi objek bisnis seperti pelanggan atau produk.
42. **Data Dictionary:** Kumpulan metadata yang mendeskripsikan struktur dan definisi data dalam basis data.
43. **Concurrency Control:** Teknik untuk mengelola akses bersama ke data dan mencegah konflik dalam transaksi yang berlangsung bersamaan.
44. **Stored Procedure:** Sekumpulan pernyataan SQL yang diberi nama dan disimpan dalam basis data untuk eksekusi berulang.
45. **Denormalisasi:** Proses mengurangi tingkat normalisasi dalam desain basis data untuk meningkatkan kinerja dengan memperkenankan redundansi data.
46. **Commit dan Rollback:** Perintah untuk menyelesaikan atau membatalkan transaksi dalam basis data.
47. **Data Compression:** Teknik untuk mengurangi ukuran penyimpanan data dengan cara mengompresinya.
48. **Schema Denormalisasi:** Pendekatan desain basis data yang mempertahankan data dalam bentuk yang lebih tidak terurai.

49. **Concurrency Lock:** Mekanisme yang mencegah akses bersamaan ke sumber daya data yang dapat mengakibatkan konflik.

50. **Star Schema:** Skema desain data warehousing yang terdiri dari satu tabel fakta dan tabel dimensi yang saling terhubung membentuk pola bintang.

51. **Snowflake Schema:** Skema desain data warehousing yang mirip dengan star schema, tetapi dimensi terpecah lebih lanjut menjadi subdimensi.

52. **Data Dictionary:** Koleksi metadata yang mendeskripsikan definisi dan struktur data dalam basis data.

53. **Load Balancing:** Distribusi beban kerja secara merata di antara server atau sumber daya komputasi untuk meningkatkan kinerja dan ketersediaan.

54. **Data Definition Language (DDL):** Bagian dari SQL yang mengatur struktur database, seperti membuat, mengubah, atau menghapus tabel.

55. **Data Manipulation Language (DML):** Bagian dari SQL yang digunakan untuk memanipulasi atau mengelola data dalam basis data, seperti mengambil atau memperbarui data.

56. **Database Administrator (DBA):** Profesional yang bertanggung jawab atas manajemen, pemeliharaan, dan keamanan basis data.

57. **ACID Transactions:** Transaksi yang mematuhi prinsip Atomicity, Consistency, Isolation, dan Durability.

58. **Data Modeling:** Proses merancang representasi struktural dan hubungan antar data dalam suatu domain bisnis.

59. **Spatial Database:** Basis data yang mendukung penyimpanan dan kueri data spasial, seperti peta dan koordinat geografis.

60. **Data Warehouse Architect:** Profesional yang merancang dan mengelola struktur data warehousing untuk analisis dan pelaporan bisnis.

61. **Surrogate Key:** Kunci buatan yang ditambahkan ke tabel untuk menjadi primary key, umumnya digunakan dalam desain basis data.

62. **Materialized View:** Hasil dari query yang disimpan sebagai tabel fisik untuk meningkatkan kinerja query yang kompleks.

63. **Data Cleansing:** Proses membersihkan dan memperbaiki data yang tidak akurat atau tidak lengkap.

64. **Change Data Capture (CDC):** Teknik untuk mendeteksi dan merekam perubahan data dalam basis data.

65. **Database Migration:** Proses memindahkan data dari satu sistem basis data ke sistem basis data lainnya.

66. **Backup Incremental:** Proses mencadangkan hanya data yang telah berubah sejak cadangan terakhir.

67. **Foreign Key Constraint:** Aturan integritas referensial yang memastikan bahwa nilai di kolom referensi selalu ada di kolom yang merujuk.

68. **Database Sharding:** Membagi data menjadi bagian-bagian lebih kecil yang disebut shard untuk meningkatkan kinerja dan skala horizontal.

69. **Data Lake:** Penyimpanan besar yang menggabungkan data terstruktur dan tidak terstruktur dari berbagai sumber.

70. **Automated Backup:** Proses pencadangan otomatis yang dijadwalkan untuk melindungi data secara berkala.

71. **Columnar Database:** Penyimpanan data yang menyimpan nilai kolom secara berurutan untuk meningkatkan kinerja analisis kolom.

72. **Temporal Database:** Penyimpanan data yang melacak perubahan data sepanjang waktu, memungkinkan akses ke versi data sebelumnya.

73. **Data Profiling:** Proses analisis data untuk mengidentifikasi kualitas, struktur, dan karakteristik data.

74. **Database Query:** Pernyataan yang digunakan untuk mengambil data dari basis data, umumnya menggunakan SQL.

75. **Data Replication:** Proses menggandakan dan mendistribusikan data ke beberapa lokasi atau server.

76. **Master Data Management (MDM):** Proses manajemen data inti, seperti data pelanggan atau produk, untuk memastikan konsistensi di seluruh organisasi.

77. **Logical Backup:** Cadangan data yang mencakup skema dan struktur data, tetapi tidak data sebenarnya.

78. **Geographic Information System (GIS):** Sistem yang mengintegrasikan data geografis dan memungkinkan analisis berbasis lokasi.

79. **Data Masking:** Proses menyembunyikan atau menyamaratakan data sensitif dalam lingkungan pengujian atau pengembangan.

80. **Database Trigger:** Aturan yang diaktifkan secara otomatis oleh peristiwa tertentu dalam basis data, seperti penambahan atau penghapusan data.

81. **Elastic Database:** Konsep penyimpanan data yang dapat diubah ukurannya secara dinamis sesuai dengan kebutuhan, sering digunakan dalam lingkungan cloud.

82. **Data Validation:** Proses memeriksa data untuk memastikan keakuratannya dan sesuai dengan aturan yang ditetapkan.

83. **Database Partitioning:** Membagi tabel atau indeks besar menjadi bagian-bagian yang lebih kecil untuk meningkatkan kinerja dan manajemen data.

84. **Data Lakehouse:** Kombinasi antara data lake dan data warehouse, menyatukan kelebihan keduanya untuk analisis data yang holistik.

85. **Database Index:** Struktur yang mempercepat pencarian dan kueri data dengan menyimpan referensi ke lokasi data.

86. **Database Schema:** Struktur organisasi dan definisi objek dalam sebuah basis data, termasuk tabel, indeks, dan keterbatasan.

87. **Data Warehousing:** Proses penyimpanan, penggabungan, dan analisis data dari berbagai sumber untuk mendukung pengambilan keputusan.

88. **Database Clustering:** Menggabungkan beberapa server atau sumber daya dalam satu kelompok untuk meningkatkan ketersediaan dan kinerja.

89. **Query Execution Plan:** Rencana yang disusun oleh pengelola basis data untuk mengeksekusi suatu query dengan efisien.

90. **Database Connection Pooling:** Teknik untuk mengelola dan membagikan koneksi database yang sudah ada untuk mengurangi overhead.

91. **Data Encryption:** Proses mengamankan data dengan mengubahnya menjadi format terenkripsi yang hanya dapat diakses dengan kunci yang benar.

92. **Database Scalability:** Kemampuan sistem basis data untuk menangani peningkatan jumlah data atau pengguna dengan menjalankan lebih banyak sumber daya.

93. **Database Triggers:** Pemicu atau aturan yang diaktifkan secara otomatis oleh perubahan data tertentu dalam basis data.

94. **Data Warehouse Star Schema:** Struktur skema yang sering digunakan dalam data warehousing, melibatkan tabel fakta dan tabel dimensi yang membentuk pola bintang.

95. **Database Table:** Struktur penyimpanan utama yang menyimpan data dalam basis data relasional.

96. **Database View:** Pandangan virtual dari data dalam satu atau lebih tabel, menyajikan data berdasarkan kriteria tertentu.

97. **Data Archiving:** Proses memindahkan data yang tidak aktif atau jarang digunakan ke penyimpanan yang lebih murah untuk menghemat ruang penyimpanan aktif.

98. **Database Connection:** Koneksi yang dibuat antara aplikasi dan basis data untuk mengakses atau memanipulasi data.

99. **Database Backup:** Salinan data yang diambil untuk tujuan pemulihan setelah kehilangan atau kegagalan sistem.

100. **Database Deadlock:** Keadaan di mana dua atau lebih transaksi saling menunggu untuk sumber daya yang dipegang oleh transaksi lain, mengakibatkan penghentian.

101. **Data Governance:** Kerangka kerja dan kebijakan yang mengatur pengelolaan dan penggunaan data dalam sebuah organisasi.

102. **Database Denormalization:** Kebalikan dari normalisasi, yaitu proses menambahkan redundansi ke dalam desain basis data untuk meningkatkan kinerja.

103. **Data Mining:** Proses menggunakan teknik statistik dan matematika untuk mengeksplorasi dan menganalisis pola dalam data besar.

104. **Database Constraint:** Aturan yang diterapkan pada data untuk memastikan konsistensi dan integritas, seperti keterbatasan unik atau keterbatasan kunci asing.

105. **Database Trigger:** Prosedur atau aturan yang dijalankan otomatis ketika suatu peristiwa tertentu terjadi dalam basis data, seperti penyisipan atau pembaruan data.

106. **Database View:** Subset dari tabel atau hasil dari query yang dapat diakses seperti tabel biasa.

107. **Data Mart:** Subsistem basis data yang menyimpan data untuk mendukung kebutuhan analisis suatu area bisnis atau departemen.

108. **Database Link:** Koneksi virtual antara dua basis data yang memungkinkan query dan manipulasi data di antara keduanya.

109. **Data Warehouse Snowflake Schema:** Bentuk skema data warehousing di mana dimensi dibagi menjadi level-level yang lebih rinci.

110. **Database Snapshot:** Salinan read-only dari basis data pada suatu titik waktu tertentu, sering digunakan untuk melacak perubahan atau sebagai titik pemulihan.

111. **Data Lake Governance:** Proses pengelolaan dan pengendalian data dalam lingkungan data lake untuk memastikan kualitas dan keamanan data.

112. **Database Audit:** Proses pemantauan dan pencatatan aktivitas pengguna dan perubahan data untuk keperluan keamanan dan kepatuhan.

113. **Data Masking:** Proses menyamarkan atau menggantikan informasi pribadi atau sensitif dalam data untuk melindungi privasi.

114. **Database Connection String:** String teks yang berisi informasi untuk mengidentifikasi dan mengakses basis data, termasuk nama host, nama basis data, dan kredensial pengguna.

115. **Data Profiling:** Analisis statistik dan evaluasi kualitas data untuk mengidentifikasi anomali atau masalah potensial.

116. **Database Schema:** Struktur logis yang mendefinisikan organisasi objek-objek, seperti tabel, indeks, dan tampilan, dalam sebuah basis data.

117. **Data Warehouse Fact Table:** Tabel yang menyimpan data fakta atau kuantitatif dalam struktur data warehousing.

118. **Database Partitioning Key:** Kolom yang digunakan untuk membagi tabel atau indeks menjadi sejumlah partisi untuk meningkatkan kinerja dan manajemen data.

119. **Database Connection Pooling:** Teknik yang mengelola dan mendaur ulang koneksi database untuk mengurangi overhead dan meningkatkan kinerja aplikasi.

120. **Data Stewardship:** Proses pengelolaan dan pemeliharaan data yang melibatkan tanggung jawab dan akuntabilitas dalam organisasi.