

BAB I DASAR PEMODELAN PERANGKAT LUNAK

⌘ KONSEP PEMODELAN PERANGKAT LUNAK

A. PEMODELAN PERANGKAT LUNAK

Pemodelan merupakan suatu proses dalam **menggambarkan secara abstrak suatu model**. Model diartikan sebagai **rencana, representasi, atau deskripsi yang menjelaskan suatu objek, sistem, atau konsep** yang berupa penyederhanaan atau idealisasi. Bentuknya dapat berupa model fisik (maket dan prototipe), model citra (gambar rancangan dan citra komputer), atau rumusan matematis. Proses pemodelan menampilkan deskripsi suatu proses dari beberapa perspektif tertentu. Proses pemodelan perangkat lunak menjadi aktivitas yang saling terkait (koheren) untuk menspesifikasikan, merancang, implementasi kode program, dan pengujian sistem perangkat lunak (Romi, 2006).

Pemodelan perangkat lunak adalah proses menciptakan representasi abstrak dari sistem perangkat lunak yang ingin kita bangun atau perbaiki.

Tujuan Pemodelan Perangkat Lunak:

1. **Memahami kebutuhan pengguna:** Pemodelan membantu kita mendefinisikan dan memahami kebutuhan pengguna serta bagaimana pengguna akan berinteraksi dengan perangkat lunak yang akan dibangun.
2. **Meningkatkan kualitas perangkat lunak:** Dengan memodelkan perangkat lunak, kita dapat menganalisis dan merancang sistem secara lebih efisien, mengurangi kesalahan dan memastikan bahwa perangkat lunak sesuai dengan kebutuhan.
3. **Komunikasi dan kolaborasi:** Pemodelan menjadi alat komunikasi yang efektif antara pengembang, pemangku kepentingan, dan tim lainnya dalam memahami dan merancang perangkat lunak.

Pemodelan dalam rekayasa perangkat lunak menjadi aktivitas yang dilakukan di tahapan awal pengembangan aplikasi dan akan memengaruhi pekerjaan lain. Beberapa pemodelan yang dikenal dalam pengembangan aplikasi atau rekayasa perangkat lunak sebagai berikut.

1. Pemodelan Proses

Model yang digunakan dalam pengembangan sistem perangkat lunak (software development life cycle process) terdiri dari tahapan spesifikasi kebutuhan perangkat lunak sampai perawatan sistem.

2. Pemodelan Analisis dan Desain

Model ini berisi spesifikasi lengkap dari persyaratan representasi desain yang komprehensif bagi perangkat lunak. Model ini harus memenuhi tiga sasaran utama, yaitu

menggambarkan yang dibutuhkan pelanggan, membangun dasar bagi pembuatan desain perangkat lunak, dan membatasi persyaratan yang dapat divalidasi.

3. Pemodelan Data

Model ini merupakan sarana melakukan abstraksi data, relasi antarentitas dalam basis data, dan konsep membuat diskripsi stuktur basis data. Model ini memuat spesifikasi operasi dasar (basic operation) dalam pengaksesan dan pembaharuan data, serta tabiat data (data behavior).

4. Pemodelan Fungsional dan Aliran Informasi

Informasi ditransformasikan saat mengalir melalui sebuah sistem berbasis komputer. Sistem menerima input dengan berbagai cara dan menghasilkan output sehingga dapat menciptakan suatu model aliran bagi setiap sistem tanpa melihat ukuran dan kompleksitasnya. Contoh: Data Flow Diagram (DFD)

5. Pemodelan Unified Modeling Language (UML)

Pemodelan Unified Modeling Language (UML) merupakan suatu bahasa yang berisi aturan-aturan, notasinotasi, dan gambar untuk membuat visualisasi model seperangkat lunak dengan pendekatan sistem berorientasi objek.

⌘ Metode Pengembangan Perangkat Lunak

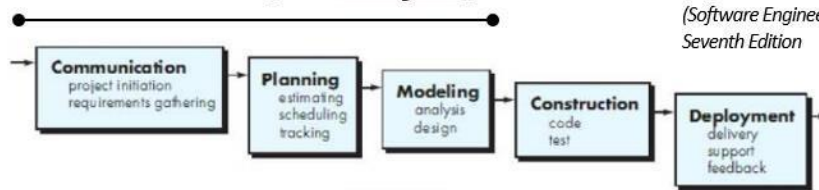
Metode Pengembangan Sistem adalah pendekatan yang digunakan untuk merancang, mengembangkan, dan mengimplementasikan sistem informasi atau perangkat lunak. Metode ini membantu mengatur langkah-langkah yang harus diikuti dalam proses pengembangan, memastikan bahwa sistem yang dihasilkan berkualitas, sesuai dengan kebutuhan, dan dapat diandalkan.

Untuk pemodelan perangkat lunak ternyata memiliki berbagai macam jenis proses yang dapat digunakan dan berbagai macam pula jenis nya berikut :

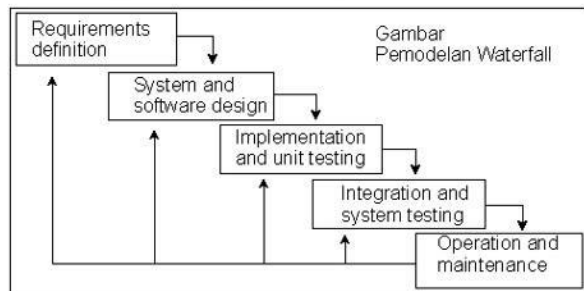
o Metode Waterfall

Metode Waterfall merupakan metode pengembangan perangkat lunak paling tradisional yang sangat sistematis dimana alurnya benar-benar baku dimana metode ini berkaitan dengan alus bisnis yang kita ketahui yaitu SDLC. Untuk bentukan alurnya kurang lebih seperti berikut:

WATERFALL (Air Terjun)



Source Gambar :
Roger S. Pressman
(Software Engineering A Practitioner's Approach)
Seventh Edition



Gambar
Pemodelan Waterfall

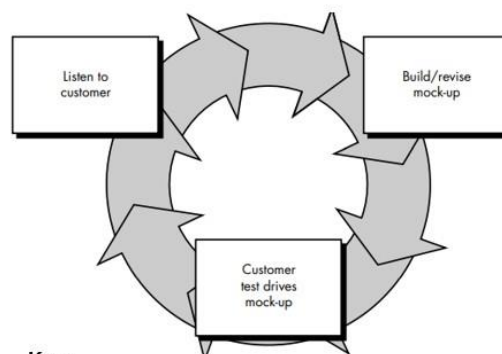
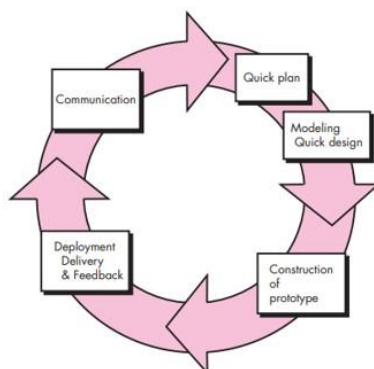
Key:

- Tahapan (Bertahap) / Linear
- Project yang cukup besar
- Jelas & Terdokumentasi
- Tidak Fleksibel
- Waktu yang lama

o Metode Prototype / Prototyping

Prototype dalam bahasa Indonesia diartikan dengan istilah purwarupa. Istilah tersebut berarti model awal atau rancangan sementara yang masih membutuhkan berbagai penyesuaian sebelum dinyatakan telah memenuhi hasil yang diinginkan.

Prototype



Key:

- Purwarupa / Rancangan Produk
- Fokus Produk
- Project singkat & tidak terlalu rumit (lingkup kecil)
- Fleksibel
- Waktu yang Singkat

o Metode Spiral

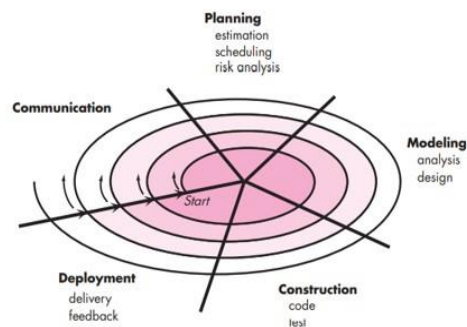
Metode spiral menggabungkan dua metode pengembangan yang telah dibahas sebelumnya, yaitu prototype dan waterfall. Pengembang melaksanakan prototyping dengan cara sistematis khas metode waterfall.

Umumnya metode spiral diterapkan dalam pengembangan perangkat lunak berskala besar, sekaligus membutuhkan sistem yang kompleks. Setiap prosesnya selalu disertai dengan analisis mendalam mengenai tingkat risiko dan keberhasilan pengembangan. Pelaksanaan metode spiral dilakukan dalam lima langkah. Pertama adalah komunikasi, yaitu pemilik proyek menyampaikan kebutuhannya kepada pengembang perangkat lunak. Dilanjutkan dengan perencanaan mendetail tentang proyek yang digarap. Langkah perencanaan diikuti dengan analisis untuk mengidentifikasi berbagai kemungkinan yang bisa terjadi selama pengembangan. Kemudian, pengembangan perangkat lunak mulai dijalankan dan setelah jadi akan mendapatkan evaluasi dari pelanggan.

SPIRAL

Note :

- Penggabungan dari "Waterfall & Prototype"
- Berfokus pada produk namun tidak melupakan alur / tahapan yang dilalui
- Cocok digunakan untuk project besar / rumit
- Semakin banyak iterasi / perulangan prosedur semakin rumit dan membutuhkan waktu yang lama



o Metode RAD

RAD merupakan singkatan dari Rapid Application Development. Metode ini juga menggunakan pendekatan iteratif dan inkremental, tetapi lebih menekankan pada tenggat waktu dan efisiensi biaya yang sesuai dengan kebutuhan. Proses pengembangan dengan Metode RAD dianggap lebih singkat. Pasalnya, semua pihak, baik pelanggan maupun pengembang, terus terlibat secara aktif dalam setiap proses hingga hasil dapat tercapai. Di samping itu, tahapan kerja pada metode ini juga lebih sedikit.

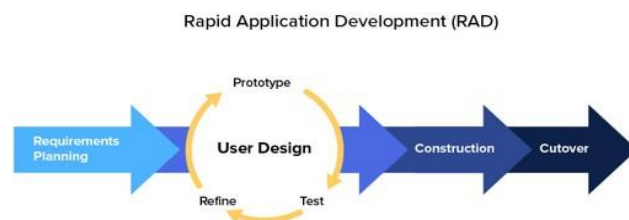
Alur kerja hanya dibagi menjadi tiga tahap yang semuanya padat. Identifikasi tujuan yang langsung diiringi dengan komunikasi dan perancangan, di mana seluruh pihak terlibat aktif dalam setiap perumusannya. Proses ini menjadi tahap awal dari Metode RAD.

Tahap kedua masih melibatkan semua pihak, yaitu proses mendesain sistem atau perangkat lunak sesuai kebutuhan. Pelanggan atau pengguna ikut terjun dalam menguji coba perangkat lunak. Perbaikan pun langsung diterapkan jika pengguna menemukan kesalahan. Ketika pengguna terpuaskan dengan desain perangkat lunak, setelah melalui berbagai perbaikan, barulah proses kerja menginjak pada tahap terakhir, yaitu implementasi. Desain perangkat lunak mulai diterjemahkan dalam bahasa mesin dan bisa digunakan. Sumber : (<https://salamadian.com/metode-pengembangan-perangkat-lunak/>)

RAD

Note :

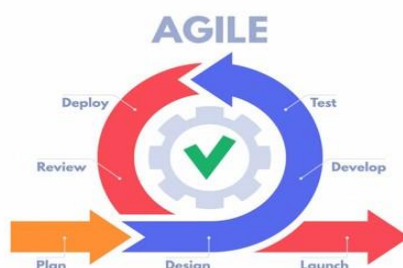
- Rapid / Cepat
- Iterasi / Pengulangan Terjadi di Perancangan Produk hingga desain disetujui "Stake Holder"
- Cocok untuk project lingkup kecil dengan waktu yang singkat
- Pembagian kerja tim yang modular



o Metode Agile

Metode ini menekankan kolaborasi tim yang kuat, responsif terhadap perubahan, dan iteratif dalam pengembangan perangkat lunak. Salah satu contoh framework Agile adalah Scrum, yang terdiri dari sprint (iterasi) dalam rentang waktu tertentu, misalnya, 2 minggu atau sekian minggu.

AGILE



Note :

- Scrum, Sprint
- Kepuasan Pelanggan
- Kanban board
- Tim Kecil (<20 orang) namun memiliki motivasi tinggi
- Terdokumentasi baik
- Manager Proyek dapat memantau langsung kinerja tim
- Apabila terjadi bottle neck proyek masih bisa berjalan karena pengerjaan sesuai sprint

BAB 2 KONSEP PEMODELAN TERSTRUKTUR DAN OBJEK

PEMODELAN TERSTRUKTUR VS PEMODELAN BERORIENTASI OBJEK

Pemodelan terstruktur dan pemodelan berorientasi objek adalah dua pendekatan yang berbeda dalam merancang dan mengembangkan sistem atau perangkat lunak. Berikut adalah perbedaan mendasar antara keduanya:

Pemodelan Terstruktur:

1. Fokus: Pemodelan terstruktur berfokus pada **pemisahan data (struktur data) dari prosedur (algoritma)**. Data dan prosedur dianggap sebagai dua entitas terpisah dalam pemodelan terstruktur. Data diolah melalui serangkaian prosedur yang dijalankan berurutan.
2. Paradigma: Pemodelan terstruktur **berbasis pada paradigma pemrograman prosedural**, di mana kode dibagi menjadi fungsi-fungsi dan bagian-bagian yang lebih kecil yang dieksekusi secara berurutan.
3. Komponen Utama: **Struktur data** (seperti array, record, atau tipe data abstrak) dan algoritma (prosedur atau fungsi) adalah komponen utama dalam pemodelan terstruktur.
4. Hubungan Data: Dalam pemodelan terstruktur, data dapat dipertukarkan secara **bebas antara prosedur-prosedur**, dan **akses ke data sering bersifat global**, yang dapat menyebabkan kompleksitas dan masalah dalam manajemen kode.

Pemodelan Berorientasi Objek:

1. Fokus: Pemodelan berorientasi objek berfokus pada penggabungan data dan perilaku (prosedur) ke dalam satu unit yang disebut **objek**. Objek merupakan representasi konkret dari entitas di dunia nyata dan memiliki atribut (data) serta metode (prosedur) yang beroperasi pada atribut tersebut.
2. Paradigma: Pemodelan berorientasi objek **berbasis pada paradigma pemrograman berorientasi objek (OOP)**, yang menekankan pada konsep abstraksi, pewarisan, enkapsulasi, dan polimorfisme.
3. Komponen Utama: **Objek sebagai komponen utama** dalam pemodelan berorientasi objek. Objek-objek ini berkomunikasi satu sama lain melalui pesan, dan setiap objek memiliki peran dan tanggung jawabnya sendiri.

4. Hubungan Data: Dalam pemodelan berorientasi objek, data dan perilaku yang terkait dengan suatu objek dikemas bersama dalam satu kesatuan. Hal ini memungkinkan enkapsulasi dan menyembunyikan detail implementasi, sehingga kode menjadi lebih bersih, mudah dipahami, dan **terhindar dari konflik data global**.

Pemodelan terstruktur seringkali digunakan dalam pemrograman prosedural, sedangkan pemodelan berorientasi objek lebih umum digunakan dalam pemrograman berorientasi objek. Pemilihan antara keduanya tergantung pada kompleksitas proyek, struktur data yang diinginkan, dan gaya pemrograman yang diadopsi.

Kemudian untuk memodelkan suatu sistem biasanya dibutuhkan seorang **System Analyst** dimana beliau bertugas untuk merancang dan mengidentifikasi keinginan dari user atau **stake holder**.

Analisis Sistem

Setelah mendapatkan rancangan atau mengidentifikasi keinginan user tugas **(System Analyst)** suatu sistem adalah menganalisa seluruh kebutuhan sistem kemudian **menggambarannya dalam sebuah notasi atau diagram agar alur bisa lebih cepat di mengerti** serta membantu **programmer** dalam membuat logika program.



Dalam menganalisis Sistem atau aplikasi kita dapat menggunakan **teknik analisis**. Dimana teknik ini dibagi menjadi 2 : yaitu **Analisis Terstruktur & Analisis Berorientasi Objek**

ANALISIS TERSTRUKTUR

Suatu proses untuk mengimplementasikan urutan langkah penyelesaian suatu masalah bentuk program, yang dilakukan dengan memperhatikan urutan dari setiap perintah yang sistematis, logis, dan tersusun berdasarkan algoritma dan dapat dengan mudah dipahami.

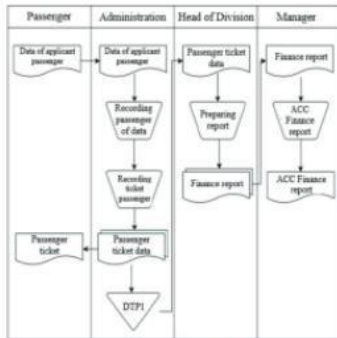
ANALISIS BERORIENTASI OBJEK

Suatu proses untuk mengimplementasikan pemecahan masalah dalam bentuk program yang berorientasikan kepada objek. Semua data dan fungsi didalamnya dibungkus kedalam kelas-kelas atau objek dimana pemrosesan data, pengiriman pesan antar objek digambarkan. Terkadang analisis ini berfokus pada objek atau bentuk-bentuk diagram yang dibuat oleh karena itu perlu adanya penyelarasan pemahaman tentang diagram yang dibuat agar mudah dipahami.

Perangkat Pemodelan yang digunakan :

Analisis Terstruktur :	Analisis Berorientasi Objek :
<ol style="list-style-type: none">1. Flowchart2. Flowmap3. ERD (Entity Relationship Diagram)4. Kamus Data	<ol style="list-style-type: none">1. Usecase Diagram2. Activity Diagram3. Sequence Diagram4. Class Diagram5. Object Diagram6. Deploymen Diagram7. Component Diagram

Pemodelan Analisis Terstruktur

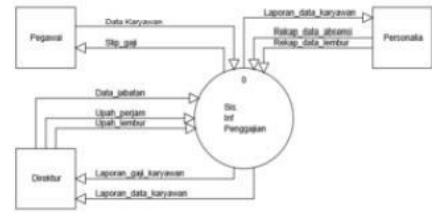


FlowMap

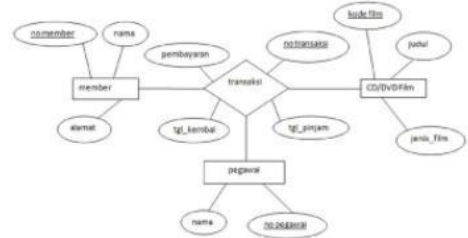
No.	Simbol	Keterangan
1	=	Dusunun atau terdiri dari
2	*	Dan
3	[]	Baik ... atau ...
4	{}*	n kali diulang bernilai banyak
5	()	Data opsional
6	*..*	Batas komentar

Sumber : Sukanto dan Shalahudin (2013:74)

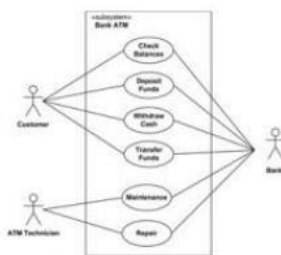
Kamus Data



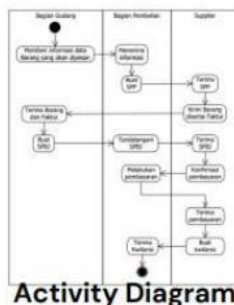
DFD (Data Flow Diagram)



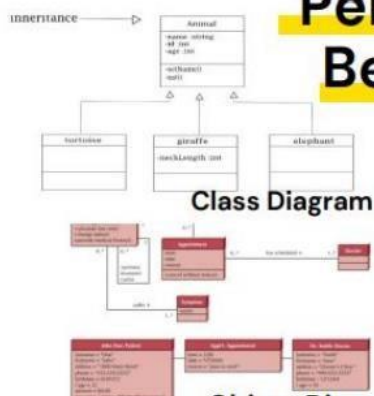
Entity Relationship Diagram



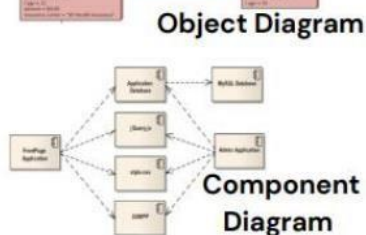
Use Case Diagram



Activity Diagram

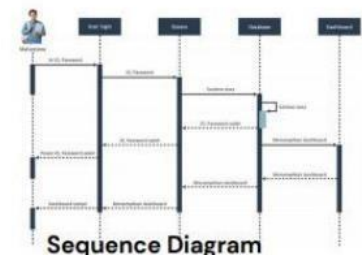


Class Diagram

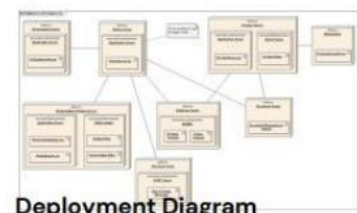


Object Diagram

Component Diagram



Sequence Diagram



Deployment Diagram

BAB 3 DIAGRAM PEMODELAN BERORIENTASI OBJEK

⌘ PEMODELAN PERANGKAT LUNAK DENGAN UML

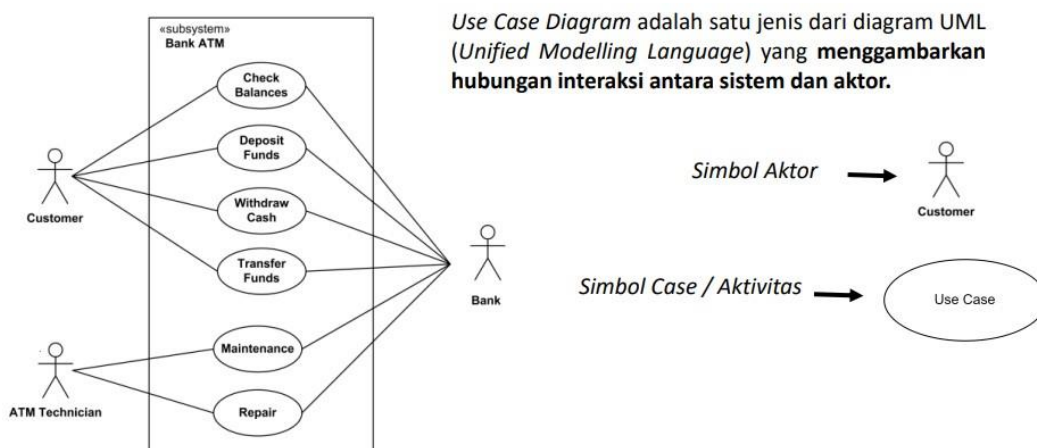
UML (Unified Modelling Language) adalah suatu metode dalam pemodelan secara visual yang digunakan sebagai sarana perancangan sistem berorientasi objek. Diagram-diagram yang digunakan dalam UML adalah diagram-diagram yang ada pada analisis berorientasi objek dan saat ini bentuk inilah yang cukup digemari oleh para pembuat rancangan aplikasi termasuk digunakan untuk pelaporan karya ilmiah atau skripsi mahasiswa Teknik Informatika.

Menurut Suendri (2018) dalam Windu dan Grace (2013), Unified Modeling Language (UML) adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak. UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem”.

Unified Modeling Language (UML) adalah sebuah bahasa yang berdasarkan grafik atau gambar untuk memvisualisasi, menspesifikasikan, membangun, dan pendokumentasian dari sebuah sistem pengembangan software berbasis OO (Object-Oriented). UML sendiri juga memberikan standar penulisan sebuah sistem blue print, yang meliputi konsep bisnis proses, penulisan kelas-kelas dalam bahasa program yang spesifik, skema database, dan komponen-komponen yang diperlukan dalam sistem software (<http://www.omg.org>).

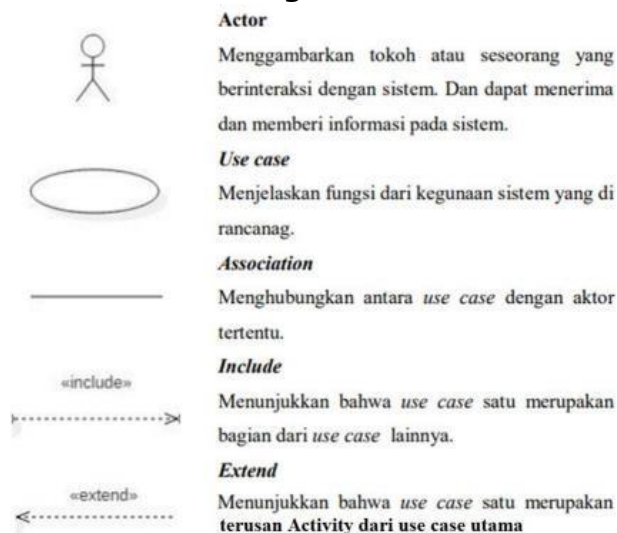
Dalam pembangunan aplikasi berorientasi objek ada banyak diagram yang dapat digunakan untuk menggambarkan sistem namun yang paling sering digunakan, yaitu use case diagram, activity diagram, sequence diagram, dan class diagram.

A. Usecase Diagram



Menurut Suendri (2018) dalam Prabowo Pudjo Widodo (2011), Use case menggambarkan external view dari sistem yang akan kita buat modelnya. Model use case dapat dijabarkan dalam diagram use case, tetapi perlu diingat, diagram tidak indetik dengan model karena model lebih luas dari diagram. Use case harus mampu menggambarkan urutan aktor yang menghasilkan nilai terukur.

Simbol-Symbol Usecase Diagram :



KEBUTUHAN FUNSIONAL :

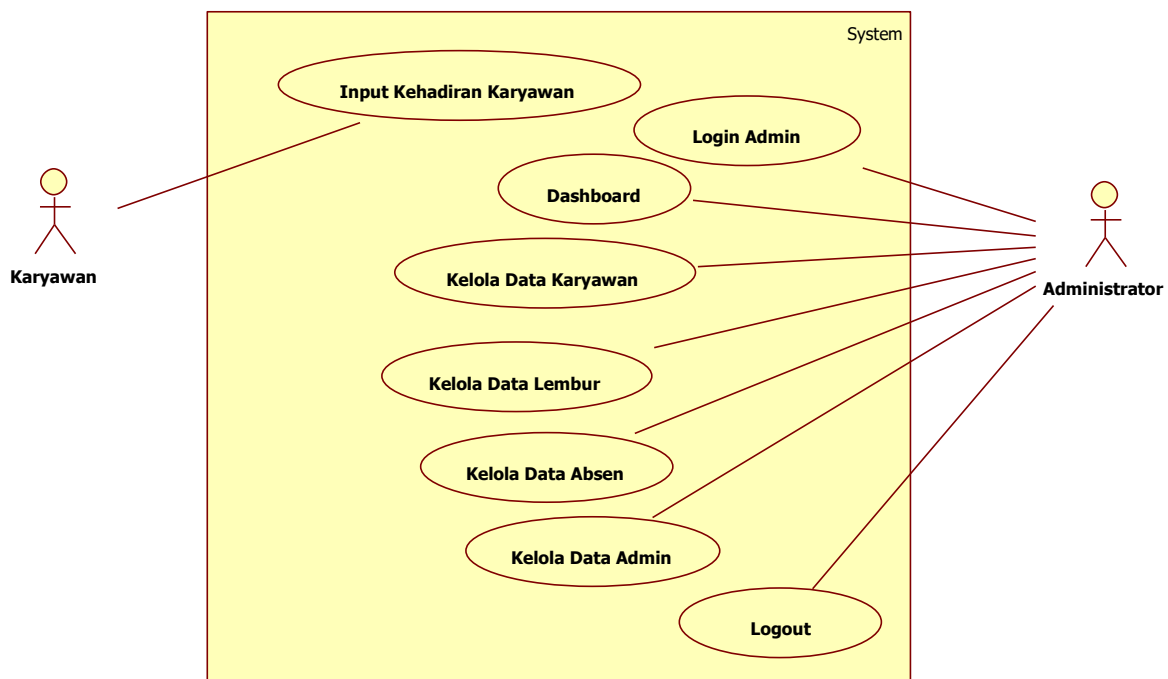
Kebutuhan fungsional merupakan gambaran dari fungsionalitas sistem yang dapat di deskripsikan dengan sebuah table dimana pada table tersebut terdapat list fungsi yang nantinya akan dijadikan fungsi sistem.

CONTOH:

Tabel Daftar Kebutuhan Fungsional

Kode	Kebutuhan Fungsional	Deskripsi
001	Login Admin	Fungsi ini digunakan untuk admin memasuki sistem / aplikasi
002	Dashboard	Menampilkan data hadir karyawan
003	Kelola Data Karyawan	Fungsi ini digunakan admin untuk mengelola data karyawan dari mulai tambah data, edit, hapus, pengisian ketidakhadiran (absen) karyawan dan

		cetak laporan kehadiran per karyawan.
004	Kelola Lembur	Fungsi ini digunakan admin untuk mengelola data lembur karyawan
005	Kelola Data Absen	Fungsi ini digunakan untuk mengelola data ketidakhadiran (absen) karyawan.
006	Kelola Data Admin	Fungsi ini dibuat untuk mengatur user atau id yang digunakan untuk masuk kedalam aplikasi pengelolaan data.
007	Input Kehadiran Karyawan	Fungsi ini adalah tampilan antar muka depan yang akan diakses karyawan untuk melakukan absensi dengan cara meletakan kartu RFID (kartu karyawan) ke alat RFID Reader
008	Logout	Fungsi ini digunakan untuk keluar dari aplikasi



Setelah Usecase dibuat, ada satu bagian lagi dari usecase yang dapat dideskripsikan kembali sebelum melanjutkan penggambaran sistem ke tahap yang lain yaitu **Scenario Usecase / Usecase Scenario**.

USECASE SCENARIO

Usecase scenario merupakan pendeskripsian alur per aktivitas yang ada didalam Usecase Diagram. Dimana alur yang dibuat nantinya dapat digambarkan dengan menggunakan **Activity Diagram**.

Hal yang perlu diperhatikan dalam membuat scenario Usecase adalah :

1. Aktor yang terlibat
2. Kondisi Awal / keadaan awal aktivitas / proses
3. Interaksi Antara Aktor dan Sistem (sesuai alur sistem yang dibuat)
4. Kondisi Akhir / keadaan akhir aktivitas / proses
5. Uncertain condition / Kondisi yang tidak menentu / pengecualian

Berikut Contoh Usecase Scenario yang dapat anda buat :

Contoh : Skenario *Login Admin*

No	001	
Nama	Login Admin	
Deskripsi	Fungsi ini digunakan untuk admin memasuki sistem / aplikasi	
Aktor	Admin	
Skenario		
Kondisi Awal	Tampilan berada di halaman login	
Aksi Aktor		Reaksi Sistem
1. Memasukan username dan password.		3. Melakukan verifikasi user dan melakukan pembatasan hak akses.
2. Menekan tombol login.		4. Menampilkan halaman admin
Kondisi Akhir	Tampilan berada di halaman admin	

B. Activity Diagram (Diagram Activity)

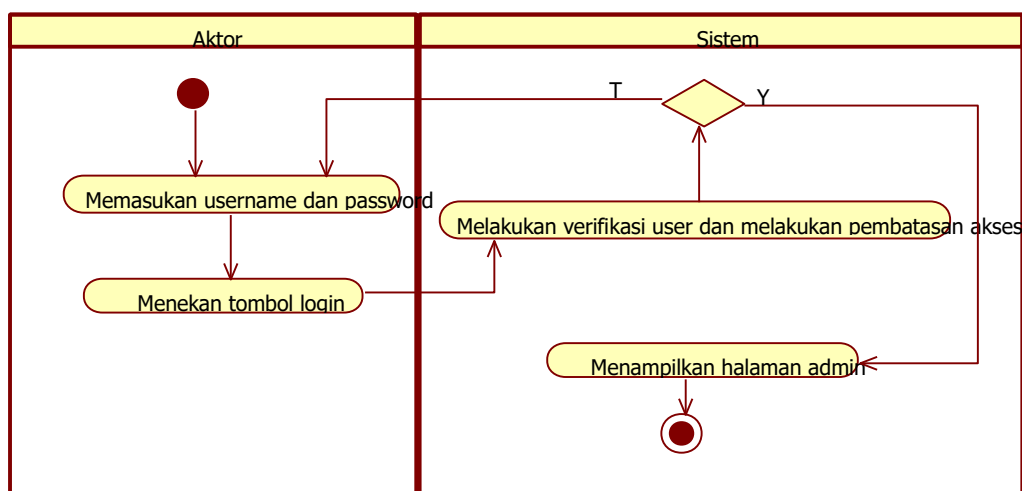
Diagram activity menunjukkan aktivitas sistem dalam bentuk kumpulan aksi-aksi, bagaimana masingmasing aksi tersebut dimulai, keputusan yang mungkin terjadi hingga berakhirnya aksi. Activity diagram juga dapat menggambarkan proses lebih dari satu aksi dalam waktu bersamaan. "Diagram activity adalah aktifitas-aktifitas, objek, state, transisi state dan event. Dengan kata lain kegiatan diagram alur kerja menggambarkan perilaku

sistem untuk aktivitas”. Notasi umum yang sering digunakan dalam activity diagram adalah sebagai berikut:

No	Simbol	Notasi	Keterangan
1	●	<i>Initial State</i>	Titik awal untuk memulai suatu aktifitas
2	⦿	<i>Final State</i>	Titik akhir untuk mengakhiri suatu aktifitas
3	▭	<i>Activity</i>	Memperlihatkan bagaimana masing – masing kelas antarmuka saling berinteraksi satu sama lain
4	◇	<i>Decision</i>	Pilihan untuk pengambilan keputusan
5	→	<i>Control Flow</i>	Arus aktifitas
6	▭	<i>Swimlane</i>	Sebuah cara untuk mengelompokan <i>activity</i> berdasarkan <i>actor</i> (mengelompokan <i>activity</i> dalam sebuah urutan yang sama)

Untuk menggambarkan Activity Diagram kita bisa mengikuti alur yang telah dibuat pada usecase scenario dengan menggunakan tools seperti draw.io / starUML / Visio.

Contoh Activity Diagram (dari Usecase Scenario pada pembahasan sebelumnya)



C. Sequence Diagram

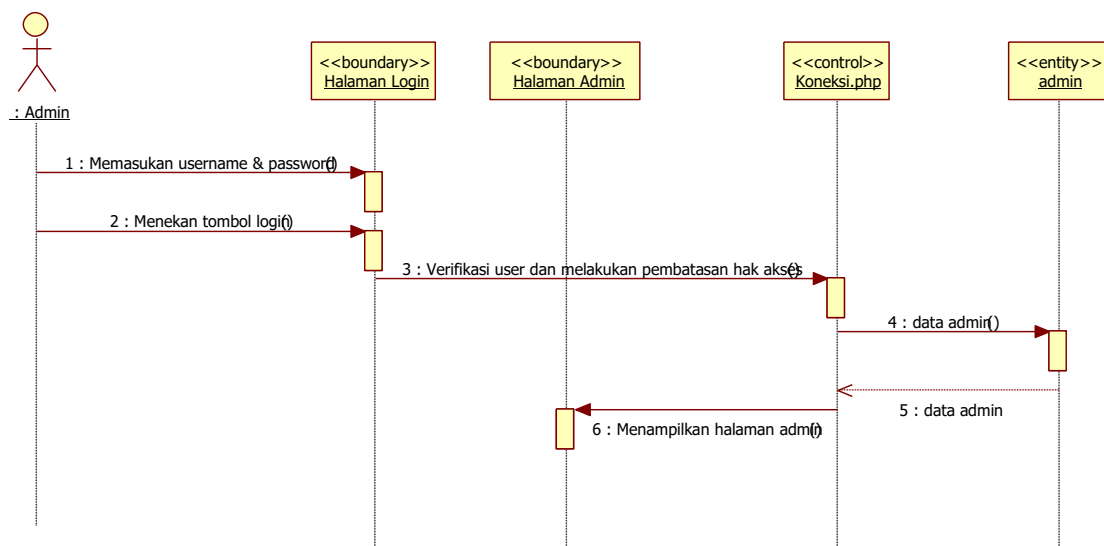
Sequence diagram adalah gambaran tahap demi tahap, termasuk kronologi (urutan) perubahan secara logis yang dilakukan oleh sistem dimana masing-masing objek saling berinteraksi dengan cara mengirimkan message.

Biasanya Sequence Diagram dapat dibuat dengan cara melihat alur proses yang telah digambarkan didalam bentuk activity diagram. Dengan memperhatikan notasi-notasinya atau

simbol-simbol yang sesuai. Notasi umum yang sering digunakan dalam Sequence diagram adalah sebagai berikut:

No	Simbol	Notasi	Keterangan
1		Object	Merupakan <i>instance</i> dari sebuah <i>class</i> yang dituliskan tersusun secara horizontal diikuti <i>lifeline</i>
2		LifeLine	Objek <i>entity</i> , antar muka yang saling berinteraksi
3		Activation	Indikasi dari lamanya objek melakukan eksekusi
4		Message	Indikasi untuk komunikasi antar objek yang (memanggil method)
5		Actor	Pengguna sistem atau yang berinteraksi langsung dengan sistem
6		Entity Class	Simbol untuk sebuah class yang menjadi entitas / tabel
7		Boundary Class	Simbol untuk sebuah class yang menjadi tampilan atau antarmuka sistem
8		Control Class	Simbol untuk sebuah class yang menjadi control penghubung antara <i>boundary</i> dan <i>entity</i>

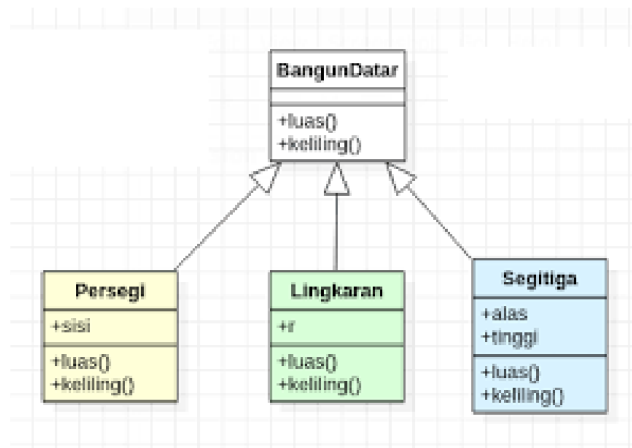
Contoh Sequence Diagram :



D. Class Diagram

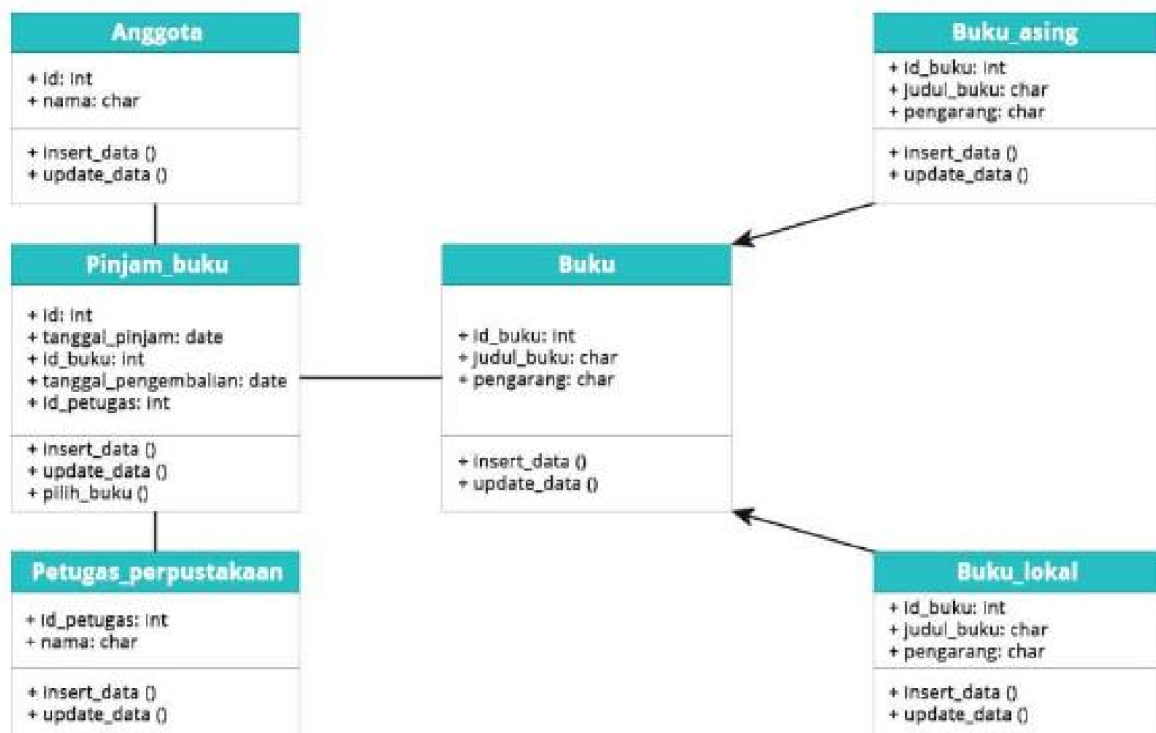
Class diagram atau diagram kelas adalah salah satu jenis diagram struktur pada UML yang menggambarkan dengan jelas struktur serta deskripsi class, atribut, metode, dan hubungan dari setiap objek. Ia bersifat statis, dalam artian diagram kelas bukan menjelaskan apa yang terjadi jika kelas-kelasnya berhubungan, melainkan menjelaskan hubungan apa yang terjadi. (Sumber : <https://www.dicoding.com/blog/memahami-class-diagram-lebih-baik/>)

Dalam penerapannya class diagram ini memiliki banyak fungsi namun untuk program yang cukup detail atau rumit terkadang karena saking banyaknya class proses ini digunakan untuk mempersingkat kita untuk menampilkan class-class yang penting dalam aplikasi kita dan hubungan-hubungan interaksi antar objek. Sebelumnya kita belajar tentang interaksi antar objek seperti ada inheritance dan generalisasi pada materi itu kita telah melihat bentuk notasi yang digunakan dalam class diagram.



Notasi / Simbol pada Class diagram :

No	Simbol	Notasi	Keterangan
1	<div style="border: 1px solid black; padding: 5px; width: fit-content;"> Nama Class -atribut -method() </div>	Class	Bagian atas class menunjukkan nama dari class, bagian tengah mengindikasikan atribut dari class, dan bagian bawah mendefinisikan method dari sebuah class
2	→	Interaction	Menunjukkan baik aliran pesan atau informasi antar class
3	⬅	Generalization	Menunjukkan jaringan semantik atau relasi pewarisan antara dua class, dimana relasi ini memungkinkan suatu class mewarisi dan method class lain



BAB I PENGENALAN SISTEM BASIS DATA

A. DATA DAN INFORMASI

Data adalah catatan atas sekumpulan fakta yang belum mempunyai arti bagi penerimanya dan masih memerlukan suatu pengolahan. Data dapat dinyatakan dalam bentuk karakter, angka, simbol, suara atau dalam bentuk simbol lainnya yang bisa digunakan sebagai bahan untuk melihat lingkungan, objek, kejadian ataupun sebuah konsep. Contoh data adalah data siswa, data guru, data mapatelajaran, dll.

Informasi merupakan hasil dari pengolahan data yang memiliki nilai tertentu dan bisa digunakan untuk pengambilan suatu keputusan. Data bisa dianggap sebagai objek dan informasi adalah suatu subjek yang bermanfaat bagi penerimanya. Misalnya : siswa sesuai dengan jurusan, jumlah siswa yang mengambil mapatelajaran basis data, dll.

B. BASIS DATA

Ramakrishnan dan **Gehrke** (2003), basis data adalah kumpulan data yang terdiri atas *entity* dan *relationship*, yang umumnya mendeskripsikan aktivitas dari satu organisasi atau lebih yang berhubungan.

Stephens dan **Plew** (2000) menyatakan, basis data sebagai mekanisme yang digunakan untuk menyimpan informasi atau data. Sedangkan menurut **Connolly** dan **Carolllyn** (2005), basis data adalah sebuah koleksi data yang dipakai bersama dan terhubung secara logis.

Jadi, basis data adalah kumpulan data yang terdiri dari atribut, entity dan relationship yang di rancang untuk memenuhi kebutuhan informasi dari sebuah organisasi.

C. KOMPONEN BASIS DATA

Sebuah basis data tersusun dari beberapa komponen, yaitu :

1. DATA

Data disimpan secara terintegrasi (*Integrated*), artinya basis data merupakan gabungan dari bermacam file aplikasi yang berbeda yang disusun dengan menghilangkan bagian – bagian yang merangkap. Sebagai alat penghubung digunakan kunci (*Key*). Data dipakai secara bersama sama (*Shared*), artinya masing – masing bagian dari suatu data dapat digunakan atau diakses secara bersama-sama dalam waktu yang bersamaan oleh pemakai untuk aplikasi yang berbeda. Komponen data dibagi menjadi 3, yaitu :

- ξ Data Operasional, yaitu data yang disimpan dalam basis data baik itu berupa data master maupun data transaksi

- ξ Data Masukan (Data Inputan) , data dari luar sistem yang dimasukkan melalui peralatan input (keyboard) yang dapat mengubah data operasional.
- ξ Data Keluaran (Data Output) yaitu data berupa laporan melalui peralatan output (scanner, printer,dll) sebagai hasil proses dari dalam sebuah sistem yang mengakses data operasional.

2. HARDWARE

Perangkat keras yang dibutuhkan untuk mengelola basis data, berupa komputer beserta kelengkapannya seperti monitor, keyboard, mouse dan lain sebagainya.

3. SOFTWARE (PERANGKAT LUNAK)

Aplikasi yang digunakan sebagai *interface* (antar muka) antara user (pengguna) dengan data fisik basis data, seperti program aplikasi (Inventory System, POS,dll) maupun Database Management System (DBMS) (MySQL, Oracle, Microsoft SQL Server, PostgreSQL, SQLite dll)

4. USER (PENGGUNA)

User pada basis data ini dibagi menjadi 4, yaitu :

a. User umum (*End User / Naïve User*)

User umum yaitu pemakai yang berinteraksi dengan sistem basis data melalui pemanggilan satu program permanen yang telah disediakan sebelumnya.

b. User Khusus (*Sophisticated User*)

User khusus yaitu pemakai yang berinteraksi dengan basis data tidak melalui program melainkan menggunakan bahasa *Query*.

c. Program Aplikasi (*Application Programmer*)

Program aplikasi yaitu user yang berinteraksi dengan basis data melalui Data Manipulation Language (DML) yang disertakan dalam program yang di tulis pada bahasa pemrograman induk seperti Visual Basic, PHP, Delphi, dll.

d. Database Administrator (*DBA*)

Database Administrator adalah tenaga ahli yang bertugas untuk mengontrol sistem basis data secara keseluruhan, meramalkan kebutuhan akan sebuah sistem basis data, mengatur dan merencanakan.

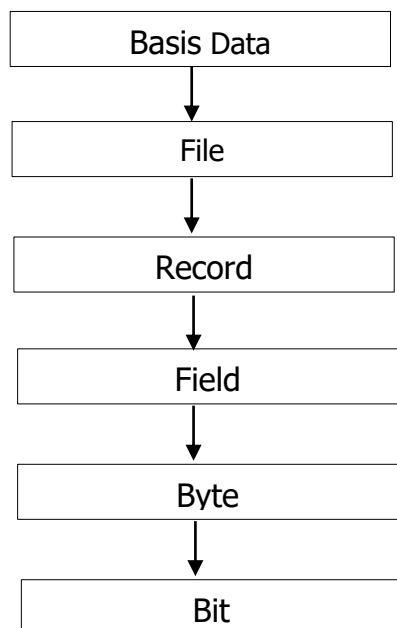
D. MANFAAT BASIS DATA

Banyak manfaat basis data dalam kehidupan kita. Jika kita amati, setiap perusahaan baik itu perusahaan yang memiliki skala kecil, menengah maupun besar kini telah menggunakan sistem informasi untuk membantu kegiatan operasionalnya. Jadi, bisa dikatakan bahwa basis data itu menjadi satu kesatuan yang tidak dapat dipisahkan dalam kehidupan sehari – hari. Pengolahan data tidak dapat dilepaskan begitu saja dalam kehidupan sehari – hari seperti ilmu pengetahuan, pendidikan, usaha dan bidang – bidang yang lainnya pasti memerlukan sistem pengolahan data agar masing – masing bidang tersebut dapat berjalan dengan lancar. Pemanfaatan basis data dilakukan dengan tujuan :

1. kecepatan dan kemudahan (speed)
2. efisiensi ruang penyimpanan (space)
3. keakuratan (accuracy)
4. ketersediaan (availability)
5. kelengkapan (completeness)
6. keamanan (security)
7. data dapat dipakai secara bersama (shareability)

E. HIERARKI BASIS DATA

Dalam basis data, urutan atau hierarki data sangatlah penting, adapun struktur hierarki data dalam basis data dari tertinggi sampai terendah sebagai berikut :



Keterangan :

- ☑ Basis Data : kumpulan dari file yang membentuk sebuah basis data.
- ☑ File atau / berkas adalah kumpulan dari record yang saling berkaitan dan memiliki format field yang sama, namun berbeda isi datanya.
- ☑ Record atau baris adalah gabungan dari sejumlah elemen data yang saling berkaitan.
- ☑ Field atau Elemen data atau atribut adalah satuan data terkecil yang tidak dapat dipecah lagi menjadi unit lain yang bermakna.
- ☑ Byte adalah atribut dari field yang berupa karakter yang membentuk nilai dari sebuah field

- ☑ Bit adalah bagian terkecil dari data secara keseluruhan, yaitu berupa karakter ASCII nol atau satu yang merupakan komponen byte.

F. BAHASA BASIS DATA

Cara berinteraksi antara pemakai dengan basis data telah diatur dalam Bahasa khusus yang ditetapkan oleh suatu perusahaan pembuat DBMS. Bahasa ini disebut Bahasa basis data, contoh dari Bahasa basis data adalah SQL, dBase, QUEL dan sebagainya.

Berdasarkan fungsinya Bahasa basis data dibedakan kedalam 3 bentuk, yaitu : DCL (Data Control Language), DDL (Data Definition Language) dan DML (Data Manipulation Language).

1. DCL (Data Control Language)

DCL (Data Control Language) merupakan bahasa SQL yang digunakan untuk mengontrol data dan server basis data.

2. DDL (Data Definition Language)

DDL (Data Definition Language) digunakan untuk membuat database / tabel, menghaous database / tabel, membuat key/index dan membuat relasi antar table. Hasil dari DDL adalah himpunan data yang disimpan secara khsuus pada kamus data (data dictionary) dan kamus data merupakansuatu metadata (suoerdata) yang mendeskripsikan data yang sesungguhnya.

3. DML (Data Manipulation Language)

DML (Data Manipulation Language) adalah bahasa basis data yang digunakan untuk melakukan manipulasi dan pengambilan data pada suatu basis data.

Manipulasi basis data dapat berupa :

- Menyisipkan atau menambah data pada file atau tabel dalam suatu basis data
- Menghapus data pada file atau tabel dalam suatu basis data
- Mengubah data pada file atau tabel dalam suatu basis data
- Mencari data pada file atau tabel dalam suatu basis data.

DML (Data Manipulation Language) merupakan bahasa yang miliki tujuan untuk mempermudah user dalam mengakses data. DML (Data Manipulation Language) memiliki 2 jenis, yaitu prosedural dan non-prosedural (deklaratif).

Query adalah pernyataan yang diajukan oleh pengguna untuk mengambil sebuah informasi di dalam suatu basis data. Query merupakan bagian dari DML yang digunakan untuk mengambil informasi yang disebut sebagai Query Language.

LATIHAN

Buatlah sebuah peta konsep mengenai materi “Basis Data”

Jawaban :

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

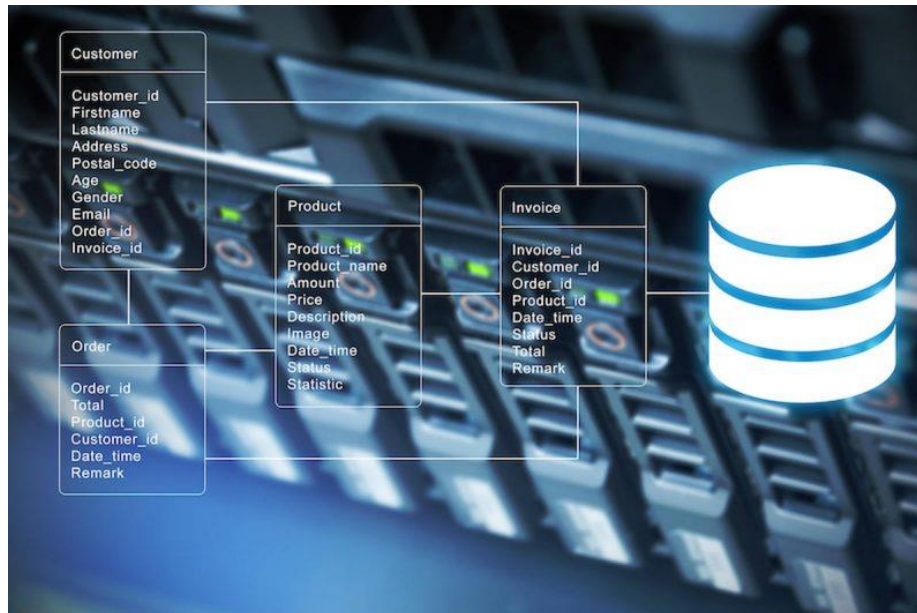
.....

Nilai	Tanda Tangan Guru

BAB 2 ENTITY RELATIONSHIP DIAGRAM (ERD)

1. KONSEP DASAR ENTITY RELATIONSHIP DIAGRAM (ERD)

Model entity relationship pertama kali dikenalkan oleh P.P.Chen pada tahun 1976. Model ini dirancang untuk menggambarkan persepsi dari user dan berisi objek – objek dasar yang disebut **entitas** dan berhubungan antar entitas disebut **relationship**.



Entity relationship diagram (ERD) merupakan notasi grafis dalam pemodelan data konseptual yang digunakan untuk memodelkan struktur data dan hubungan antar data. ERD juga menggambarkan antar hubungan

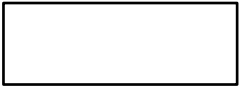
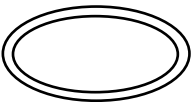
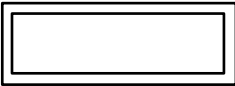
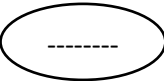



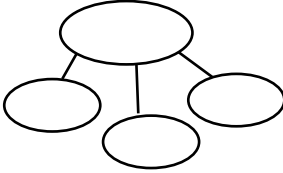
antara satu entitas dengan entitas yang lainnya yang memiliki sejumlah atribut dalam sistem yang terintegrasi.

Entity relationship diagram (ERD) digunakan oleh perancang sistem untuk memodelkan data yang nantinya akan dikembangkan menjadi basis data (database). Model Entity relationship diagram (ERD) ini akan membantu pada saat akan melakukan analisis dan perancangan basis data karena model Entity relationship diagram (ERD) ini menunjukkan bermacam – macam data yang dibutuhkan.

2. SIMBOL ENTITY RELATIONSHIP DIAGRAM (ERD)

Dalam membuat entity relationship diagram (ERD) digunakan beberapa simbol yaitu : entitas, relasi, atribut dan penghubung.

Berikut merupakan simbol – simbol yang digunakan dalam entity relationship diagram (ERD) :

SIMBOL	NAMA	SIMBOL	NAMA
	Entitas atau entity		Atribut multi-value
	Weak entity		Atribut Tamu / Foreign Key
	Relationship atau relasi		Atribut Kunci / Primary Key
	Atribut		Atribut komposit

A. ENTITAS (ENTITY)

Entitas (entity) adalah objek – objek dasar yang terkait di dalam sistem. Objek dasar dapat berupa orang, benda atau hal lain yang keterangannya perlu disimpan dalam basis data. Entitas disajikan dalam bentuk persegi panjang. Entitas diberi nama dengan kata benda dan dapat dikelompokkan dalam 4 (empat) kelas yaitu : role (peran), events (kejadian), locations (lokasi), tangible things / concepts (sesuatu yang tidak nyata / konsep). Contoh entitas : siswa, guru, mata pelajaran, dll. Contoh detail dari suatu entitas disebut instance. Contoh instance : siswa bernama cica, riri.

Entitas dibagi menjadi 2 tipe yaitu :

1. Entitas Kuat

Entitas kuat adalah entitas yang dapat berdiri sendiri, tidak bergantung kepada entitas lain, memiliki primary key.

2. Entitas Lemah

Entitas lemah adalah entitas yang tidak dapat berdiri sendiri, entitas yang bergantung kepada entitas lain dan tidak memiliki primary key.

B. RELASI (RELATIONSHIP)

Relasi adalah penghubung antara entitas yang satu dengan entitas yang lain dan merupakan bagian yang sangat penting dalam mendesign basis data. Bagian dari suatu relasi adalah :

1. Representasi Relasi

Relasi digambarkan dalam bentuk garis yang akan menghubungkan entitas – entitas yang saling berelasi. Nama relasi umumnya berupa kata kerja dan nama relasi harus unik dalam satu entity relationship diagram (ERD).

2. Derajat Relasi

Derajat relasi menunjukkan jumlah entitas yang terhubung dalam suatu relasi. Entitas – entitas yang terhubung dalam suatu relasi disebut partisipan. Derajat relasi dibagi menjadi 3, yaitu :

- Relasi biner yaitu relasi berderajat dua.
- Relasi terner yaitu relasi berderajat tiga artinya terdapat tiga entitas yang berpartisipasi dalam relasi terner.
- Relasi kompleks yaitu relasi yang menghubungkan lebih dari 2 entitas. Contoh: terdapat suatu kasus yang memiliki empat relasi atau berderajat empat yang dinamakan relasi quartener.

C. ATRIBUT

Atribut adalah property atau karakteristik dari entitas atau relationship yang menyediakan penjelasan detail tentang entitas atau relationship tersebut. Nilai atribut merupakan suatu data aktual atau informasi yang disimpan pada suatu atribut dalam suatu entitas atau relationship.

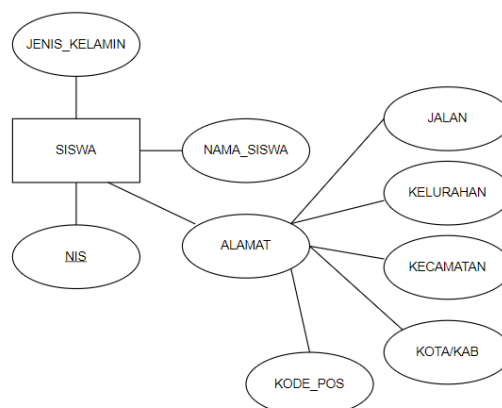
Berdasarkan karakteristik atau sifatnya, atribut dibagi beberapa kelompok yaitu :

1. Simple Attribute

Simple attribute adalah atribut terkecil yang tidak dapat dibagi lagi menjadi atribut yang lebih kecil. Contoh : jenis kelamin

2. Composite Attribute

Composite Attribute adalah atribut yang dapat dibagi menjadi atribut yang lebih kecil. Contoh : alamat



3. Single Valued Attribute

Single value attribute adalah suatu atribut yang hanya mempunyai satu nilai. Contoh: NIS, nama siswa, alamat pada tabel siswa merupakan atribut yang memiliki nilai tunggal.

4. Multi Valued Attribute

Multi valued attribute adalah atribut yang memiliki lebih dari satu nilai yang jenisnya sama dari sebuah data tunggal. Contoh : atribut hobi pada tabel siswa yang memiliki nilai ganda.

Contoh :

NIS	Nama_Siswa	Tgl_lahir	Alamat	Hobby
10109345	Fitrianingsih	05-05-2001	Jln. Pagarsih Gg. Siti Mariah No. 565/86	Membaca novel, nonton, traveling
10109332	Riza Hardian	19-01-2001	Jln. Permata Kopo No. 71/D	Futsal, renang
10109113	Siti Nurbaya	31-12-2001	Jln. Sadang No. 34 Kab. Bandung	Traveling, nonton

5. Mandatory Attribute dan Non-Mandatory Attribute

Mandatory attribute adalah atribut yang harus memiliki nilai (tidak boleh null)

Non-mandatory attribute adalah atribut yang boleh tidak memiliki nilai (boleh null)

Contoh :

NIS	Nama_Siswa	Tgl_lahir	Alamat	Hobby
10109345	Fitrianingsih			
10109332	Riza Hardian			
10109113	Siti Nurbaya			

6. Key Attribut

Key attribut adalah satu atau beberapa atribut yang memiliki nilai unik sehingga dapat di digunakan untuk membedakan data pada suatu baris/record dengan baris/record yang lainnya pada suatu entitas. Key attribute di bagi menjadi beberapa bagian, yaitu:

a. Foreign key

Foreign key adalah atribut yang melengkapi suatu relasi, biasanya foreign key ditempatkan pada entitas anak dan sama dengan primary key pada entitas induk dan kemudian direlasikan.

b. Primary key

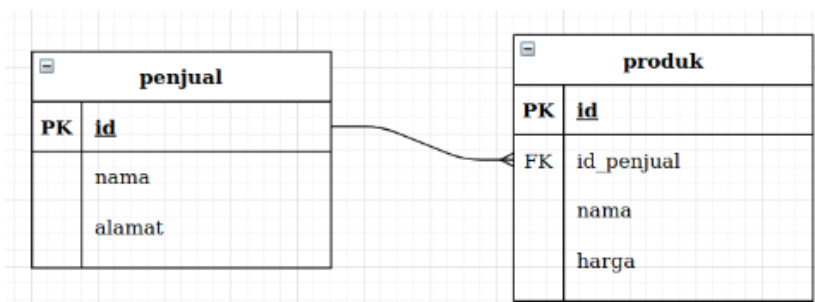
Primary key adalah kandidat kunci yang dipilih sebagai kunci utama untuk mendefinisikan baris pada tabel, primary key memiliki nilai atribut yang unik dan dipakai untuk membedakan satu kolom dengan kolom yang lainnya.

D. KARDINALITAS

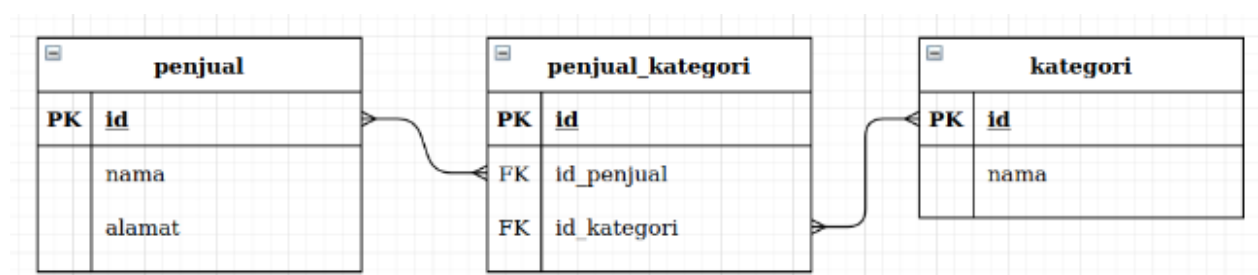
Kardinalitas adalah tingkatan / derajat pada sebuah entitas yang dapat menghubungkan antara entitas yang satu dengan entitas yang lainnya. Secara umum ada 3 bentuk kardinalitas antar himpunan entitas, yaitu :

1. Satu ke satu (one to one) artinya satu anggota entitas hanya boleh berelasi dengan satu anggota entitas yang lainnya.
2. Satu ke banyak (one to many) atau banyak ke satu (many to one), kardinalitas one to many maupun many to one di anggap sama karena kardinalitas selalu dilihat dari dua sisi.

Contoh :



3. Banyak ke banyak (many to many) artinya setiap anggota entitas diperbolehkan memiliki hubungan yang banyak dengan entitas lainnya. Contoh :



3. METODOLOGI ENTITY RELATIONSHIP DIAGRAM (ERD)

Membuat ERD tidak dapat dilakukan begitu saja, akan tetapi diperlukan tahapan yang disebut metodologi, yaitu :

1. Tentukan entitas terlebih dahulu, lakukan dengan cermat, teliti dan baik sehingga dapat menjawab persoalan yang ada.
2. Tentukan relasi antar entitas, bagaimana hubungan antara entitas satu dengan yang lainnya.
3. Gambarlah ERD yang sifatnya temporal atau sementara, hal ini diperlukan untuk memberikan gambaran secara umum.
4. Buat atau isi kardinalitas, hal ini diperlukan untuk menentukan jumlah kejadian.
5. Tentukan *primary key* atau kata kunci utama dalam sebuah entitas.

6. Menggambar ERD berdasarkan kunci utama.
7. Tentukan atribut yang dibutuhkan dari setiap entitas yang ada.
8. Menggambar ERD secara lengkap dengan atribut.
9. Lakukan evaluasi dan pemeriksaan terhadap hasil, apabila terdapat kekeliruan dapat mengulang langkahnya.

.....LATIHAN 1.....

Pada saat mendaftar menjadi anggota perpustakaan di smk marhas margahayu, dicatatlah nama siswa, nomor idnuk siswa, kelas siswa dan alamat siswa. Setelah itu mereka baru bisa meminjam buku di perpustakaan. Buku-buku yang dimiliki perpustakaan smk marhas margahayu sangat banyak sekali jumlahnya. Tiap buku memiliki data nomor buku, judul, pengarang, penerbit, tahun terbit. Satu buku bisa ditulis oleh beberapa pengarang. Tentukan *entitas, atribut dan relasi* dari deskripsi di atas, dengan menggambar ERDnya.

Jawaban :

.....

.....

.....

.....

.....

.....

Nilai	Tanda Tangan Guru

.....LATIHAN 2.....

Toko X adalah sebuah toko dimana gudang untuk menyimpan barang-barangnya lebih dari satu. Barang-barang yang ada di dalam gudang tersebut dikirim oleh beberapa supplier. Toko tersebut menjual banyak jenis barang, diantaranya adalah barang pangan dan barang sandang. Tentukan *entitas, atribut dan relasi* dari deskripsi di atas, dengan menggambar ERDnya.

Jawaban :

.....

.....

.....

.....

.....

.....

.....

.....

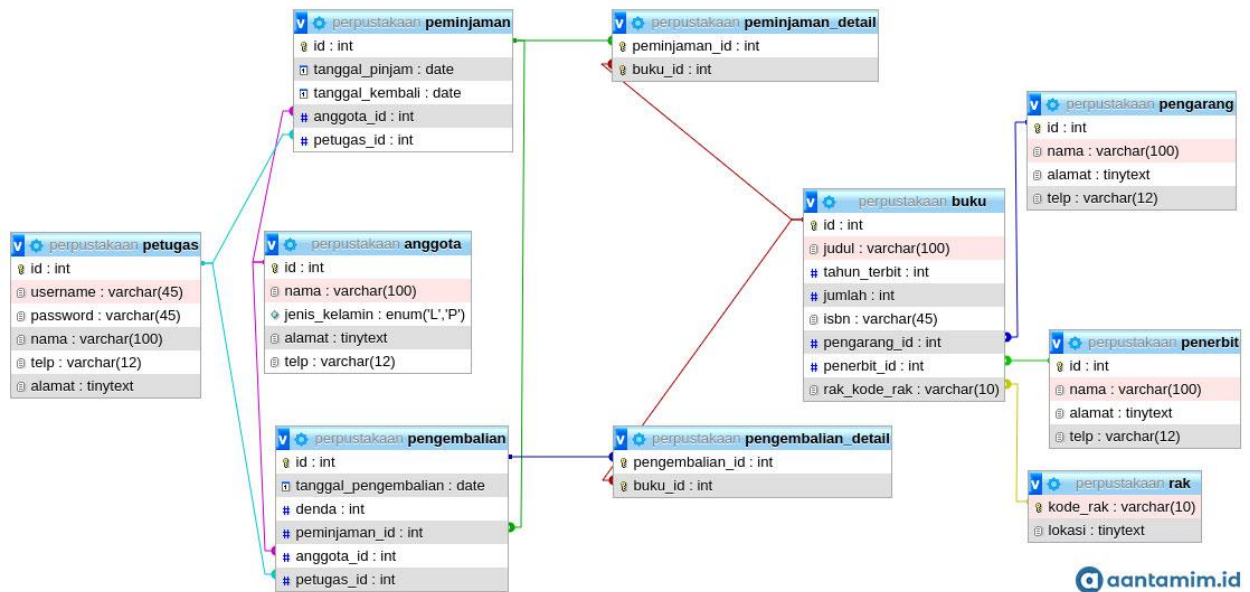
.....

.....

Nilai	Tanda Tangan Guru

.....LATIHAN 3.....

Buatlah ERD dan Kamus datanya dari skema relasi dibawah ini

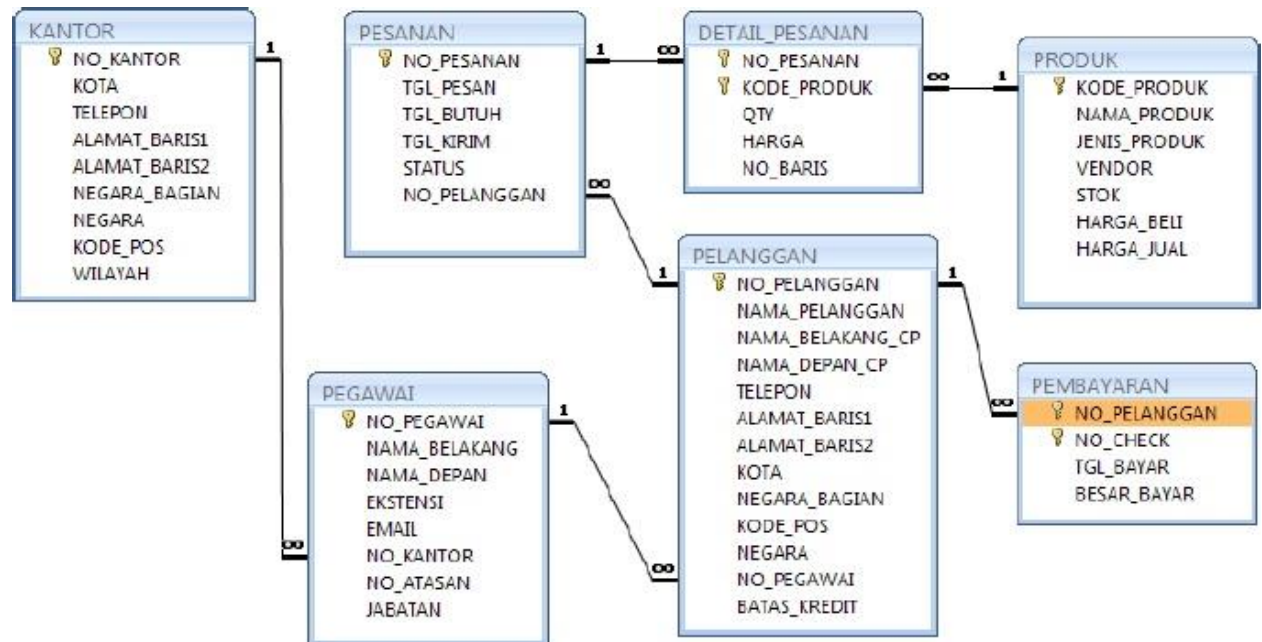


Jawaban :

Nilai	Tanda Tangan Guru

.....LATIHAN 4.....

Buatlah ERD dan Kamus datanya dari skema relasi dibawah ini



Jawaban :

.....

.....

.....

.....

.....

.....

.....

Nilai	Tanda Tangan Guru

BAB 3 TIPE DATA PADA BASIS DATA

Pada bab ini kita akan mempelajari mengenai definisi tipe data dalam basis data serta tipe data dan fungsinya pada aplikasi DBMS SQL-Server, MySQL dan Microsoft Access. Masing – masing aplikasi tersebut mempunyai tipe data sendiri namun fungsi tipe data yang sama dalam aplikasi yang berbeda dapat mempunyai fungsi yang sama.

1. TIPE DATA

Tipe data mendefinisikan jenis nilai yang dapat dimiliki suatu kolom. Tipe data dapat berupa integer data, character data, date and time data, binary strings dan masih banyak lagi. Tipe data pada setiap atribut (kolom) digunakan untuk keperluan penentuan struktur setiap tabel. Penetapan tipe data ini akan berpengaruh pada nilai yang akan disimpan atau di input pada setiap atribut (kolom) tersebut.

2. MACAM – MACAM TIPE DATA

Tipe data digunakan untuk mendefinisikan suatu field atau kolom. Setiap kolom yang dibuat harus didefinisikan terlebih dahulu. Adapun jenis – jenis tipe data yang digunakan pada basis data ada bermacam – macam, yaitu sebagai berikut :

a. TIPE DATA PADA SQL SERVER

Tipe data pada SQL Server memiliki beberapa jenis, yaitu :

TIPE DATA	KETERANGAN
INT	Tipe data bilangan bulat yang dapat menerima nilai mulai dari -2^{31} hingga $2^{31} - 1$. Tipe data ini menghabiskan 4 bytes untuk menyimpan data pada harddisk.
BIT	Tipe data BIT hanya bisa menerima inputan angka 1 dan 0 sebagai nilai (atau bisa juga null yang artinya tidak ada nilai). Tipe data ini dapat membantu untuk mendapatkan nilai (output) yang bernilai true atau false, yes atau no.
SMALLINT	Tipe data SMALLINT mirip dengan tipe data INT, hanya saja pada tipe data SMALLINT nilai yang diterima lebih kecil dari tipe data INT. tipe data ini dapat menerima nilai mulai dari -2^{15} hingga $2^{15}-1$. SMALLINT membutuhkan 50% memori yang digunakan INT yakni 2 byte.

TIPE DATA	KETERANGAN
BIGINT	Tipe data BIGINT mirip juga dengan tipe data INT hanya saja tipe data ini dapat menerima nilai mulai dari -2^{63} hingga $2^{63} - 1$. Tipe data ini menghabiskan 8 bytes untuk menyimpan data pada harddisk.
DECIMAL	Tipe data DECIMAL menggunakan 2 parameter untuk menentukan tingkat presisi nilai yang diterima yaitu precision dan scale. Precision adalah jumlah digit yang bisa diterima oleh field dan scale adalah jumlah angka dibelakang koma yang bisa diterima oleh field. Sebagai contoh : misalnya kita membuat parameter precision sebanyak 5 dan scale 2 maka fieldnya bisa menerima nilai seperti 123,45. Tipe data decimal menerima nilai mulai dari -10^{38} hingga $10^{38} - 1$. Tipe data decimal menghabiskan 5-17byte untuk menyimpan data pada harddisk, tergantung pada tingkat presisi nilai yang diinputkan.
NUMERIC	Tipe data NUMERIC pada dasarnya sama dengan tipe data decimal namun tipe data numeric ini untuk mendefinisikan angka pecahan baik fixed decimal maupun floating point. Nilai n adalah jumlah bytes total dan p adalah presisi angka dibelakang koma.
DATE TIME	Tipe data date time dapat menerima nilai tanggal dan waktu yang berfungsi untuk mendefinisikan tanggal, menyimpan tahun, bulan, hari, jam, menit, detik dan seperseribu detik (mili seconds). Tipe data ini menghabiskan 8 bytes untuk menyimpan data pada harddisk.
MONEY	Tipe data money merupakan bilangan pecahan dengan 4 angka dibelakang koma, digunakan untuk perhitungan mata uang. Tipe data ini menghabiskan 8 bytes untuk menyimpan data pada harddisk.
TEXT	Tipe data text dapat menyimpan teks sampai dengan 2GB. Text disebut juga dengan binary large objects (BLOBs).
IMAGE	Tipe data image berfungsi untuk mendefinisikan binary data untuk menyimpan image, seperti JPG, GIF, TIFF dan lain lain.
CHAR	Tipe data char digunakan untuk mendefinisikan string sepanjang mempunyai karakter dan dapat digunakan untuk menginputkan data karakter non-unicode dengan jumlahkarakter yang fix. Tipe data char dapat menerima nilai hingga 8000 karakter dan jumlah bytes yang dibutuhkan tergantung dari jumlah karakter yang diinputkan. Apabila

TIPE DATA	KETERANGAN
	jumlah karakter yang di inputkan adalah 1 karakter maka membutuhkan 1 byte.
VARCHAR	Tipe data varchar ini dapat mendefinisikan string sepanjang variable n. tipe data varchar dapat digunakan bagi user yang tidak mengetahui secara pasti jumlah karakter yang akan di inputkan oleh user. Tipe data varchar dapat menerima nilai hingga 800 karakter. Jadi, jika pada tipe data char user mendefinisikan chat(5) maka user akan selalu membutuhkan 5 bytes untuk menyimpan data pada harddisk walaupun jumlah karakter yang di inputkan hanya 1 hingga 4 karakter maka pada tipe data varchar ini jumlah bytes yang dibutuhkan akan lebih fleksible.
FLOAT	Tipe data float ini mirip dengan tipe data decimal tapi hanya saja parameter scale pada tipe data float bisa menerima nilai yang tak terhingga, seperti pada nilai pi (π). Float berfungsi untuk mendefinisikan angka pecahan (floating point). Nilai n adalah jumlah angka yang di tampung. Tipe data float menghabiskan 8 bytes untuk menyimpan data pada harddisk.
BINARY	Tipe data binary dapat menerima nilai dengan maksimum 8000 bytes data. Tipe data ini diinterpretasikan sebagai string dari bit dan berfungsi untuk menyimpan bit pattern seperti heksadesimal.

b. TIPE DATA PADA MICROSOFT ACCESS

Tipe data pada microsoft access ini memiliki beberapa tipe data, seperti berikut:

TIPE DATA	KETERANGAN	KAPASITAS
Text	Digunakan untuk teks atau kombinasi antara teks dan bilangan	Panjang max sampai dengan 255 karakter
Memo	Digunakan untuk jumlah teks yang lebih	Menyimpan hingga 65.536 karakter
Byte	Memungkinkan bilangan bulat dari 0 hingga 255	1 bytes
Integer	Memungkinkan bilangan bulat antara -32.768 dan 32.767	2 bytes

TIPE DATA	KETERANGAN	KAPASITAS
Long	Memungkinkan bilangan bulat antara -2,147,483,648 dan 2,147,483,647.	4 bytes
Single	Presisi tunggal floating-point. Akan menangani sebagian besar desimal.	4 bytes
Double	Presisi ganda floating-point. Akan menangani sebagian besar desimal.	8 bytes
Currency	Digunakan untuk mata uang. Menyimpan hingga 15 digit seluruh dolar, ditambah 4 angka desimal.	8 bytes
Autonumber	Penomoran otomatis yang biasanya dimulai dari 1.	4 bytes
Date/Time	Digunakan untuk tanggal dan waktu	8 bytes
Yes/No	Tipe data logika dan digunakan untuk menampilkan yes/no, true/false, atau on/off. Jika dalam kode biasanya menggunakan konstanta benar dan salah (setara dengan 1 dan 0)	1 bit
Ole Object	Digunakan untuk menyimpan gambar, audio, video, atau BLOB lainnya (binary large objects)	1 bytes
Hyperlink	Berisi tautan ke file lain, termasuk halaman web	Up to 1 GB
Lookup Wizard	Tipe data untuk daftar opsi, yang kemudian dapat dipilih dari daftar drop-down.	4 bytes

c. TIPE DATA PADA MySQL

Tipe data MySQL dibagi menjadi beberapa bagian, yaitu :

1. TIPE DATA BILANGAN

Tipe data bilangan dibagi menjadi beberapa bagian, yaitu :

TIPE DATA	KETERANGAN
TINYINT	Digunakan untuk menyimpan data bilangan bulat positif dan negatif. Jangkauan : -128 s.d 127 dan ukuran : 1 byte (8bit)
SMALLINT	Digunakan untuk menyimpan data bilangan bulat positif dan negatif. Jangkauan : -8.388.608 s/d 8.388.607 dan ukuran : 3 byte (24 bit)
INT	Digunakan untuk menyimpan data bilangan bulat positif dan negatif. Jangkauan : -2.147.483.648 s/d 2.147.483.647
BIGINT	Digunakan untuk menyimpan data bilangan bulat positif dan negatif. Jangkauan : $\pm 9,22 \times 10^{18}$ s/d 8 byte (64 bit)

TIPE DATA	KETERANGAN
FLOAT	Digunakan untuk menyimpan data bilangan pecahan positif dan negatif presisi tunggal. Jangkauan : -3.402823466E+38 - 1.175494351E38,0, dan 1.175494351E-38 s/d 3.402823466E+38 dan untuk ukurannya : 4 byte (32 bit)
DOUBLE	Digunakan untuk menyimpan data bilangan pecahan positif dan negatif presisi ganda. Jangkauan : -1.79...E+308 s/d -2.22...E-308,0
REAL	Merupakan sinonim dari double
DECIMAL	Digunakan untuk menyimpan data bilangan pecahan positif dan negatif. Jangkauan : -1.79...E+308 s/d -2.22...E-308,0, dan 2.22...E-308 s/d 1.79...E+308 dan ukurannya : 8 byte (64 bit)

2. TIPE DATA JAM, TANGGAL, DAN HARI

Tipe data jam, tanggal dan hari dibagi menjadi beberapa bagian, yaitu :

TIPE DATA	KETERANGAN
DATE	Digunakan untuk menyimpan data tanggal. Jangkauan : 1000-01-01 s/d 9999-12-31 (YYYY-MMDD) Ukuran : 3 byte
TIME	Digunakan untuk menyimpan data waktu Jangkauan : -838:59:59 s/d +383:59:59 (HH:MM:SS) Ukuran : 3 byte
DATETIME	Digunakan untuk menyimpan data tanggal dan waktu. Jangkauan : '1000-01-01 00:00:00' s/d '9999-12-31 23:59:59' Ukuran : 8 byte
TIMESTAMP	Kombinasi tanggal dan jam saat tabel / data diakses dengan jangkauan '1970-01-01 00:00:00 s/d 2037'
YEAR	Digunakan untuk menyimpan data tahun dari tanggal. Jangkauan : 1900 s/d 2155 Ukuran : 1 byte

3. TIPE DATA LAINNYA

Tipe data lainnya pada MySQL adalah sebagai berikut :

TIPE DATA	KETERANGAN
CHAR	Digunakan untuk menyimpan data string ukuran tetap Jangkauan : 0 s/d 255 karakter
VARCHAR	Digunakan untuk menyimpan data string ukuran dinamis Jangkauan : 0 s/d 255 karakter (versi 4.1), 0 s/d 65.535 (versi 5.0.3)
TINYBLOB, TINYTEXT	L+2 byte, dengan L<224. Tipe TEXT atau BLOB dengan panjang maksimum 1677215 karakter
MEDIUMBLOB, MEDIUMTEXT	L+2 byte dengan L<32. Tipe TEXT atau BLOB dengan panjang maksimum 4294967295 karakter
ENUM	Digunakan untuk enumerasi (kumpulan data)
SET	Digunakan untuk combination (himpunan data). Jangkauan : sampai dengan 255 string anggota

BAB 4 STRUCTURE QUERY LANGUAGE (SQL)



SQL (***Structured Query Language***) merupakan suatu bahasa yang terstruktur karena SQL memiliki beberapa aturan yang telah distandarkan oleh asosiasi yang bernama ANSI

(***American National Standard Institute***). SQL memungkinkan user untuk mengakses informasi tanpa harus mengetahui dimana lokasinya atau bagaimana informasi tersebut disusun. SQL didukung oleh DBMS seperti MySQL Server, MsSQL, PostgreSQL, Interbase, dan Oracle. Selain itu, SQL juga didukung oleh basis data yang bukan server seperti Ms. Access maupun Paradox.

Berdasarkan fungsinya Bahasa basis data dibedakan kedalam 3 bentuk, yaitu : DCL (Data Control Language), DDL (Data Definition Language) dan DML (Data Manipulation Language).

1. DCL (DATA CONTROL LANGUAGE)

DCL merupakan sub bahasa SQL yang digunakan untuk mengontrol data dan server basis data. Yang termasuk kedalam perintah dasar DCL adalah **GRANT** dan **REVOKE**.

a. **GRANT**

GRANT digunakan untuk mengecek hak akses yang terdapat pada user di MySQL,

```
SHOW GRANTS FOR "nama_user"@"lokasi_user";
```

Contoh :

```
SHOW GRANTS FOR "admin_dosen"@"localhost" \G
```

§ **Catatan** : User dan lokasi user diatas saya dapatkan dari artikel sebelumnya. Hak Akses pada user **admin_dosen** yaitu dapat Melihat, Menginput, Mengedit dan Menghapus data dari tabel **daftar_dosen** yang terdapat pada **database idmysql**.

b. **REVOKE**

REVOKE digunakan untuk menghapus hak akses pada user MySQL. Penghapusan hak akses ini terdapat suatu kebijakan pada user pengguna atau user root mengadakan maintenance user. Selain itu, kita tidak harus menghapus user pada MySQLnya apa bila terjadi perubahan data tapi kita hanya cukup menghapus hak aksesnya saja yang terdapat pada user tersebut.

```
REVOKE hak_akses ON nama_database.nama_tabel FROM  
"nama_user"@"lokasi_user";
```

§ **Keterangan** : **hak_akses** adalah Jenis hak akses yang akan dihapus.
nama_database adalah database yang akan dilakukan action.
nama_tabel adalah tabel yang hak aksesnya akan dihapus.
nama_user adalah user yang akan dihapus hak aksesnya.
lokasi_user adalah lokasi user seperti localhost dan IP Address.

§ Contoh :

```
REVOKE insert, update, delete ON idmysql.daftar_dosen FROM  
"admin_dosen"@"localhost";
```

2. DDL (DATA DEFINITION LANGUAGE)

Struktur basis data yang menggambar atau mewakili desain basis data secara keseluruhan dispesifikasikan dengan bahasa khusus, yaitu DDL. Melalui bahasa ini kita dapat membuat table baru (**create table**), indeks, mengubah table, menentukan struktur penyimpanan table dan lainnya. Hasil dari perintah DDL adalah himpunan definisi data yang disimpan secara khusus pada kamus data (**data dictionary**). Kamus data merupakan suatu metadata (superdata), yaitu data yang mendeskripsikan data yang sesungguhnya. Adapun perintah dasar DDL adalah **CREATE, ALTER, RENAME, DROP**.

a. CREATE

CREATE digunakan untuk membuat database, table. Perintah SQL CREATE sebagai berikut :

```
♥ CREATE DATABASE nama_database;  
♥ CREATE TABLE nama_tabel (nama_kolom1 tipe_data,  
nama_kolom2 tipe_data, ...);
```

Contoh :

```
♥ CREATE DATABASE DB_Marhas;  
♥ CREATE TABLE Siswa (NIS Int(15), Nama_siswa  
Varchar(100), Tempat_Lahir_siswa Varchar(50),  
Jenis_kelamin_siswa Enum('L', 'P'), Alamat_siswa  
Varchar(100), No_Tlp INT(20));
```

b. ALTER TABEL

ALTER TABEL digunakan untuk menambah atau menghapus kolom dalam suatu table. Perintah SQL ALTER TABEL sebagai berikut :

```
♥ ALTER TABLE nama_tabel ADD nama_kolom tipe_data; //  
untuk menambah kolom  
♥ ALTER TABLE nama_tabel DROP COLUMN nama_kolom; //  
untuk menghapus kolom
```

Contoh :

```
♥ ALTER TABLE Siswa ADD Kelas VARCHAR;  
♥ ALTER TABLE Siswa DROP Kelas;
```

c. RENAME

RENAME digunakan untuk merubah nama objek. Perintah SQL RENAME sebagai berikut :

```
♥ RENAME TABLE nama_tabel TO nama_tabel_baru;
```

Contoh :

♥ **RENAME TABLE Siswa TO Siswa_Marhas;**

d. DROP

DROP digunakan untuk menghapus database dan tabel. Perintah SQL DROP sebagai berikut :

♥ **DROP TABLE nama_tabel;**
♥ **DROP DATABASE nama_database;**

Contoh :

♥ **DROP TABLE Siswa;**
♥ **DROP DATABASE DB_Marhas;**

3. DML (DATA MANIPULATION LANGUAGE)

Data Manipulation Language (DML) adalah bentuk basis data untuk melakukan manipulasi dan pengambilan data pada suatu basis data. Manipulasi data pada basis data dapat berupa :

a. Mengambil data (**SELECT**) pada file atau table dalam suatu basis data.

SELECT digunakan untuk mengambil data dari table dalam basis data

Perintah SQL SELECT sebagai berikut :

♥ **SELECT * FROM nama_tabel;**
♥ **SELECT nama_kolom FROM nama_tabel;**

Contoh :

ξ **SELECT * FROM Siswa;**
ξ **SELECT Nama_siswa, No_Tlp FROM Siswa;**

♥ **SELECT WHERE**

Select where digunakan untuk kriteria seleksi. Perintah SQL SELECT WHERE sebagai berikut :

♥ **SELECT nama_kolom FROM nama_tabel WHERE kolom operator nilai;**

Contoh :

ξ **SELECT * FROM Siswa WHERE Tempat_Lahir_siswa = 'Bandung';**

Berdasarkan klausa WHERE diatas, operator yang dapat digunakan adalah sebagai berikut :

OPERATOR	KETERANGAN
=	Sama dengan
>	Lebih besar dari
<	Kurang dari
<>	Tidak sama dengan
>=	Lebih besar sama dengan
<=	Kurang dari sama dengan
BETWEEN	Diantara
LIKE	Mencari suatu pola

☑ **CATATAN**

Tanda petik tunggal ('...') pada SQL digunakan untuk nilai teks sedangkan untuk nilai numerik tidak diberi tanda petik tunggal ('...').

♥ **SELECT DISTINCT**

SELECT DISTINCT digunakan jika ada nilai yang sama kemudian SQL akan mengambil salah satunya. Perintah SQL SELECT DISTINCT sebagai berikut:

♥ **SELECT DISTINCT nama_kolom FROM nama_tabel;**

Contoh :

ξ **SELECT DISTINCT Nama_siswa FROM Siswa;**

♥ **SELECT LIKE**

SELECT LIKE digunakan untuk menentukan pencarian data berdasarkan pola tertentu pada suatu kolom. Perintah SQL SELECT LIKE sebagai berikut :

```
SELECT nama_kolom FROM nama_tabel WHERE nama_kolom LIKE pola;
```

Contoh :

```
ξ SELECT Nama_siswa FROM Siswa WHERE  
Tempat_Lahir_siswa LIKE 'B%';
```

☑ **CATATAN**

Simbol '%' digunakan untuk menentukan wildcard (sembarang huruf), baik sebelumnya maupun sesudah pola

- b. Penghapusan data (**DELETE**) pada file atau table dalam suatu basis data.

Perintah SQL DELETE sebagai berikut :

```
DELETE FROM nama_tabel WHERE nama_kolom = nilai;
```

Contoh :

```
DELETE FROM Siswa WHERE Nama_siswa = 'Fitria';
```

Dan untuk menghapus semua data pada table tanpa menghapus table dengan menggunakan perintah SQL sebagai berikut :

```
DELETE * FROM nama_tabel; atau  
DELETE FROM nama_tabel;
```

Contoh :

```
DELETE * FROM Siswa; atau  
DELETE FROM Siswa;
```


- c. Pengubahan data (**UPDATE**) pada file atau table dalam suatu basis data.

UPDATE digunakan untuk memodifikasi baris yang ada dalam sebuah table.

Perintah SQL UPDATE sebagai berikut :

```
UPDATE nama_tabel SET Nama_kolom = nilai_baru  
WHERE nama_kolom = nilai;
```

Contoh :

```
UPDATE Siswa SET Tempat_Lahir_siswa = 'Cirebon' WHERE  
Nama_siswa = 'FITRIA';
```

- d. Menyisipkan data baru (**INSERT**) pada file atau table dalam suatu basis data.

Pernyataan **INSERT** digunakan untuk menambahkan baris data baru ke table.

Perintah SQL INSERT sebagai berikut :

```
♥ INSERT INTO nama_tabel VALUES (nilai1, nilai2, ...);  
♥ INSERT INTO nama_tabel (kolom1, kolom2, ...) VALUES (nilai1,  
nilai2, ...); // untuk menentukan kolom yang akan diisi dengan data  
baru
```

Contoh :

```
ξ INSERT INTO Siswa VALUES ('10109345', 'Fitrianingsih2',  
'Cimahi', 'P', 'Jl. Pagarsih', '087878685533');
```

```
ξ INSERT INTO Siswa (Nama_siswa, Tempat_Lahir_Siswa)  
Values ('Sakhiya', 'Bandung');
```

DML merupakan bahasa SQL yang bertujuan untuk memudahkan user untuk mengakses data – data. Ada 2 jenis DML, yaitu :

- Procedural, digunakan untuk memudahkan user menentukan data apa yang diinginkan dan bagaimana cara mendapatkan data tersebut. Contoh : dBase, FoxBase.
- Nonprosedural (**deklaratif**), digunakan untuk memudahkan user menentukan data apa yang diinginkan tanpa menyebutkan bagaimana cara untuk mendapatkan data tersebut. Contoh : SQL, QBE.

Query adalah perintah SQL yang digunakan untuk mengambil informasi di dalam suatu basis data. Query yang merupakan bagian dari DML yang digunakan untuk mengambil informasi disebut **Query Language**.