BAB II
TINJAUAN PUSTAKA

II.1. Pengertian Sistem Pakar

Menurut (Rosnelly Rika ; 2012 : 2) Sistem Pakar (*Expert System*) adalah sistem komputer yang ditujukan untuk meniru semua aspek (*emulates*) kemampuan pengambilan keputusan (*decision making*) seorang pakar. Sistem pakar memanfaatkan secara maksimal pengetahuan khusus selayaknya seorang pakar untuk memecahkan masalah.

Menurut (Maruli Tua Nahampun ; 2014 : 56) Sistem Pakar adalah salah satu cabang dari *Aftifical Intelligence* yang membuat penggunaan secara *knowledge* (pengetahuan) yang khusus untuk menyelesaikan masalah tingkat manusia yang pakar (ahli). Seorang pakar adalah orang memiliki keahlian dalam bidang tertentu , yaitu pakar yang mempunyai pengetahuan atau kemampuan khusus yang orang lain tidak mengetahui atau mampu dalam bidang yang dimilikinya.

Sistem Pakar adalah sebuah perangkat lunak komputer yang memiliki basis pengetahuan untuk domain tertentu dan menggunakan penalaran inferensi menyerupai seorang pakar dalam memecahkan masalah.

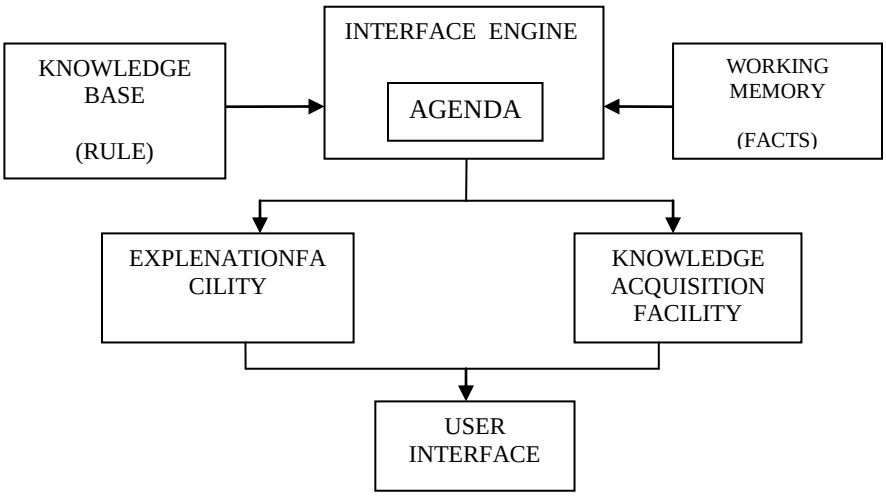
Secara umum, sistem pakar (*Expert System*) adalah sistem yang berusaha mengadopsi pengetahuan manusia ke komputer, agar komputer dapat menyelesaikan masalah seperti yang biasa dilakukan oleh para ahli. Dengan sistem pakar ini, orang awam pun dapat menyelesaikan masalah yang cukup rumit

yang sebenarnya hanya dapat diselesaikan dengan bantuan para ahli. Tujuan pengembangan sistem pakar sebenarnya tidak untuk menggantikan peran para pakar, namun untuk mengimplementasikan pengetahuan para pakar ke dalam bentuk perangkat lunak, sehingga dapat digunakan oleh banyak orang dan tanpa biaya yang besar. Untuk membangun sistem yang difungsikan untuk menirukan seorang pakar manusia harus bisa melakukan hal-hal yang dapat dikerjakan oleh para pakar. Untuk pembangun sistem yang seperti itu maka komponen-komponen dasar yang minimal harus dimiliki adalah sebagai berikut:

1. Antar muka (user interface).
2. Basis pengetahuan (knowledge base).
3. Mesin inferensi (Inference Engine). (Yasidar Nur Istiqomah; 2013 : 34).

II.1.1. Struktur Sistem Pakar

Adapun struktur sistem pakar dapat dilihat pada Gambar II.1.



Gambar II.1. : Struktur Sistem Pakar

Sumber : (Rosnelly Rika ; 2012 : 13)

Komponen yang terdapat dalam struktur sistem pakar ini adalah *knowledge base (rules), inference engine, working memory, explanation facility, knowledge acquisition facility, dan user interface*. (Rika Rosnelly; 2012: 13).

II.2. Metode Dempster-shafer

Teori *Dempster-shafer* (DST) merupakan teori matematika dari *evidence.* Teori ini dapat memberikan sebuah cara untuk menggabungkan *evidence* dari beberapa sumber dan mendatangkan atau memberikan tingkat kepercayaan (direpresentasikan melalui fungsi kepercayaan) dimana mengambil dari seluruh *evidence* yang tersedia.

Secara Umum teori *Dempster –Shafer* ditulis dalam suatu interval : **[*Belief, Plausibility*]** Belief (*Bel*) adalah ukuran kekuatan evidence dalam

mendukung suatu himpunan proposisi. Jika bernilai 0 sampai 1 maka mengindikasikan bahwa tidak ada *evidence* , dan jika bernilai 1 menunjukkan adanya kepastian.

$$PI(S) = 1 - Bel(\overline{S})$$

Pada teori *Dempster – Shafer* dikenal adanya *frame of discrement* yang dinotasikan dengan θ. *Frame* ini merupakan semesta pembicaraan dari sekumpulan *hipotesis*. Tujuan nya adalah mengaitkan ukuran kepercayaan elemen-elemen θ. Tidak semua evidence secara langsung mendukung tiap-tiap elemen. (Anis Mistanti, 2014 : 76).

Menurut (Maruli Tua Nahampun ; 2014 : 57) Studi kasus Metode Dempster Shafer Untuk menganalisis gejala-gejala yang diberikan oleh pasien untuk mendapatkan kemungkinan nama penyakitnya, dilakukan dengan menghitung nilai densitas dari gejala dengan menghitung nilai kepercayaan menggunakan rumus Demster-shafer.

$$m3(Z) = \frac{\sum x \cap y \blacksquare Z\ m^1(x)\ m^2(y)}{1 - \sum x \cap y = o\ m^1(x)\ m^2(y)}$$

Seorang pasien mengalami gejala penyakit pada kelapa sawit dengan diagnosa dokter penyakit yang mungkin dideritanya adalah Garis Kuning, Busuk Kuncup atau Tanjuk. Gejala-1 : Daun tampak mengering dan gugur apabila diketahui nilai kepercayaan setelah dilakukan observasi nyeri ulu hati sebagai gejala daripenyakit Garis Kuning (GK) dan Busuk Kuncup (BK) adalah:

$$m1\{GK, BK,DBM\} = 0,8$$

$$m1\{\theta\} = 1 - 0,8 = 0,2$$

Gejala-2 : Tampak Bercak-bercak lonjong berwarna kuning dan ditengahnya berwarna coklat kemudian jika diketahui nilai kepercayaan setelah dilakukan observasi terhadap rasa terbakar pada tenggorokan sebagai gejala dari penyakit Tanjuk (PT) adalah :

$$m2\{GK, BK, PT\} = 0,9$$
$$m2\{\theta\} = 1 - 0,9 = 0,1$$

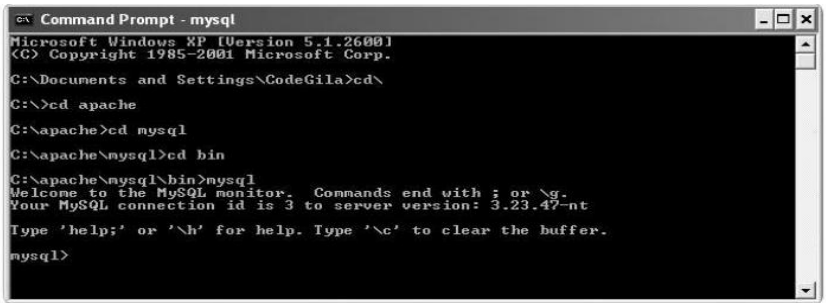
II.3. PHP

PHP singkatan dari *hypertext preprocessor* yaitu bahasa pemrograman *web server-side* yang bersifat *open source*. PHP merupakan *script* yang terintegrasi dengan HTML dan berada pada server (*server side HTML embedded scripting*). PHP adalah *script* yang digunakan untuk membuat halaman *website* yang dinamis. Dinamis berarti halaman yang akan ditampilkan dibuat saat halaman itu diminta oleh client. Mekanisme ini menyebabkan informasi yang diterima client selalu yang terbaru/*up to date*. Semua *script* PHP dieksekusi pada *server* dimana *script* tersebut dijalankan.(Anhar ; 2010 : 3)

PHP adalah singkatan dari *PHP Hypertext Preprocessor*. Saat pertama kali dikembangkan oleh programmer bernama Rasmus Lerdoff, PHP awalnya adalah singkatan dari *Personal Home Page Tools*. Namun setelah dikembangkan oleh Zeev Suraski dan Andi Gutmans, dan fiturnya bertambah, maka PHP diubah singkatannya menjadi yang sekarang ini. Di tinjau dari segi sintak bahasa nya, PHP

mirip dengan C, bagi mereka yang sudah berpengalaman dengan C akan mudah memahami PHP, tapi jangan khawatir, yang belum berpengalaman pun akan cukup mudah memahami PHP (Tim EMS; 2014 :59)

II.4. MySQL



Gambar II.2. : DBMS My Sql

Sumber : (Anhar ; 2010 : 47)

MySQL (*My Structure Query Language*) adalah salah satu *Database Management System* (DBMS) dari sekian banyak DBMS seperti *Oracle*, *MS SQL*, *Postgre SQL*, dan lainya. Mysql berfungsi untuk mengolah database menggunakan bahasa SQL. Mysql bersifat *open source* sehingga kita bisa menggunakan secara gratis. Pemrograman PHP juga sangat mendukung/support dengan database MySQL (Anhar ; 2010 : 45).

My SQL adalah salah satu program RDBMS yang sangan terkenal berikut ini beberapa fitur My SQL :

1. My SQL adalah sistem database yang lazim digunakan di lingkungan Web.
2. My SQL adalah sistem database yang berjalan di server.
3. My SQL cocok untuk aplikasi kecil dan besar
4. My SQL cukup cepat, bisa dihandalkan dan mudah dipakai.
5. My SQL dikembangkan, didistribusikan, dan pendukung secara open source oleh Oracle Corporation (Tim EMS;2014: 129).

II.5. Database

Database adalah sekumpulan tabel-tabel yang berisi data dan merupakan kumpulan dari *field* atau kolom. Struktur file yang menyusun sebuah database adalah *Data Record* dan *Field* (Anhar ; 2010 : 45).

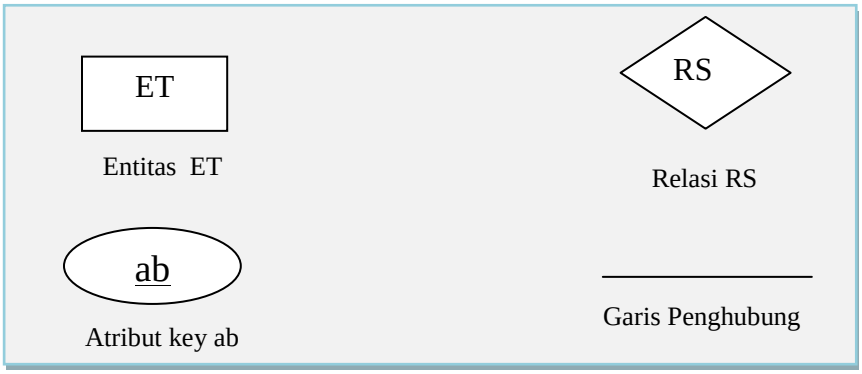
a. Data adalah satu satuan informasi yang akan diolah. Sebelum diolah, data dikumpulkan di dalam suatu *file database*.
b. *RECORD* adalah data yang isinya merupakan satu kesatuan seperti NamaUser dan *Password.* Setiap keterangan yang mencakup NamaUser dan *Password* dinamakan satu *record*. Setiap *record* diberi nomor urut yang disebut nomor record *(Record Number)*.
c. *FIELD* adalah sub bagian dari *record*. Dari contoh isi *record* di atas, maka terdiri dari 2 *field*, yaitu: *field* NamaUser dan *Password.*

II.5.1. Pemodelan Data

Menurut **(**Yudi Priyadi ; 2014 : 10) Terdapat beberapa penjelasan mengenai pemodelan basis data. Suatu basis data dapatdigunakan secara bebas untuk menggambarkan dan memberikan deskripsi mengenai kumpulan informasi uang tersimpan dalam *data storage* komputer. Secara sederhana, defenisi untuk model basis data adalah sekumpulan notasi atau simbol untuk menggambarkan data dan relasinya, berdasarkan suatu konsep dan aturan tertentu suatu pemodelan.

II.5.2. Notasi Diagram E-R

Menurut **(**Yudi Priyadi ; 2014 : 20) Pemodelan basis data dengan menggunakan diagram relasi antar entitas, dapat dilakukan dengan menggunakan suatu pemodelan basis datayang bernama Diagram *Entity-Relational* (selanjutnya disingkat Diagram E-R). Pada Gambar II.2, terdapat suatu simbol/notasi dasar yang digunakan pada Diagram E-R,yaitu entitas,relasi, atribut, dan garis penghubung.



Gambar II.3 : Notasi Dasar Diagram E-R

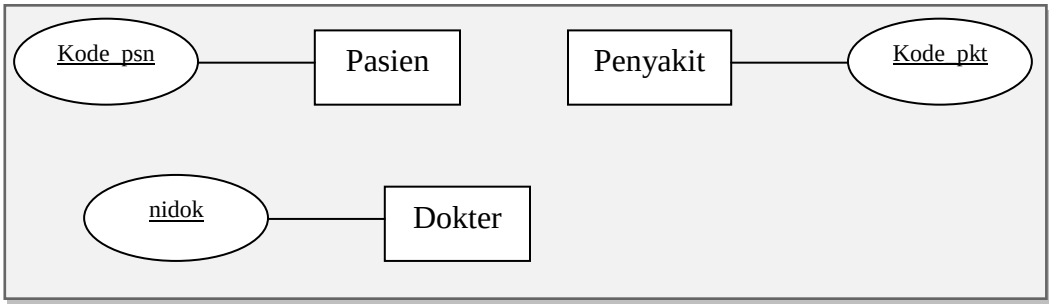
(Sumber : Yudi Priyadi ; 2014 : 20)

1. Entitas

   Merupakan notasi untuk mewakili suatu objek dengan karakteristik sama, yang dilengkapi oleh atribut, sehingga pada suatu lingkungan nyata setiap objek akan berbeda dengan objek lainya. Pada umumnya, objek dapat berupa benda, pekerjaan, tempat dan orang.

2. Atribut

   Merupakan notasi yang menjelaskan karakteristik suatu entitas dan juga relasinya. Atribut dapat sebagai key yang bersifat unik, yaitu *Primary Key* atau *Foreign Key.* Selain itu, atribut juga dapat sebagai atributdeskriptif saja, yaitu sebagai pelengkap deskripsi suatu entitas dan relasi.

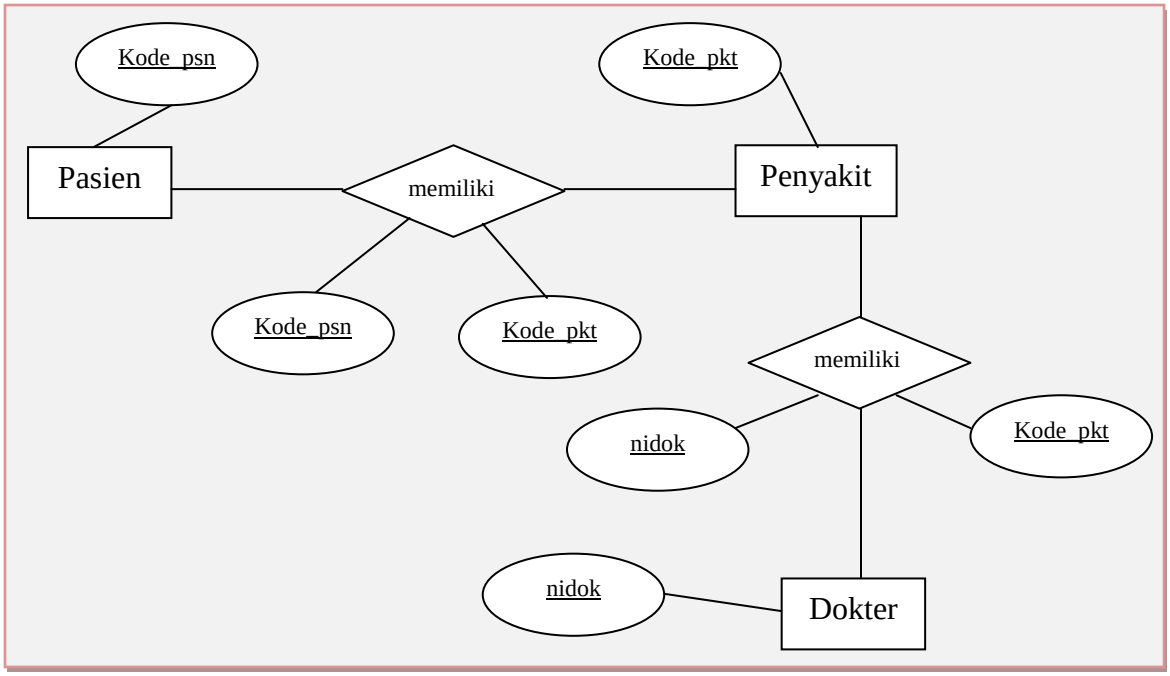


Gambar II.4 : Atribut Key pada Entitas

(Sumber : Yudi Priyadi ; 2014 : 23)

3. Relasi

   Merupakan notasi yang digunakan untuk menghubungkan beberapa entitas berdasarkan fakta pada suatu lingkungan. Adapun Gambar II.4. Pemilihan Relasi untuk entitas dapat dilihat dibawah ini:



Gambar II.5 : Pemilihan Relasi untuk Entitas

(Sumber : Yudi Priyadi ; 2014 : 25)

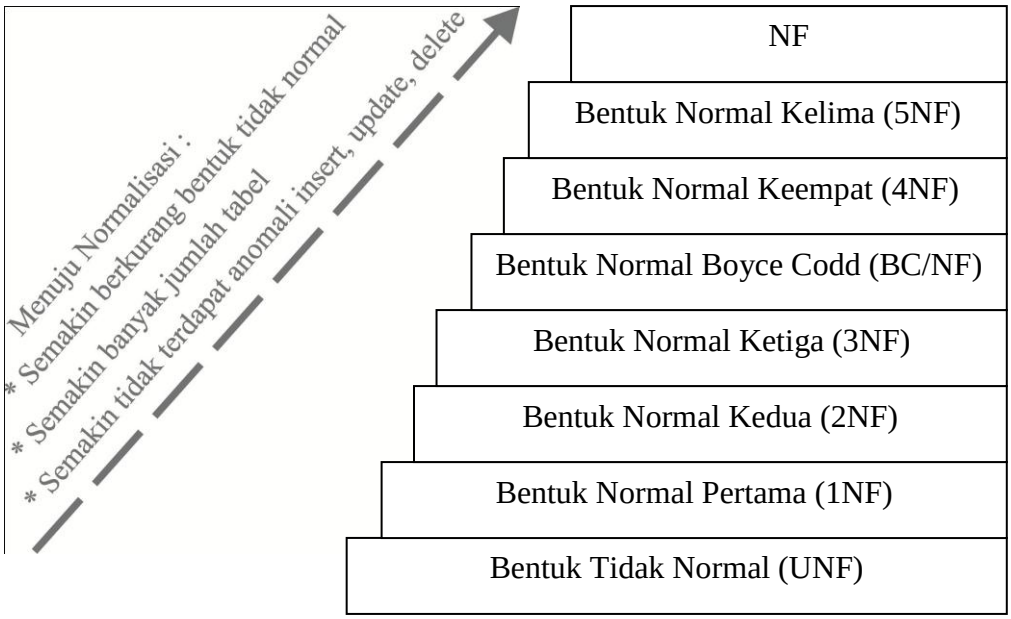
4. Garis penghubung

   Merupakan notasi untuk merangkaikan keterkaitan antar notasi yang digunakan dalam Diagram E-R, yaitu entitas, relasi dan atribut.

II.5.3. Normalisasi

Menurut **(**Yudi Priyadi ; 2014 : 67) Normalisasi merupakan proses sistematis yang dilakukan pada struktur tabel basis data menjadi struktur tabel yang memiliki integritas data, sehingga tidak memiliki data anomali pada saat melakukan *insert*, *delete*, dan *update*. Pada Gambar II.5, tahapan proses sistematis yang dilakukan

mulai dari bentuk tidak normal menjadi bentuk normal memiliki suatu syarat yang harus dipenuhi pada saat menuju suatu bentuk yang lebih baik (*well structured relation*).



Gambar II.6 : Tahapan Proses Bentuk Normalisasi

(Sumber : Yudi Priyadi ; 2014 : 67)

Setiap syarat dalam tahapan suatu bentuk normal memiliki keterkaitan, hal ini disebabkan karena pada setiap bentuk normal mengalami penyempurnaan untuk bentuk normal selanjutnya. Bentuk tidak normal akan semakin berkurang, setelah melalui tahapan berubahan bentuk normalisasi, sehingga berdampak pada jumlah tabel yang semakin banyak, tetapi menuju perbaikan ke dalam bentuk *well structured relation.* Hal ini terjadi akibat dari pengelompokan data suatu tabel agar memiliki ketergantungan secara fungsional.

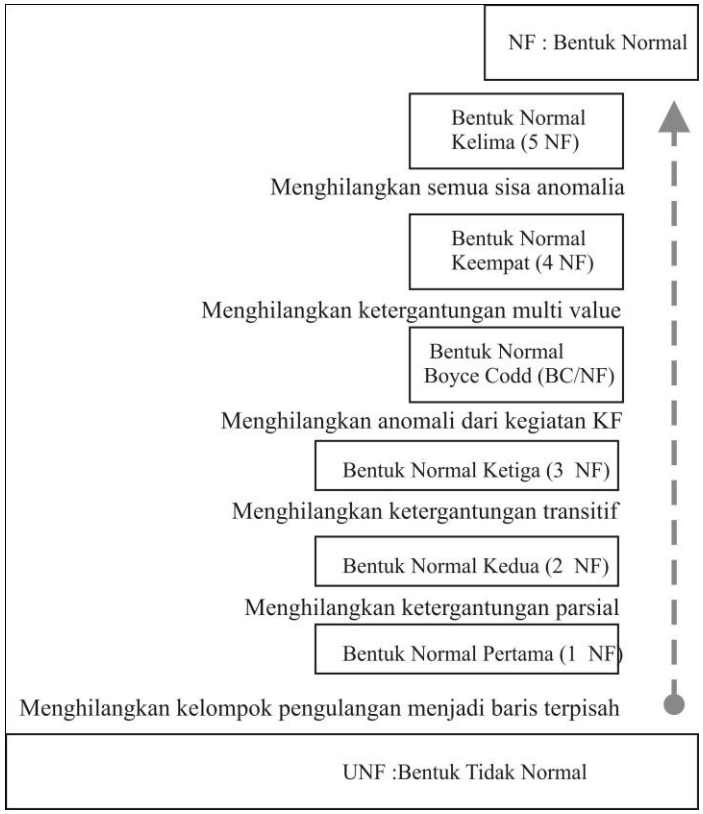
II.5.4. Aturan Proses Normalisasi

Menurut **(**Yudi Priyadi ; 2014 : 68) Secara sederhana, kegiatan normalisasi adalah melakukan dekomposisi atau penguraian tabel beserta datanya, menjadi tabel yang normal menurut konsep RDBMS. Merujuk pada gambar II.6, dekomposisi diawali dengan melakukan analisis pada suatu tabel atau beberapa contoh formulir yang sudah memiliki data lengkap dalam basis data, tetapi masih dalam bentuk yang tidak normal (UNF). Oleh karena itu agar dapat memenuhi syarat bentuk normal pertama (1NF), pada setiap barisnya diisikan suatu *value* dengan kelompok data yang sama, berdasarkan suatu atribut key. Dengan demikian, kelompok pengulangan dalam suatu baris dapat dihilangkan, karena sudah tidak terdapat *value* yang kosong untuk setiap *field* dan *record*nya

Setelah memenuhi syarat bentuk normal pertama (1NF), proses berikutnyaadalah menghilangkan ketergantungan secara parsial, yaitu dengan cara melakukan dekomposisi tabel menjadi beberapa kelompok tabel berdasarkan field yang memiliki status sebagai key. Hal ini dapat dilakukan oleh salah satu field saja, dengan tetap tidak mengubah arti relasi dan ketergantungannya. Oleh sebab itu, disebut ketergantungan fungsional sebagian (*partiallly functional*), sehingga syarat bentuk normal kedua (2NF) sudah tercapai.

Bentuk normal kedua (2NF) merupakan syarat yang harus dimiliki untuk menuju bentuk normal ketiga (3NF). Pada proses ini, dilakukan dengan menghilangkan ketergantungan secara transitif, yaitu suatu konsep untuk tabel dari hasil relasi yang didalamnya terdapat ketergantungan secara tidak langsung pada

beberapa atributnya. Pada umumnya proses normalisasi sudah dapat tercapai pada bentuk normal ketiga (3NF), yaitu dengan menghasilkan tabel yang tidak mengalami anomali basis data pada saat proses *insert, delete,* dan *update.*



Gambar II.7 : Tahapan Aturan Proses Normalisasi

(Sumber : Yudi Priyadi ; 2014 : 69)

II.6. Unified Modeling Language (UML)

Menurut (Rosa A.S & M. Shalahuddin ; 2011 : 118) Pada perkembangan teknik pemrograman berorientasi objek, muncullah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakanteknik pemrograman berorientasi objek, yaitu *Unified Modeling Language* (UML). UML muncul karena adanya kebutuhan pemodelan visual untuk

menspesifikasikan, menggambarkan, membangun dan dokumentasi dari sistem perangkat lunak. UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung.

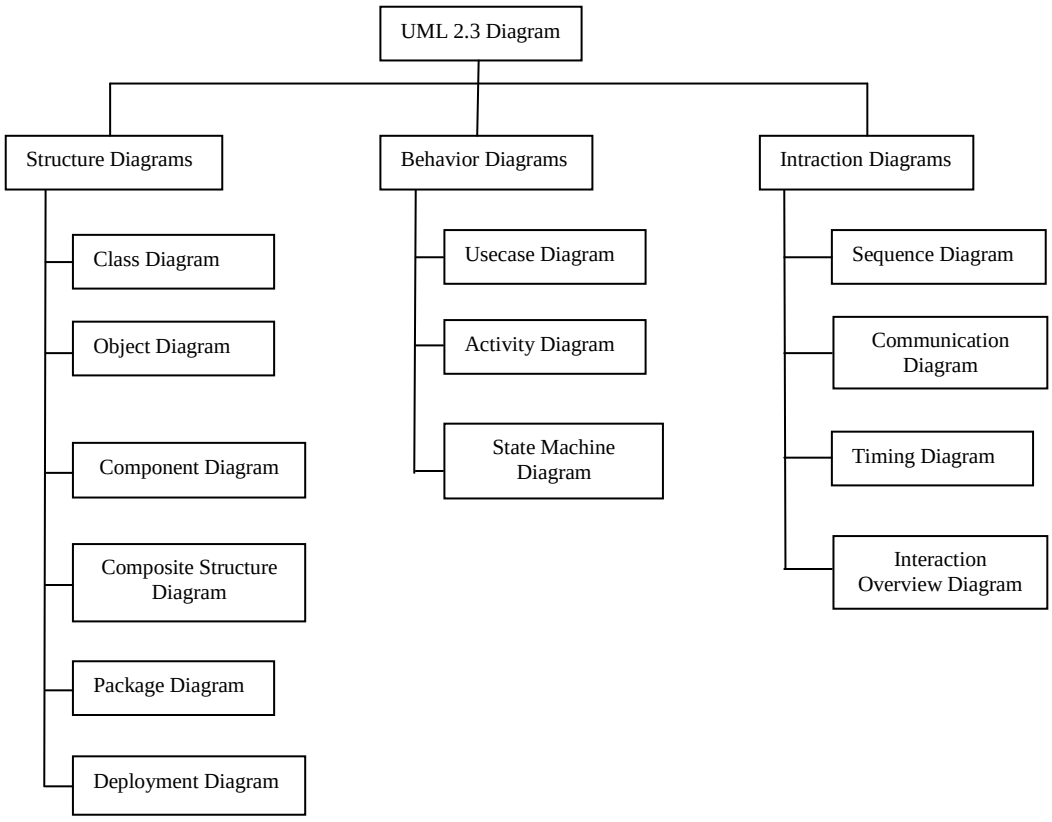
UML hanya berfungsi untuk melakukan pemodelan, jadi penggunaan UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataannya UML paling banyak digunakan pada metode berorientasi objek.

Unified Modelling Language (UML) biasanya digunakan untuk:

1. Menggambarkan batasan sistem dan fungsi-fungsi sistem secara umum.
2. Menggambarkan kegiatan atau proses bisnis yang dilaksakan secara umum.
3. Menggambarkan representasi stuktur statik sebuah sistemdalam bentuk *class diagram*.
4. Membuat model *behavior* "yang menggambarkan sifat atau sebuah sistem.
5. Menyatakan arsitektur implementasi fisik menggunakan *component* dan *development diagram.*
6. Menyampaikan atau memperluas *fungsionalty* dengan *stereotypes.* (Yuni Sugiarti; 2013: 36).

II.6.1 Diagram-Diagram UML

Menurut (Rosa A.S & M. Shalahuddin ; 2011 : 120) Pada UML 2.3 terdiri dari 13 macam diagram yang dikelompokan dalam 3 kategori. Pembagian kategori dan macam-macam diagram tersebut dapat dilihat pada gambar II.7 di bawah ini :



Gambar II.8 : Diagram UML

(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 121)

Berikut ini penjelasan singkat dari pembagian kategori tersebut

1. *StructureDiagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.
2. *Behavior Diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.

3. *Interaction Diagrams* yaitu kumpulan diagram yang digunakan untukmenggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.

A. *Class Diagram*

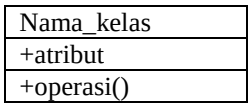
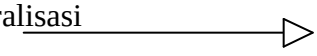
Diagram kelas atau *Class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem.

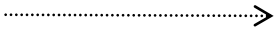
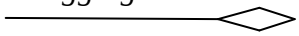
Kelas memiliki apa yang disebut atribut dan metode atau operasi.

1) Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas
2) Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas

Berikut Tabel II.8 menerangkan simbol-simbol pada diagram kelas :

Tabel II.1. *Class Diagram*

Simbol	Deskripsi
Kelas  	Kelas pada struktur sistem
Antarmuka / *interface*  Nama_*interface*	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi / *association* Asosiasi berarah/ *directed association* 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity* Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*
Generalisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum khusus)

Kebergantungan 	Relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi / *aggregation* 	Semua bagian (*whole part*)

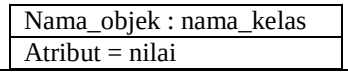
(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 124)

B. *Object Diagram*

Diagram objek menggambarkan struktur sistem dari segi penamaan objek dan jalannya objek dalam sistem. Pada diagram objek harus dipastikan semua kelas yang sudah didefenisikan pada diagram kelas harus dipakai objeknya, karena jika tidak, pendefenisian kelas itu tidak dapat dipertanggungjawabkan. Untuk apa mendefenisikan sebuah kelas sedangkan pada jalannya sistem, objeknya tidak pernah dipakai.

Berikut adalah Tabel II.9 menerangkan simbol-simbol diagram objek

Tabel II.2. *Diagram Objek*

Simbol	Deskripsi
Objek 	Objek dari kelas yang berjalansaat sistem dijalankan
Link	Relasi antar objek

(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 124)

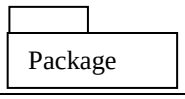
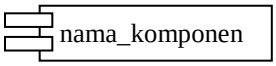
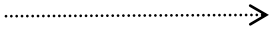
C. *Component Diagram*

Diagram komponen atau component diagram dibuatuntuk menunjukkan organisasi dan ketergantungan di antara kumpulan komponen dalam sebuah sistem. Diagram komponen fokus pada komponen sistem yang dibutuhkan dan ada didalam sistem. Komponen dasar yang biasanya ada dalam suatu sistem adalah sebagai berikut :

1) Komponen *user interface* yang menangani tampilan
2) Komponen *bussiness procesiing* yang menangani fungsi-fungsi proses bisnis
3) Komponen data yang menangani manipulasi data
4) Komponen *security* yang menangani keamanan sistem

Komponen lebih terfokus pada penggolongan secara umum fungsi-fungsi yang diperlukan, berikut Tabel II.10 yang menerangkan simbol-simbol yang ada pada diagram komponen.

Tabel II.3. *Komponen Diagram*

Simbol	Deskripsi
Package 	*Package* merupakan sebuah bungkusan dari satu atau lebih komponen
Komponen 	Komponen Sistem
Kebergantungan / *dependency* 	Kebergantungan antar komponen, arah panah mengarah pada komponen yang dipakai
Antar muka / *interface*	Sama dengan konsep *interface*pada

 nama_interface	pemrograman berorientasi objek, yaitu sebagai antarmuka komponen agar tidak mengakses langsung komponen
Link _____	Relasi antar komponen

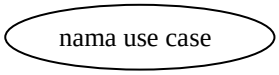
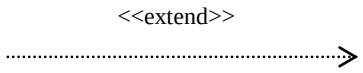
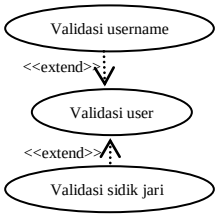
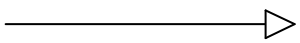
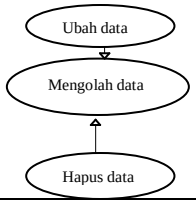
(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 126)

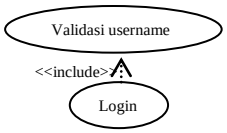
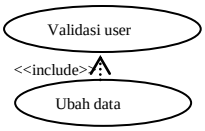
D. *Use Case Diagram*

Use case atau diagram *use case* merupakan pemodelan untuk kelakuan (*behaviour)*sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar,*use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Syarat penamaan pada *use case* adalah nama didefenisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu pendefenisian apa yang disebut aktor dan *use case*.

1) Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
2) *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

Tabel II.4. *Use Case Diagram*

Simbol	Deskripsi
Use case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*
Aktor / *actor*  nama aktor	Orang, proses, atau sistem yang lain berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan di buat itu sendiri
Asosiasi / *association*	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case*, atau *usecase* memiliki interasi dengan aktor
Ekstensi / *extend* 	Relasi usecase tambahan ke subuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanppa *use case* tambahan itu, mirip dengan prinsip inheritance pada pemrograman berorientasi objek, biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan misal  arah panah mengarah pada *use case* yang ditambahkan
Generalisasi / *generalization* 	Hubungan generalisasi dan spesialisasi (umum – khusus) antara dua buah use case dimana fungsi yang satu adalah fungsi yang lebih umum dari lainya misalnya : 

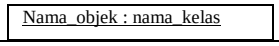
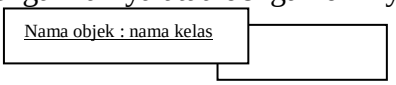
	Arah panah mengarah pada use case yang menjadi generalisasinya (umum)
Menggunakan / *include* / *uses* <<include>> <<uses>>	Relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya atau sebagai syarat dijalankan use case ini Ada 2 sudut pandang yang cukup besar mengenai include di usecase 1. include berarti use case yang ditambahkan akan selalu dipanggil saat use case dijalankan misal pada kasus berikut :  2. include berarti use case yang tambahan akan selalu melakukan pengecekan apakah use case yang di tambahkan telah di jalankan sebelum use case tambahan di jalankan, misal pada kasus berikut :  Kedua interpretasi di atas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan.

(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 131)

E. Communication Diagram

Diagram komunikasi mengelompokkan message pada kumpulan diagram sekuen menjadi sebuah diagram. Dalam diagram komunikasi yang dituliskan adalah operasi / metode yang di jalankan antara objek yang satu dengan objek lainnya secara keseluruhan, oleh karna itu dapat di ambil dari jalanya interaksi pada semua diagram sekuen. Berikut adalah Tabel II.12 yang menerangkan simbol-simbol yang ada pada diagram komunikasi :

Tabel II.5. ***Komunikasi Diagram***

Simbol	Deskripsi
Objek 	Objek yang melakukan interaksi pesan
Link	Relasi antar objek yang menghubungkan objek satu dengan lainya atau dengan dirinya sendiri 
Arah pesan / stimulus 	Arah pesan yang terjadi, jika pada suatu link ada dua arah pesan yang berbeda, maka arah juga deigambarkan dua arah pada dua sisi link 

(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 140)

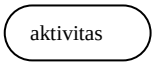
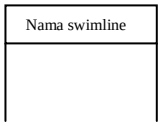
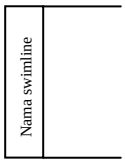
F. *Activity Diagram*

Diagram aktivitas atau activity diagram menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Diagram aktivitas juga banyak digunakan untuk mendefenisikan hal-hal berikut :

1) Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sitemyang didefenisikan
2) Urutan atau pengelompokan tampilan dari sistem/user interface dimana setiap aktivitasdianggap memiliki sebuah rancangan antarmuka tampilan
3) Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefenisikan kasus ujinya.

Berikut adalah Tabel II.13 yang menggambarkan simbol-simbol yang ada pada diagram aktivitas :

Tabel II.6. ***Activity Diagram***

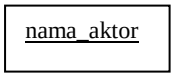
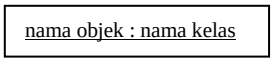
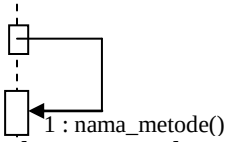
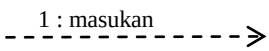
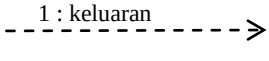
Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki status awal
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
Percabangan / decesion 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
Penggabungan / join	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
Swimlane  atau 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

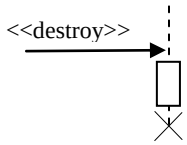
(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 134)

G. Sequence Diagram

Diagram sekuen menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan messasge yang dikirimkan dan diterima antar objek. Banyaknya diagram objek yang digambarkan adalah sebanyak pendefinisian use case yang memiliki proses sendiri atau yang penting semua use case yang telah didefenisikan interaksi jalanya pesan sudah dicakup dapa diagram sekuen sehingga semakin banyak use case yang didefenisikan maka diagram sekuen yang harus dibuat juga semakin banyak. Berikut adalah Tabel II.14 yang menerangkan simbol-sombol yang ada pada diagram sequence :

Tabel II.7. *Sequence Diagram*

Simbol	Deskripsi
Aktor  atau  tampa waktu aktif	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya di nyatakan menggunakan kata benda di awali frase nama aktor
Garis hidup / lifeline	Menyatakan kehidupan suatu objek
Objek 	Menyatakan objek yang berintaraksi pesan
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi pesan
Pesan tipe create	Objek yang lain, arah panah mengarah pada objek yang dibuat
Pesan tope call 	Menyatakan suatu objek memanggil operasi / metode yang ada pada objek lain atau dirinya sendiri  Arah panah mengarah pada objek yang memiliki operasi / metode, karena ini memanggil operasi / metode maka operasi / metode yang di panggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi
Pesan tipe send 	Menyatakan bahwa suatu objek mengirimkan data / masukan / informasi ke objek lainya, arah panah mengarah pada objek yang dikirimi
Pesan tipe return 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasiatau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian

Pesan tipe destroy  	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada create maka ada destroy

(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 138)