BAB II
LANDASAN TEORI

2.1 *Data Mining*

Data mining adalah suatu istilah yang digunakan untuk menguraikan penemuan pengetahuan di dalam *database*. *Data mining* adalah proses yang menggunakan teknik statistik, matematika, kecerdasan buatan, dan *machine learning* untuk mengekstraksi dan mengidentifikasi informasi yang bermanfaat dan pengetahuan yang terakit dari berbagai *database* besar (Turban, dkk. 2005).

Menurut Gartner Group *data mining* adalah suatu proses menemukan hubungan yang berarti, pola, dan kecenderungan dengan memeriksa dalam sekumpulan besar data yang tersimpan dalam penyimpanan dengan menggunakan teknik pengenalan pola seperti teknik statistik dan matematika (Larose, 2005).

Data mining adalah serangkaian proses untuk menggali nilai tambah dari suatu kumpulan data berupa pengetahuan yang selama ini tidak diketahui secara manual (Pramudiono, 2006).

Menurut (Kusrini dan Luthfi, 2009) menyatakan tentang hal penting yang terkait dengan *data mining* antara lain :

1. *Data mining* merupakan suatu proses otomatis terhadap data yang sudah ada.
2. Data yang akan diproses berupa data yang sangat besar.
3. Tujuan *data mining* adalah mendapatkan hubungan atau pola yang mungkin memberikan indikasi yang bermanfaat.

2.1.1 Pengelompokan Data Mining

Data mining dibagi menjadi beberapa kelompok berdasarkan tugas yang dapat dilakukan, yaitu (Larose, 2005) :

1. Deskripsi

   Terkadang peneliti dan analis secara sederhana ingin mencoba mencari cara untuk menggambarkan pola dan kecendrungan yang terdapat dalam data. Sebagai contoh, petugas pengumpulan suara mungkin tidak dapat menemukan keterangan atau fakta bahwa siapa yang tidak cukup profesional akan sedikit didukung dalam pemilihan presiden. Deskripsi dari pola dan kecendrungan sering memberikan kemungkinan penjelasan untuk suatu pola atau kecendrungan.

2. Estimasi

   Estimasi hampir sama dengan klasifikasi, kecuali variabel target estimasi lebih ke arah numerik daripada ke arah kategori. Model dibangun menggunakan record lengkap yang menyediakan nilai dan variabel target sebagai nilai prediksi. Selanjutnya, pada peninjauan berikutnya estimasi nilai dari variabel target dibuat berdasarkan nilai variabel prediksi. Sebagai contoh yaitu estimasi indeks prestasi kumulatif mahasiswa program pascasarjana dnegan melihat nilai indeks prestasi mahasiswa tersebut pada saat mengikuti program sarjana.

3. Prediksi

   Prediksi hampir sama dengan klasifikasi dan estimasi, kecuali bahwa dalam prediksi nilai dari hasil akan ada di masa mendatang.

   Contoh prediksi dalam bisnis dan penelitian adalah :

   - Prediksi harga beras dalam tiga bulan yang akan datang.
   - Prediksi persentase kenaikan kecelakaan lalu lintas tahun depan jika batas bawah kecepatan dinaikan.

   Beberapa metode dan teknik yang digunakan dalam klasifikasi dan estimasi dapat pula digunakan (untuk keadaan yang tepat) untuk prediksi.

4. Klasifikasi

   Dalam klasifikasi, terdapat target variabel kategori. Sebagai contoh penggolongan pendapatan dapat dipisahkan dalam tiga kategori, yaitu pendapatan tinggi, pendapatan sedang dan pendapatan rendah.

   Contoh lain klasifikasi dalam bisnis dan penelitian adalah :

   - Menentukan apakah suatu transaksi kartu kredit merupakan transaksi yang curang atau bukan.
   - Memperkirakan apakah suatu pengajuan hipotek oleh nasabah merupakan suatu kredit yang baik atau buruk.
   - Mendiagnosis penyakit seorang pasien untuk mendapatkan termasuk kategori penyakit apa.

5. Pengklusteran

   Pengklusteran merupakan pengelompokan record, pengamatan, atau memperhatikan dan membentuk kelas objek-objek yang memiliki kemiripan. Kluster adalah kumpulan record yang memiliki kemiripan satu dengan yang lainnya dan memiliki ketidakmiripan dengan record-record dalam kluster lain.

   Pengklusteran berbeda dengan klasifikasi yaitu tidak adanya variabel target dalam pengklusteran. Pengklusteran tidak mencoba untuk melakukan klasifikasi, mengestimasi atau memprediksi nilaidari variabel target. Akan tetapi, algoritma pengklusteran mencoba untuk melakukan pembagian terhadap keseluruhan data menjadi kelompok-kelompok yang memiliki kemiripan (homogen), yang mana kemiripan record dalam satu kelompok akan bernilai maksimal, sedangkan kemiripan dengan record dalam kelompok lain akan bernilai minimal.

   Contoh pengklusteran dalam bisnis dan penelitian adalah :

   - Mendapatkan kelompok-kelompok konsumen untuk target pemasaran dari suatu poduk bagi perusahaan yang tidak memiliki dana pemasaran yang besar.

- Untuk tujuan audit akuntansi, yaitu melakukan pemisahan terhadap perilaku finansial dalam baik dan mencurigakan.
- Melakukan pengklusteran terhadap ekspresi dari gen, untuk mendapatkan kemiripan perilaku dari gen dalam jumlah besar.

6. Asosiasi

   Tugas asosiasi dalam data mining adalah menemukan atribut yang muncul dalam satu waktu. Dalam dunia bisnis lebih umum disebut analisis keranjang belanja.

   Contoh asosiasi dalam bisnis dan penelitian adalah :

   - Meneliti jumlah pelanggan dari perusahaan telekomukasi seluler yang diharapkan untuk memberikan respons positif terhadap penawaran *upgrade* layanan yang diberikan.
   - Menemukan barang dalam supermarket yang dibeli secara bersamaan dan barang yang tidak pernah dibeli secara bersamaan.

2.1.2 Proses *Data Mining*

Secara sistematis, ada 3 (tiga) langkah utama dalam data mining (Gorunescu, 2011) yaitu :

1) Eksplorasi atau pemrosesan awal data

   Eksplorasi atau pemrosesan awal data terdiri dari pembersihan data, normalisasi data, transformasi data, penanganan data yang salah, reduksi dimensi, pemilihan subset fitur, dan sebagainya.

2) Membangun model dan melakukan validasi terhadapnya

   Membangun model dan melakukan validasi terhadapnya berarti melakukan analisis berbagai model dan memilih model dengan kinerja prediksi yang terbaik. Dalam langkah ini digunakan metode-metode seperti Klasifikasi, Regresi, Analisis Cluster, Deteksi Anomali, Asosiasi, Analisis Pola Sekuensial, dan sebagainya. Dalam beberapa referensi, Deteksi Anomali juga masuk dalam langkah eksplorasi. Akan

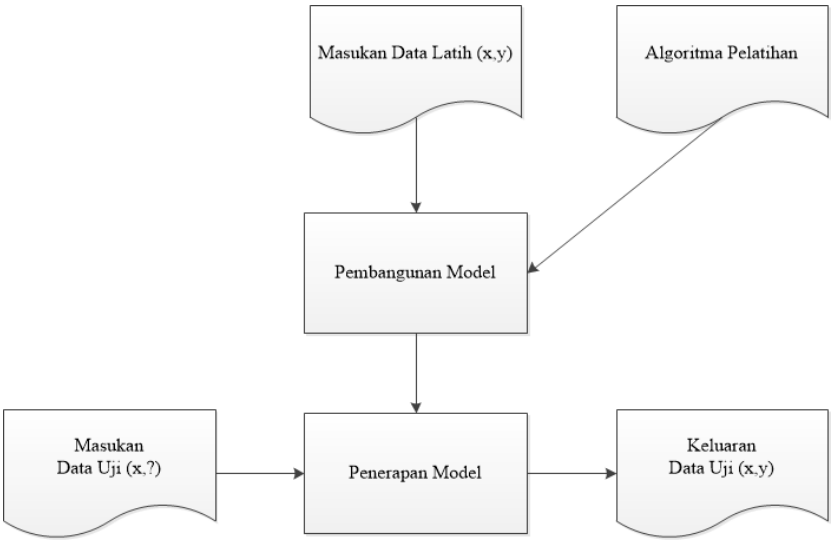
tetapi, Deteksi Anomali juga dapat digunakan sebagai algoritma utama, terutama untuk mencari data yang spesial.

3) Penerapan

Penerapan berarti menerapkan model pada data yang baru untuk menghasilkan perkiraan atau prediksi masalah yang diinvestigasi.

2.1.3 Konsep Klasifikasi

Menurut (Prasetyo, 2014) menyatakan bahwa klasifikasi dapat didefinisikan secara detail sebagai suatu pekerjaan yang melakukan pelatihan atau pembelajaran terhadap fungsi target *f* yang memetakan setiap vektor (sub fitur) *x* ke dalam satu dari sejumlah label kelas *y* yang tersedia. Pekerjaan pelatihan tersebut akan menghasilkan suatu model yang kemudian disimpan sebagai memori.



Gambar 2.1 Proses Pekerjaan Klasifikasi (Presetyo, 2014).

Kerangka kerja klasifikasi ditunjukkan pada gambar 2.1. Pada gambar tersebut disediakan sejumlah data latih (*x*,*y*) untuk digunkaan sebagai data pembangun model, kemudian menggunakan model tersebut untuk memprediksi kelas dari data uji (*x*,?) sehingga data uji (*x*,?) diketahui kelas *y* yang seharusnya.

Model yang sudah dibangun pada saat pelatihan kemudian dapat digunakan untuk memprediksi label kelas dari data baru yang belum diketahui label kelasnya. Dalam pembangunan model selama proses pelatihan tersebut diperlukan adanya suatu algoritma untuk membangunnya yang disebut sebagai algoritma pelatihan. Ada banyak algoritma pelatihan yang sudah dikembangkan oleh para peneliti seperti *Decision Tree, K-Nearest Neighbor, Artificial Neural Network* dan sebagainya. Setiap algoritma mempunyai prinsip yang sama yaitu melakukan suatu pelatihan sehingga di akhir pelatihan model dapat memetakan (memprediksi) setiap vektor masukan ke label kelas keluaran yang benar.

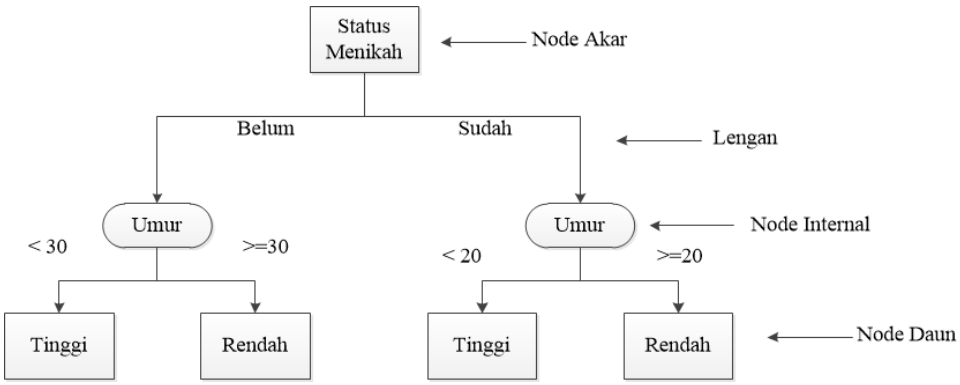
2.1.4 Pohon Keputusan

Menurut (Kusrini dan Luthfi, 2009) pohon keputusan merupakan metode klasfikasi dan prediksi yang sangat kuat dan terkenal. Metode pohon keputusan mengubah fakta yang sangat besar menjadi pohon keputusan yang merepresentasikan aturan. Aturan dapat dengan mudah dipahami dengan bahasa alami. dan mereka juga dapat dieskpresikan dalam bentuk bahasa basis data seperti *Structured Query Language* untuk mecari *record* pada kategori tertentu.

Sebuah pohon keputusan mungkin dibagun dengan saksama secara manual atau dapat tumbuh secara otomatis dengan menerapkan salah satu atau beberapa algoritma pohon keputusan untuk memodelkan himpunan data yang belum terklasifikasi. Banyak algoritma yang dapat dipakai dalam pembentukan pohon keputusan, antara lain ID3, CART dan C4.5 (Larose,2005).

Data dalam pohon keputusan biasanaya dinyatakan dalam bentuk tabel dengan atribut dan *record*. Atribut menyatakan suatu parameter yang dibuat sebagai kriteria dalam pembentukan pohon. Misalkan untuk menentukan main tenis, kriteria yang diperhatikan adalah cuaca, angin dan

temperatur. Proses pada pohon keputusan adalah mengubah bentuk data (tabel) menjadi model pohon, mengubah model pohon menjadi *rule* dan menyederhanakan *rule* (Basuki dan Syarif, 2003).



Gambar 2.2 Contoh Pohon Keputusan (Prasetyo, 2014).

Karakteristik dari *decision tree* seperti gambar 2.2, dibentuk sejumlah elemen sebagai berikut (Tan, 2006) :

- Node akar, tidak mempunyai lengan masukan dan mempunyai nol atau lebih lengan keluaran.
- Node internal, setiap node yang bukan daun yang mempunyai tepat satu lengan masukan dan dua atau lebih lengan keluaran. Node ini menyatakan pengujian yang didasarkan pada nilai fitur.
- Lengan, setiap cabang menyatakan nilai hasil pengujian di node bukan daun.
- Node daun (terminal), node yang mempunyai tepat satu lengan masukan dan tidak mempunyai lengan keluaran. Node ini menyatakan label kelas (keputusan).

2.1.5 Algoritma C4.5

Algoritma C4.5 merupakan algoritma yang digunakan untuk membentuk pohon keputusan. Menurut (Kusrini dan Lutfhi, 2009) menyatakan bahwa secara umum algoritma C4.5 untuk membangun pohon keputusan adalah sebagai berikut :

1) Pilih atribut sebagai akar.
2) Buat cabang untuk tiap-tiap nilai.
3) Bagi kasus dalam cabang.
4) Ulangi proses untuk setiap cabang sampai semua kasus pada cabang memiliki kelas yang sama.

Untuk memilih atribut sebagai akar, didasarkan pada nilai *gain* tertinggi dari atribut-atribut yang ada. Untuk menghitung gain digunakan rumus seperti tertera dalam persamaan 1 berikut :

$$Gain\ (S, A) = Entropy\ (S) - \sum_{i=l}^{n} \frac{|si|}{|s|} * Entropy \quad (1)$$

Keterangan :

S : himpunan kasus
A : atribut
n : jumlah partisi atribut A
|Si| : jumlah kasus pada partisi ke-i
|S| : jumlah kasus dalam S

Sementara itu, penghitungan nilai entropi dapat dilihat pada persamaan 2 berikut :

$$Entropy\ (S) = \sum_{i=l}^{n} - pi * log_2\ pi \quad (2)$$

Keterangan :

S : himpunan kasus
A : atribut
n : jumlah partisi S
pi : proporsi dari Si terhadap S

sebagai contoh, untuk membentuk pohon keputusan dapat dilakukan dengan menentukan atribut sebagai akar lalu mencari nilai *entropy* dan *gain*. Misalkan ditampilkan 14 dataset pengujian untuk klasifikasi C4.5 sebagai berikut :

Tabel 2.1 Data Set Pengujian Klasifikasi (Kusrini dan Luthfi, 2009).

No	Outlook	Temperature	Humidity	Windy	Play
1	Sunny	Hot	High	FALSE	No
2	Sunny	Hot	High	TRUE	No
3	Cloudy	Hot	High	FALSE	Yes
4	Rainy	Mild	High	FALSE	Yes
5	Rainy	Cool	Normal	FALSE	Yes
6	Rainy	Cool	Normal	TRUE	Yes
7	Cloudy	Cool	Normal	TRUE	Yes
8	Sunny	Mild	High	FALSE	No
9	Sunny	Cool	Normal	FALSE	Yes
10	Rainy	Mild	Normal	FALSE	Yes
11	Sunny	Mild	Normal	TRUE	Yes
12	Cloudy	Mild	High	TRUE	Yes
13	Cloudy	Hot	Normal	FALSE	Yes
14	Rainy	Mild	High	TRUE	No

Untuk dapat menentukan nilai-nilai g*ain* dan *entropy* dari masing-masing atribut di atas, maka data tersebut harus di konverisikan ke dalam bentuk tabel klasifikasi sebagai berikut :

Tabel 2.2 Klasifikasi Perhitungan *Gain* dan *Entropy* (Kusrini dan Luthfi, 2009).

Node	Atribut	Nilai Atribut	Jml Kasus (S)	Tidak (S_1)	Ya (S_2)	Entropy	Gain
1	Total		14	4	10	0.8631	
	Outlook						0.2582
		Cloudy	4	0	4	0	
		Rainy	5	1	4	0.72192	
		Sunny	5	3	2	0.9709	

Tabel 2.2 (Lanjutan).

Node	Atribut	Nilai Atribut	Jml Kasus (S)	Tidak (S_1)	Ya (S_2)	Entropy	Gain
	Temperature						0.1838
		Cool	4	0	4	0	
		Hot	4	2	2	1	
		Mild	6	2	4	0.9182	
	Humidity						0.3705
		High	7	4	3	0.9852	
		Normal	7	0	7	0	
	Windy						0.0059
		False	8	2	6	0.8112	
		True	6	4	2	0.9182	

Setelah tabel perhitungan *gain* dan *entropy* dibuat, maka langkah selanjutnya adalah melakukan perhitungan dengan mencari nilai *entropy* menggunakan persamaan (2). Selelah menentukan nilai *entropy* langkah selanjutnya yaitu mencari nilai *gain* menggunakan persamaan (1).

Langkah pertama yang harus dilakukan yaitu dengan mencari nilai *entropy* total. Baris total kolom *entropy* pada tabel 2.1 dihitung dengan rumus persamaan (2) sebagai berikut :

$$Entropy\ (Total) = (-\frac{4}{14} * \log_2(\frac{4}{14})) + (-\frac{10}{14} * \log_2(\frac{10}{14}))$$

$$Entropy\ (Total) = 0.863120569$$

Langkah selanjutnya adalah mencari nilai *entropy* untuk masing-masing atribut misalnya pada *Outlook* (*Cloudy*, *Rainy*, *Sunny*) sebagai berikut :

$$EntropyOutlook\ (Cloudy) = (-\frac{0}{4} * \log_2(\frac{0}{4})) + (-\frac{4}{4} * \log_2(\frac{4}{4}))$$

$$EntropyOutlook\ (Cloudy) = 0$$

$$EntropyOutlook\ (Rainy) = (-\frac{1}{5} * \log_2(\frac{1}{5})) + (-\frac{4}{5} * \log_2(\frac{4}{5}))$$

$$EntropyOutlook\ (Rainy) = 0.721928095$$

$$EntropyOutlook\ (Sunny) = (-\frac{3}{5} * \log_2(\frac{3}{5})) + (-\frac{2}{5} * \log_2(\frac{2}{5}))$$

$$EntropyOutlook\ (Sunny) = 0.970950594$$

Perhitungan selanjutnya adalah menhitung nilai *gain* total untuk *Outlook* dengan menggunakan persamaan (1) berdasarkan nilai *entropy* dari masing-masing sebagai berikut :

$$Gain(Total, Outlook) = (Entropy(Total) - \sum_{i=l}^{n} \frac{|Outlook|}{|Total|} * Entropy(Outlook)$$

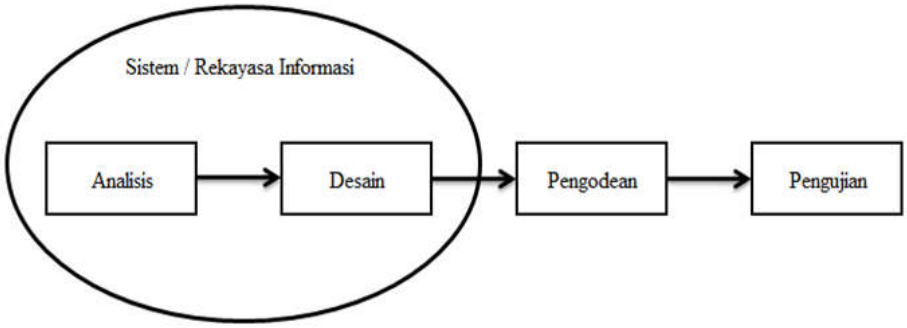
$$Gain(Total, Outlook) = 0.863120569 - ((\frac{4}{14} * 0) + (\frac{5}{14} * 0.723) + (\frac{5}{14} * 0.97))$$

$$Gain(Total, Outlook) = 0.2582$$

Perhitungan nilai *entropy* dan *gain* dilakukan sampai atribut terakhir sehingga diperoleh nilai-nalai *gain* dan *entropy* tertentu. Hasil dari nilai *gain* kemudian dibandingan dengan nilai *gain* lainnya untuk mencari nilai tertinggi sehingga *gain* yang merupakan nilai tertinggi akan dijadikan acuan untuk melakukan proses perhitungan kembali pada proses pembentukan struktur pohon keputusan (Kusrini dan Luthfi, 2009).

2.2 Metode *Waterfall*

Menurut (Rosa dan Shalahuddin, 2016) menjelaskan bahwa model SLDC air terjun (*waterfall*) sering juga disebut model sekuensial *linier* (*sequential liner*) atau alur hidup klasik (*classic life cycle*). Model air terjun menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut dimulai dari analisis, desain, pengkodean, pengujian dan tahap pendukung (*support*). Berikut ini adalah gambar model air terjun :



Gambar 2.3 Metode *Waterfall* (Rosa dan Shalahuddin, 2016).

1) Analisis kebutuhan perangkat lunak

   Proses pengumpulan kebutuhan dilakukan secara intensif untuk mespesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh *user*. Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu untuk di-dokumentasikan.

2) Desain

   Desain perangkat lunak adalah proses multi langkah yang fokus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antarmuka dan prosedur pengodean. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan ke representasi desain agar dapat di implementasikan menjadi program pada tahap selanjutnya. Desain perangkat lunak yang dihasilkan pada tahap ini juga perlu didokumentasikan.

3) Pembuatan kode program

   Desain harus ditranslasikan ke dalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan desain yang telah dibuat pada tahap desain.

4) Pengujian

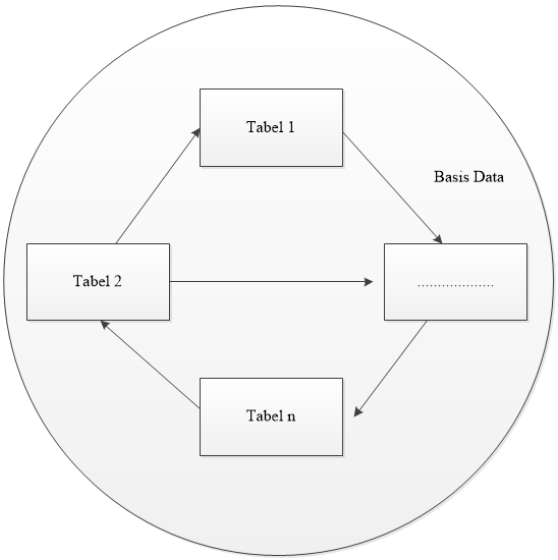
   Pengujian fokus pada perangkat lunak secara dari segi lojik dan fungsional dan memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalisir kesalahan (*error*) dan memastikan keluaran yang dihasilkan sesuai dengan yang diinginkan.

5) Pendukung (*support*) atau pemeliharaan (*maintenance*)

Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah di kirimkan ke *user*. Perubahan bisa terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak harus beradaptasi dengan lingkungan baru. Tahap pendukung atau pemeliharaan dapat mengulangi proses pengembangan mulai dari analisis spesifikasi untuk perubahan perangkat lunak sudah ada, tapi tidak untuk membuat perangkat baru.

2.3 Basis Data

Menurut (Rosa dan Shalahuddin, 2016) Basis data adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat. Basis data dapat diimplementasikan dengan tabel-tabel yang saling memiliki relasi seperti gambar berikut.



Gambar 2.4 Ilustrasi Basis Data (Rosa dan Shalahuddin, 2016).

Kebutuhan basis data dalam sistem informasi meliputi :

- Memasukkan, menyimpan dan mengambil data.
- Membuat laporan berdasarkan data yang telah disimpan.

Tujuan dari dibuatnya tabel-tabel disini adalah untuk menyimpan data ke dalam tabel-tabel agar mudah diakses. Oleh karena itu, untuk merancang tabel-tabel yang akan dibuat maka dibutuhkan pola pikir penyimpanan data nantinya jika dalam bentuk baris-baris data (*record*) dimana setiap baris terdiri dari beberapa kolom.

2.3.1 DBMS

Menurut (Rosa dan Shalahuddin, 2016) menyatakan bahwa DBMS (*Database Management System*) adalah suatu sistem aplikasi yang digunakan untuk menyimpan dan menampilkan data. Suatu sistem aplikasi disebut DBMS jika memenuhi persyaratan minimal sebagai berikut :

- Menyediakan fasilitas untuk mengelola akses data
- Mampu menangani integritas data
- Mampu menangani *backup* data

DBMS sudah mulai berkembang sejak tahun 1960an. Kemudian sekitar tahun 1970an mulai berkembang teknologi *Relational* DBMS yaitu DBMS berbasis relasional model. Relasional model pertama kali dikembangkan oleh Edgar J. Codd pada tahun 1970. Secara sederhanan relasional model dapat dipahami sebagai suatau model yang memandang data sebagai sekumpulan tabel yang saling terkait. Hampir semua DBMS komersial dan *open source* saat ini berbasis *Relational* DBMS atau RDBMS.

Berikut ini adalah 4 macam DBMS versi komersial yang paling banyak digunakan di dunia saat ini, yaitu :

- Oracle
- Microsoft SQL Server
- IBM DB2
- Microsoft Access

Sedangkan DBMS versi *open source* yang cukup berkembang dan paling banyak digunakan saat ini adalah sebagai berikut :

- MySQL

- PostgreSQL
- Firebird
- SQLite

Hampir semua DBMS mengadopsi SQL sebagai bahasa untuk mengelola data pada DBMS.

2.3.2 SQL

SQL (*Struktured Query Language*) adalah bahasa yang biasa dipakai pada sistem database relasional untuk mengakses data. Melalui SQL inilah data dalam MySQL dapat diakses melalui PHP (Kadir, 2009).

Menurut (Rosa dan Shalahuddin, 2016) menyatakan bahwa SQL (*Struktured Query Language*) adalah bahasa yang digunakan untuk mengelola data pada RDBMS. SQL awalnya dikembangkan berdasarkan teori aljabar relasional dan kalkulus. SQL mulai berkembang pada tahun 1970an. SQL mulai digunakan sebagai standar yang resmi pada tahun 1986 oleh ANSI (*American National Standards Insitute*) pada tahun 1987 oleh ISO (*International Organization For Stantardization*) dan disebut SQL-86. Berikut ini adalah contoh pengaksesan data pada DBMS dengan SQL secara umum terdiri dari 4 hal sebagai berikut :

- Memasukkan Data (*insert*)

  INSERT INTO Tabel_mahasiswa (nim, nama, tanggal_lahir) *VALUES* (‘13501058’,’Rosa’,’1986-01-01’;

  Query di atas digunakan untuk memasukkan data mahasiswa dengan NIM 13501058, nama Rosa dan tanggal lahir 1 Januari 1986 ke tabel “Tabel_Mahasiswa”.
- Mengubah Data (*update*)

  UPDATE Tabel_mahasiswa

  SET tanggal_lahir = ‘1990-03-04’

  WHERE nim = ‘13501058’;

Query di atas digunakan untuk mengubah data tanggal lahir mahasiswa dengan NIM = 13501058 menjadi 4 Maret 1990 dalam tabel "Tabel_Mahasiswa".

- Menghapus Data (*delete*)
  DELETE FROM Tabel_mahasiswa
  WHERE nim = '13501058';
  Query di atas digunakan untuk menghapus data mahasiswa dengan NIM = 13501058 dari tabel "Tabel_Mahasiswa".
- Menampilan Data (*select*)
  SELECT nim, nama
  FROM Tabel_mahasiswa
  WHERE nim = '13501058';
  Query di atas digunakan untuk menampilkan data mahasiswa yang tersimpan dalam "Tabel_Mahasiswa dengan NIM = 13501058.

2.3.3 CDM

Menurut (Rosa dan Shalahuddin, 2016) menyatakan bahwa CDM (*Conceptual Data Model*) atau model konsep data merupakan konsep yang berkaitan dengan pandangan pemakai terhadap data yang disimpan dalam basis data. CDM dibuat sudah dalam bentuk tabel-tabel tanpa tipe data yang menggambarkan relasi antar tabel untuk keperluan implementasi ke basis data. CDM merupakan hasil penjabaran lebih lanjut dari ERD.
Berikut adalah simbol-simbol yang ada pada CDM.:

Tabel 2.3 Simbol-simbol Pada PDM.

Simbol	Deskripsi
Entitas / Tabel nama_tabel	Entitas atau tabel yang menyimpan data dalam basis data.

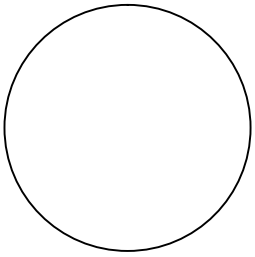
Tabel 2.3 (Lanjutan).

Node	Atribut
Relasi	Relasi antar tabel yang terdiri atas nama relasi dan *multiplicity*.

2.4 *Data Flow Diagram* (DFD)

Menurut (Rosa dan Shalahuddin, 2016) menyatakan bahwa *Data Flow Diagram* (DFD) atau dalam bahasa Indonesia menjadi Diagram Alir Data (DAD) adalah representasi grafik yang menggambarkan aliran informasi dan transformasi yang diaplikasikan sebagai data yang mengalir dari masukan (*input*) dan keluaran (*output*). DFD dapat digunakan untuk merepresentasikan sebuah sistem atau perangkat lunak pada beberapa level abstraksi. DFD dapat dibagi menjadi beberapa level yang lebih detail. DFD menyediakan mekanisme untuk pemodelan fungsional ataupun pemodelan aliran informasi. Oleh karena itu, DFD lebih sesuai digunakan untuk memodelkan fungsi-fungsi perangkat lunak yang akan diimplementasikan menggunkan pemrograman terstuktur karena pemrograman terstuktur membagi-bagi bagiannya dengan fungsi-fungsi prosedur-prosedur. Notasi-notasi pada DFD (Edward Yourdon dan Tom DeMarco) adalah sebagai berikut :

Tabel 2.4 Notasi *Data Flow Diagram* (DFD).

Notasi	Keterangan
	Proses atau fungsi atau prosedur pada pemodelan perangkat lunak yang akan diimplementasikan dengan pemograman terstruktur, maka pemodelan notasi inilah yang harusnya menjadi fungsi atau prosedur didalam kode program

Tabel 2.4 (Lanjutan).

Notasi	Keterangan
_____ _____	*File* atau basis data atau penyimpanan (*storage*) pada pemodelan perangkat lunak yang akan diimplementasikan dengan pemrograman terstruktur, maka pemodelan notasi inilah yang harusnya dibuat mejadi table-tabel basis data yang dibutuhkan, table-tabel ini juga harus sesuai dengan perancangan table-tabel pada basis data (ERD)
	Entitas luar (*external entity*) atau masukan (*input*) atau keluaran (*output*) atau orang yang memakai atau berinteraksi dengan perangkat lunak yang dimodelkan atau sistem lain yang terkait dengan aliran data dari sistem yang dimodelkan
	Aliran data, merupakan data yang dikirim antar proses, dari penyimpanan ke proses atau dari proses kemasukan *(input)* atau keluaran *(ouput)*

2.5 Aplikasi Web

Menurut (Kadir, 2009) aplikasi web adalah jenis aplikasi yang diakses melalui *browser*, misalnya Internet Explorer dan Mozilla Firefox. Aplikasi web yang paling dasar ditulis dengan menggunakan HTML. Sebagaimana diketahui, HTML (*Hyper Text Markup Language*) adalah bahasa standar untuk membuat halaman-halaman web.

2.5.1 PHP

Menurut (Kadir, 2009) menyatakan bahwa PHP adalah skrip yang dijalankan di *server*. Jadi, konsepnya berbeda dengan JavaScript yang

dijalankan pada sisi klien. Keuntungan penggunaan PHP, kode yang menyusun program tidak perlu dibagikan ke pemakai yang berarti bahwa kerahasiaan kode dapat dilindungi. Hal menarik yang didukung oleh PHP tetapi tidak mungkin dilakukan oleh JavaScript adalah kenyataan bahwa PHP bisa digunakan untuk mengakses berbagai macam database, seperti Access, Oracle, MySQL, dan lain-lain. Seperti halnya JavaScript, kode PHP dapat disisipkan pada kode HTML. Selain itu PHP juga bisa digunakan untuk menghasilkan kode-kode HTML. Skrip PHP diawali dengan <?php dan diakhiri dengan ?>. Di dalam pasangan tanda tersebut terdapat pernyataan-pernyataan PHP. Seperti halnya pada JavaScript, antar pernyataan harus dipisahkan oleh tanda titik-koma (;).

2.5.2 HTML

Menurut (Kadir, 2009) menyatakan bahwa HTML adalah bahasa dengan tanda-tanda khusus yang digunakan di awal era web untuk menyajikan informasi. Kode HTML di awali dengan <html> dan diakhiri dengan </html>. Namun sebelum <html> diharuskan untuk diberikan deklarasi <!DOCTYPE>. Fungsi deklarasi ini adalah sebagai *Public Text Identifier*. Beberapa hal penting dalam kode HTML :

1. Tanda <> menyatakan sebuah tag.
2. Pada umumnya tag berpasangan. Contoh, <html> dengan </html>, <head> dengan </head> dan <body> dengan </body>.
3. Tag yang tidak berpasangan antara lain
 dan <hr/>
4. Pada tag yang berpasangan, tag yang berkedudukan sebagai tag penutup mempunyai bentuk berupa </ >.

2.5.3 Javascript

Menurut (Kadir, 2009) menyatakan bahwa Javascript adalah bahasa skrip (bahasa yang kodenya ditulis menggunakan teks biasa) yang ditempelkan pada dokumen HTML dan diproses pada sisi klien. Dengan adanya bahasa ini, kemampuan dokumen HTML menjadi semakin luas. Sebagai contoh,

dengan menggunakan JavaScript dimungkinkan untuk memvalidasi masukan-masukan pda formulir sebelum formulir dikirm ke *server*. Selain itu, dengan menggunakan JavaScript juga dimungkinkan untuk mengimplementasikan tugas yang bersifat interaktif tanpa berhubungan dengan *server*. Beberapa contoh yang bisa dilakukan melalui JavaScript :

1. Memanpilkan jam lokal pada halaman web.
2. Mengatur warna latar belakang halaman web.
3. Mengganti gambar ketika pemakai menempatkan penunjuk *mouse* ke suatu gambar.
4. Memvalidasi keabsahan data yang dimasukkan oleh pemakai.

2.5.4 CSS

Menurut (Prasetio, 2012) "*Cascading Style Sheet* (CSS) adalah suatu teknologi yang di gunakan untuk memperindah halaman *website* (situs)". CSS mempunyai 2 bagian utama yaitu selectors dan deklarasi. Yang dimaksud selectors biasanya element HTML yang ingin diubah, sedangkan deklarasi biasanya terdiri dari properti dan nilai. Properti sendiri adalah atribut style yang di ingin diubah, dan setiap properti memiliki nilai. *Cascading Style Sheet* (CSS) merupakan aturan untuk mengatur beberapa komponen dalam sebuah web sehingga akan lebih terstruktur dan seragam.

2.5.5 *Framework* Bootstrap

Menurut (Tim Litbang Wahana Komputer, 2016) menyatakan bahwa Bootstrap adalah sebuah *framework* CSS dari Twitter yang menyediakan komponen-komponen antarmuka siap pakai dan telah dirancang sedemikian rupa untuk keperluan desain halaman *website* yang artistik. Sebagai CSS Framework, Bootstrap tergolong paket lengkap. Teknologi HTML, CSS, JavaScript yang ada pada Bootstrap, tidak hanya dapat melakukan *styling* dengan CSS saja, akan tetapi juga dapat menggunakan komponen-komponen seperti *icon*, tombol dan navigasi dengan desain unik khas Bootstap. Bootstrap adalah *framework* CSS yang gratis dengan

menggunakan lisensi MIT. MIT adalah sebuah lisensi *open source* dimana semua kalangan bebas menggunakan *framework* ini secara gratis tanpa takut terkena masalah legalitas..

2.5.6 MySQL

Menurut (Anhar, 2010) mengatakan bahwa MySQL (*My Structured Query Language*) adalah sebuah program pembuat dan pengelola database atau yang sering disebut DBMS (*Database Management System*), sifat dari DBMS ini adalah Open Source dan ini didapatkan gratis pada alamat http://www.mysql.com. MySQL awalnya dibuat oleh perusahaan konsultan bernama TcX yang berlokasi di Swedia dan dulunya MySQL berjalan pada Platform Linux, dengan adanya perkembangan dan banyaknya pengguna, serta lisensi dari database ini adalah *Open Source*, maka para ahli pengembang merilisnya dalam versi Windows.

2.6 *Black-Box Testing* (Pengujian Kotak Hitam)

Menurut (Rosa dan Shalahuddin, 2016) *Black-box testing* yaitu menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan. Pengujian kotak hitam dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan. Kasus uji yang dibuat untuk melakukan pengujian kotak hitam harus dibuat dengan kasus benar dan kasus salah, misal untuk proses *login* maka kasus uji yang dibuat adalah :

1. Jika *user* memasukan nama pemakai (*username*) dan kata sandi (*password*) yang benar.
2. Jika *user* memasukan nama pemakai (*username*) dan kata sandi (*password*) yang salah, misalnya nama pemakai benar tapi kata sandi salah, atau sebaliknya, atau keduanya salah.