BAB II

LANDASAN TEORI

2.1. Pengertian Sistem

Sistem adalah suatu himpunan dari berbagai bagian atau elemen yang saling berhubungan secara teorganisasi berdasarkan fungsi-fungsinya, menjadi satu kesatuan. (Hartono, 2013).

Sistem adalah suatu kumpulan dari unsur, komponen, atau variabel yang terorganisasi, saling berinteraksi, saling tergantung satu sama lain, yang berfungsi bersama-sama untuk mencapai tujuan tertentu. (Sutabri, 2012).

Jadi, dapat disimpulkan bahwa sistem adalah komponen yang saling berinteraksi satu sama lain yang bersama sama untuk mencapai tujuan tertentu.

2.2. Sistem Pendukung Keputusan

Sistem pendukung keputusan (SPK) adalah bagian dari sistem informasi berbasis komputer termasuk sistem berbasis pengetahuan atau manajemen pengetahuan yang digunakan untuk mendukung pengambilan keputusan dalam suatu organisasi atau perusahaan. Dapat juga dikatakan sebagai sistem pendukung berbasis komputer bagi para pengambil

keputusan manajemen yang menangani masalah-masalah yang tidak terstruktur (Widodo, 2011).

Pengertian diatas dapat dijelaskan bahwa SPK bukan merupakan alat pengambilan keputusan, melainkan merupakan sistem yang membantu pengambil keputusan dengan melengkapi mereka dengan informasi dari data yang telah diolah dengan relevan dan diperlukan untuk membuat keputusan tentang suatu masalah dengan lebih cepat dan akurat. Sistem ini tidak dimaksudkan untuk menggantikan pengambilan keputusan dalam proses pembuatan keputusan.

2.3. Logika Fuzzy

Logika fuzzy adalah suatu cara yang tepat untuk memetakan suatu ruang input ke dalam suatu ruang output. Terdapat beberapa alasan orang menggunakan logika fuzzy (Widodo & Handayanto, 2012), antara lain adalah.

a. Konsep logika fuzzy mudah dimengerti. Konsep matematis yang mendasari penalaran fuzzy sangat sederhana dan mudah dimengerti.
b. Logika fuzzy sangat fleksibel.
c. Logika fuzzy memiliki toleransi terhadap data-data yang tidak tepat.
d. Logika fuzzy mampu memodelkan fungsi-fungsi nonlinear yang sangat kompleks.

e. Logika fuzzy dapat membangun dan mengaplikasikan pengalaman-pengalaman para pakar secara langsung tanpa harus melalui proses pelatihan.

f. Logika fuzzy dapat bekerja sama dengan teknik-teknik kendali secara konvensional.

g. Logika fuzzy didasarkan pada bahasa alami.

Secara umum, sistem logika fuzzy memiliki 4 elemen yaitu.

1. Basis aturan yang berisi aturan-aturan yang bersumber dari pakar.
2. Suatu mekanisme pengambilan keputusan dimana pakar mengambil keputusan dengan menerapkan pengetahuan yang dimiliki.
3. Proses fuzzifikasi (*fuzzification*) yang merubah besaran tegas (*criso*) kedalam besaran fuzzy.
4. Proses defuzzifikasi (*defuzzification*), merupakan kebalikan dari proses fuzzifikasi yaitu besaran fuzzy hasil dari *inference engine*, menjadi besaran tegas (*crisp*).

Cara kerja logika fuzzy memiliki 3 bagian, yaitu fuzzyfikasi, inferensi fuzzy, dan defuzzyfikasi. Namun, proses defuzzyfikasi disini bersifat optimal yaitu apabila kesimpulan sudah memenuhi atau sesuai dengan yang diharapkan, maka tidak perlu dilakukan proses defuzzyfikasi tetap dilakukan.

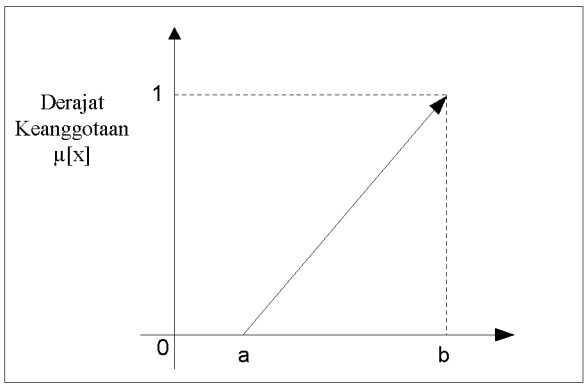
Logika Fuzzy juga memiliki Fungsi Keanggotaan (*membership function*) yaitu suatu kurva yang menunjukkan pemetaan titik-titik input data ke dalam nilai keanggotaannya (sering juga disebut dengan derajat

keanggotaan) yang memiliki interval antara 0 sampai 1. Salah satu cara yang dapat digunakan untuk mendapatkan nilai keanggotaan adalah dengan melalui pendekatan fungsi. Ada beberapa fungsi yang bisa digunakan.

2.3.1 Representasi Linear

Representasi Linear adalah pemetaan input ke derajat keanggotannya digambarkan sebagai suatu garis lurus. Pada representasi linear terdapat 2 kemungkinan, yaitu:

1. Kenaikan himpunan dimulai pada nilai domain yang memiliki derajat keanggotaan nol (0) bergerak ke arah kanan menuju nilai domain yang memiliki derajat keanggotaan lebih tinggi.



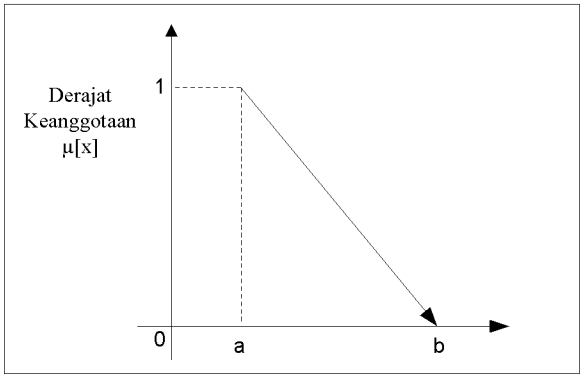
Gambar 2.1 Representasi Kurva *Linear* Naik

Fungsi keanggotaan :

$$\mu[x,a,b] = \begin{cases} 0; & x \leq a \\ \frac{x-a}{b-a}; & a \leq x \leq b \\ 1; & x \geq b \end{cases}$$

2. Penurunan himpunan dimulai dari nilai domain dengan derajat keanggotaan tertinggi pada sisi kiri, kemuadian bergerak menurun ke nilai domain yang memiliki derajat keanggotaan

lebih rendah.



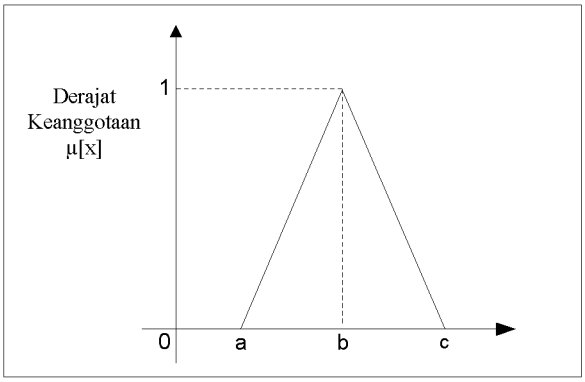
Gambar 2.2 Representasi Kurva *Linear* Turun

Fungsi keanggotaan :

$$\mu[x,a,b] = \begin{cases} \frac{b-x}{b-a}; & a \leq x \leq b \\ 0; & x \geq b \end{cases}$$

2.3.2 Representasi Kurva Segitiga

Kurva segitiga pada dasarnya terbentuk dari gabungan antara 2 garis (*linear*).



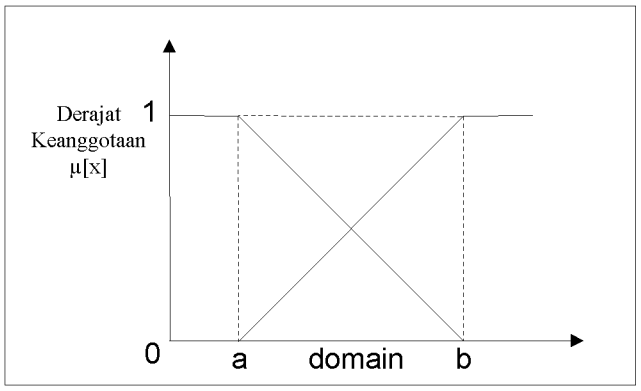
Gambar 2.3 Representasi Kurva Segitiga

Fungsi Keanggotaan :

$$\mu[x,a,b] = \begin{cases} 0; & x \leq a \; atau \; \geq c \\ \frac{x-a}{b-a}; & a \leq xb \\ \frac{c-x}{c-b}; & b \leq x \geq c \end{cases}$$

2.3.3 Representasi Kurva Bentuk Bahu

Daerah yang terbentuk di tengah-tengah suatu variabel yang direpresentasikan dalam bentuk kurva segitiga, pada sisi kanan dan kirinya akan naik turun. Tetapi terkadang salah satu sisi dari variabel tersebut tidak mengalami perubahan. Himpunan *fuzzy* "bahu", digunakan untuk mengakhiri variabel suatu daerah *fuzzy*.



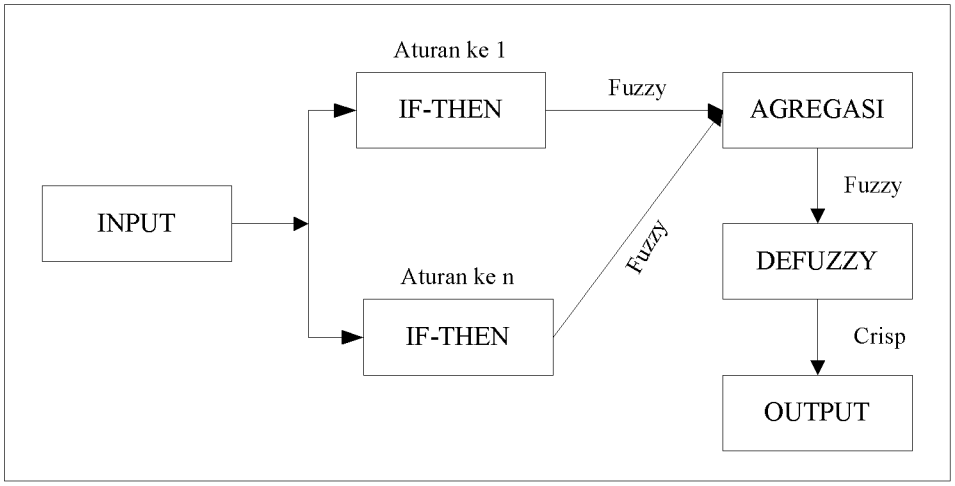
Gambar 2.4 Representasi Kurva Bentuk Bahu

Fungsi Keanggotaan :

$$\mu[x,a,b] = \begin{cases} 0; & x \geq b \\ \frac{b-x}{b-a}; & x \leq x \leq b \\ 1; & x \geq a \\ 0; & x \leq a \\ \frac{x-a}{b-a}; & a \leq x \leq b \\ 1; & x \geq b \end{cases}$$

2.4. Fuzzy Tsukamoto

Sistem Inferensi Fuzzy merupakan suatu kerangka komputasi yang didasarkan pada teori himpunan fuzzy, aturan fuzzy berbentuk IF-THEN, penalaran fuzzy. Secara garis besar, diagram blok proses inferensi fuzzy.



Gambar 2.5 Diagram blok sistem inferensi Fuzzy Tsukamoto

Sistem inferensi fuzzy menerima input crisp. Input ini kemudian dikirim ke basis pengetahuan yang berisi n aturan fuzzy dalam bentuk IF-THEN. Fire strength akan dicari pada setiap aturan. Apabila jumlah aturan lebih dari satu, maka akan dilakukan agregasi dari semua aturan. Selanjutnya, pada hasil agregasi akan dilakukan defuzzy untuk mendapatkan nilai crisp sebagai output sistem.

Pada dasarnya, metode tsukamoto mengaplikasikan penalaran monoton pada setiap aturannya. Kalau pada penalaran monoton, sistem hanya memiliki satu aturan, pada metode tsukamoto, sistem terdiri atas beberapa aturan. Karena menggunakan konsep dasar penalaran monoton, pada metode tsukamoto, setiap konsekuen pada aturan yang berbentuk IF-THEN harus direpresentasikan dengan suatu himpunan fuzzy dengan fungsi keanggotaan yang monoton. Output hasil inferensi dari tiap-tiap aturan diberikan secara tegas (*crisp*) bedasarkan α-predikat (*fire strength*).

Proses agregasi antar aturan dilakukan, dan hasil akhirnya diperoleh dengan menggunakan defuzzy dengan konsep rata-rata terbobot.

Misalkan ada variabel input, yaitu x dan y, serta satu variabel output yaitu z. Variabel x terbagi atas 2 himpunan yaitu A1 dan A2, variabel y terbagi atas 2 himpunan juga, yaitu B1 dan B2, sedangkan variabel output Z terbagi atas 2 himpunan yaitu C1 dan C2. Tentu saja himpunan C1 dan C2 harus merupakan himpunan yang bersifat monoton. Diberikan 2 aturan sebagai berikut :

IF x is A1 and y is B2 THEN z is C1

IF x is A2 and y is B2 THEN z is C1.

2.5. Pengertian *Unified Modeling Language* (*UML*)

Unified Modeling Language (*UML)* adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan requirement, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek. (Sukamto dan Shalahuddin, 2013).

UML (*Unified Modeling Language*) menyediakan serangkaian gambar dan diagram yang sangat baik. Beberapa diagram memfokuskan diri pada ketangguhan teori *object-oriented* dan sebagian lagi memfokuskan pada detail rancangan dan konstruksi. Semua dimaksudkan sebagai sarana komunikasi antar *team programmer* maupun dengan pengguna. (Widodo dan Herlawati, 2011), "*UML* diaplikasikan untuk

maksud tertentu", biasanya antara lain untuk :

1. Merancang perangkat lunak.
2. Sarana komunikasi antara perangkat lunak dengan proses bisnis.
3. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem.
4. Mendokumentasi sistem yang ada, proses-proses dan organisasinya.

Tabel 2.1 Tipe Diagram *Unified Modeling Language* (*UML*)

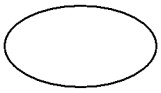
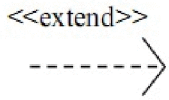
No.	Diagram	Tujuan
1	*Class*	Memperlihatkan himpunan kelas-kelas, antarmuka – antarmuka, kolaborasi-kolaborasi, serta relasi-relasi
2	*Package*	Memperlihatkan kumpulan kelas-kelas, merupakan dari diagram komponen
3	*Use case*	Diagram ini memperlihatkan himpunan *use case* dan aktor-aktor (suatu jenis khusus dari kelas)
4	*Sequence*	Diagram interaksi yang menekankan pada pengiriman pesan dalam suatu waktu tertentu
5	*Communication*	Sebagai pengganti diagram kolaborasi *UML* 1.4 yang Menekankan organisasi struktural dari obyek-obyek yang menerima serta mengirim pesan
6	*Statechart*	Diagram status memperlihatkan keadaan – keadaan pada sistem, memuat status *(state)*, transisi, kejadian serta aktivitas
7	*Activity*	Tipe khusus dari diagram status yang memperlihatkan aliran dari suatu aktivitas ke aktivitas lainnya dalam suatu sistem
8	*Component*	Memperlihatkan organisasi serta kebergantungan sistem atau perangkat lunak pada komponen - komponen yang telah ada sebelumnya
9	*Deployment*	Memperlihatkan konfigurasi saat aplikasi dijalankan *(run-time)*

2.6 Jenis-Jenis Diagram *Unified Modeling Language* (*UML*)

a. *Use Case Diagram*

Use case atau diagram *use case* merupakan pemodelan untuk kelakuan *(behavior)* sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. (Sukamto dan Shalahuddin, 2013). Adapun simbol-simbol yang digunakan dalam *use case* adalah sebagai berikut:

Tabel 2.2 Simbol-simbol *Use case* Diagram

No	Simbol	Nama	Keterangan
1		*Actor*	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri.
2		*Use case*	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.
3		*Association*	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
4		*Extend*	Relasi *use case* tambahan ke sebuah *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu.
5		*Generalization*	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.

Tabel 2.3 Lanjutan Simbol-simbol *Use case* Diagram

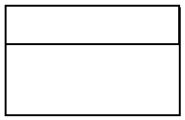
6	<<include>> -------->	*Include*	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini.

b. *Class Diagram*

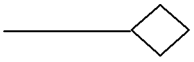
Class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. (Sukamto dan Shalahuddin, 2013).

Diagram kelas dibuat agar pembuat program atau *programmer* membuat kelas-kelas sesuai rancangan di dalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak sinkron. Adapun simbol-simbol yang digunakan dalam *class* diagram adalah sebagai berikut.

Tabel 2.4 Simbol-simbol *Class* Diagram

No	Simbol	Nama	Keterangan
1		*Class*	Kelas pada stuktur sistem.
2		*Interface*	Sama dengan konsep *interface* dalam pemrograman berorientasi objek.
3		*Association*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*.

Tabel 2.5 Lanjutan Simbol-simbol *Class* Diagram

4		*Directed association*	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain.
5		*Generalization*	Relasi antar kelas dengan makna generalisasi – spesialisasi (umum khusus).
6		*Dependency*	Relasi antar kelas dengan makna kebergantungan antar kelas.
7		*Aggregation*	Relasi antar kelas dengan makna semua bagian *(whole-part)*.

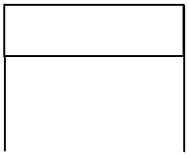
c. *Activity Digaram*

Activity diagram menggambarkan aliran kerja (*workflow*) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. (Sukamto dan Shalahuddin, 2013). Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem. Adapun simbol-simbol yang digunakan dalam *activity* diagram adalah sebagai berikut.

Tabel 2.6 Simbol-simbol *Activity* Diagram

No	Simbol	Nama	Keterangan
1		Status awal	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
2		Aktivitas	Aktivitas yang dilakukan sistem, biasanya diawali dengan kata kerja.

Tabel 2.7 Lanjutan Simbol-simbol *Activity* Diagram

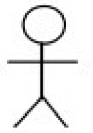
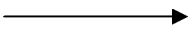
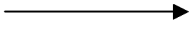
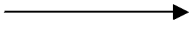
3		Decision	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
4		*Join*	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
5		Status akhir	Status akhir yang dilakukan sebuah sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
6		*Swimlane*	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

d. ***Sequence Diagram***

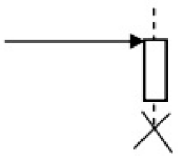
Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. (Sukamto dan Shalahuddin, 2013).

Sequence diagram menunjukkan urutan *event* kejadian dalam suatu waktu. Komponen *sequence* diagram terdiri atas obyek yang dituliskan dengan kotak segiempat bernama. *Message* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan progress vertikal. Simbol-simbol yang digunakan dalam *sequence* diagram adalah.

Tabel 2.8 Simbol-simbol *Sequence* Diagram

No	Simbol	Nama	Keterangan
1		Aktor atau Tanpa Waktu Aktif	Digunakan untuk menggambarkan *user*/pengguna.
2		Garis hidup atau *lifeline*	Menyatakan kehidupan suatu objek.
3		*Boundary*	Digunakan untuk menggambarkan sebuah *form.*
4		*Control*	Digunakan untuk menghubungkan *boundary* dengan tabel.
5		*Entity*	Digunakan untuk menggambarkan hubungan kegiatan yang akan dilakukan.
6		Waktu Aktif	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya.
7	<<create>> 	Pesan tipe *create*	Menyatakan suatu objek membuat objek lain, arah panah mengarah pada ojek yang dibuat.
8	1:nama_metode() 	Pesan tipe *call*	Arah panah mengarah pada objek yang memiliki operasi atau metode, karena ini memanggil operasi atau metode maka operasi atau metode yang dipanggil harus ada pada diagram kelas sesuai denga kelas objek yang berinteraksi.
7	1:masukan 	Pesan tipe *send*	Menyatakan bahwa suatu objek mengirimkan data atau masukan atau informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi.

Tabel 2.9 Lanjutan Simbol-simbol *Sequence* Diagram

8	1:keluaran	Pesan tipe *return*	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.
9	<<destroy>> 	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*.

2.7. Basis Data

Basis Data merupakan kumpulan data yang berlasi yang disusun, diorganisasikan dan disimpan secara sistematik dalam media simpan komputer mengacu pada metode-metode tertentu sedemikian rupa sehingga dapat diakses secara cepat dan mudah menggunakan program/aplikasi komputer untuk memperoleh data dari basis data tersebut.(*Fathansyah,2011*).

2.8. MySql

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL atau DBMS (*Database Management System*) yang *multi-thread* dan *multi-user* dengan sekitar 6 juta instalasi di seluruh dunia.MySQL tersedia sebagai perangkat lunak gratis dibawah lisensi GNU *General Public License* (GPL), tetapi mereka juga menjual di bawah lisensi

komersial untuk kasus-kasus dimana penggunaannya tidak cocok dengan penggunaan GPL (Aditya, 2011). MySQL memiliki sejumlah fitur seperti yang akan dijelaskan dibawah ini.

1. Mutliplatform

   MySQL tersedia pada beberapa platform (windows, linux, unix, dan lainlain).

2. Kuat, cepat dan mudah digunakan.

   MySQL tergolong sebagai database server (server yang melayani permintaan terhadap database) yang andal, dapat menangani database database yang besar dengan kecepatan tinggi. Mendukung banyak sekali fungsi untuk mengakses database dan sekaligus mudah untuk digunakan.

3. Jaminan keamanan akses

   MySQL mendukung pengamanan database dengan berbagai kriteria pengaksesan. Sebagai gambaran, dimungkinkan untuk mengatur user tertentu agar bisa mengakses data yang bersifat rahasia (misal gaji pegawai), sedangkan user lain tidak boleh sesuai dengan hak aksesnya.

4. Dukungan SQL

   Seperti tersirat namanya, SQL mendukung perintah SQL (*Structured Query Language*). Sebagaimana diketahui SQL merupakan bahasa standar dalam pengaksesan database rasional.

2.9. Borland Delphi 7.0

Delphi adalah sebuah piranti pengembangan aplikasi berbasis windows yang dikeluarkan oleh Borland Internasional. Perangkat lunak ini sangat dikenal di lingkungan pengembangan aplikasi karena mudah untukdipelajari dan dapat digunakan untuk dipelajari yang dapat digunakan untuk menangani berbagai hal, dari aplikasi matematika, permainan(*games*), Hingga database. Pada penanganan database, Delphi menyediakan fasilitas yang memungkinkan program dapat berinteraksi dengan database seperti dBase, Paradox,Oracle, MySQL dan Acces (Abdul, 2012). Dalam perkembangannya, aplikasi Borland Delphi 7.0 memiliki berbagai kemudahan dan keunggulan dibandingkan dengan beberapa produk sejenis yang ada. Hal inilah yang menjadi alasan menggunakan Borland Delphi 7,0 dalam pembuatan pemrograman dekstop, berikut adalah beberapa kelebihan yang dimiliki oleh Borland Delphi.

a. *Freeware*

Borland Delphi 7.0 adalah software yang bisa di download dan digunakan dengan gratis, tetapi source code software tersebut tidak bisa dilihat karena masih memiliki batasan atas hak cipta pengguna.

b. Mempunyai desain yang *user friendly* terhadap para programmer *beginer*.

c. Mempunyai komponen yang sangat komplek untuk pembuatan *software* aplikasi sampai database.

d. Mempunyai aplikasi *plugin* database bawaan (BDE).

e. Mempunyai kecepatan kompilasi yang cepat.

f. Aplikasi yang dihasilkan bisa merupakan *File Executable portable* dan *Executable installer.*

g. Mudah untuk membuat koneksi ke berbagai aplikasi database, misalnya Acces, MySql database lainnya.