BAB II

TINJAUAN PUSTAKA

II.1. Sistem

Sistem berasal dari bahasa latin (*systēma*) dan bahasa Yunani (*sustēma*) adalah suatu kesatuan yang terdiri komponen atau elemen yang dihubungkan bersama untuk memudahkan aliran informasi, materi atau energi. Istilah ini sering dipergunakan untuk menggambarkan suatu set entitas yang berinteraksi, dimana suatu model matematika seringkali bisa dibuat. (Pipi Wijia Astuti, 2015)

II.2. Keputusan

Keputusan adalah suatu reaksi terhadap beberapa solusi alternatif yang di lakukan secara sadar dengan cara menganalisa kemungkinan-kemungkinan dari alternatif tersebut bersama konsekuensinya. (Pipi Wijia Astuti, 2015)

II.3 Sistem Pendukung Keputusan

Sistem Pendukung Keputusan merupakan sistem informasi interaktif yang menyediakan informasi, pemodelan, dan pemanipulasian data. Sistem ini digunakan untuk membantu pengambilan keputusan dalam situasi yang semiterstruktur dan situasi yang tidak terstruktur, dimana tidak seorang pun tahu secara pasti bagaimana keputusan seharusnya dibuat.

SPK juga dapat didefinisikan sebagai “sistem berbasis komputer interaktif, yang membantu para pengambil keputusan untuk menggunakan data dan berbagai

model untuk memecahkan masalah tidak terstruktur". SPK dirancang untuk menunjang seluruh tahapan pembuatan keputusan yang dimulai dari tahap mengidentifikasi masalah, memilih data yang relevan, menentukan pendekatan yang digunakan dalam proses pembuatan keputusan, sampai pada kegiatan mengevaluasi pemilihan alternatif.

Turban dkk, memberikan pengertian yang mencakup semua SPK mulai dari dasar sampai yang paling ideal. Turban dkk, menyatakan bahwa SPK merupakan suatu pendekatan (metodologi) untuk mendukung pengambilan keputusan. SPK menggunakan CBIS (*Computer Based Information System*) yang fleksibel, interaktif dan dapat diadaptasi yang dikembangkan untuk mendukung solusi untuk masalah manajemen spesifik yang tidak terstruktur.

Dari uraian diatas dapat disimpulkan bahwa, sistem pendukung keputusan adalah suatu sistem berbasis komputer yang dapat menghasilkan alternatif terbaik yang telah ditentukan berdasarkan kriteria- kriteria tertentu untuk membantu para pengambil keputusan dalam menentukan keputusan secara objektif. (Dwi Citra Hartini dkk, 2013)

II.3.1. Karakteristik Sistem Pendukung Keputusan

Beberapa karakteristik yang membedakan Sistem Pendukung Keputusan dengan sistem informasi lainnya yaitu:

1. Sistem Pendukung Keputusan dirancang untuk membantu pengambilan keputusan dalam memecahkan masalah yang sifatnya semi terstruktur ataupun tidak terstruktur.

2. Dalam proses pengolahannya, sistem pendukung keputusan mengkombinasikan model-model analisis dengan teknik pemasukan dan konvensional secara fungsi-fungsi pencarian informasi.
3. Sistem Pendukung Keputusan dirancang sedemikian rupa sehingga dapat digunakan atau dioperasikan dengan mudah oleh orang-orang yang tidak memiliki dasar kemampuan pengoperasian komputer yang tinggi. Oleh karena itu pendekatan yang digunakan biasanya model interaktif.
4. Sistem Pendukung Keputusan dirancang dengan menekankan pada aspek fleksibilitas serta kemampuan adaptasi yang tinggi. Sehingga mudah disesuaikan dengan berbagai perubahan lingkungan yang terjadi pada kebutuhan pemakai. (Dwi Citra Hartini dkk, 2013)

II.3.2. Ciri-Ciri Sistem Pendukung Keputusan

Menurut Kusrini (2007), dalam Jurnal Asep Sujarwadi dan Dodo Zaenal Abidin (2016) dengan judul "Perancangan Sistem Pendukung Keputusan Dengan Metode S*imple Additive Weighting* (SAW) Dalam Penentuan Tunjangan Kinerja Pegawai Pada Kepolisian Resort Kota (POLRESTA) Jambi". Adapun kriteria atau ciri-ciri dari suatu keputusan adalah sebagai berikut:

1. Banyak alternatif atau pilihan;
2. Ada kendala atau syarat;
3. Mengikuti suatu pola atau model, baik yang terstruktur maupun tidak terstruktur;
4. Banyaknya input atau variabel;

5. Adanya faktor resiko;
6. Dibutuhkan kecepatan, ketepatan dan keakuratan.

II.3.3. Komponen Sistem Pendukung Keputusan

Sistem Pendukung Keputusan terdiri dari empat komponen utama, yaitu :

1. Subsistem Manajemen Data

   Pada subsistem manajemen data terdapat basis data yang berisikan data-data yang relevan dengan situasi yang ada dan dikelola menggunakan perangkat lunak yang disebut *Database Management System* (DBMS). Biasanya data disimpan dan diakses melalui suatu *database web server*. Kemampuan yang dibutuhkan dari manajemen basis data dapat disimpulkan sebagai berikut:

   a. Kemampuan untuk mengkombinasikan berbagai variasi data melalui pengambilan dan ekstraksi data.
   b. Kemampuan untuk menambahkan sumber data secara cepat dan mudah.
   c. Kemampuan untuk menggambarkan struktur data logikal sesuai dengan pengertian pemakai.
   d. Kemampuan menangani data secara personal, sehingga pemakai dapat mencoba berbagai alternatif penanganan data.
   e. Kemampuan mengelola berbagai variasi data.

2. Subsistem Manajemen Model

   Salah satu keunggulan SPK adalah kemampuan mengintegrasikan akses data dan model keputusan. Hal ini dapat dilakukan dengan menambahkan model keputusan ke dalam sistem informasi yang menggunakan database sebagai

mekanisme integrasi dan komunikasi antar model. Kemampuan yang dibutuhkan pada subsistem manajemen model meliputi:

a. Kemampuan untuk menciptakan model-model baru secara cepat dan mudah.
b. Kemampuan untuk mengakses dan mengintegrasikan model-model keputusan.
c. Kemampuan untuk mengelola basis model dengan fungsi manajemen yang analog dan manajemen *database*.

3. Subsistem Antarmuka Pengguna

Fleksibilitas dan kekuatan karakteristik SPK ialah adanya kemampuan berinteraksi antara sistem dan pemakai, yang dinamakan subsistem *user interface* (antarmuka pengguna). Subsistem ini dapat dibagi menjadi tiga bagian yaitu:

a. Bahasa aksi, meliputi apa yang dapat digunakan oleh pemakai dalam berkomunikasi dengan sistem.
b. Bahasa tampilan dan presentasi, meliputi apa yang harus diketahui oleh pemakai.
c. Basis pengetahuan, meliputi apa yang harus diketahui pemakai agar penggunaan sistem pendukung keputusan bisa efektif.

Kemampuan yang harus dimiliki oleh subsistem pendukung keputusan ini meliputi:

1. Kemampuan menangani versi dialog, sesuai kondisi pemakai.

2. Kemampuan mengakomodasi tindakan pemakai dengan berbagai alat masukan.
3. Kemampuan menampilkan data dengan berbagai variasi format dan alat keluaran.
4. Kemampuan untuk mendukung dan mengetahui basis pengetahuan pemakai.

4. Subsistem Basis Pengetahuan

   Subsistem basis pengetahuan adalah subsistem yang sifatnya opsional, namun akan sangat menguntungkan apabila digunakan untuk menunjang tiga subsistem utama. Subsistem ini menggunakan kecerdasan buatan sehingga sistem dapat mengambil tindakan secara otomatis sesuai dengan keinginan pengguna. (Dwi Citra Hartini dkk, 2013)

II.3.4. Tujuan Sistem Pendukung Keputusan

Tujuan dari Sistem Pendukung Keputusan menurut Turban dkk adalah sebagai berikut:

1. Membantu manajer dalam pengambilan keputusan atas masalah semi terstruktur.
2. Memberikan dukungan atas pertimbangan manajer dan bukannya di maksudkan untuk menggantikan fungsi manajer.
3. Meningkatkan efektivitas keputusan yang di ambil manajer lebih daripada perbaikan efisiensinya.

4. Kecepatan *komputasi*. Komputer memungkinkan para pengambil keputusan untuk melakukan banyak *komputasi* secara cepat dengan biaya yang rendah.
5. Peningkatan produktivitas. Membangun suatu kelompok pengambil keputusan, terutama para pakar, bisa sangat mahal. Pendukung terkomputerisasi bisa mengurangi ukuran kelompok dan memungkinkan para anggotanya untuk berada di berbagai lokasi yang berbeda-beda (menghemat biaya perjalanan). Selain itu, produktivitas staf pendukung (misalnya analisis keuangan dan hukum) bisa di tingkatkan. Produktivitas juga bisa di tingkatkan menggunakan peralatan optimasi yang menentukan cara terbaik untuk menjalankan sebuah bisnis.
6. Dukungan kualitas. Komputer bisa meningkatkan kualitas keputusan yang di buat. Sebagai contoh, semakin banyak data yang di akses, makin banyak juga alternatif yang bisa di evaluasi. Analisis resiko bisa di lakukan dengan cepat dan pandangan dari para pakar (beberapa dari mereka berada di lokasi yang jauh) bisa dikumpulkan dengan cepat dan dengan biaya yang lebih rendah. Keahlian bahkan bisa di ambil langsung dari sebuah sistem komputer melalui metode kecerdasan tiruan. Dengan *computer*, para pengambil keputusan bisa melakukan simulasi yang kompleks, memeriksa banyak skenario yang memungkinkan, dan menilai berbagai pengaruh secara cepat dan ekonomis. Semua kapabilitas tersebut mengarah kepada keputusan yang lebih baik.
7. Berdaya saing. Manajemen dan pemberdayaan sumber daya perusahaan. Tekanan persaingan menyebabkan tugas pengambilan keputusan menjadi sulit. Persaingan di dasarkan tidak hanya pada harga, tetapi juga pada kualitas,

kecepatan, kustomasi produk, dan dukungan pelanggan. Organisasi harus mampu secara sering dan cepat mengubah mode operasi, merekayasa ulang proses dan struktur, memberdayakan karyawan, serta berinovasi. Teknologi pengambilan keputusan bisa menciptakan pemberdayaan yang signifikan dengan cara memperbolehkan seseorang untuk membuat keputusan yang baik secara cepat, bahkan jika mereka memiliki pengetahuan yang kurang.

8. Mengatasi keterbatasan *kognitif* dalam pemrosesan dan penyimpanan. Menurut Simon (1977), otak manusia memiliki kemampuan yang terbatas untuk memproses dan menyimpan informasi. Orang-orang kadang sulit mengingat dan menggunakan sebuah informasi dengan cara yang bebas dari kesalahan. (Dwi Citra Hartini dkk, 2013)

II.4. Metode *Analitycal Hierarchy Process*

Metode AHP atau Proses Hirarki Analitik merupakan salah satu metode pengambilan keputusan dimana faktor-faktor logika, intuisi, pengalaman, pengetahuan, emosi, dan rasa dicoba untuk dioptimasikan dalam suatu proses yang sistematis. Sistem yang dibangun didukung oleh metode AHP. Adapun langkah-langkah dalam perhitungan AHP adalah sebagai berikut :

1. Mendefinisikan masalah dan menentukan solusi yang diinginkan.
2. Membuat struktur hirarki yang diawali dengan tujuan utama.
3. Menjumlahkan setiap kolom (∑kolom) pada matriks perbandingan.
4. Normalisasi matriks, dengan membagi setiap kolom matriks dengan jumlah kolom (∑kolom), kemudian dijumlahkan setiap barisnya (∑baris).

5. Menghitung *Total Priority Value* (TPV) untuk mendapatkan bobot subkriteria.
6. Menghitung uji konsistensi.

   Tahapan dalam melakukan uji konsistensi adalah sebagai berikut :

   a. Mengalikan nilai TPV dengan nilai kolom matriks pada nilai matriks perbandingan kemudian jumlahkan tiap barisnya.
   b. Mencari *Consistency Index* (CI) dengan rumus :

$$C = \frac{\lambda m - n}{n-1} \qquad (2.1)$$

   Dimana :

   CI = *Consistency Index*

   N = Banyaknya elemen yang dibandingkan

   λ max = *Eigen Value Maksimum*

   c. Mencari *Consistency Rasio* (CR) dengan rumus :

$$C = \frac{C}{R} \qquad (2.2)$$

   Dimana :

   CR = *Consistency Ratio*

   CI = *Consistency Index*

   RI = *Random Index*

Kriteria dan alternatif dinilai melalui perbandingan berpasangan. Menurut Thomas L. Saaty, untuk berbagai persoalan, skala 1 sampai 9 adalah skala terbaik dalam mengekspresikan pendapat. Nilai dan definisi pendapat kualitatif dari skala perbandingan Saaty dapat dilihat pada tabel berikut :

Tabel II.1. Skala Perbandingan

Intensitas Kepentingan	Definisi
1	Kedua elemen sama pentingnya
3	Elemen yang satu sedikit lebih penting daripada elemen yang lainnya
4	Elemen yang satu lebih penting daripada yang lainnya
7	Satu elemen jelas lebih mutlak penting daripada elemen lainnya
9	Satu elemen mutlak penting daripada elemen lainnya
2,4,6,8	Nilai-nilai antara dua nilai pertimbangan-pertimbangan yang berdekatan

(Sumber : Astri Herdiyanti dan Utami Dewi Widianti, 2013)

Sebuah nilai rasio dikatakan konsisten jika bernilai $0 \leq$ rasio ≤ 0.1, dengan demikian hasil perhitungan data dapat dibenarkan. Untuk menentukan rasio konsisten atau tidak dapat menggunakan tabel konsistensi rasio sebagai berikut :

Tabel II.2. Konsistensi Rasio

N	*RI*
1	*0.00*
2	*0.00*
3	*0.58*
4	*0.90*
5	*1.12*
6	*1.24*
7	*1.32*
8	*1.41*
9	*1.45*
10	*1.49*
11	*1.51*
12	*1.48*
13	*1.56*
14	*1.57*
>=15	*1.59*

(Sumber : Astri Herdiyanti dan Utami Dewi Widianti, 2013)

II.5. Metode *Simple Additive Weighting*

Metode SAW sering juga dikenal istilah metode penjumlahan terbobot. Konsep dasar metode SAW adalah mencari penjumlahan terbobot dari rating kinerja pada setiap alternatif pada semua atribut. Metode SAW membutuhkan proses normalisasi matriks keputusan (X) ke suatu skala yang dapat diperbandingkan dengan semua ranting alternatif yang ada.

Langkah Penyelesaian SAW sebagai berikut :

1. Menentukan kriteria-kriteria yang akan dijadikan acuan dalam pengambilan keputusan, yaitu Ci.
2. Menentukan rating kecocokan setiap alternatif pada setiap kriteria.
3. Membuat matriks keputusan berdasarkan kriteria (Ci), kemudian melakukan normalisasi matriks berdasarkan persamaan yang disesuaikan dengan jenis atribut (atribut keuntungan ataupun atribut biaya) sehingga diperoleh matriks ternormalisasi R.
4. Hasil akhir diperoleh dari proses perankingan yaitu penjumlahan dari perkalian matriks ternormalisasi R dengan vektor bobot sehingga diperoleh nilai terbesar yang dipilih sebagai alternatif terbaik (Ai) sebagai solusi.

$$r_{ij} = \begin{cases} \dfrac{X_{ij}}{\underset{i}{\text{Max}}\, X_{ij}} \\ \dfrac{\underset{i}{\text{Min}}\, X_{ij}}{X_{ij}} \end{cases}$$

Keterangan :

r_{ij} = nilai rating kinerja ternormalisasi

x_{ij} = nilai atribut yang dimiliki dari setiap kriteria

Max x_{ij} = nilai terbesar dari setiap kriteria

Min x_{ij} = nilai terkecil dari setiap kriteria

Benefit = jika nilai terbesar adalah nilai terbaik

Cost = jika nilai terkecil adalah terbaik

Dimana r_{ij} adalah rating ternormalisasi dari alternatif Ai pada atribut Cj; i=1,2,…,m dan j=1,2,…,n. Nilai prefensi untuk setiap alternatif (Vi) diberikan sebagai :

$$V = \sum_{j-1}^{n} W j \; R ii$$

Keterangan :

Vi = rangking untuk setiap alternatif

W_j = nilai bobot dari setiap kriteria

r_{ij} = nilai rating kinerja ternormalisasi

Nilai Vi yang lebih besar mengidentifikasikan bahwa alternatif Ai lebih terpilih. (Fitriani, 2015)

II.6. *Visual Basic Net* 2010

Visual Basic Net 2010 adalah salah satu bahasa pemrograman yang berorientasi objek. *Microsoft Visual Basic* .NET adalah sebuah alat untuk

mengembangkan dan membangun aplikasi yang bergerak di atas sistem .NET *Framework,* dengan menggunakan bahasa *BASIC.*

Bahasa *Visual Basic* telah digunakan secara luas karena kemudahan penggunaannya bagi orang awam dan penulisan kode di dalamnya tidak terlalu rumit dibandingkan bahasa C, Delphi, dan Java. (M. Rosidi Zamroni, 2014)

II.7. *SQL Server* 2008

SQL Server 2008 adalah sebuah terobosan baru dari *Microsoft* dalam bidang *database*. *SQL Server* adalah DBMS (*Database Management System*) yang dibuat oleh *Microsoft* untuk ikut berkecimpung dalam persaingan dunia pengolahan data menyusul pendahulunya seperti IBM dan Oracle. *SQL Server* 2008 dibuat pada saat kemajuan dalam bidang *hardware* sedemikian pesat. Oleh karena itu sudah dapat dipastikan bahwa *SQL Server* 2008 membawa beberapa terobosan dalam bidang pengolahan dan penyimpanan data. (Wahana Komputer ; 2010 : 2)

II.8. *Database*

Database atau basis data adalah sekumpulan data yang memiliki hubungan secara logika dan diatur dengan susunan tertentu serta disimpan dalam media penyimpanan komputer. Data itu sendiri adalah representasi dari semua fakta yang ada pada dunia nyata. *Database* sering digunakan untuk melakukan proses terhadap data-data tersebut untuk menghasilkan informasi tertentu. Misalnya dari data nama siswa dan tanggal lahir siswa Anda bisa mendapatkan informasi nama

siswa yang berulang tahun pada hari ini. Tentu saja informasi tersebut akan Anda dapatkan dari software pemroses *database* dengan cara Anda memberikan perintah dalam bahasa tertentu yaitu SQL (*Structured Query Language*).(Wahana Komputer ; 2010 : 24)

II.9. *Unified Modelling Language*

Menurut Windu Gata (2013 : 4) Hasil pemodelan pada OOAD terdokumentasikan dalam bentuk *Unified Modelling Language* (UML). 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. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem.

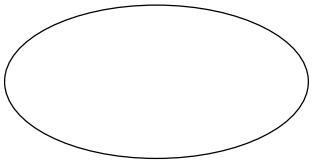
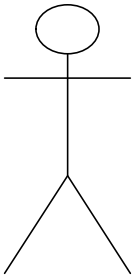
Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut :

1. *Use Case* Diagram

   Use Case Diagram merupakan pemodelan untuk melakukan *(behavior)* sistem informasi yang akan dibuat. *Use Case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan *use case* digunakan untuk mengetahui fungsi apa saja yang ada di

dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case* diagram,

Tabel II.3. Simbol *Use Case*

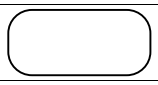
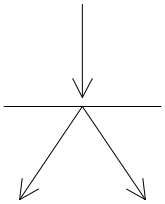
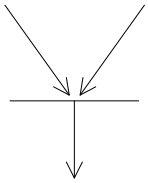
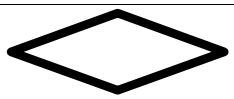
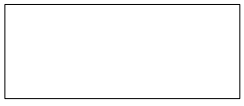
Gambar	Keterangan
	*Use Case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukan pesan antar unit dengan actor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case.*
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan actor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa actor berinterakssi dengan *use case*, tetapi tidak memiliki control terhadap *use case*.
	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengindikasikan bila actor berinteraksi secara pasif dengan sistem.
	Include, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	Extend, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

(Sumber : Windu Gata ; 2013 : 4)

2. Diagram Aktivitas *(Activity Diagram)*

Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram*, yaitu :

Tabel II.4. Simbol *Activity Diagram*

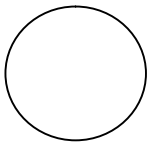
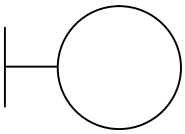
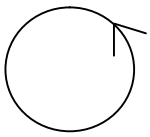
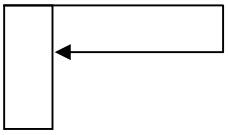
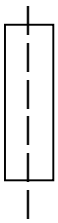
Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activities*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.
	*Join* (Penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true*, *false*.
	*Swimlane*, pembagian *activity* diagram untuk menunjukkan siapa melakukan apa.

(Sumber : Windu Gata ; 2013 : 6)

3. Diagram Urutan (*Sequence Diagram*)

Sequence Diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram*, yaitu :

Tabel II.5. Simbol *Sequence Diagram*

Gambar	Keterangan
	*Entity Class*, merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class*, berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih actor dengan sistem, seperti tampilan formentry dan *form* cetak.
	*Control Class*, suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
	Message, simbol mengirim pesan antar *class*.
	*Recursive*, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Activation*, *Activation* mewakili sebuah ekseskusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

(Sumber : Windu Gata ; 2013 : 7)

4. *Class Diagram* (Diagram Kelas)

   Merupakan hubungan antar kelas dan penjelasaan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggung jawab entitas yang menentukan perilaku sistem. *Class Diagram* juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan

constraint yang berhubungan dengan objek yang dikoneksikan. *Class Diagram* secara khas meliputi : Kelas (*Class*), Relasi, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/Method*), *Visibility*, tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *multiplicity* atau kardinaliti.

Tabel II.6. *Multiplicity Class Diagram*

Multiplicity	Penjelasan
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4

(Sumber : Windu Gata ; 2013 : 9)

II.10. *Entity Relationship Diagram*

Entity Relationship Diagram adalah alat pemodelan data utama dan akan membantu mengorganisasi data dalam suatu proyek ke dalam entitas-entitas dan menentukan hubungan antarentitas. Proses memungkinkan analis menghasilkan struktur basis data yang baik sehingga data dapat disimpan dan diambil secara efesien. (Janner Simarmata dan Iman Paryudi ; 2010 : 67)

II.11. Normalisasi

Normalisasi adalah teknik perancangan yang banyak digunakan sebagai pemandu dalam merancang basis data relasional. Pada dasarnya, normalisasi

adalah proses dua langkah yang meletakkan data dalam bentuk tabulasi dengan menghilangkan kelompok berulang lalu menghilangkan data yang terduplikasi dari tabel rasional.

Teori normalisasi didasarkan pada konsep bentuk normal. Sebuah tabel relasional dikatakan berada pada bentuk normal tertentu jika tabel memenuhi himpunan batasan tertentu. Ada lima bentuk normal yang telah ditemukan.

II.11.1.Bentuk-Bentuk Normalisasi

1. Bentuk Normal Pertama (1NF)

Contoh yang kita gunakan di sini adalah sebuah perusahaan yang mendapatkan barang dari sejumlah pemasok. Masing-masing pemasok berada pada satu kota. Sebuah kota dapat mempunyai lebih dari satu pemasok dan masing-masing kota mempunyai kode status tersendiri. Masing-masing pemasok bisa menyediakan banyak barang. Tabel relasionalnya dapat dituliskan sebagai berikut :

PEMASOK (p#, status, kota, b#, qty) dimana

p# : kode pemasok (kunci utama)

status : kode status kota

kota : nama kota

b# : barang yang dipasok

qty : jumlah barang yang dipasok

Supaya bisa menggabungkan jumlah barang yang di pasok (qty) secara unik dengan barang (b#) dan pemasok (#p), kita menggunakan kunci utama gabungan yang tersusun dari b# dan p#.

Sebuah tabel relasional secara definisi selalu berada dalam bentuk normal pertama. Semua nilai pada kolom-kolom nya adalah atomik. Ini berarti kolom-kolom tidak mempunyai nilai berulang . Gambar II.1 menunjukkan tabel pemasok dalam 1NF.

PEMASOK

P#	Status	Kota	B#	Qty
P1	20	Yogyakarta	B1	300
P1	20	Yogyakarta	B2	200
P1	20	Yogyakarta	B3	400
P1	20	Yogyakarta	B4	200
P1	20	Yogyakarta	B5	100
P1	20	Yogyakarta	B6	100
P2	10	Medan	B1	300
P2	10	Medan	B2	400
P3	10	Medan	B2	200
P4	20	Yogyakarta	B2	200
P4	20	Yogyakarta	B2	300
P4	20	Yogyakarta	B5	400

Gambar II.1. Bentuk Normal Pertama (1NF)

2. Bentuk Normal Kedua (2NF)

Definisi bentuk normal kedua menyatakan bahwa tabel dengan kunci utama gabungan hanya dapat berada pada 1NF, tetapi tidak pada 2NF. Sebuah tabel relasional berada pada bentuk normal kedua jika dia berada pada bentuk normal kedua jika dia berada pada 1NF dan setiap kolom bukan kunci yang sepenuhnya tergantung pada seluruh kolom yang membentuk kunci utama. Ini berarti bahwa setiap kolom bukan kunci harus tergantung pada seluruh kolom yang membentuk kunci utama. Tabel pemasok berada pada INF, tetapi tidak pada

2NF karena status dan kota tergantung secara fungsional hanya pada kolom p# dari kunci gabungan (p#, b#).

p# ⟶ kota, status
kota ⟶ status
(p#, b#) ⟶ qty

Pada contoh, kita memindahkan kolom p#, status, dan kota ke tabel baru yang disebut PEMASOK2. Kolom p# menjadi kunci utama tabel ini. Gambar 2.4 menunjukkan hasilnya.

Pemasok2

P#	Status	Kota
P1	20	Yogyakarta
P2	10	Medan
P3	10	Medan
P4	20	Yogyakarta
P5	30	Bandung

Barang

P#	B#	Qty
P1	B1	300
P1	B2	200
P1	B3	400
P1	B4	200
P1	B5	100
P1	B6	100
P2	B1	300
P2	B2	400
P3	B2	200
P4	B2	200
P4	B2	300
P4	B5	400

Gambar II.2. Bentuk Normal Kedua (2NF)

3. Bentuk Normal Ketiga (3NF)

Bentuk normal ketiga mengharuskan semua kolom pada tabel relasional tergantung hanya pada kunci utama. Secara definisi, sebuah tabel berada pada bentuk normal ketiga (3NF) jika tabel sudah berada pada 2NF dan setiap kolom yang bukan kunci tidak tergantung secara transitif pada kunci utamanya.

Untuk mengubah PEMASOK2 menjadi 3NF, kita membuat tabel baru yang disebut KOTA_STATUS dan memindahkan kolom kota dan status ke tabel

baru. Status dihapus dari tabel asal, kota tetap dibiarkan karena akan berfungsi sebagai kunci asing bagi KOTA_STATUS, dan tabel asal diberi nama baru PEMASOK_KOTA. Gambar II.3 menunjukkan hasilnya.

PEMASOK_KOTA

P#	Kota
P1	Yogyakarta
P2	Medan
P3	Medan
P4	Yogyakarta
P5	Bandung

KOTA_STATUS

Kota	Status
Yogyakarta	20
Medan	10
Bandung	30
Semarang	40
Bandung	30

Gambar II.3. Bentuk Normal Ketiga (3NF)

4. Bentuk Normal Boyce-Code (BCNF)

Setelah 3NF, semua masalah normalisasi hanya melibatkan tabel yang mempunyai tiga kolom atau lebih dan semua kolom adalah kunci. Banyak praktisi berpendapat bahwa menempatkan entitas pada 3NF sudah cukup karena sangat jarang entitas yang berada pada 3NF bukan merupakan 4NF dan 5NF.

5. Bentuk Normal Keempat (4NF)

Sebuah tabel relasional berada pada bentuk normal keempat (4NF) jika dia dalam BCNF dan semua ketergantungan multivalue merupakan ketergantungan fungsional.

Bentuk normal keempat (4NF) didasarkan pada konsep ketergantungan multivalue (MVD). Sebuah ketergantungan multivalue terjadi ketika dalam sebuah tabel relasional yang mengandung setidaknya tiga kolom, satu kolom mempunyai banyak baris bernilai sama, tetapi kolom lain bernilai berbeda.

Definisi secara formal diberikan oleh CJ Date, yaitu : Misalnya, ada sebuah tabel relasional R dengan kolam A, B dan C, maka R.A~~>~~ R.B (kolom A

menentukkan kolom B). Adalah benar jika dan hanya jika himpunan nilai B yang cocok dengan pasangan nilai A dan nilai C pada R hanya tergantung pada nilai A dan tidak tergantung pada nilai C.

MVD selau terjadi dalam pasangan, yaitu R.A $\rightarrow$ R.C dipenuhi pula. Misalnya, pegawai ditugaskan kebanyak proyek dan ia mempunyai banyak keahlian. Jika kita mencatat informasi ini pada satu tabel, ketiga atribut harus digunakan sebagai kunci karena tidak ada satu atribut pun yang dapat secara unik mengidentifikasi sebauh *record*.

Untuk mengubah sebuah tabel dengan ketergantungan *multivalue* ke dalam 4NF, pindahkan masing-masing pasangan MVD ke tabel baru. Gambar II.4 menunjukkan hasilnya.

PEGAWAI_PROYEK

Peg#	#pry
1211	P1
1211	P3

PEGAWAI_AHLI

Peg#	Ahli
1211	Analisis
1211	Perancangan
1211	Pemrograman

Gambar II.4. Bentuk Normal Keempat (4NF)
(Sumber : Janner Simarmata dan Iman Paryudi, 2010 : 86)

6. Bentuk Normal Kelima (5NF)

Sebuah tabel berada pada bentuk normal kelima (5NF) jika ia tidak dapat mempunyai dekomposisi *lossless* menjadi sejumlah tabel lebih kecil. Empat bentuk normal pertama berdasarkan pada konsep ketergantungan fungsional, sedangkan bentuk normal kelima berdasarkan pada konsep ketergantungan gabungan (*join dependence*). Contoh, misalkan kita mempunyai pegawai yang menggunakan keahlian perancangan pada suatu proyek lainnya. (Janner Simarmata dan Iman Paryudi, 2010 : 85 - 86).