BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terkait

Berikut adalah beberapa jurnal terdahulu dengan beberapa judul yang menggunakan metode *Profile Matching* dapat dilihat dibawah ini :

1. Penelitian implementasi metode *Profile Matching* untuk evaluasi potensi akademik penjurusan siswa MAN 2 Kota Kediri. Tujuan penelitian ini adalah menerapkan metode *profile matching* untuk evaluasi potensi akademik penjurusan siswa. Metode *profile matching* digunakan dengan menganalisa kriteria penilaian akademik dan non akademik. Kriteria nilai akademik meliputi nilai rata-rata rapor dan nilai Ujian Nasional sedangkan data nilai non akademik siswa meliputi minat siswa, minat orang tua, tes IQ dan catatan prestasi siswa. Adapun rekomendasi jurusan meliputi jurusan IPA, IPS dan Bahasa. Selanjutnya kriteria penilaian disesuaikan dengan nilai gap kompetensi jurusan berdasarkan kategori *core factor* dan *secondary factor*. Perancangan aplikasi menggunakan bahasa pemrograman *java* dan *database mySQL*. Hasil penelitian berupa aplikasi yang dapat menunjukkan rekomendasi jurusan siswa di Madrasah Aliyah Negeri 2 Kediri, (Intan Nur Farida:2016 ;1). Sedangkan penelitian yang dilakukan oleh penulis dalam pemilihan sekolah pindahan bisa dikategorikan sebuah keputusan multi kriteria, dimana sekolah terfavorit yang terpilih haruslah memenuhi beberapa kriteria. Dalam pemilihan sekolah pindahan biasanya akan terdapat beberapa

kandidat dengan nilai yang berbeda pada setiap kriteria, dimana masing-masing kriteria memiliki prioritas yang berbeda-beda. Hal tersebut seringkali menjadi permasalahan yang cukup kompleks. Saat ini terdapat banyak metode untuk membantu dalam pengambilan keputusan. Salah satu metode yang telah teruji dan banyak digunakan *Profile Matching*. PM bekerja dengan cara membandingkan tingkat kepentingan dari setiap kriteria, dan nilai kandidat pada setiap kriteria, dan nilai kandidat pada kriteria guna menghasilkan keputusan kandidat terbaik.

2. Penelitian sistem pendukung keputusan pemilihan jabatan area supervisor pada PT. Indomarco Prismatama dengan metode *Profile Matching.* Seleksi ini digunakan untuk menilai kemampuan calon Area Supervisor secara objektif bukan subjektif. Penelitian ini mengambil data dari PT Indomarco Prismatama Medan berupa nilai bobot setiap atribut seperti aspek intelektual, aspek sikap kerja dan aspek perilaku kemudian dilakukan proses perangkingan yang akan menentukan alternatif yang optimal yaitu kepala toko yang layak menjadi Area supervisor. Metode yang digunakan dalam penelitian ini adalah *Profile Matching*. Konsep dari metode ini adalah membandingkan kemampuan individu kepala toko terhadap nilai ideal dari perusahaan. Metode ini dipilih karena mampu menyeleksi kepala toko yang layak dan berpotensi menjadi area supervisor dari sejumlah kepala toko yang ada. Hasil rekomendasi ini dapat digunakan untuk membantu dalam hal pengambilan keputusan yang disajikan dalam bentuk peringkat. Penelitian ini menghasilkan sebuah sistem informasi yang dapat membantu Manajer HRD

dalam menentukan kepala toko yang layak menjadi area supervisor, (Kardiaman Lius: 2014 ;1). Sedangkan penelitian yang dilakukan oleh penulis dalam pemilihan sekolah pindahan bisa dikategorikan sebuah keputusan multi kriteria, dimana sekolah terfavorit yang terpilih haruslah memenuhi beberapa kriteria. Dalam pemilihan sekolah pindahan biasanya akan terdapat beberapa kandidat dengan nilai yang berbeda pada setiap kriteria, dimana masing-masing kriteria memiliki prioritas yang berbeda-beda. Hal tersebut seringkali menjadi permasalahan yang cukup kompleks. Saat ini terdapat banyak metode untuk membantu dalam pengambilan keputusan. Salah satu metode yang telah teruji dan banyak digunakan *Profile Matching*. PM bekerja dengan cara membandingkan tingkat kepentingan dari setiap kriteria, dan nilai kandidat pada setiap kriteria, dan nilai kandidat pada kriteria guna menghasilkan keputusan kandidat terbaik.

3. Penelitian perancangan aplikasi peningkatan jenjang karir menggunakan metode *Profile Matching* studi kasus; CV. Indaco Tranding Company dalam proses penentuan jabatan ini dibutuhkan sebuah sistem pendukung keputusan. Salah satu metode yang dapat digunakan untuk sistem pendukung keputusan adalah dengan menggunakan metode *Profile Matching*. Pada penelitian ini akan diangkat suatu kasus yaitu mencari solusi terbaik bedasarkan kompetensi karyawan pada suatu perusahaan menggunakan metode *Profile Matching*. Metode ini dipilih karena mampu menyeleksi kandidat terbaik dari sejumlah karyawan yang ada, dalam hal ini kandidat yang dimaksudkan yaitu karyawan yang berhak menduduki jabatan yang tersedia berdasarkan kriteria-

kriteria yang ditentukan, (Wini Lestari: 2016; 1). Sedangkan penelitian yang dilakukan oleh penulis dalam pemilihan sekolah pindahan bisa dikategorikan sebuah keputusan multi kriteria, dimana sekolah terfavorit yang terpilih haruslah memenuhi beberapa kriteria. Dalam pemilihan sekolah pindahan biasanya akan terdapat beberapa kandidat dengan nilai yang berbeda pada setiap kriteria, dimana masing-masing kriteria memiliki prioritas yang berbeda-beda. Hal tersebut seringkali menjadi permasalahan yang cukup kompleks. Saat ini terdapat banyak metode untuk membantu dalam pengambilan keputusan. Salah satu metode yang telah teruji dan banyak digunakan *Profile Matching*. PM bekerja dengan cara membandingkan tingkat kepentingan dari setiap kriteria, dan nilai kandidat pada setiap kriteria, dan nilai kandidat pada kriteria guna menghasilkan keputusan kandidat terbaik.

II.2. Sistem Pendukung Keputusan (SPK)

Sistem pendukung keputusan (SPK) 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. Tujuan dari SPK adalah untuk membantu pengambil keputusan memilih berbagai alternatif keputusan yang merupakan pengolahan informasi-informasi yang diperoleh atau tersedia dengan menggunakan model pengambilan keputusan. Ciri utama sekaligus keunggulan dari sistem pendukung keputusan tersebut adalah kemampuannya untuk

menyelesaikan masalah-masalah yang tidak erstruktur. Pengambilan keputusan merupakan proses pemilihan alternatif tindakan untuk mencapai tujuan atau sasaran tertentu. Pengambilan keputusan dilakukan dengan pendekatan sistematis terhadap permasalahan melalui proses pengumpulan data menjadi informasi serta ditambah dengan faktor-faktor yang perlu dipertimbangkan dalam pengambilan keputusan. Tahap-tahap yang harus dilalui dalam proses pengambilan keputusan sebagai berikut :

a. Tahap Pemahaman (*Intelligence Phase*) Tahap ini merupakan proses penelusuran dan pendeteksian dari lingkup problematika serta proses pengenalan masalah. Data masukan diperoleh, diproses, dan diuji dalam rangka mengidentifikasikan masalah.
b. Tahap Perancangan (*Design Phase*) Tahap ini merupakan proses pengembangan dan pencarian alternatif tindakan atau solusi yang dapat diambil. Tersebut merupakan representasi kejadian nyata yang disederhanakan, sehingga diperlukan proses validasi dan verifikasi untuk mengetahui keakuratan model dalam meneliti masalah yang ada.
c. Tahap Pemilihan (*Choice Phase*) Pada tahap ini dilakukan pemilihan terhadap berbagai alternatif solusi yang dimunculkan pada tahap perencanaan agar ditentukan atau dengan memperhatikan kriteria-kriteria berdasarkan tujuan yang akan dicapai. Tahap Implementasi (*Implementation Phase*) Pada tahap ini dilakukan penerapan terhadap rancangan sistem yang telah dibuat pada tahap perancangan serta pelaksanaan alternatif tindakan yang telah dipilih pada tahap pemilihan, (Dyna Marisa Khairina; 2016 : 17).

II.3. Metode *Profile Matching* (PM)

Metode *profile matching* merupakan proses membandingkan antara kompetensi individu dengan kompetensi jabatan sehingga dapat diketahui perbedaan kompetensinya (disebut juga *gap*), semakin kecil *gap* yang dihasilkan maka bobot nilainya semakin besar yang berarti memiliki peluang lebih besar untuk karyawan menempati posisi tersebut. Adapun Rumus *Profile Matc*hing adalah :

1. Menghitung dan mengelompokan *core factor* dan *secondary factor*

   Core factor merupakan aspek (kompetensi) yang paling menonjol atau paling dibutuhkan oleh suatu jabatan yang diperkirakan dapat menghasilkan kinerja optimal. Perhitungan *core factor* dapat ditunjukan pada rumus:

$$NCF = \frac{\sum NC\ (i,s,p)}{\sum IC} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

   Keterangan :

   NCF : nilai rata-rata *core factor*

   NC (i,s,p) : jumlah total nilai *core factor* (intelektual, sikap kerja, perilaku)

   IC : jumlah item *core factor*

   Secondary factor adalah item-item selain aspek yang ada pada *core factor* (*fcator* pendukung). Perhitungan *Secondary factor* dapat ditunjukan pada rumus :

$$\text{NSF} = \frac{\sum NS(i,s,p)}{\sum IS} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2)$$

   Keterangan :

NSF : nilai rata-rata *secondary factor*

NS (i,s,p) : jumlah total nilai *secondary factor*

IS : jumlah item *secondary factor*

2. Menghitung nilai total Tiap aspek

   Dari perhitungan dari tiap aspek tersebut kemudian dihitung nilai total berdasarkan presentase dari *core factor* dan *secondary factor* yang diperkirakan berpengaruh terhadap kinerja tiap-tiap profil. Perhitungan dapat dilihat pada rumus :

   Nilai total = 60% NCF + 40% NSF . . . . . (3)

   Keterangan :

   NCF = Nilai rata-rata *core factor*

   NSF = Nilai rata *secondary factor*

3. Menghitung hasil akhir (Rangking)

   Hasil akhir dari proses *profile matching* adalah rangking dari kandidat yang dapat mengisi suatu jabatan tertentu. Penentuan rangking mengacu pada hasil perhitungan tertentu, (Kardiaman Lius Sarumaha; 2014 : 64).

II.4. Sistem

Sistem merupakan kumpulan elemen yang saling berhubungan satu sama lain yang membentuk satu kesatuan dalam usaha mencapai satu tujuan. Di dalam perusahaan, yang dimaksud elemen dari sistem adalah departemen-departemen internal seperti persediaan barang mentah, produksi, persediaan barang jadi, promosi, penjualan, keuangan, personalia, serta pihak eksternal seperti *suppiler*,

dan konsumen yang saling terkait satu sama lain dan membentuk suatu kesatuan usaha, (Kardiaman Lius Sarumaha; 2014 : 64).

II.5. Sistem Informasi

Sistem Informasi adalah kumpulan elemen yang saling berhubungan satu sama lain yang berbentuk satu kesatuan untuk mengintegrasikan data, memproses dan menyimpan serta mendistribusikan informasi. Sistem informasi dapat didefinisikan sebagai suatu sistem yang dibuat oleh manusia yang terdiri dari beberapa komponen dalam organisasi untuk mencapai suatu tujuan yaitu menyajikan informasi. Komponen sistem informasi terdiri dari :

a. *Hardware* (perangkat keras), terdiri dari komputer, printer dan jaringan.
b. *Software*, kumpulan perintah yang ditulis dengan aturan untuk memerintah komputer melaksanakan tugas tertentu.
c. Data, merupakan komponen dasar dari informasi yang akan diproses lebih lanjut untuk menghasilkan informasi.
d. Manusia, yang terlibat dalam komponen manusia seperti operator dan pimpinan.
e. Prosedur, dokumentasi proses sistem, buku penuntun operasional (aplikasi) dan teknis, (Nursahid; 2015: 56).

II.6. Data Dan Informasi

Data merupakan deskripsi tentang benda, kejadian, aktivitas, dan transaksi yang tidak mempunyai makna atau tidak berpengaruh secara langsung kepada makna pemakai. Data juga dapat diartikan suatu bahan mentah yang kelak dapat diolah lebih lanjut untuk menjadi sesuatu yang lebih bermakna. Dan deta inilah yang nantinya akan disimpan dalam d*atabase*. Sedangkan informasi adalah data yang telah diolah menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengambilan keputu n saat ini atau saat mendatang, (Muhammad Taufiq; 2013 : 50).

II.7. *Microsoft Visual Studio* 2010

Microsft Visual Studio merupakan sebuah perangkat lunak lengkap (*suite*) yang dapat digunakan untuk melakukan pengembangan aplikasi, baik itu aplikasi bisnis, aplikasi personal, ataupun komponen aplikasinya, dalam bentuk aplikasi *console*, aplikasi *Windows*, ataupun aplikasi *Web*. *Visual Studio* mencakup kompiler, *Software Development Kit* (SDK), *Integrated Development Environment* (IDE), dan dokumentasi (umumnya berupa MSDN *Library*). Kompiler yang dimasukkan ke dalam paket *Visual Studio* antara lain *Visual C++, Visual C#, Visual Basic, Visual Basic .NET, Visual InterDev, Visual J++, Visual J#, Visual FoxPro*, dan *Visual SourceSafe*. *Microsoft Visual Studio* dapat digunakan untuk mengembangkan aplikasi dalam *native code* (dalam bentuk bahasa mesin yang berjalan di atas *Windows*) ataupun *Managed Code* (dalam bentuk *Microsoft Intermediate Language* di atas *.NET Framework*). Selain itu,

Visual Studio juga dapat digunakan untuk mengembangkan aplikasi *Silverlight*, aplikasi *Windows Mobile* (yang berjalan di atas *.NET Compact Framework*). *Visual Studio* sebelumnya versi *Visual Studio* 9.0.21022.08, atau dikenal dengan sebutan *Microsoft Visual Studio* 2008 yang diluncurkan pada 19 November 2007, yang ditujukan untuk *platform Microsoft .NET Framework* 3.5. Versi sebelumnya, *Visual Studio* 2005 ditujukan untuk *platform .NET Framework* 2.0 dan 3.0. *Visual Studio* 2003 ditujukan untuk *.NET Framework* 1.1, dan *Visual Studio* 2002 ditujukan untuk *.NET Framework* 1.0. Versi-versi tersebut di atas kini dikenal dengan sebutan *Visual Studio .NET*, karena memang membutuhkan *Microsoft .NET Framework*. Sementara itu, sebelum muncul *Visual Studio .NET*, terdapat *Microsoft Visual Studio* 6.0 (VS1998), (Herpendi;2016:1).

II.8. *Database Management System* (DBMS)

Database Management System atau disingkat DBMS adalah perangkat lunak atau (*software)* yang berfungsi untuk mengelola *database*. Mulai dari mengelola *database* sampai dengan proses yang berlaku dalam *database* tersebut, baik berupa *entry*, *edit*, hapus, *query* terhadap data, membuat laporan dan lain sebgainya secara efektif dan efesien, (Nurlaila Hasyim ; 2014: 2 - 3).

II.9. *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. *Microsoft* merilis *SQL Server* 2008 dalam beberapa versi yang disesuaikan dengan segment-segment pasar yang dituju. Versi-versi tersebut adalah sebagai berikut. Menurut cara pemrosesan data pada prosesor maka *Microsoft* mengelompokkan produk ini berdasarkan 2 jenis yaitu :

a. Versi 32-bit(x86), yang biasanya digunakan untuk komputer dengan *single prosesor* (*Pentium* 4) atau lebih tepatnya *prosesor* 32 bit dan sistem operasi *Windows XP*.
b. Versi 64-bit(x64), yang biasanya digunakan untuk komputer dengan lebih dari satu *prosesor* (Misalnya *Core* 2 *Duo*) dan *system* operasi 64 bit seperti *Windows* XP 64, *Vista*, dan *Windows* 7. Sedangkan secara keseluruhan terdapat versiversi seperti berikut ini:
c. Versi *Compact*, ini adalah versi "Tipis" dari semua versi yang ada. Versi ini seperti versi desktop pada *SQL Server* 2000. Versi ini juga digunakan pada *handled drvice* seperti *Pocket* PC, PDA, *SmartPhone, Tablet* PC.
d. Versi *Express*, ini adalah versi "Ringan" dari semua versi yang ada(tetapi versi ini berbeda dengan versi *compact*) dan paling cocok untuk latihan. *Express Mana*ger standar, integrasi dengan CLR dan XML, (Agus Tinus Setiawan;2016:54).

II.10. Normalisasi

Normalisasi *database* merupakan suatu pendekatan sistematis untuk meminimalkan redundansi data pada suatu *database* agar *database* tersebut dapat beekrja dengan optimal. Jika anda seorang *database administrator* ketika terjadi suatu pada database seperti penurunan kinerja, mungkin anda akan ditanya apakah database tersebut telah dinormalisasi. Tujuan normalisasi *database* adalah untuk menghilangkan dan mengurangi redudansi data dan tujuan yang kedua adalah memastikan dependensi data (Data berada pada tabel yang tepat), jika data dalam database tersebut belum dinormalisasi maka akan terjadi 3 kemungkinan yang akan merugikan sistem secara keseluruhan :

1. *Insertion Anomaly* adalah proses melakukan penambahan *recod* baru akan tetapi mempengaruhi *user* untuk terjadinya duplikasi data.
2. *Deletion Anomaly* adalah proses melakukan penghapusan *record* akan tetapi akan menyebabkan hilangnya data yang akan dibutuhkan pada *record* lain.
3. *Modification Anomaly* adalah proses merubah data pada sebuah *record* mempengaruhi perubahan pada *record* lain karena adanya duplikasi.

Normalisasi database terdiri dari banyak bentuk, dalam ilmu basis data setidaknya 9 bentuk normalisasi yang ada yaitu 1NF, 2NF, 3NF, EKNF, BCNF, 4NF, 5NF, DKNF dan 6NF. Namun dalam prakteknya dalam dunia industri bentuk normalisasi ini yang paling sering digunakan ada sekitar 5 bentuk :

1. Normal Form

   Data yang direkam dan dimasukan secara mentah dalam satu tabel pada bentuk ini sangat mungkin terjadi inkonsistensi dan anomali data.

   Contoh Normal Form :

Tabel.II.1. Contoh Normal Form

IDBuku	Judul_Buku	Tgl_Terbit	IDPenerbit	Alamat_Penerbit
801	Blogging.co.id	20-jan-11	P01	Jl 1 kaltim
802	Info Blog	22-jan-11	P02	Jl 2 kaltim
803	Database Design	20-sep-11	P03	Jl 3 kaltim
804	Blog Indonesia	20-mar-11	P04	Jl 4 kaltim

(Sumber :Eka ; 2014 : 2)

Pada bentuk ini ada beberapa ciri-ciri yang penting, yang pertama adalah akan terjadi anomali *insert, update,* dan *delete*. Hal ini menyebabkan beberapa fungsi DML dalam SQL tidak dapat berjalan dengan baik. Sebagai contoh jika ingin menghapus peminjam, maka data penerbit dan buku yang harusnya tidak terhapus akan ikut hilang.

2. First Normal Form (1 NF)

   Bentuk normal yang pertama atau 1NF mensyaratkan beberapa kondisi dalam sebuah database, berikut adalah fungsi dari bentuk normal pertama ini, menghilangkan duplikasi kolom dari tabel yang sama, buat tabel terpisah untuk masing-masing kelompok data terkait dan mengidentifikasi setiap baris dengan kolom yang unik (*Primary Key*).

   Contoh normalisasi database 1NF:

Tabel.II.2. Contoh Normalisasi Database 1NF

IDBuku	Judul_Buku	Tgl_Terbit
801	Blogging.co.id	20-jan-11
802	Info Blog	22-jan-11
803	Database Design	20-sep-11
804	Blog Indonesia	20-mar-11

(Sumber :Eka ; 2014 : 3)

Tabel.II.3. Contoh Normalisasi Database 1NF

IDPenerbit	Alamat_Penerbit
P01	Jl 1 kaltim
P02	Jl 2 kaltim
P03	Jl 3 kaltim
P04	Jl 4 kaltim

(Sumber :Eka ; 2014 : 3)

Pada intinya bentuk normalisasi 1NF ini mengelompokkan beberapa tipe data atau kelompok data yang sejenis agar dapat dipisahkan sehingga anomali data dapat diatasi. Contoh adalah ketika kita ingin menghapus, meng*update*, atau menambahkan data peminjam, maka kita tidak bersinggungan dengan data buku atau data penerbit, sehingga inkonsisten data dapat mulai dijaga.

3. Second Normal Form (2 NF)

   Syarat untuk menerapkan normalisasi bentuk kedua ini adalah data telah dibentuk dalam 1NF, berikut adalah beberapa fungsi normalisasi 2NF, menghapus beberapa subset data yang ada pada tabel dan menempatkan merekaa pada tabel terpisah, menciptakan hubungan antara tabel baru dan tabel lama dengan menciptakan *foreign key*, dan tidak ada atribut dalam

tabel yang secara fungsional bergantug pada *candidate key* tabel tersebut.

Contoh normalisasi database bentuk 2NF:

Tabel.II.4. Contoh Normalisasi Database Bentuk 2NF

IdTrx	Judul_Buku	IdBuku	IdPeminjam	IdPenerbit	Nama_Penerbitt
1111	Blogging.co.id	B01	P01	P01	Pt.aneka
2222	Blogging.co.id	B01	P01	P01	Pt.aneka

(Sumber :Eka ; 2014 : 4)

Tabel.II.5. Contoh 2NF Dari Tabel Diatas

IdTrx	IdBuku	IdPeminjam	IdPenerbit
1111	B01	P01	P01
2222	B01	P01	P01

(Sumber :Eka ; 2014 : 4)

Contoh diatas kita menggunkan tabel bantuan yaitu tabel transaksi, pada intinya bentuk kedua ini adalah tidak boleh ada *field* yang berhubungan dengan *field* lainnya secara fungsional. Contoh judul buku tergantung dengan idbuku sehingga dalam bentuk 2NF judul buku dapat dihilangkan karena telah memiliki tabel master tersendiri.

4. Third Normal Form (3 NF)

   Normalisasi *database* dalam bentuk 3NF bertujuan untuk menghilangkan seluruh atribut atau *field* yang tidak berhubungan dengan *primmary key*. Dengan demikian tidak ada ketergantungann transitif pada setiap kandidat *key*. Syarat dari bentuk normal ketiga atau 3NF adalah, memenuhi semua persyaratan dari bentuk normal kedua, dan menghapus kolom yang tidak tergantung pada *primary key*.

   Contoh normalisasi database bentuk 3NF:

Tidak semua atau tabel dapat kita sesuaikan dengan berbagai bentuk normalisasi ini, untuk contoh 3NF kita akan mengambila contoh dari tabel order.

Tabel.II.6. Contoh Normalisasi Database Bentuk 3NF

orderId	custId	Harga	Jumlah	Total
112	C111	1000	21	21000
113	C112	1000	22	22000

(Sumber :Eka ; 2014 : 5)

Tabel.II.7. Contoh Normalisasi Database Bentuk 3NF

orderId	custId	Harga	Jumlah
112	C111	1000	21
113	C112	1000	22

(Sumber :Eka ; 2014 : 5)

Pada tabel pertama diatas, apakah semua kolom sepenuhnya tergantung pada *primary key*, tentu tidak, hanya saja ada satu *field* yaitu total yang bergantung pada harga dan jumlah, total dapat dihasilkan dengan mengalikan harga dan jumlah. Bentuk 3NF dalam tabel diatas dapat dilakukan dengan membuang field total, (Eka ; 2014 : 5).

II.11. UML (*Unified Modeling Language*)

Unified Modeling 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

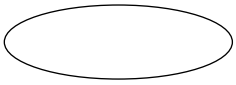
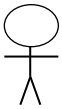
dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem.

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

1. *Use case* Diagram

   Use case diagram merupakan pemodelan untuk kelakukan (*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, yaitu :

Tabel II.8. Simbol *Use Case*

Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukan pesan antar unit dengan aktor, 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 aktor, 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 aktor berinteraksi 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

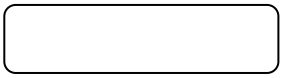
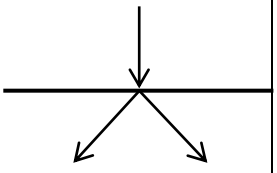
	yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan bila aktor berinteraksi secara pasif dengan sistem.
	Include, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	Extend, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

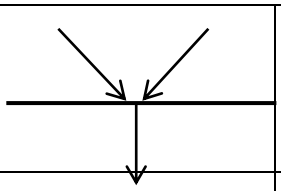
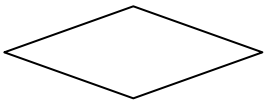
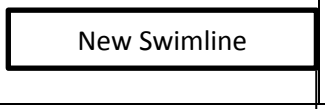
(Sumber : Gellysa, Helmi ; 2015 : 93-94)

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.9. Simbol *Activity Diagram*

Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.

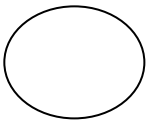
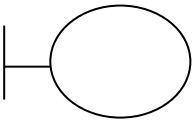
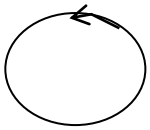
	*Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points,* menggambarkan pilihan untuk pengambilan keputusan, *true, false.*
	*Swimlane*, pembagian *activity* diagram untuk menunjukkan siapa melakukan apa.

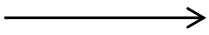
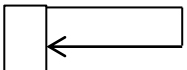
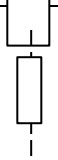
(Sumber : Gellysa, Helmi ; 2015 : 94)

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.10. 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 aktor 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 eksekusi 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 : Gellysa, Helmi ; 2015 : 95)

4. *Class Diagram* (Diagram Kelas)

   Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggng 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.11. ***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 : Gellysa, Helmi ; 2015 : 95)