BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terdahulu

Berdasarkan penelitian yang dilakukan oleh Riangga Duta Jayapana dan Yuniarsi Rahayu(2015) mengenai penerapan algoritma *apriori* untuk menganalisis pola pembelian obat-obatan. Riangga Duta Jayapana dan Yuniarsi Rahayu menyimpulkan bahwa pengolahan *dataset* Apotek Rahayu Jepara menggunakan aplikasi WEKA dapat menghasilkan pola frekuensi tinggi yaitu 2 *itemset* maupun 3 *itemset*, pola frekuensi tinggi 2 *itemset* yang didapat yaitu "Jika membeli obat avarin maka membeli obat dexa". Dan pola transaksi 3 *itemset* yang didapat yaitu "Jika membeli avarin dan troviacol maka membeli obat dexa". (Riangga Duta Jayapana dan YuniarsiRahayu;2015)

Berdasarkan penelitian yang dilakukan oleh Fitri Nurchalifatun (2015) mengenai penerapan metode *asosiasion data mining* dengan algoritma *apriori* untuk mengetahui kombinasi antar *itemset.* Fitri Nurchalifatun menyimpulkan bahwa pola kombinasi yang diperoleh dengan metode asosiasi, yaitu mempunyai pola kombinasi *confidence* tertinggi adalah chokimisu maka tiramisu dengan nilai *confidence* 53,85%, roti umbul maka vanila dengan nilai *confidence* 47,06%. (Fitri Nurchalifatun;2015)

Berdasarkan penelitian yang dilakukan oleh Kennedi Tampubolon (2013) mengenai implementasi *data mining* algoritma *apriori* pada sistem persediaan

alat-alat kesehatan. Kennedi Tampubolon menyimpulkan bahwa *data mining* dapat di implementasi dengan menggunakan *database* penjualan alat-alat kesehatankarena dapat menemukan kecenderungan pola kombinasi itemset untuk menunjang pengambilan keputusan persiapan stok barang yang diperlukan. (Kennedi Tampubolon;2013;105)

II.2. Landasan Teori

II.2.1. *Data mining*

”*Data mining* adalah proses yang mempekerjakan satu atau lebih teknik pembelajaran komputer (*machine learning*) untuk menganalisis dan mengekstraksi pengetahuan (*knowledge*) secara otomatis” (Hermawati, 2009:3).

Definisi lain diantaranya adalah pembelajaran berbasis induksi (*induction-based learning*) adalah proses pembentukan definisi-definisi konsep umum yang dilakukan dengan cara mengobservasi contoh-contoh spesifik dari konsep-konsep yang akan dipelajari. *Knowledge Discovery in Databases* (KDD) adalah penerapan metode saintifik pada *data mining*. Dalam konteks ini, *data mining* merupakan satu langkah dari proses KDD. (Hermawati, 2009:3)

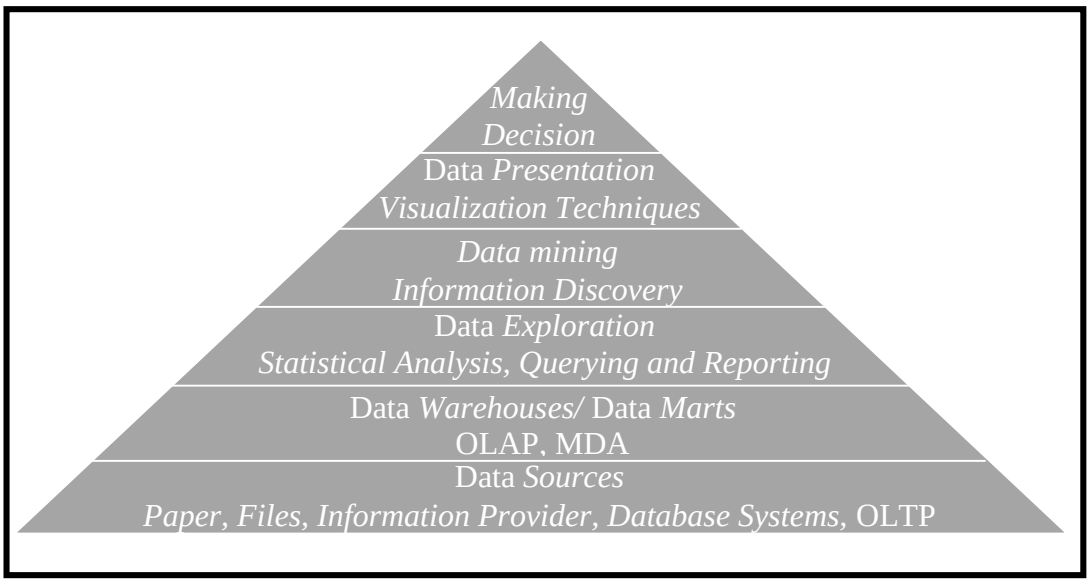
Data mining merupakan proses iteratif dan interaktif untuk menemukan pola atau model baru yang sahih (sempurna), bermanfaat dan dapat dimengerti dalam suatu *database* yang sangat besar (*massive databases*).

- Sahih, yaitu dapat digeneralisasi untuk masa yang akan datang.
- Baru, yaitu apa yang sedang tidak diketahui.
- Bermanfaat, yaitu dapat digunakan untuk melakukan suatu tindakan.

- Iteratif, memerlukan sejumlah proses yang diulang.
- Interaktif, memerlukan interaksi manusia dalam prosesnya.

(Hermawati, 2009:3)

Data mining berisi pencarian *trend* atau pola yang diinginkan dalam *database* besar untuk membantu pengambilan keputusan di waktu yang akan datang. Pola-pola ini dikenali oleh perangkat tertentu yang dapat memberikan suatu analisa data yang berguna dan berwawasan yang kemudian dapat dipelajari dengan lebih teliti, yang mungkin saja menggunakan perangkat pendukung keputusan yang lainnya.(Hermawati, 2009:3)



Gambar II.1 *Data mining* dan Teknologi *Database* Lainnya
(Sumber : Hermawati ; 2009:4)

Dari gambar di atas terlihat bahwa teknologi data *warehouse* digunakan untuk melakukan OLAP, sedangkan *data mining* digunakan untuk melakukan *information discovery* yang informasinya lebih ditujukan untuk seorang *Data Analyst* dan *Business Analyst* (dengan ditambah visualisasi tentunya). Dalam prakteknya, *data mining* juga mengambil data dari data *warehouse*. Hanya saja

aplikasi dari *data mining* lebih khusus dan lebih spesifik dibandingkan OLAP mengingat *database* bukan satu-satunya bidang ilmu yang mempengaruhi *data mining*, banyak lagi bidang ilmu yang turut memperkaya *data mining* seperti : *information science* (ilmu informasi), *high performance computing*, visualisasi, *machine learning*, statistik, *neural network (*jaringan syaraf tiruan), pemodelan matematika, *information retrieval* dan *information extraction* serta pengenalan pola. Bahkan pengolahan citra (*image processing*) juga digunakan dalam rangka melakukan *data mining* terhadap data *image/ spatial*.(Hermawati, 2009:3)

Alasan mengapa melakukan *data mining* dari sudut pandang komersial karena :

1. Meledaknya volume data yang dihimpun dan disimpan dalam data *warehouse* seperti data *web, e-commerce*, penjualan di *departement store*, transaksi bank/ *credit card*.
2. Proses komputasi yang dapat diupayakan.
3. Kuatnya tekanan kompetitif untuk dapat menyediakan yang lebih baik, layanan-layanan *custom*-isasi dan informasi sedang menjadi produk yang berarti.

(Hermawati, 2009:3)

II.2.2. Operasi *Data mining*

Operasi *data mining* menurut sifatnya dibedakan menjadi dua yaitu :

1. Prediksi (*prediction driven*), untuk menjawab pertanyaan apa dan sesuatu yang bersifat transparan. Operasi prediksi digunakan untuk validasi

hipotesis, *querying* dan pelaporan, analisis multidimensi, OLAP (*Online Analytic Processing*) serta analisis statistik.

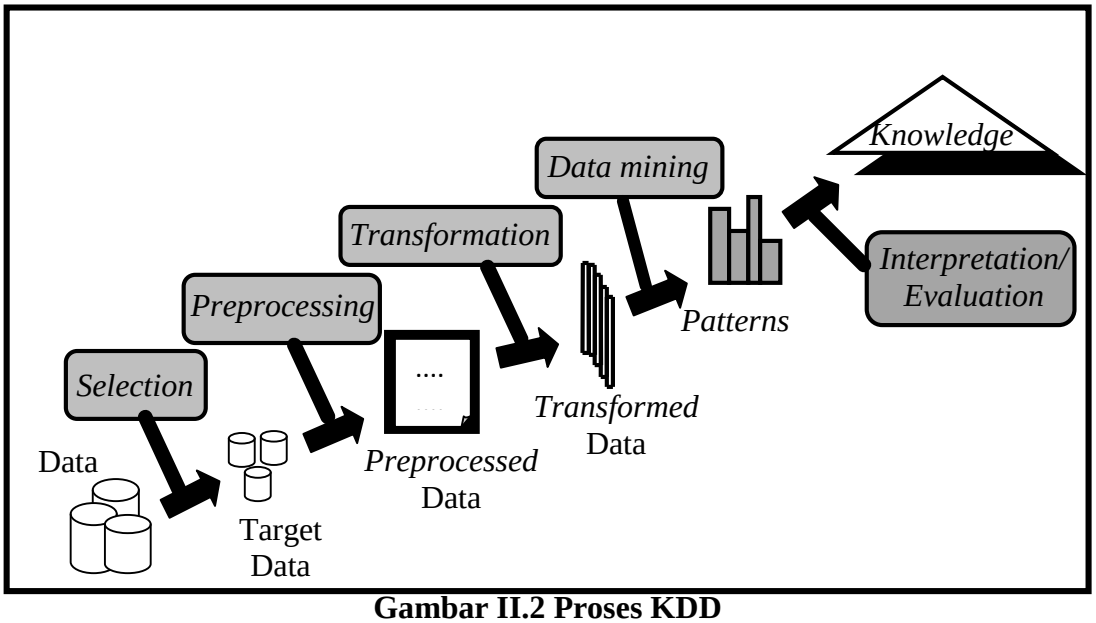
2. Penemuan (*discovery driven*) bersifat transparan dan untuk menjawab pertanyaan “mengapa?”. Operasi penemuan digunakan untuk analisis data eksplorasi, pemodelan prediktif, segmentasi *database*, analisis keterkaitan (*link analysis*) dan deteksi deviasi.

(Hermawati, 2009:5)

Tahapan proses dalam penggunaan *data mining* yang merupakan proses *Knowledge Discovery in Databases* (KDD) seperti yang terlihat pada gambar 2.2 dapat diuraikan sebagai berikut :

- Memahami *domain* aplikasi untuk mengetahui dan menggali pengetahuan awal serta apa sasaran pengguna.
- Membuat target data-set yang meliputi pemilihan data dan fokus pada sub-set data.
- Pembersihan dan transformasi data meliputi eliminasi derau, *outliers*, *missing value* serta pemilihan fitur dan reduksi dimensi.
- Penggunaan algoritma *data mining* yang terdiri dari asosiasi, sekuensial, klasifikasi, klasterisasi, dll.
- Interpretasi, evaluasi dan visualisasi pola untuk melihat apakah ada sesuatu yang baru dan menarik dan dilakukan iterasi jika diperlukan.

(Hermawati, 2009:5)



Gambar II.2 Proses KDD
(Sumber : Hermawati ; 2009:5)

II.2.3. Tantangan Dalam *Data mining*

Tantangan dalam *data mining* meliputi :

1. *Scalability*, yaitu besarnya ukuran basis data yang digunakan.
2. *Dimensionality*, yaitu banyaknya jumlah atribut dalam data yang akan diproses.
3. *Complex and Heterogeneous Data*, yaitu data yang kompleks dan mempunyai variasi yang beragam.
4. Data *Quality*, kualitas data yang akan diproses seperti data yang bersih dari *noise*, *missing value*, dsb.
5. *Data Ownership and Distribution*, yaitu siapa yang memiliki data dan bagaimana distribusinya.
6. *Privacy Preservation*, yaitu menjaga kerahasiaan data yang banyak diterapkan pada data nasabah perbankan.
7. *Streaming Data*, yaitu aliran data itu sendiri.(Hermawati ; 2009:19)

II.2.4. Teknik *Data mining*

Beberapa teknik dan sifat *data mining* adalah sebagai berikut :

1. *Classification*
2. *Clustering*
3. *Association Rule Discovery*
4. *Sequential Pattern Discovery*
5. *Regression*(Hermawati ; 2009:14)

II.2.5. *Association Rules*

Mendeteksi kumpulan atribut-atribut yang muncul bersamaan (*co-occur*) dalam frekuensi yang sering dan membentuk sejumlah kaidah dari kumpulan-kumpulan tersebut. Contoh : 90% orang yang berbelanja di suatu supermarket yang membeli roti juga membeli selai dan 60% dari semua orang yang berbelanja membeli keduanya (Hermawati, 2009:17).

Jika diberikan sekumpulan *record* yang masing-masing terdiri dari sejumlah *item* dari kumpulan yang diberikan, akan menghasilkan aturan ketergantungan (*dependency rules*) yang akan memprediksi kejadian dari satu item berdasarkan kejadian item lainnya.

Contoh aplikasi kaidah asosiasi adalah sebagai berikut :

1. *Marketing and Sales Promotion*

   Misalkan diketahui aturan ketergantungan dimana :

   {*Bagels, ...* } → {*Potato Chips*}

Potato Chips sebagai *consequent*, dapat digunakan untuk menentuka apa yang dapat dilakukan untuk meningkatkan penjualan.

Bagels in the antecedent, dapat digunakan untuk melihat produk mana yang akan terkena dampak jika toko tersebut tidak lagi menjual *bagels*.(Hermawati ; 2009:17)

Bagels in antecedent and Potato Chips in consequent, dapat digunakan untuk melihat produk apa yang harus dijual dengan *bagels* untuk mempromosikan penjualan *potato chips*.

2. *Supermarket Shelf Management*

   Tujuan : Untuk mengenali item-item yang dibeli bersama-sama oleh cukup banyak pelanggan.

   Aturan klasik : Jika seorang pelanggan membeli *diaper* dan susu maka dia juga akan membeli *beer*. Sehingga jangan kaget jika menemukan enam *pack beer* yang ditumpuk dekat *diapers*.

3. *Inventory Management*

   Tujuan : Seorang pelanggan perusahaan perbaikan peralatan mengharapkan keaslian dari perbaikan produk konsumen dan menjaga pelayanan dengan menggunakan suku cadang yang baik untuk mengurangi jumlah kunjungan ke rumah pelanggan. (Hermawati ; 2009:18)

II.2.6. Metodologi Dasar Analisis Asosiasi

Analisis Asosiasi atau *Association rule mining* adalah teknik *data mining* untuk menemukan aturan asosiasi antara kembinasi item (Heroe Santoso, dkk, 2016: 21).Metodologi dasar analisis asosiasi terbagi menjadi dua tahap, yaitu :

1. Analisis Pola Frekuensi Tinggi

Tahapan ini mencari kombinasi *item* yang memenuhi syarat minimum dari nilai *support* dalam *database*. Nilai *support* sebuah *item* diperoleh dengan menggunakan rumus berikut :

$$Support\ A = \frac{Jumlah\ transaksi\ mengandung\ A}{Total\ transaksi}\ x\ 100\% \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

Pada rumus 1 menjelaskan bahwa nilai *support* diperoleh dengnn cara mencari jumlah transaksi yang mengandung *item A* dengan jumlah seluruh transaksi. Sedangkan nilai *support* dari 2 item diperoleh dari rumus 2 berikut :

$$Support\ (A,B) = \frac{\sum Transaksi\ Mengandung\ A\ dan\ B}{\Sigma\ Transaksi} x\ 100\% \ldots\ldots\ldots\ldots\ldots\ldots\ldots (2)$$

Pada rumus 2 menjelaskan bahwa nilai *support* diperoleh dengan cara mencari jumlah transaksi yang mengandung *item A* dan *item B* (*item* pertama bersama dengan *item* yang lain) dibagi dengan keseluruhan transaksi.

2. Pembentukan Aturan Asosiasi

Setelah semua pola frekuensi tinggi ditemukan, barulah dicari aturan asosiasi yang memenuhi syarat minimum untuk *confidence* dengan

menghitung *confidence* aturan asosiatif "jika A maka B".Nilai *confidence* dariaturan"jika A maka B"diperoleh dengan rumus berikut *:*

$$Confidence\ P(B|A = \frac{\sum Total\ transaksi\ mengandung\ A\ dan\ B}{\sum Transaksi\ mengandung\ A}\ x\ 100\% \ldots\ldots\ldots\ldots (3)$$

Untuk menentukan aturan asosiasi yang akan dipilih maka harus diurutkan berdasarkan *Support* × *Confidence*. Aturan diambil sebanyak *n* aturan yang memiliki hasil terbesar (Heroe Santoso, dkk, 2016: 21).

II.2.7. Algoritma *Apriori*

Algoritma *apriori* termasuk jenis aturan asosiasi pada data *mining*. Aturan yang menyatakan asosiasi antara beberapa atribut sering disebut *affinity analysis* atau *market basket analysis*. Analisis asosiasi atau *association rule mining* adalah teknik *data mining* untuk menemukan aturan suatu kombinasi *item.* Contoh aturan asosiatif dari analisis pembelian di suatu pasar swalayan adalah dapat diketahuinya berapa besar kemungkinan seorang pelanggan membeli roti bersamaan dengan susu. (Kusrini, Emha Taufiq Luthfi 2009: 149).

Penting tidaknya suatu asosiasi dapat diketahui dengan dua tolak ukur, yaitu *support* dan *confidence*. *Support* (nilai penunjang) adalah persentase kombinasi *item* tersebut dalam *database*, sedangkan *confidence* (nilai kepastian) adalah kuatnya hubungan antar *item* dalam aturan asosiasi. (Heroe Santoso, dkk, 2016: 20).

Proses utama yang dilakukan dalam algoritma *apriori* untuk mendapat *frequent itemset* yaitu :

1. *Join* (penggabungan)
   Proses ini dilakukan dengan cara pengkombinasian *item* dengan yang *item* lainya hingga tidak bisa terbentuk kombinasi lagi.

2. *Prune* (pemangkasan)
   Proses pemangkasan yaitu hasil dari *item* yang telah dikombinasikan kemudian dipangkas dengan menggunakan minimum *support* yang telah ditentukan .

(Heroe Santoso, dkk, 2016: 20)

II.2.8. Tentang Persediaan Barang

Persediaan dalam perusahaan pengertian atau prosesnya tergantung dari jenis perusahaan tersebut. Jika perusahaan termasuk dalam kelompok perusahaan manufaktur berarti persediaan yang akan dikelola meliputi persediaan produk jadi, persediaan produk dalam proses, persediaan bahan baku, persediaan bahan penolong dan lainnya. Sedangkan jika perusahaan termasuk dalam kelompok perusahaan dagang, maka persediaan yang dikelola hanya satu macam saja yaitu persediaan barang dagangan yang merupakan barang yang dibeli dan kemudian dijual kembali. Dari pengertian tersebut, maka dapat disimpulkan bahwa pengelolaan persediaan tergantung dari jenis perusahaan. Lembaga pendidikan, termasuk universitas, merupakan organisasi / perusahaan yang tidak menggunakan persediaan untuk dijual kembali ataupun diolah dan kemudian dijual kembali.

Sehingga pengelolaan persediaan yang dimiliki dapat dikatakan hanya sebatas membeli dan kemudian digunakan untuk kegiatan sehari-hari. Maka dapat diambil kesimpulan sementara bahwa pengelolaan persediaan / pencatatan persediaan dilakukan saat pembelian dan pengeluaran barang saja(Siti Munawaroh;2006;125).

II.2.9. Nugget

Nugget adalah suatu bentuk produk olahan daging yang terbuat dari daging giling yang dicetak dalam bentuk potongan empat persegi dan dilapisi tepung berbumbu (battered dan braded) (Maghfiroh, 2000). Nugget dikonsumsi setelah proses penggorengan rendam (deep fat frying) (Saleh et.al, 2002). Nugget dibuat dari daging giling yang diberi bumbu, dicampurbahan pengikat, kemudian dicetak membentuk tertentu, dikukus, dipotong dan dilumuri perekat tepung (batter) dan diselimuti tepung roti (breading).

Nugget digoreng setengah matang dan dibekukan untuk mempertahankanmutunya selama penyimpanan (Astawan, 2007). Nugget merupakan salahsatu bentuk produk makanan beku siap saji, yaitu produk yang telahmengalami pemanasan sampai setengah matang (precooked), kemudiandibekukan (Afrisanti, 2010). Produk beku siap saji ini hanya memerlukanwaktu penggorengan selama 1 menit pada suhu 150º C. Tekstur nuggettergantung dari bahan asalnya (Astawan, 2007).Standarisasi kualitas untuk bahan pangan untuk nugget meliputi sifatkimia dan organoleptik. Persyaratan untuk menguji kualitas bahan pangan menurut Badan Standarisasi Nasional

(2002) menggunakan uji kualitas kimia meliputi kadar lemak, air, abu, protein dan karbohidrat. Uji kualitas organoleptik meliputi aroma, rasa, dan tekstur. Badan Standarisasi Nasional(BSN) (2002) pada SNI.01-6638-2002 mendefinisikan nugget ayam sebagai produk olahan ayam yang dicetak, dimasak, dibuat dari campuran daging ayam giling yang diberi bahan pelapis dengan atau tanpa penambahan bahan makanan lain dan bahan tambahan makanan yang diizinkan. Sebagai pedoman standar karakteristik nugget keong, mengacu pada SNI.01–6638–2002 (BSN, 2002) yang membahas tentang standar kualitas nugget ayam.
Sumber: *(http://digilib.unimus.ac.id/files/disk1/129/jtptunimus-gdl-trilistian-6432-3-babii.pdf)*

II.3. *Microsoft SQL Server* 2008

II.3.1. *SQL Server* 2008 R2

SQL Server merupakan suatu *Relational Database Management Systems* (RDBMS) yang digunakan untuk menyimpan data. Data yang disimpan pada *database* bisa dalam skala kecil maupun besar. Selain itu, penyajiannya merupakan penyajian pada level fisik karena kita akan menyimpan langsung data pada *database* dengan kondisi yang sebenarnya, yaitu disimpan pada tabel apa, kolom mana, dan menggunakan tipe data saat penyimpanan (Benardo, dkk, 2015: 94).

Pada *SQL Server* 2008 terdapat fitur-fitur yang dapat mengembangkan performa dari *database* tersebut. Beberapa fitur tersebut, yaitu :

1. *Date Data Type*

Digunakan untuk menyimpan data tanggal saja sehingga akan menghemat *space* pada *server*.

2. *Data Compression*

Digunakan untuk melakukan *compress* data sehingga ukuran data yang disimpan dalam hal *space hardisk* akan lebih kecil.

3. *Sparse Column*

Digunakan untuk menyimpan data yang memiliki lebih banyak data *NULL* dengan lebih efisien.

4. *Row Constructor*

Digunakan untuk melakukan *insert* beberapa data sekaligus dengan satu perintah *INSERT*.

5. *Table Valued Parameter*

Digunakan untuk melakukan *parsing array* pada bahasa pemrograman, dimana satu variable diberikan data-data yang akan diproses setelahnya (Benardo, dkk, 2015: 94).

II.3.2 Konsep *Database*

Pangkalan data atau basis data (bahasa Inggris : *database*), atau sering pula dieja basis data, adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. Perangkat lunak yang digunakan untuk mengelola dan memanggil kueri (*query*)

basis data disebut sistem manajemen basis data (*database management system*, DBMS). Sistem basis data dipelajari dalam ilmu informasi(Benardo, dkk, 2015).

Selain itu juga dalam mengambil data dari *server* lain akan mengalami penurunan performa. Tetap dengan menggunakan terdistribusi, bisa dengan cepat melakukan akses untuk data pada *database server* yang didistribusikan. Sedangkan untuk tersentralisasi, karena *database*-nya hanya satu dan terpusat (misalnya di *head office*) maka seluruh *client* dari manapun akan mengambil data tersebut dari satu *database*. Dengan demikian data yang diambil tidak akan bermasalah dalam hal konsistensi karena berada dalam satu sumber, tetapi akan membutuhkan *hardware* yang jauh lebih besar dan *bandwidth* yang lebih tinggi. Hal ini dikarenakan *server* tersebut berfungsi untuk menampung penggunaan connection yang sangat banyak.

Database atau basis data adalah kumpulan data yang disimpan secara sistematis di dalam komputer dan dapat diolah atau dimanipulasi menggunakan perangkat lunak (program aplikasi) untuk menghasilkan informasi. Pendefinisian basis data meliputi spesifikasi berupa tipe data, struktur, dan juga batasan-batasan data yang akan disimpan. Basis data merupakan aspek yang sangat penting dalam sistem informasi dimana basis data merupakan gudang penyimpanan data yang akan diolah lebih lanjut. Basis data menjadi penting karena dapat menghidari duplikasi data, hubungan antar data yang tidak jelas, organisasi data, dan juga update yang rumit. Untuk penyimpanan *database*, biasanya digunakan *relational database*, yaitu suatu mekanisme penyimpanan data pada suatu tabel tertentu yang terhubung antara tabel yang satu dengan tabel lainnya dengan menggunakan

references data. Data tersebut berupa *field* atau kolom pada tabel yang menghubungkan tabel yang satu dengan tabel yang lain (Benardo, dkk, 2015:).

II.4. *Microsoft Visual Studio* 2010

Visual Basic.Net 2010 merupakann *core* dari pembuatan aplikasi berbasis *.Net*. yang merupakan lingkungan pemrograman yang mempermudah tahapan *desain, development, debugging*, dan *deployment* dari aplikasi berbasis *.Net* dan *XMLweb service*, serta meningkatkan efisiensi *developer* dengan menyediakan lingkungan pemrograman yang sudah biasa digunakan.

.NET Framework adalah teknologi inti yang menyediakan berbagai library untuk digunakan oleh aplikasi di atasnya. Komponen inti *.NET Framework* adalah *Common Language Runtime* (CLR) yang menyediakan *run time environment* untuk aplikasi yang dibangun menggunakan *Visual Studio.NET*, terlepas dari jenis bahasa pemrogramannya (Benardo, dkk, 2015: Hal. 93).

II.5. Pemodelan Sistem

II.5.1. *Unified Modelling Language (UML)*

Pada perkembangan teknologi perangkat lunak, diperlukan adanya bahasa yang digunakan untuk memodelkan perangkat lunak yang akan dibuat dan perlu adanya standarisasi agar orang diberbagai negara dapat mengerti pemodelan perangkat lunak (Shalahuddin, M. dan Rosa A.S, 2014:137). Seperti yang kita ketahui bahwa menyatukan banyak kepala untuk menceritakan sebuah ide dengan tujuan untuk memahami hal yang sama tidaklah mudah, oleh karena itu

diperlukan sebuah bahasa pemodelan perangkat lunak yang dapat dimengerti oleh banyak orang.

Banyak orang yang telah membuat bahasa pemodelan pembangunan perangkat lunak sesuai dengan teknologi pemrograman yang berkembang pada saat itu, misalnya yang sempat berkembang dan digunakan oleh banyak pihak adalah *Data Flow Diagram* (DFD) untuk memodelkan perangkat lunak yang menggunakan pemrograman prosedural atau struktural, kemudian juga ada *State Transition Diagram* (STD) yang digunakan untuk memodelkan sistem *real time* (waktu nyata).

Pada perkembangan teknik pemrograman berorientasi objek, muncullah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek, ysaitu *Unified Modelling Language* (UML). UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasi, menggambarkan, membangun dan dokumentasi dari sistem perangkat lunak.

“UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung” (Shalahuddin, M. dan Rosa A.S, 2014:137).

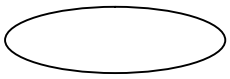
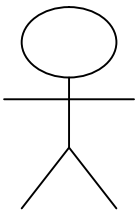
UML hanya berfungsi untuk melakukan pemodelan. Jadi penggunaan UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataannya UML paling banyak digunakan pada metodologi berorientasi objek. Seperti yang kita ketahui bahwa banyak hal di dunia sistem informasi yang tidak dapat dibakukan, semua tergantung kebutuhan, lingkungan dan konteksnya. Begitu juga dengan

perkembangan penggunaan UML bergantung pada *level* abstraksi penggunaannya. Jadi belum tentu pandangan yang berbeda dalam penggunaan UML adalah suatu yang salah, tapi perlu ditelaah dimanakah UML digunakan dan hal apa yang ingin divisualkan(Shalahuddin, M. dan Rosa A.S, 2014:137).

II.5.2. *Use Case Diagram*

Use case diagram merupakan pemodelan untuk tingkah laku sisteminformasiyang dibuat. Use case diagram mendeskripsikan sebuah interaksi antara satu ataulebih actor dengan sistem informasi yang dibuat (Shalahuddin, M. dan Rosa A.S, 2014:156). Secara kasar, Use casedigunakan untuk mengetahuai fungsi apa saja yang ada didalam sebuah sisteminformasi dan siapa saja yang berhak menggunakan fungsi itu. Syarat penamaanuse case nama didefinisikan sesederhana mungkin dan dapat dipahami (Rosa danSalahudin, 2011). Berikut adalah simbol-simbol yang ada pada diagram *use case* :

Tabel II.1 Simbol-Simbol *Use CaseDiagram*

Simbol	Nama	Keterangan
	*Use Case*	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case.*
	*Actor*	Orang, proses, atau sistem lain yang berorientasi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya

		dinyatakan menggunakan kata benda di awal frase nama aktor.
———————	*Association*	Komunikasi antar aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
<<extend>> - - - - - - - - - - - - ▶	*Extend*	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu; mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek; biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan.
——————▶	*Generalization*	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.
<<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.

(Sumber : Shalahuddin, M. dan Rosa A.S, 2014:156, Rekayasa Perangkat Lunak)

II.5.3. *Activity Diagram*

Diagram aktivitas atau *activity diagram* menggambarkan *work flow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram

aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan *actor*, jadi aktivitas yang dapat dilakukan oleh sistem.

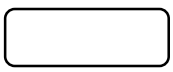
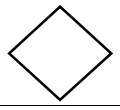
Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut :

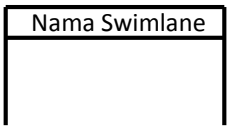
1. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokan tampilan dari sistem/ *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.
3. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.
4. Rancangan menu yang ditampilkan pada perangkat lunak.

(Shalahuddin, M. dan Rosa A.S, 2014:162)

Berikut adalah simbol-simbol yang ada pada diagram aktivitas :

Tabel II.2 Simbol *ActivityDiagram*

Simbol	Nama	Keterangan
	Status Awal	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
	Aktivitas	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
	Percabangan/ *Decision*	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.

	Penggabungan/ *Join*	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
	Status Akhir	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
	Swimlane	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

(Sumber : Shalahuddin, M. dan Rosa A.S, 2014:162, Rekayasa Perangkat Lunak)

II.5.4. *Class Diagram*

Diagram kelas atau *class diagram* menggambarkan struktur system dari segi pendefenisian kelas-kelas yang akan dibuat untuk membangun sistem(Shalahuddin, M. dan Rosa A.S, 2014:137).Kelas memiliki apa yang disebut atribut dan metode atau operasi

a. Atribut merupakan variable-variabel yang dimiliki oleh suatu kelas
b. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

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(Shalahuddin, M. dan Rosa A.S, 2014:142).

Banyak berbagai kasus, perancangan kelas yang dibuat tidak sesuaidengan kelas-kelas yang dibuat pada perangkat lunak, sehingga tidaklah ada gunanya lagi sebuah perancangan karena apa yang dirancang dan hasil jadinya tidak sesuai.

Kelas-kelas yang ada pada struktur system harus dapat melakukan fungsi-fungsi-fungsi sesuai dengan kebutuhan system sehingga pembuat perangkat lunak

dapat membuat kelas-kelas didalam program perangkat lunak sesuai dengan perancangan diagram kelas. Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut :

a. Kelas main : kelas yang memiliki fungsi awal dieksekusi ketika system dijalankan.
b. Kelas yang menangani tampilan system (*view*) : kelas yang mendefenisikan dan mengatur tampilan ke pemakai.
c. Kelas yang diambil dari pendefenisian *use case* (*controller*) : kelas yang menangani fungsi-fungsi yang harus ada diambil dari pendefenisian *use case*, ,kelas ini biasanya disebut dengan kelas proses yang menangani proses bisnis pada perangkat lunak.
d. Kelas yang diambil dari pendefenisian data (*model)* : kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data. Semua table yang dibuat di basis data dapat dijadikan kelas, namun untuk table dari hasil ralasi atau atribut mutivalue pada ERD dapat dijadikan kelas tersendiri dapat juga tidak asalkan pengaksesannya dapat dipertanggungjawabkan atau tetap ada didalam perancangan kelas.

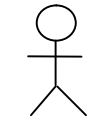
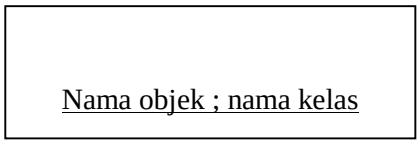
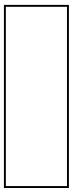
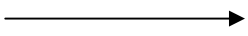
(Shalahuddin, M. dan Rosa A.S, 2014:142).

II.5.5. *Sequence Diagram*

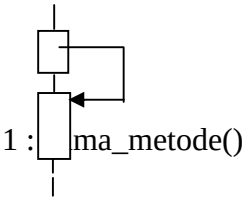
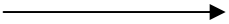
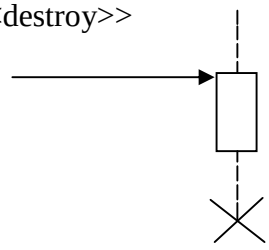
Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirmkan dan diterima antar objek(Shalahuddin, M. dan Rosa A.S, 2014:165).Oleh karena itu untuk menggambar diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimilik kelas yang diinstansiasi menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat scenario yang ada pada *use case*.

Banyaknya diagram sekuen yang harus digambar adalah minimal sebanyak pendefenisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefenisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak *use case* yang didefenisikan maka diagram sekuen yang harus dibuat juga semakin banyak. Berikut adalah simbol-simbol yang ada pada diagram sekuen:

Tabel II.3. Simbol *Sequence Diagram*

Simbol/ Arti	Deskripsi
 Nama aktor Atau Nama aktor Tanpa waktu aktif	Orang, proses, atau sistem lain yanng berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama actor
Garis hidup / *lifeline*	Menyatakan kehidupan suatu objek
Objek  Nama objek ; nama kelas	Menyatakan objek yang berinteraksi pesan
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan didalamnya.
Pesan tipe create <<create>> 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat.

(Sumber : Shalahuddin, M. dan Rosa A.S, 2014:165-167, Rekayasa Perangkat Lunak)

Pesan tipe call 1 : nama_metode() 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri,  Arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi/metode maka operasi/metode yanng dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi.
Pesan tipe send 1 : masukan 	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi.
Pesan tipe return 1 : keluaran 	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
Pesan tipe destroy <<destroy>> 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhri, sebaiknya jika ada create maka ada destroy.

(Sumber : Shalahuddin, M. dan Rosa A.S, 2014:165-167, Rekayasa Perangkat Lunak)