BAB 2
TINJAUAN PUSTAKA

2.1. Tinjauan Tempat Penelitian

Pada bagian tinjauan tempat penelitian ini akan dijelaskan beberapa hal mengenai tempat penelitian. Tinjauan penelitian meliputi profil perusahaan, logo perusahaan, visi dan misi, serta struktur organisasi perusahaan.

2.1.1. Profil Perusahaan

Blackjack Industries adalah sebuah Perusahaan yang berjalan di bidang produksi dan penjualan pakaian jadi seperti T-Shirt, Jaket, Sweaters, Celana, Kemeja, Rompi, Dompet, Sabuk, Topi, Sandal, Tas, dll, dengan system penjualan retail, konsinyasi dan beli putus. berdiri sejak tahun 1999 di bandung dengan awal produk berupa pakaian jadi hingga sekarang berkembang tidak hanya pakaian jadi, kami juga memproduksi merchandise, percetakan dan berbagai macam aksesoris. kualitas produk sangat kami utamakan di dukung dengan sumber daya yang berkualitas dan berpengalaman, Blackjack Industries beralamat di Jl. Trunojoyo. 28 Bandung . Saat ini Blackjack Industries telah memiliki 20 outlet resmi yang berada di beberapa kota-kota besar di Indonesia.

Blackjack Industries tidak hanya mendapatkan tanggapan positif dari para penggemarnya, tetapi banyak dilirik juga oleh media cetak dan elektronik yang kemudian banyak meliput kegiatan usaha Blackjack Industries, adapun media – media yang pernah meliput Blackjack Industries antara lain BandungTV, Tabloid Peluang Usaha, Business Opportunity, KOMPAS, Pikiran Rakyat, SINDO, dan lain-lain. Liputan tersebut menambah tanggapan pasar menjadi jauh lebih besar sehingga banyak orang yang tertarik untuk menjadi distributor di kotanya masing-masing.

Pada saat ini Blackjack Industries mengadakan perubahan manajemen dalam rangka lebih memfokuskan potensi dalam pengadaan produk dalam jumlah besar demi memenuhi permintaan pasar yang masih sangat potensial perkembangannya.

2.1.2. Logo Perusahaan

Logo dari Blackjack Industries dapat dilihat pada Gambar 2.1.



Gambar 2.1 Logo Blackjack Industries

Makna dari logo pada Gambar 2.1 adalah sebagai berikut: :

1. Nama Distro Blackjack ini diambil dari tanggal lahir pemilik blackjack, yang mana dalam permainan kartu *blackjack* bernilai 21(dua puluh satu), sesuai dengan tanggal lahir pemilik.
2. Gambar “mahkota” di logo bermakna simbol kekuasaan dan bertujuan agar Blackjack Industries akan selalu menjadi *brand* yang menguasai pangsa pasar di Indonesia

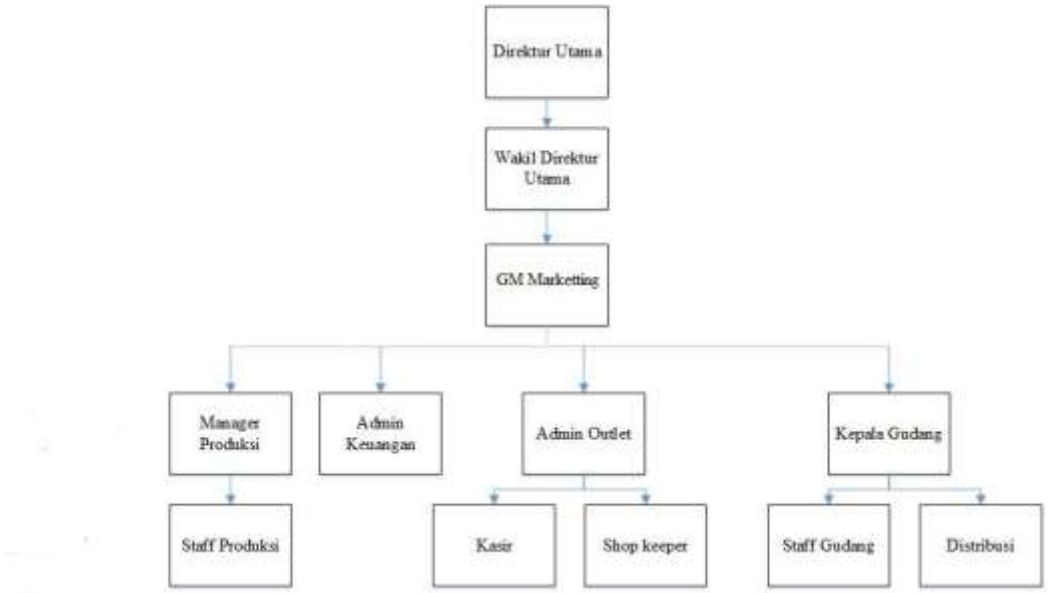
2.1.3. Visi dan Misi

Berikut ini merupakan visi dan misi dari Blackjack Industries :

- Visi distro adalah Memperluas dan mempertahankan usaha yang sedang di jalankan, serta mencari keuntungan sebesar mungkin dengan modal seminim mungkin.
- Misi distro adalah mepertahankan produk asli dan membuat produk sesuai dengan perkembangan jaman

2.1.4. Struktur Organisasi Perusahaan

Struktur organisasi Blackjack Industries dapat dilihat pada Gambar 2.2.



Gambar 2.2 Struktur Organisasi Blackjack Industries

Keterangan posisi :

1. Direktur Utama

   Direktur Utama mempunyai tugas untuk menawarkan visi dan misi, memimpin rapat umum, mengambil keputusan, menjalankan tanggung jawab, melakukan koordinasi peningkatan mutu pelayanan, penelitian, pengabdian dan kerja sama, bertanggung jawab atas semua proses bisnis perusahaan.

2. Wakil Direktur Utama

   Wakil Dirut bertugas utama menyelesaikan permasalahan perusahaan yang tidak bisa di kerjakan oleh direktur utama seperti masalah-masalah internal lainnya.

3. GM Marketting

   GM marketting mempunyai beberapa tugas yaitu :

   1. Bertanggung jawab terhadap Manager Umum

2. Menetapkan tujuan dan sasaran jalannya operasional perusahaan dan strategi pemasaran yang tepat kepada konsumen (strategi bisnis perusahaan).

3. Membuat analisa terhadap pangsa pasar.

4. Menganalisis laporan yang dibuat oleh bawahannya.

4. Manajer Produksi

Manajer produksi mempunyai tugas dalam membuat perencanaan produksi, membuat rencana pembuatan produk serta kegiatan-kegiatan lainnya untuk periode tertentu.

5. Staff Produksi

Bertugas untuk menjalankan produksi yang sedang berjalan..

6. Admin Keuangan

Bertugas untuk mengatur pengeluaran dan pemasukan perusahaan dan membuat laporan yang bersangkutan dengan kegiatan perusahaan.

7. Admin Outlet

Bertugas untuk mengatur kegiatan toko dan meningkatan penjualan di toko.

8. Kasir

Bertugas untuk menerima pembayaran dari pelanggan toko.

9. Shop Keeper

Bertugas menjaga toko dan melayani pelanggan yang datang.

10. Kepala Gudang

Bertugas untuk mengatur pengiriman barang, stock barang yang ada di gudang.

11. Staff Gudang

Bertugas untuk menjalankan kegiatan di gudang.

12. Distribusi

Bertugas untuk melakukan pengiriman barang ke semua outlet.

2.2 . Landasan Teori

Pada bagian landasan teori ini akan menjelaskan teori-teori yang akan digunakan sebagai acuan dalam penelitian yang akan dilakukan.

2.2.1. Data

Data adalah representasi fakta dunia nyata yang mewakili suatu objek seperti manusia (pegawai, siswa, pembeli, pelanggan), barang, hewan, peristiwa, konsep, keadaan, dan sebagainya, yang direkam dalam bentuk angka, huruf, simbol, teks, gambar, bunyi, atau kombinasinya[5]. *Data* dapat juga digunakan sebagai input dan menghasilkan sebuah informasi. *Data* juga merupakan sesuatu yang belum memiliki arti dan masih membutuhkan suatu pengolahan . Dalam *data* terdapat himpunan *data* yang merupakan kumpulan dari objek dan atributnya. Atribut merupakan sifat atau karakteristik dari suatu objek yang biasanya dikenal sebagai *variable, field,* karakteristik atau fitur. Salah satu himpunan *data* adalah *record data* , yaitu data yang terdiri dari sekumpulan *record* yang masing-masing terdiri dari suatu set atribut yang tetap. Salah satu yang termasuk dalam tipe data *record* yaitu data transaksi. *Data* transaksi merupakan sebuah tipe khusus dari *record data,* dimana tiap *record* (transaksi) meliputi satu *set item* [6].

2.2.2. Basis Data

Basis Data dapat didefinisikan dalam sejumlah sudut pandang seperti himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasi sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah. Basis Data dapat diartikan pula sebagai kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redudansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan [4]. Operasi dalam basis data:

1. Pembuatan basis data baru (*create database*).
2. Penghapusan basis data (*drop database*).
3. Pembuatan file/tabel baru ke suatu basis data (*create table*).
4. Penghapusan file/tabel dari suatu basis data (*drop table*).

5. Penambahan/pengisian data baru ke sebuah file/tabel di sebuah basis data (*insert*).
6. Pengambilan data dari sebuah file/tabel (*retrieve/search*).
7. Pengubahan data dari sebuah file/tabel (*update*).
8. Penghapusan data dari sebuah file/tabel (*delete*).

2.2.3. *Database Management System*

Database Management System (DBMS) atau dalam bahasa Indonesia sering disebut sebagai Sistem Manajemen Basis Data adalah suatu sistem aplikasi yang digunakan untuk menyimpan, mengelola, dan menampilkan data. Suatu sistem aplikasi disebut DBMS jika memenuhi persyaratan minimal sebagai berikut [4] :

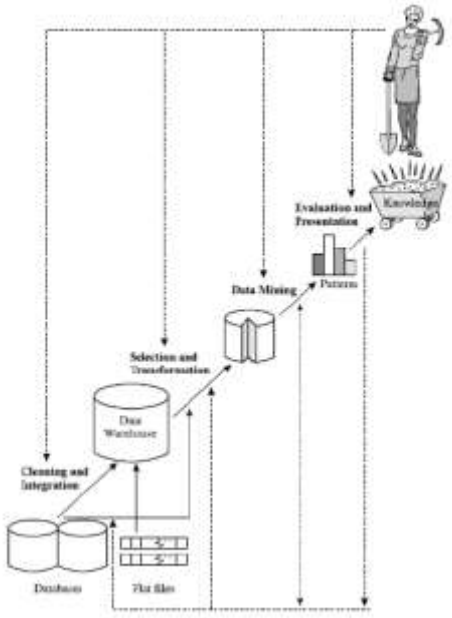
1. Menyediakan fasilitas untuk mengelola akses data.
2. Mampu menangani integritas data.
3. Mampu menangani akses data yang dilakukan secara bersamaan.
4. Mampu menangani *backup* data.

2.2.4. Data Mining

Data *mining* adalah kegiatan mengekstraksi atau menambang pengetahuan dari data yang berukuran atau berjumlah besar. Nama lain yang sering digunakan dan memiliki makna yang sama dengan data *mining* adalah *knowledge mining from data*, *knowledge extraction*, *pattern analysis*, *data archaeology*, *data dredging*.

Data *mining* berusaha memecahkan masalah dengan menganalisis sekumpulan data yang sudah ada dalam basis data. Sedangkan hasil dari proses dari data mining adalah penemuan pengetahuan, informasi, pola, atau model.

Sebagian lain melihat data *mining* hanya sebagai satu langkah penting pada proses KDD (*Knowledge Discovery from Data*). Gambar 2.3 merupakan langkah-langkah dari proses KDD[5].



Gambar 2.3 Tahapan KDD [5]

1. Data *cleaning and Integration*

   Data *cleaning* berguna untuk menghilangkan *noise* data yang tidak konsisten serta mengintegrasikan data, di mana sumber data yang terpecah dapat disatukan.
2. Data *selection and transformation*

   Memilih data yang sesuai dengan analisis yang akan dilakukan dan mengubahnya agar sesuai untuk proses data *mining*..
3. Data *mining*

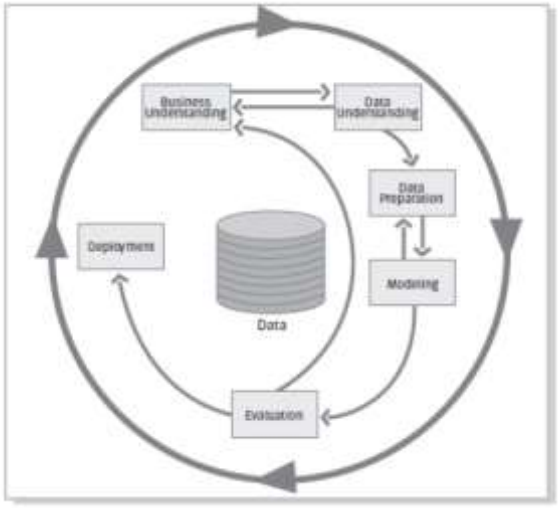
   Proses penerapan metode yang digunakan untuk menambang pola atau pengetahuan dari data.
4. *Evalution and Knowledge Presentation*

   Mengidentifikasi pola yang benar-benar menarik yang mewakili pengetahuan serta memvisualisasikannya kepada pengguna.

2.2.5. Metode CRISP - DM

CRISP – DM (*Cross – Industry Standard of Data Mining*). CRISP – DM merupakan metodologi data *mining* yang digambarkan dengan model proses dan

di dalamnya terdapat tugas-tugas yang harus dilakukan untuk melakukan proyek data *mining*[2] . Tahapan dari CRISP – DM dapat dilihat pada gambar 2.4.



Gambar 2.4 Model Proses CRISP – DM [2]

CRoss-Industry Standard Process for Data Mining (CRISP-DM) adalah metode populer yang digunakan untuk meningkatkan keberhasilan suatu proyek *data mining* [5]. Model proses CRISP-DM terdiri dari enam tahap yang fleksibel.Model ini telah menjadi proses standar proyek *data mining* di banyak perusahaan untuk mendukung keputusan bisnis. Berikut adalah tahap-tahap pada model proses CRISP-DM.

1. Tahap Pemahaman Bisnis

Tahap pemahaman bisnis fokus pada mengidentifikasi tujuan proyek dan kebutuhan dari sudut pandang bisnis. Kemudian mendefinisikan masalah *data mining*. Beberapa kegiatan pada tahap pemahaman bisnis adalah sebagai berikut.

a. Melakukan identifikasi tujuan bisnis,
b. Melakukan penilaian situasi,
c. Menentukan tujuan *data mining*

2. Tahap Pemahaman Data

Tahap pemahaman data dimulai dengan pengumpulan data awal, mempelajari data-data tersebut, dan menilai kualitas data. Kemudian dilakukan

pendeteksian bagian dari data yang mungkin mengandung informasi tersembunyi. Beberapa kegiatan pada tahap pemahaman data adalah sebagai berikut.

a. Mengumpulkan data awal,
b. Mendeskripsikan data yang diperoleh,
c. Mengeksplorasi data, dan
d. Melakukan verifikasi terhadap kualitas data.

3. Tahap Persiapan Data

Tahap persiapan data digunakan untuk membangun data final dari sekumpulan data mentah. Pada tahap ini dilakukan *data preprocessing* agar data dapat digunakan untuk proses pemodelan. Kegiatan pada tahap ini mungkin dilakukan beberapa kali dan tidak dalam urutan yang ditentukan. Beberapa kegiatan tersebut adalah sebagai berikut.

a. Melakukan pemilihan data,
b. Melakukan pembersihan data,
c. Melakukan pembangunan data,
d. Mengintegrasikan data dari berbagai sumber, dan
e. Mentransformasikan data sehingga siap untuk diproses.

4. Tahap Pemodelan

Tahap pemodelan merupakan tahap utama pada proses *data mining.* Pada tahap ini diterapkan teknik pemodelan pada data yang telah disiapkan. Beberapa kegiatan pada tahap pemodelan adalah sebagai berikut.

a. Memilih teknik pemodelan yang sesuai dengan data,
b. Menjelaskan prosedur teknik pemodelan yang digunakan,
c. Melakukan penerapan teknik pemodelan, dan
d. Menilai model yang dihasilkan.

5. Tahap Evaluasi

Setelah model terbentuk, perlu dilakukan evaluasi terhadap langkah-langkah yang dilakukan sebelumnya. Hal tersebut dilakukan untuk memastikan model sesuai dengan tujuan bisnis yang ditetapkan. Beberapa kegiatan pada tahap evaluasi adalah sebagai berikut.

a. Mengevaluasi model yang dihasilkan,

b. Mengkaji ulang proses-proses yang dilakukan, dan

c. Menentukan keputusan penggunaan hasil *data mining.*

6. Tahap Penerapan

Tahap ini merupakan tahap implementasi hasil proses *data mining.* Model yang telah dihasilkan perlu diorganisir dan disajikan kepada pengguna. Hal tersebut biasanya berguna untuk proses pengambilan keputusan perusahaan. beberapa kegiatan pada tahap penerapan adalah sebagai berikut,

a. Menentukan rencana penerapan hasil *data mining,*

b. Menentukan rencana pengawasan dan pemeliharaan,

c. Membuat laporan akhir, dan

d. Melakukan ulasan terhadap proyek yang telah dilakukan.

2.2.6. Data *Preprocessing*

Proses pemodelan pada *data mining* memerlukan data yang berkualitas. Namun data pada sebuah perusahaan sangat mungkin untuk tidak akurat, tidak lengkap dan tidak konsisten. Hal tersebut dapat disebabkan oleh banyak faktor, salah satunya adalah kesalahan manusia atau program saat proses memasukkan data. *Data preprocessing* dapat dilakukan untuk memperbaiki kualitas data, sehingga dapat meningkatkan akurasi dan efisiensi hasil *data mining* [8]. Beberapa kegiatan pada *data preprocessing* adalah sebagai berikut.

1. Menangani nilai kosong

Keberadaan nilai kosong pada data adalah masalah yang sering terjadi. Nilai kosong ini akan mempengaruhi hasil analisis data. Nilai kosong pada data biasanya disebabkan oleh kesalahan input atau suatu atribut yang memang tidak memiliki sebuah nilai. Oleh karena itu, nilai kosong pada data perlu ditangani dengan metode yang sesuai. Salah satu cara yang dapat digunakan adalah menghapus data yang memiliki nilai kosong [8]. Setiap objek yang mengandung nilai kosong akan dihapus agar tidak mempengaruhi informasi yang terdapat dalam data.

2. Menghapus *noise*

Data yang salah, data yang tidak memiliki arti dan *outliers* dapat direpresentasikan sebagai *noise*. Seperti halnya nilai kosong, *noise* juga akan mempengaruhi hasil pemodelan *data mining.* Untuk itu *noise* perlu dihapus agar menghasilkan model yang berkualitas. Cara yang dapat digunakan untuk menghapus *noise* adalah dengan melakukan *smoothing.* Salah satu teknik untuk melakukan *smoothing* adalah dengan menggunakan metode *binning* [8].

Metode *binning* digunakan untuk membagi sekumpulan nilai numerik ke dalam beberapa partisi atau *bin*. *Binning* dimulai dengan mengurutkan setiap nilai pada sebuah atribut. Kemudian setiap nilai dipartisi ke dalam *bin* yang kurang lebih memiliki frekuensi yang sama (*equal-frequency partitioning*). Setelah itu, data pada setiap *bin* diganti dengan nilai batas *bin* terdekat (*smoothing by bin boundaries*). Nilai batas *bin* merupakan nilai minimum dan maksimum pada setiap *bin.*

3. Pembangunan atribut

Pembangunan atribut (*attribute construction*) merupakan proses pembentukan atribut-atribut baru dari atribut yang sudah ada. Teknik tersebut digunakan untuk membantu dalam proses pembentukan model. Penurunan atribut juga berguna untuk menyeragamkan tipe data atribut, sehingga akan lebih mudah dalam proses pemodelan.

2.2.7. Pengelompokan *(Clustering)*

Clustering merupakan pekerjaan yang memisahkan data / *vector* ke dalam sejumlah kelompok (*cluster*) menurut karakteristiknya masing – masing. Data – data yang memiliki kemiripan karakteristik yang berkumpul dalam *cluster* yang sama, dan data – data dengan karakteristik berbeda akan terpisah dalam *cluster* yang berbeda. Tidak diperlukan label baru bisa diberikan ketika *cluster* sudah terbentuk. Karena tidak adanya target label kelas untuk setiap data maka

clustering sering disebut juga pembelajaran tidak terbimbing (*unsupervised learning*) [4].

Teknik *clustering* banyak diterapkan dalam berbagai bidang, seperti kedoktrean, teknik *clustering* dapat digunakan untuk mengelompokan jenis – jenis penyakit berbahaya berdasarkan karakteristik atau sifat – sifat penyakit pasien. Dalam bidang keshatan dapat digunakan untuk mengelompokan jenis makanan berdasarkan kandungan kalori, vitamin, protein. Dalam bidang klimatologi dapat digunakan untuk mengetahui pola angin dan kondisi cuaca di udara sehingga bias diketahui wilayah – wilayah yang retang terhadap cuaca buruk, dan sebagainya.

Clustering dapat dibedakan menjadi dua tujuan, yaitu *clustering* untuk pemahaman dan *clustering* untuk penggunaan. Jika tujuanya untuk pemahaman maka *cluster* yang terbentuk harus menangkap struktur alami data. Biasanya proses *clustering* dalam tujuan ini hanya sebagai proses awal untuk kemudian dilanjutkan dengan pekerjaan inti seperti *summarization* (rata – rata, standar deviasi), pelebelan kelas pada setiap kelompok untuk kemudian digunakan sebagai data latih klasifikasi, dan sebaginya. Sementara jika tujuanya untuk penggunaan, biasanya tujuan utama untuk mencari *prototype cluster* yang paling representative terhadap data dan memberikan abstraksi dari setiap objek data dalam *cluster* di mana sebuah data terletak di dalamnya.

Berdasarkan strukturnya, *clustering* terbagi menjadi dua, yaitu hirarki dan partisi. Dalam pengelompokan berbasis hirarki (*hierarchical clustering*), satu data tunggal bisa dianggap sebuah cluster, dua atau lebih *cluster* kecil dapat bergabung menjadi sebuah *cluster* besar. Begitu seterusnya hingga semua data dapat bergabung menjadi sebuah *cluster*. Pengelompokan berbasis partisi membagi set data ke dalam sejumlah *cluster* yang tidak bertumpang – tindih antara satu *cluster* dengan *cluster* yang lain, artinya setiap data hanya menjadi anggota satu *cluster* saja.

Teknik *clustering* berdasarkan keanggotaanya dibedakan menjadi dua bagian, yaitu eklusif dan tumpang-tindih. Dalam kategori eklusif, sebuah data bias dipastikan hanya menjadi satu *cluster* dan tidak menjadi anggota di *cluster* yang lain. Metode clustering yang termasuk kategori ini adalah *K – Means, DBSCAN*,

dan *SOM.* Sementara yang termasuk kategori tumpang-tindih adalah metode *clustering* yang membolehkan sebuah data menjadi anggota di lebih dari satu *cluster*, misalnya *Fuzzy C-Means.*

Hal yang terpenting dalam *clustering* adalah matriks yang digunakan untuk mengukur ketidak miripan data yang dikelompokan. Penggunaan matriks yang berbeda dapat memberikan hasil yang berbeda, tergantung kasus yang diselesaikan. Matriks yang paling banyak digunakan adalah matriks *Euclidean.* Secara geometris matriks ini memberikan jarak terpendek antara dau data [4].

Menghitung jarak dengan menggunakan Euclidean :

$$d = \sqrt{\sum_{i=1}^{r}(X\,i - Yi)^2}$$ Persamaan 2.1

Dimana :

d = Jarak

r = jumlah fitur dalam vektor

Xi = nilai fitur *X* ke i

Yi = nilai fitur *Y* ke i

2.2.7.1. Teknik *Clustering*

Tujuan utama dari metode *cluster* adalah pengelompokan sejumlah data atau obyek ke dalam *cluster* (grup) sehingga dalam setiap *cluster* akan berisi data yang semirip mungkin. Dalam *clustering* akan berusaha untuk menempatkan obyek yang mirip (jaraknya dekat) dalam satu *cluster* sangat mirip satu sama lain dan berbeda dengan obyek dalam klaster – klaster yang lain[4].

Ada dua pendekatan dalam teknik *clustering*, yaitu partisi dan hirarki. Dalam teknik partisi akan mengelompokan obyek *X1, X2, X3,…., Xn* ke dalam *k cluster.* Ini bias dilakukan dengan menentukan pusat *cluster* awal, lalu dilakukan relokasi obyek berdasarkan kriteria tertentu samapi dicapai pengelompokan yang optimum. Dalam *cluster* hirarki dimulai dengan membuat *m cluster* dimana setiap *cluster* beranggotakan satu obyek dan berakhir dengan satu *cluster* dimana anggotanya adalah *m* obyek. Pada setiap tahap dalam prosedurnya, satu *cluster*

digabung satu *cluster* yang lain. Penghentian atau *cut-off* bias dilakukan ketika jumlah *cluster* yang diinginkan sudah tercapai.

2.2.7.2. Analisis *Cluster* Berbasis Hirarki

Dalam statistik, pengelompokan berbasis hirarki adalah metode analisis *cluster* yang berusaha untuk membangun sebuah hirarki *cluster*. Strategi untuk mengelompokan berbasis hirarki pada umumnya terdapat dua jenis, yaitu *agglomerative* dan *divisive*.

Agglomerative merupakan sebuah metode pengelompokan berbais hirarki dengan pendekatan *bottom up*, yaitu proses pengelompokan dimulai dari masing – masing data sebagai satu buah *cluster,* kemudian secara rekursif mencari *cluster* terdekat sebagai pasangan untuk bergabung sebagai *cluster* yang lebih besar. Proses terbut diulang terus sehingga tampak bergerak ke atas membentuk hirarki.

Divisive merupakan metode pengelompokan data berbasis hirarki dengan pendekatan *top down*, yaitu proses pengelompokan dimulai dari satu *cluster* yang berisi semua data, kemudian secara rekursif memecah *cluster* menjadi dua *cluster* sampai *cluster* hanya berisi data tunggal (data itu sendiri). Untuk cara ini, dibutuhkan adalah keputusan *cluster* yang manakah yang akan dipecah pada setiap langkah dan bagaimana cara memechakanya.

2.2.7.3. *Agglomerative Hierarchical Clustering* (AHC)

Agglomerative hierarchical clustering merupakan algoritma yang digunakan untuk mengukur kedekatan diantara dua *cluster.* Terdapat tiga cara yang digunakan AHC, diantaranya adalah, metode *single linkage, complete linkage, average linkage.* [9]. Peneliti menggunakan metode complete linkage untuk mencari jarak paling jauh.

Single linkage memberikan hasil apabila *cluster – cluster* digabungkan menurut jarak yang paling dekat diantara dua *cluster*. *Complete linkage* terjadi apabila kelompok – kelompok digabung menurut jarak antara yang paling jauh diantara dua *cluster*. *Average linkage* digabungkan menurut jarak rata – rata

antara pasangan – pasangan anggota masing – masing himpunan diantara dua *cluster.*

Berikut ini merupakan tahapan dari algoritma *Agglomerative* :

1. Merepresentasikan setiap outlet sebagai satu kelompok, kemudian menghitung jarak (*distance measure*) antar kelompok-kelompok tersebut.
2. Cari dua *cluster* yang mempunyai jarak paling dekat dan pilih nilai terbesar untuk menentukan nilai baru dari dua cluster yang terbentuk dengan menggunakan metode *complete linkage* yang terdapat pada persamaan (2.2).
3. Ulangi langkah kedua sampai tersisa satu kelompok (kumpulan semua outlet) akan berhenti ketika jumlah *cluster* yang terbentuk sama dengan 1.
4. Ketika Jumlah *cluster* berjumlah 1 maka tahap ke 3 telah selesai dan Hasil akhir dari perulangan berdasarkan ukuran dapat dilihat pada lampiran. Berikut ini merupakan dendrogram dari hasil pengelompokan

Pada metode *compleye linkage* (*Maks*), kedekatan diantara dua *cluster* ditentukan dari jarak terjauh (terbesar) diantara pasangan diantara dua dari *cluster* berbeda (satu dari *cluster* pertama satu dari *cluster* lainnya) atau disebut juga dengan kemiripan yang paling maksimal. Dengan cara seperti ini maka akan dimulai dari masing – masing data sebagai *cluster*, kemudain mencari tetangga terdekat dan menggunakan *complete linkage* untuk menggabungkan dua *cluster* berikutnya hingga semuanya bergabung menjadi satu *cluster.*

Berikut ini merupakan persamaan *single linkage* :

$d(U,V)$=maks$\{d(U,V)\}; d(U,V) \in D$ persamaan 2.2

D = Matriks kedekatan jarak antar *cluster*.

2.2.7.4. Contoh Penerapan *Clustering* Menggunakan Algortima AHC

Contoh diberikan 5 data dan pengukuran jarak menggunakan jarak *Euclidean*, dengan menggunakan metode *complete linkage.*

Tabel 2.1 Tabel Pengukuran Jarak Menggunakan *complete Linkage*

Data ke - i	Fitur x	Fitur y

1	1	1
2	4	1
3	1	2
4	3	4
5	5	4

Yang pertama dilakukan adalah menghitung jarak dengan menggunakan *euclidean* pada semua pasangan dua data. Hasil dari perhitungan dengan menggunakan *euclidean* adalah sebagai berikut :

$$D = d(U,V) = \begin{bmatrix} 0 & 3 & 1 & 5 & 7 \\ 3 & 0 & 4 & 4 & 4 \\ 1 & 4 & 0 & 4 & 6 \\ 5 & 4 & 4 & 0 & 2 \\ 7 & 4 & 6 & 2 & 0 \end{bmatrix}$$

Gambar 2.5 Hasil Perhitungan Euclidean

Dengan memperlakukan setiap data sebagai *cluster* maka *cluster* yang terbentuk pada saat ini adalah sebanyak 5 *cluster* atau jumlah *cluster K* = jumlah data *n,* tahapan selanjutnya adalah memilih jarak dua *cluster* yang paling kecil. min{$d(U,V)$}=d13=1

Dari proses perhitungan dengan menggunakan metode *complete linkage* maka didapatkan bahwaa jarak yang paling minimum adalah *cluster* 1 dan *cluster* 3, maka kedua *cluster* ini akan digabung. Untuk melanjutkan tingkat pengelompokan berikutnya maka jarak – jarak antara *cluster* (13) dengan *cluster* yang tersisa 2,4, dan 5 dihitung kembali dengan menggunakan metode *single linkage*. Jarak – jarak yang didapatkan adalah:

d(1,3)2 = maks {d12 ,d(32)} = maks {3,4} = 4

d(1,3)4 = maks {d14 ,d(34)} = maks {5,4} = 5

d(1,3)5 = maks {d15 ,d(35)} = maks {7,6} = 7

Dengan menghapus baris – baris dan kolom – kolom matriks D yang bersesuain dengan *cluster* 1 dan 3 dan menambahkan baris dan kolom untuk

cluster 1 dan 3, dan menambahkan baris dan kolom untuk *cluster* (13), maka didapatkan matriks yang baru :

$$D = d(U,V) = \begin{matrix} (13) \\ 2 \\ 4 \\ 5 \end{matrix} \begin{bmatrix} 0 & 4 & 5 & 7 \\ 3 & 0 & 4 & 4 \\ 4 & 4 & 0 & 2 \\ 6 & 4 & 2 & 0 \end{bmatrix}$$

Gambar 2.6 Matriks Hasil Perhitungan Jarak *Complete Linkage (1)*

Tahapan selanjutnya adalah memilih kembali jarak dua *cluster* yang paling kecil.

$$\min\{d(U,V)\} = d45 = 2$$

Maka *cluster* yang terpilih adalah *cluster* 4 dan 5, maka *cluster* 4 dan *cluster* 5 digabung. Kemudian untuk menghitung jarak – jarak antara *cluster* (45) dengan *cluster* lain yang tersisa yaitu (13) dan 2 dihitung kembali dengan menggunakan metode *complete linkage*. Jarak – jarak yang didapatkan adalah :

d(45)(13) = maks {d41 ,d43, d51, d53)} = maks {5,4,7,6} = 7

d(45)2 = maks {d42 ,d(52)} = maks {4,4} = 4

dengan menghapus baris – baris dan kolom – kolom matriks D yang bersesuaian dengan *cluster* 4 dan *cluster* 5 dan menambahkan baris dan kolom untuk *cluster* (45), maka didapatkan matriks jarak yang baru :

$$D = d(U,V) = \begin{matrix} (45) \\ (13) \\ 2 \end{matrix} \begin{bmatrix} 0 & 7 & 4 \\ 4 & 0 & 3 \\ 4 & 3 & 0 \end{bmatrix}$$

Gambar 2.7 Matriks Hasil Perhitungan Jarak *Complete Linkage (2)*

Selanjutnya dipilih kembali jarak dua *cluster* yang paling kecil $\min\{d(U,V)\}=d(13)2=3$. Terpilih *cluster* (13) dan 2, maka *cluster* (13) dan 2 digabung. Untuk melanjutkan tingkat *clustering* berikutnya, maka jarak – jarak antara *cluster* (13) dan 2 dengan *cluster* yang lain yang tersisisa yaitu (45) dihitung kembali dengan menggunakan metode *complete linkage*. Jarak – jarak yang didapatkan adalah :

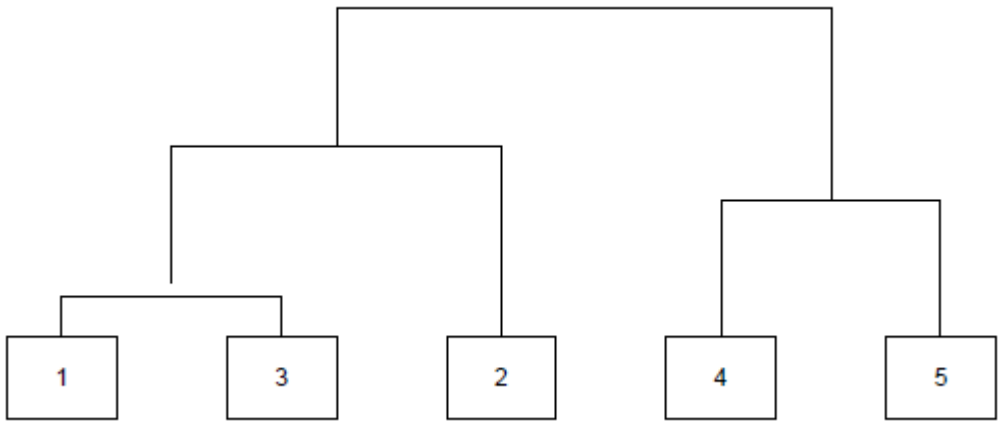
d(123)(45) = maks {d14 ,d43, d24, d25, d34, d35)} = maks {5,7,4,4,4,6} = 7. Dengan menghapus baris – baris dan kolom – kolom matriks D yang bersesuaian dengan *cluster* (13) dan 2, dan menambahkan baris dan kolom untuk *cluster* (123), maka matriks yang baru yang didapatkanya adalah :

$$D = d(U,V) = \begin{matrix} \\ (123) \\ (45) \end{matrix} \begin{bmatrix} 0 & 7 \\ 7 & 0 \end{bmatrix}$$

Gambar 2.8 Matriks Hasil Perhitungan Jarak *Complete Linkage (3)*

Jadi *cluster* (123) dan (45) digabung membentuk *cluster* tunggal dari semua 5 data, (12345), ketika jarak terdekat mencapai 7.

Hasil *clustering* yang didapatkan dapat ditampilkan dalam bentuk dendogram, seperti terlihat dalam gambar 2.10 :



Gambar 2.9 Dendrogram Hasil *Clustering* Berbasis Hirarki

2.2.7.5. *Agglomeration Schedule*

Agglomeration schedule adalah ringkasan numerik dari solusi *cluster* yang dihasilkan dari proses *clustering* menggunakan metode hierarki. [12] Berikut ini merupakan contoh dari *agglomeration schedule* :

Stage	Cluster Combined		Coefficients	Stage Cluster First Appears		Next Stage
	Cluster 1	Cluster 2		Cluster 1	Cluster 2	
1	8	11	1.260	0	0	7
2	6	7	1.579	0	0	4
3	2	9	1.625	0	0	7
4	1	6	2.318	0	2	6
5	3	5	2.619	0	0	8
6	1	10	3.670	4	0	10
7	2	8	4.420	3	1	8
8	2	3	4.505	7	5	9
9	2	4	4.774	8	0	10
10	1	2	5.718	6	9	0

Gambar 2.10 Contoh *Agglomeration Schedule*

Penjelasan tabel *agglomeration schedule* :

1. *Stage* merupakan tahapan atau iterasi yang telah dilakukan pada proses *clustering* menggunakan metode hierarki.
2. *Cluster combined* merupakan *cluster* yang digabungkan pada setiap *stage*nya.
3. *Cooficients* merupakan jarak dari dua *cluster* yang digabungkan pada setiap *stagenya* yang dicari dengan menggunakan rumus jarak.
4. *Stage cluster first appear* merupakan *stage* apabila *cluster* yang digabungkan telah digabungkan dengan *cluster* lain pada *stage* sebelumnya. Jika *cluster* yang digabungkan belum pernah digabungkan pada *stage* sebelumnya, maka akan diisi dengan nilai 0.
5. *Next stage* merupakan *stage* selanjutnya saat *cluster* akan digabungkan dengan *cluster* lain. [12]

2.2.8. *Object Oriented Programming*

Pemograman berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Menggunakan

pendekatan berorientasi objek akan memandang sistem yang akan dikembangkan sebagai suatu kumpuln objek yang berkorespondensi dengan objek dunia nyata.

Dalam rekayasa perangkat lunak, konsep pendekatan berorientasi objek dapat diterapkan pada tahap analisis, perancangan, pemograman, dan pengujian perangkat lunak. Berikut adaah beberapa konsep dasar yang harus dipahami pemograman berorientasi objek [3] :

1. Kelas (*Class*)

   Kelas adalah kumpulan objek-objek dengan karakteristik yang sama. Kelas merupakan definisi statik dan himpunan objek yang sama yang mungkin lahir atau diciptakan dari kelas tersebut. Sebuah kelas akan mempunyai sifat (atribut), kelakuan (operasi/metode), hubungan (*relationship*) dan arti. Suatu kelas dapat diturunkan dan kelas semula dapat diwariskan ke kelas yang baru.

2. Objek (*Object)*

   Objek adalah abstraksi dan sesuatu yang mewakili dunia nyata seperti benda, manusia, satuan organisasi, tempat, kejadian, strutur, status, atau hal-hal lain yang bersifat abstrak. Objek merupakan suatu entitas yang mampu menyimpan informasi (status) dan mempunyai operasi (kelakuan) yang dapat diterapkan atau dapat berpengaruh pada status objeknya. Objek mempunyai siklus hidup yaitu diciptakan, dimanipulasi, dan dihancurkan.

3. Metode (*Method*)

   Operasi atau metode pada sebuah kelas hampir sama dengan fungsi atau prosedur pada terstruktur. Sebuah kelas boleh memiliki lebih dari satu metode atau operasi. Metode atau operasi yang berfungsi untuk memanipulasi objek itu sendiri.

4. Atribut (*Attribute*)

   Atribut dari sebuah kelas adalah variabel global yang dimiliki sebuah kelas. Atribut dapat berupa nilai atau elemen-elemen data yang dimiliki oleh objek dalam kelas objek. Atribut secara individual oleh sebuah objek, misalnya berat, jenis, nama, dan sebagainya.

5. Abstraksi (*Abstraction*)

   Prinsip untuk merepresentasikan dunia nyata yang kompleks menjadi satu bentuk model yang sederhana dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan permasalahan.

6. Enkapulasi (*Encapsulation*)

   Pembungkusan atribut data dan layanan (operasi-operasi) yang dipunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerja.

7. Pewarisan (*Inheritance*)

   Mekanisme yang memungkinkan satu objek mewarisi sebagian atau seluruh definisi dan objek lain sebagai bagian dari dirinya.

8. Antarmuka (*Interface*)

   Antarmuka atau interface sangat mirip dengan kelas, tetapi tanpa atribut kelas dan tanpa memiliki metode yang dideklarasikan. Antarmuka biasanya digunakan agar kelas lain tidak langsung mengakses ke suatu kelas.

9. *Reusability*

   Pemanfaatan kembali objek yang sudah didefinisikan untuk suatu permasalahan pada permasalahan lainnya yang melibatkan objek tersebut.

10. Generalisasi dan Spesialisasi

    Menunjukkan hubungan antara kelas dan objek yang umum dengan kelas dan objek yang khusus. Misalnya kelas yang lebih umum (generalisasi) adalah kendaraan darat dan kelas khususnya (spesialisasi) adalah mobil, motor, dan kereta.

11. Komunikasi antar objek

    Komunikasi antar-objek dilakukan lewat pesan (*message*) yang dikirim dan satu objek ke objek lainnya.

12. Polimorfisme (*Polymorphism*)

    Kemampuan suatu objek untuk digunakan dibanyak tujuan yang berbeda dengan nama yang sama sehingga menghemat baris program.

13. *Package*

    Package adalah sebuah kontainer atau kemasan yang daoat digunakan untuk mengelompokkan kelas-kelas sehingga memungkinkan beberapa kelas yang bernama sama disimpan dalam package yang berbeda.

2.2.9. *Unified Modelling Language* (UML)

UML singkatan dari *Unifed Modeling Language* yang berarti bahasa pemodelan standar. UML merupakan bahasa standar untuk merancang dan mendokumentasikan perangkat lunak dengan cara berorientasi objek. Ada beberapa diagram yang digunakan proses pembuatan perangkat lunak berorientasi objek diantaranya, *use case diagram, activity diagram, class diagram* dan *sequence diagram* [3].

2.9.1. Use Case Diagram

Use case diagram merupakan pemodelan untuk tingkah laku (*behavior*) pada sistem yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. *Use case diagram* digunakan untuk mengetahui fungsi apa saja yang terdapat pada sistem. Terdapat dua hal utama yang diperlukan dalam pembentukan suatu *use case diagram* yaitu aktor dan *use case*.

1. Aktor merupakan orang, benda maupun sistem lain yang berinteraksi dengan sistem yang akan dibangun.
2. *Use Case* merupakan fungsionalitas atau layanan yang disediakan oleh sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

2.9.2. Class Diagram

Class diagram menggambarkan interaksi dan relasi antar kelas yang ada di dalam suatu sistem. Kelas memiliki atribut dan metode. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas. Metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas. Atribut dan metode dapat memiliki salah satu sifat sebagai berikut:

1. *Private,* tidak dapat dipanggil dari luar kelas yang bersangkutan.
2. *Protected*, hanya dapat dipanggil oleh kelas yang bersangkutan dan anak-anak yang mewarisinya.
3. *Public,* dapat dipanggil oleh siapa saja.

Class diagram menggambarkan relasi atau hubungan antar kelas dari sebuah sistem. Berikut ini beberapa gambaran relasi yang ada dalam *class diagram*:

1. *Association*

   Hubungan antar class yang statis. *Class* yang mempunyai relasi asosiasi menggunakan *class* lain sebagai atribut pada dirinya.

2. *Aggregation*

   Relasi yang membuat *class* yang saling terikat satu sama lain namun tidak terlalu berkegantungan.

3. *Composition*

   Relasi agregasi dengan mengikat satu sama lain dengan ikatan yang sangat kuat dan saling berkegantungan.

4. *Dependency*

   Hubungan antar *class* dimana *class* yang memiliki relasi *dependency* menggunakan *class* lain sebagai atribut pada *method*.

5. *Realization*

   Hubungan antar *class* dimana sebuah *class* memiliki keharusan untuk mengikuti aturan yang ditetapkan *class* lainnya.

2.9.3. Sequence Diagram

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan *sequence diagram* maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu.

Banyaknya *sequence diagram* yang harus digambar adalah sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada

sequence diagram sehingga semakin banyak *use case* yang didefinisikan maka *sequence diagram* yang harus dibuat juga semakin banyak. Penomoran pesan berdasarkan urutan iteraksi pesan. Penggambaran letak pesan harus berurutan, pesan yang lebih atas dari lainnya adalah pesan yang berjalan terlebih dahulu.

2.9.4. Activity Diagram

Activity diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem, proses bisnis atau menu yang ada pada perangkat lunak. Setiap *use case* yang telah dibentuk digambarkan aktivitasnya dalam *activity diagram*, mulai dari peran aktor, peran sistem, dan *decision*. *Activity diagram* juga banyak digunakan untuk mendefinisikan hal-hal berikut:

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

2.2.10 Java

Java adalah bahasa pemrograman yang dapat membuat seluruh bentuk aplikasi, desktop, web, mobile dan lainnya, sebagaimana dibuat dengan menggunakan bahasa pemrograman konvensional yang lain. Bahasa Pemrograman Java ini berorientasi objek (OOP-Object Oriented Programming), dan dapat dijalankan pada berbagai platform sistem operasi. Perkembangan Java tidak hanya terfokus pada satu sistem operasi, tetapi dikembangkan untuk berbagai sistem operasi dan bersifat open source. Dengan slogannya ***"Write once, run anywhere"*** [7].

2.2.12. MySQL

Database MySQL merupakan sistem manajemen basis data SQL yang sangat terkenal dan bersifat *open source*. MySQL dibangun, didistribusikan, dan didukung oleh MySQL AB. MySQL AB merupakan perusahaan komersial yang dibiayai oleh pengembang MySQL. MySQL mempunyai dua macam lisensi yaitu lisensi yang bersifat *open source* dengan menggunakan GNU (*General Public License*) dan lisensi kedua berupa *Standard Commercial License* yang dapat dibeli dari MySQL AB [4].

Dalam konteks bahasa SQL, informasi disimpan dalam tabel-tabel yang secara logis merupakan struktur dua dimensi yang tersimpan atas baris-baris data (*row* atau *record*) yang berada dalam satu atau lebih kolom (*column*). Baris pada tabel disebut *insance* dari data sedangkan kolom sering disebut sebagai *atributes* atau *field*. MySQL yang berjalan di sistem operasi *Windows* terbukti sangat stabil. MySQL versi *Windows* mempunyai fitur yang sama dengan MySQL versi *Unix*.